Techniques for Calibrating Pedestrian Dead Reckoning Parameters using Smartphones

DOCTORAL THESIS

Thomas Moder, Dipl.-Ing.

Supervisor

Ao.Univ.-Prof. Dipl.-Ing. Dr.techn. Manfred Wieser

Institute of Geodesy

Graz University of Technology

Graz, July 2019

AFFIDAVIT

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present doctoral thesis.

Date

Signature

Acknowledgements

I feel, writing a thesis is very much comparable to climbing an alpine face. The objective is quite appealing and attractive, looking at it from the outside. Once you started climbing, or writing, there are points where retreating is possible in a safe way, without loosing too much time. However, there is a certain point where you are so committed, that you basically have to push on and finish it. At times you do suffer, you are fatigued, you forget to drink enough and you solely live off cookies. But reaching the summit is always a huge relief. And even when making it to the summit is always magical, you usually start to question yourself, why you even started this objective in the first place, and if it really is worth all the trouble and risks. However, once you survive it, and make it back home safe, it was always worth it, and you would always do it again.

Similarly, during writing this thesis, I suffered, not so much physically but mentally, and I did question if it is worth it at times. However, now that it is almost finished, I start to feel a huge relief. I consider myself glad that I was supported, consistently encouraged, and pushed by my whole Working Group, friends and family in this phase. In all seriousness, I doubt that I would have finished this work without all your encouragement.

Herewith, I sincerely want to thank my colleagues from Graz University of Technology, for enabling me to work on and write this thesis. Mainly, Dr. Manfred Wieser, who supervised me during the whole time of this work, and constantly and actively supported me working on and writing and finishing this thesis. In all honesty, and especially true for the last year, I could not have asked for a more encouraging supervisor. Also, I want to sincerely thank Dr. Günther Retscher, for evaluating and supporting this thesis. His honest interest in my work and his useful inputs helped me finish this manuscript. Next to Dr. Manfred Wieser and Dr. Günther Retscher, I want to thank my colleagues Clemens Reitbauer for our collaboration within graph-based applications, Karin Wisiol for our collaboration within motion recognition and Roman Wilfinger for our collaboration within application development, as well as Eva Reitbauer and Dr. Norbert Kühtreiber for gently pushing me in the right direction. Additionally, I want to thank my former colleague Dr. Petra Hafner from Graz University of Technology, and Dr. John-Olof Nilsson, research fellow from the KTH Stockholm, both for their inputs during my work.

Next to the people from my Working Group, my roped party for this ascent consisted of my closest friends and my family. I want to thank all my friends who went through similar experiences and supported me in this phase. Mainly Robert, who always encouraged me with his opinions, Heather, who threatened to chain me to my desk to finish my manuscript and who would let me rant all night if necessary, and Matthias, who tried his best to make it seem that writing a thesis is no big deal in reality. I also want to thank my family, for enabling me to be able to study on and work in jobs I actually wanted. For my whole life so far, I was able to only work and focus on things I truly enjoyed.

Last but foremost, my thanks go to Elisabeth and Laura. Elisabeth, for always trying to be there and having my back, no matter what. Your selfless way of supporting me helped me climb higher than I ever imagined. And Laura, for setting my perspectives right. Whenever I reach a summit or finish an objective, the fog seems to vanish, and life and the world suddenly appear easy and clear to me. This is always the point, when I can feel and see what is actually important in life. I doubt that anything will ever be more important to me as you and Elisabeth are now.

Abstract

The rise of smartphones as universally present device opens possibilities for new location-based services, and especially for pedestrian navigation. Pedestrians, already carrying a smartphone, can make use of the integrated sensors to provide navigation. One real threat for position estimation is the disturbance of radio signals usable for positioning. Therefore, a self-contained position solution, not dependent on radio signals, is necessary for some location-based services for pedestrian navigation.

The main concept in self-contained positioning approaches utilizing smartphone sensors is pedestrian dead reckoning. Pedestrian dead reckoning comprises detecting steps and estimating the step length and step orientation, both using the acceleration and angular rate sensors of a smartphone. The most robust way to estimate steps in real time is to use a signal segmentation for the total acceleration signal. For estimating the step length of the detected steps, multiple models exist. However, not the chosen model itself is of main importance, but the correct calibration of the model parameters. For the step orientation, the relative change in orientation, computed out of the leveled and bias-corrected angular rates, is utilized. Additionally, the detection of the current locomotion mode of the pedestrian, using machine learning, is a huge benefit in improving the pedestrian dead reckoning.

For optimal results, the smartphone sensors are calibrated. For the acceleration sensors, the scale and bias errors are stable and are pre-calibrated. For the angular rate sensors, the bias errors are estimated on-route. More importantly, the model parameters for step length estimation as well as locomotion recognition are calibrated. An algorithm to estimate these model parameters in an automated fashion, utilizing reference sensors, is developed and presented. This calibration method results in errors in traveled distance for the step length estimation of about 1.6%, compared to 5% when using universal models. An automated classification process for locomotion detection based on machine learning is developed and trained, which correctly identifies the locomotion mode in about 95% of samples.

A calibrated pedestrian dead reckoning, using locomotion recognition, is used as input for an optimal position filtering for pedestrian navigation. Using raster maps, a particle filter is utilized. Using vector maps based on a routing graph, matching of the estimated positions of the pedestrian on the graph is utilized. Especially for people with visual impairments, where existing tactile paths are modeled as a graph, the application of pedestrian dead reckoning is directly used for navigation.

Zusammenfassung

Die Entwicklung von Smartphones eröffnet neue Möglichkeiten für die Navigation von Fußgängern. Fußgänger, welche ohnehin ein Smartphone bei sich tragen, können die enthaltenden Sensoren nutzen, um eine Positionsbestimmung zu berechnen. Eine Gefahr hierfür sind die immer stärker werdenden Möglichkeiten zur Verfälschung von Radiosignalen. Deswegen ist es notwendig, eigenständige Positionierungsverfahren zu verwenden, welche nicht von Radioquellen abhängig sind.

Das bekannteste Konzept für eine eigenständige Positionierung von Fußgängern mit Smartphone-Sensoren heißt *Pedestrian Dead Reckoning*. *Pedestrian Dead Reckoning* beinhaltet die Detektierung von Schritten, sowie die Schätzung der Länge und Orientierung dieser Schritte. Der direkteste Weg zur Schrittdetektierung ist die Signalsegmentierung des Beschleunigungssignals. Für die Schätzung der Schrittlänge existieren verschiedene Modelle, jedoch ist es am wichtigsten, das verwendete Modell richtig zu kalibrieren. Für die Schrittorientierung werden die kalibrierten Drehratensensoren zur Schätzung der relativen Richtungsänderung verwendet. Zusätzlich ist die Erkennung der aktuellen Fortbewegungsart ein wichtiger Parameter für *Pedestrian Dead Reckoning*.

Für eine optimale Positionsschätzung werden die Smartphone-Sensoren kalibriert. Für die Beschleunigungssensoren werden die Bias-Fehler und Skalenfaktoren langfristig bestimmt. Für die Drehratensensoren werden die Bias-Fehler in Echtzeit bestimmt. Am wichtigsten ist die Kalibrierung der Modellparameter für die Schrittlängenschätzung und die Erkennung der Fortbewegungsart. Ein Algorithmus zur automatischen Schätzung dieser Modellparameter, unter Verwendung von Referenzsensoren, wird entwickelt und angewandt. Diese Kalibrierung der Schrittlängenparameter erzielt Abweichungen in der zurückgelegten Distanz im Bereich von 1.6%, während mit der Verwendung von globalen Parametern 5% zu erwarten sind. Die automatische Klassifizierung der Fortbewegungsart, durch Techniken des maschinellen Lernens, erzielt 95% Genauigkeit.

Ein kalibriertes *Pedestrian Dead Reckoning* einschließlich der Erkennung der Fortbewegungsart, dient als Ausgangspunkt für eine Positionsbestimmung für die Fußgängernavigation. Mit Rasterkarten wird ein Partikelfilter angewendet. Mit der Verwendung von Vektorkarten wird die Methode der Graphen-gestützten Positionierung angewendet. Speziell für Menschen mit Sehbeeinträchtigungen wird eine solche Graphen-gestützte Positionierung, unter der Verwendung von taktilen Wegen als Graph, zur Navigation verwendet.

Contents

1 Introduction

Navigation generally comprises some sort of route finding, the estimation of the current position as well as the guiding process. It is often described with $navigation = routing + positioning + guidance$, in whichever order you might want to discuss them. Concepts for routing exist thoroughly, and mostly the currentness of the used map data is of importance for a correct routing. In this work, the main focus, when discussing navigation, will be the positioning part of the equation. Similarly, concepts for guidance exist, and the main aspect for a robust guidance is a robust and correct position estimation.

Historically, the evolution of positioning happened in phases, over multiple decades. Broadly speaking, positioning approaches may be divided into using landmarks, angle and distance measurements, dead reckoning, radio signals and inertial signals. Note, that dead reckoning and the development of radio signals overlap in their time line.

The break through in position estimation for navigation, was the research done on using radio signals for measuring distances. Starting with using very low frequency, land-based approaches nearly 100 years ago, this evolved to using low frequency, land-based approaches, designed for ship navigation and known as Long Range Navigation Systems. Following, the concept of satellite-based radio navigation was developed nearly 40 years ago, and is known with the implementation of the Global Positioning System. Today, virtually all techniques used for absolute position estimation for navigation, are somehow based on measuring radio signals, or their signal strength.

The applications of navigation can be divided into the mode of transportation, and the importance of a correct navigation. The development started with navigation systems for ships, followed by planes, large land vehicles and then cars. Since some years, concepts for pedestrian navigation exist. Where for ships, and especially planes, the safety critical part of navigation is overall present, this is not necessarily true, in the same depth, for pedestrian navigation. The presented evolution from ships to pedestrians is directly correlated with the evolution in accuracy and miniaturization regarding positioning and computing technologies. Especially for pedestrian navigation, the main aspect was the introduction of smartphones as universally available devices. Today, smartphones possess the necessary sensors needed for positioning, as well as the computational performance needed for computing routing and guidance processes, in real time.

1.1 Pedestrian and indoor navigation

Pedestrian navigation is a sub task of navigation, which developed to be of importance within the last decade. While specially tailored solutions exist for longer than this, e.g. solutions tailored for people with restrictions in their mobility, easily available navigation solutions for pedestrians became of importance since the development of digital personal assistants, and later, the development of smartphones.

Strongly correlated to the evolution of pedestrian navigation is the evolution of solutions for indoor navigation. This is mostly due to the fact, that pedestrians are indoors as well as outdoors. Where ships and planes only navigate outdoors, cars can be indoors as well. However, typical indoor areas for car navigation, comprise e.g. tunnels, which can be easily modeled. Contrary, indoor situations for pedestrians cannot be accounted for using simple models. Therefore, positioning approaches for indoors are needed for pedestrian navigation. Where the concepts of routing and guidance do not differ between outdoors and indoors, the concepts for positioning do differ vastly.

Therefore, indoor positioning and pedestrian positioning is an emerging field within research in the last couple of years. This is accommodated for by the creation of the International Conference of Indoor Positioning and Indoor Navigation, which is a yearly research conference. The first edition of this conference took place in 2010. From 2014 to 2018, five conference contributions, where the author of this work acted as main author of these contributions, were presented at the conference in the field of indoor and pedestrian positioning, see Moder et al. (2014), Moder et al. (2015), Moder et al. (2016), Moder et al. (2017) and Moder et al. (2018). These contributions are the base for the presented innovations in the following chapters of this work.

The device smartphone, is the obvious platform for pedestrian navigation. On the one hand, the evolution of smartphones enables multiple approaches for positioning and navigation for pedestrians, but on the other hand, forces solutions to work with sensors already present in smartphones. The most stable and robust positioning approaches, usually strive to combine absolute and relative positioning techniques, in order to use the main advantages from both approaches. In the domain of absolute positioning techniques for smartphones, multiple forms of radio positioning are possible. In the domain of relative positioning for pedestrians, dead reckoning and inertial navigation techniques are possible. Using smartphones for relative positioning for pedestrians, the main approach known today is called Pedestrian Dead Reckoning (PDR). PDR is a dead reckoning method, computed with the inertial sensors of the smartphones. Starting as a small area of interest in the research community, in 2017 and 2018, PDR was the session with the most contributions in the conference for indoor positioning and indoor navigation.

1.2 Motivation

With the development of the smartphone, the development of the sensors built into it, and the possibility to access the raw sensor data and apply algorithms to that data, opened new possibilities for positioning approaches and therefore navigation applications. Within multiple applications for pedestrian navigation, the need for self-contained, relative positioning arises. Multiple approaches have been proposed, relying on highly priced equipment, but the main concept already available, is PDR. Today, PDR is already widely spread, and the most basic concept of PDR is already usable and used, e.g. utilizing the built in *Google step detector*.

This rise of PDR using smartphones, as area of interest for the research community, and as positioning concept already present and used within applications, raises multiple research questions, including:

- How do approaches for pedestrian positioning work?
- Is it possible to improve the current concepts for pedestrian positioning?
- What is needed to improve current models for PDR?
- Is it necessary to calibrate the parameters used within the PDR models?
- Can these model parameters be estimated in an automated fashion?
- How can these PDR approaches be optimally applied for pedestrian navigation?

The presented work strives to answer these research questions.

Concluding, the main research motivation behind this work, is to explain and improve the current concept of PDR using smartphones for pedestrian navigation. The shown approaches are small improvements and innovations, however, they might be usable and useful for some future applications.

1.3 Structure of thesis

The presented work is mainly structured in three blocks. Starting, two chapters, outlining the basic fundamentals needed for understanding the concepts of pedestrian navigation, i.e. chapters 2 and 3 are given. Then chapters 4 and 5 explain the developed innovations regarding pedestrian positioning and the calibration and automation of the developed approaches. Closing, two chapters, where the general and specific application of the presented approaches are discussed, are given with chapters 6 and 7.

Within chapter 2, the basics of pedestrian navigation, such as reference systems, transformations and rotations, basics of least squares and filtering, particle filter and graph-based approaches for positioning, basics of pattern recognition and their application within pedestrian navigation, are outlined.

Within chapter 3, the development of smartphones, in general, as well specifically as navigation device for pedestrians, is described. Additionally, the smartphone applications developed and used within the following chapters are discussed.

Chapter 4 describes the possible modes of positioning for pedestrians, using smartphone sensors. The focus is, describing possible modes of positioning and mainly approaches which work indoors for pedestrians. Additionally, the framework established to compute and estimate the main modes of positioning, specifically for pedestrian positioning, are outlined and explained.

Within chapter 5, the developed calibration approaches for calibrating the smartphone sensors, as well as PDR models, are shown. Additionally, the possible automation of these calibrations is developed and discussed.

In chapter 6, the main applications for using PDR are shown and discussed. The effects of calibrating PDR parameters as well as the automation of these calibrations are shown in front of the distinct stages of PDR, as well as in front of position filtering, using raster and vector maps.

Following, chapter 7 shows and explains the specific application developed within this work, using PDR as a tool for graph-based filtering, utilizing tactile paths for the navigation of people with visual impairments.

Closing, chapter 8 provides a summary, a conclusion and discussion, and an outlook of possible further developments. Practical applications of PDR, including the shown calibration approaches, are outlined.

2 Basics and fundamentals of pedestrian navigation

The following chapter outlines the basic fundamentals of pedestrian navigation. Starting with the explanation of navigation frames, transformations and rotations within pedestrian navigation and inertial and magnetic navigation, and the sensors used for computing position solutions are described. Followed by the mathematical concept of least squares used within parameter estimation within certain sensor- and algorithmic calibrations, as well the basics of filtering and graphs in the position domain are described. Finally, the basic constraints within pedestrian positioning and pedestrian guidance are being discussed.

2.1 Navigation frames and transformations

Navigation of pedestrians describes the positioning and guiding process in and with reference to a coordinate frame. In order to make use of the computed and derived positions postulated within this thesis, the received position estimates and orientation angles are given in certain reference frames. The basic reference frames used and discussed herewith are the instrument or sensor frame, the body frame, the local level frame, the earth centered terrestrial frame and the inertial frame. In between the presented reference frames, the orientation angles, rotations and transformations are outlined.

2.1.1 Reference frames

Sensor frame:

The sensor or instrument frame is not always utilized within navigation applications, because most often it is desired to mount the instrument directly to the body used. Examples therefore are a camera on a mobile mapping platform, a radio antenna on an airplane or a wheel sensor in a car. However, within pedestrian navigation utilizing smartphones, the sensor, respectively the smartphone, is not necessarily fixed to the body, the pedestrian. Therefore the introduction of the instrument or smartphone frame is necessary. The origin is usually the center of the sensor, with the directions being defined along the sensor axis. Specifically

for smartphones with *Android* operating systems, the axis are defined within the operating system according to developers.android.com (2018), see figure 2.1.

Body frame:

The body frame is usually defined with the body executing the motions, for example cars or airplanes, and therefore in other sources also described as vehicle frame. This is true for pedestrian navigation as well, the desired position and guidance instructions are those for the pedestrian carrying the smartphone, not the smartphone itself. The origin itself is usually the center of the pedestrian, with the forward axis being defined as the usual walking direction of the pedestrian and the up axis being defined as the body axis of the pedestrian, and the right axis being rectangular.

For some applications, it is assumed that the instrument frame equals the body frame, which is technically not true. However, in most mounting scenarios, the relative orientation and position of the sensor or smartphone usually stays constant for a certain period.

Figure 2.2 shows the body system of a pedestrian carrying a smartphone with its sensor system. The pedestrian itself is then navigating within a local-level system, which is connected to an earth-centered and earth-fixed terrestrial system. Finally, the terrestrial system, specifically on earth, exists within a quasi-inertial system, where inertial measurements are observable without being affected ba any accelerations or rotations.

Local level frame:

The local level frame or local tangent frame describes the reference frame tangent to the earths surface at any given point on or near the surface of the earth, see figure 2.3.

The origin is any given point, with the x or north axis pointing to north, the z or up axis pointing to the zenith for the left-handed or down for a right-handed system, and the y or right axis being rectangular to the other axes.

Terrestrial frame:

The terrestrial frame or earth centered earth fixed frame describes the reference frame fixed to the earth. The origin is the center of the earth, the x axis pointing to the defined meridian

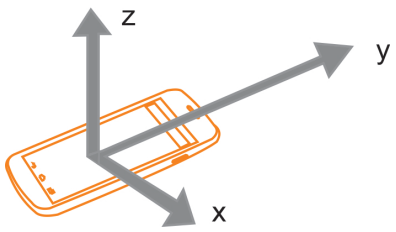

Figure 2.1: Sensor system for smartphones according to *Android*

Figure 2.2: Reference systems within pedestrian positioning

of Greenwich, the z axis being defined as the mean rotational axis of the earth, and the y axis being rectangular.

Inertial frame:

Finally, the inertial frame or earth-centered inertial frame, respectively quasi-inertial frame, is the reference frame with respect to an inertial system. Such frames are defined according to Kepler's laws. Such frames are true inertial frames if not underlying any accelerations, and pointing to the exact same directions in space at any given time. Since an earth-centered inertial frame is centered in the earth, and therefore underlying at least small accelerations during the yearly movement of the earth around the sun, it is only described as quasi-inertial frame. The frame is centered in the origin of the earth, with the x axis pointing to the vernal equinox and the y axis being defined as the mean rotational axis of the earth, again with the y axis being rectangular.

2.1.2 Rotations and transformations

Since all the presented reference frames are used within this thesis, consequently the basic rotations and transformations between the frames are outlined. Much more detailed explanations of rotations and transformations between coordinate frames as used within the presented algorithms and approaches are presented in e.g. Groves (2013), Wieser et al. (2007) or Hofmann-Wellenhof et al. (2008).

The basic rotation in a two dimensional space, using one rotational angle α for a right handed rotation, can be describe with

$$R = \begin{bmatrix} cos\alpha & sin\alpha \\ -sin\alpha & cos\alpha \end{bmatrix}, \tag{2.1}$$

and

$$x_T = R \cdot x. \tag{2.2}$$

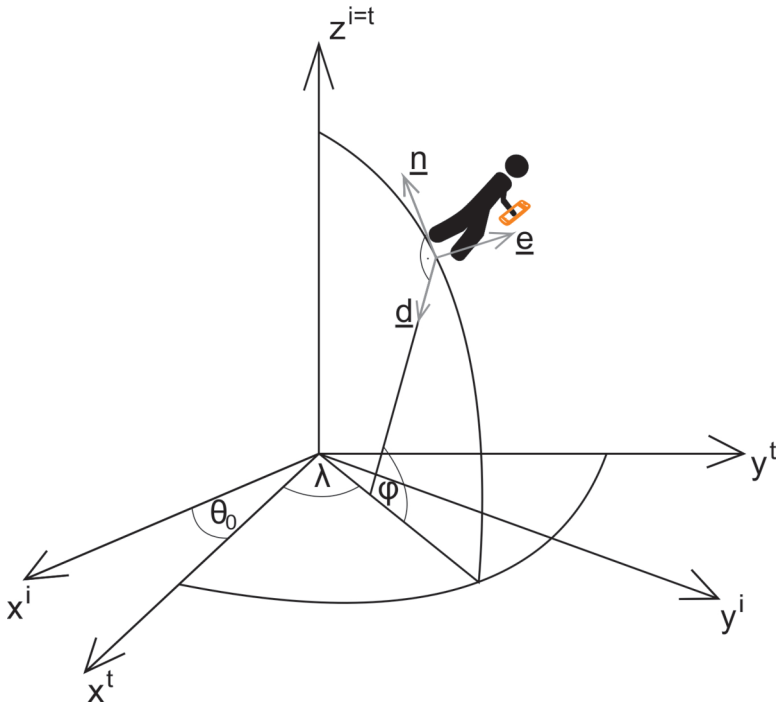

Figure 2.3: Definition of the local level system within the terrestrial and inertial system

Corresponding, the three-dimensional rotations around the x, y and z axis of a coordinate frame can be described with

$$R_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\alpha & sin\alpha \\ 0 & -sin\alpha & cos\alpha \end{bmatrix} , R_2 = \begin{bmatrix} cos\alpha & 0 & -sin\alpha \\ 0 & 1 & 0 \\ sin\alpha & 0 & cos\alpha \end{bmatrix} , R_3 = \begin{bmatrix} cos\alpha & sin\alpha & 0 \\ -sin\alpha & cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} . \quad (2.3)$$

A consecutive rotation around x, y and z would be

$$R = R_1(\alpha_1)R_2(\alpha_2)R_3(\alpha_3) = \quad (2.4)$$

$$= \begin{bmatrix} cos(\alpha_2)cos(\alpha_3) & cos(\alpha_2)sin(\alpha_3) & -sin(\alpha_2) \\ sin(\alpha_1)sin(\alpha_2)cos(\alpha_3) - cos(\alpha_1)sin(\alpha_3) & sin(\alpha_1)sin(\alpha_2)sin(\alpha_3) + cos(\alpha_1)cos(\alpha_3) & sin(\alpha_1)cos(\alpha_2) \\ cos(\alpha_1)sin(\alpha_2)cos(\alpha_3) + sin(\alpha_1)sin(\alpha_3) & cos(\alpha_1)sin(\alpha_2)sin(\alpha_3) - sin(\alpha_1)cos(\alpha_3) & cos(\alpha_1)cos(\alpha_2) \end{bmatrix} . \quad (2.5)$$

A basic seven parameter transformation of three dimensional vectors consists out of a consecutive rotation of the vector with an added scale factor μ as well as an transformation with c,

$$x_T = c + \mu Rx \quad , \quad (2.6)$$

where

$$x_T = \begin{bmatrix} x_T \\ y_T \\ z_T \end{bmatrix} \quad , \quad c = \begin{bmatrix} c_x \\ c_y \\ c_z \end{bmatrix} \quad , \quad R = R_1(\alpha_1)R_2(\alpha_2)R_3(\alpha_3) \quad , \quad x = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad . \tag{2.7}$$

The transformation between the sensor and body frame is based on the application itself, how the sensor is fixed to the body and if this changes within the application.

The transformation between the sensor or body frame to the local level frame is described by a rotation by the so called attitude angles roll r, pitch p and yaw y. The attitude describes the orientation of a sensor or object from one frame to another. Within navigation, the attitude is usually defined between the sensor frame or body frame to a local level frame, see figure 2.4 for a right-handed definition of the body frame of a smartphone with respect to a local level frame.

For rotating the local level $l - frame$ to the body $b - frame$, a consecutive rotation around the defined roll, pitch and yaw angles has to be conducted,

$$R_l^b = R_1(r)R_2(p)R_3(y). \tag{2.8}$$

R_l^b is a function of the attitude angles, and reversely the attitude angles may be computed knowing R_l^b.

$$R_l^b = \begin{bmatrix} cos(p)cos(y) & cos(p)sin(y) & -sin(p) \\ sin(r)sin(p)cos(y) - cos(r)sin(y) & sin(r)sin(p)sin(y) + cos(r)cos(y) & sin(r)cos(p) \\ cos(r)sin(p)cos(y) + sin(r)sin(y) & cos(r)sin(p)sin(y) - sin(r)cos(y) & cos(r)cos(p) \end{bmatrix} \tag{2.9}$$

The transformation of a vector given in a local level frame to an terrestrial frame can be conducted with a rotation based on the origin of the local frame, given in φ, λ. The rotation R_l^t consists out of the north n, east e and down d vectors,

$$n = -\frac{d\mathbf{d}}{d\varphi} \quad , e = -\frac{1}{cos\varphi}\frac{d\mathbf{d}}{d\lambda} \quad , d = \begin{bmatrix} -cos(\varphi)cos(\lambda) \\ -cos(\varphi)sin(\lambda) \\ -sin(\varphi) \end{bmatrix} . \tag{2.10}$$

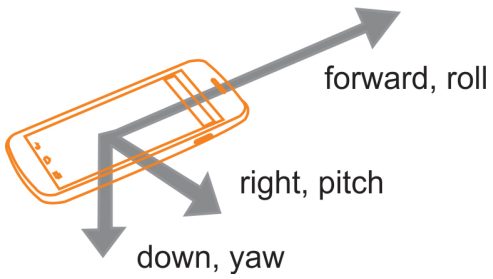

Figure 2.4: Attitude angles within pedestrian positioning using smartphones

With $R_l^t = \Big[n, e, d\Big]$, corresponding to graphic 2.3,

$$R_l^t = \begin{bmatrix} -sin(\varphi)cos(\lambda) & -sin(\varphi) & -cos(\varphi)cos(\lambda) \\ -sin(\varphi)sin(\lambda) & cos(\lambda) & -cos(\varphi)sin(\lambda) \\ cos(\varphi) & 0 & -sin(\varphi) \end{bmatrix}. \tag{2.11}$$

The transformation between an terrestrial frame and an quasi-inertial frame consists out of a rotation around the z axis about the Greenwich sidereal time θ_0 with $= R_t^e(-\theta_0)$.

2.2 Inertial and magnetic navigation

Inertial navigation is the computation of positioning measures with respect to an inertial frame. On earth, we can only use a non-inertial frame, being influenced by the earths rotation about the sun. Within an inertial frame, self-supported sensors measuring the acceleration in a known direction can be used to compute changes in position through integrating the acceleration twice over time, $\iint a\,dt\,dt$. This can either be used in pedestrian navigation to directly estimate relative position changes, e.g., using supported approaches like foot-mounted navigation. Contrary, the inertial signals can be used to gather information about the general gait cycle, by looking at the signal as a whole, which is used in Pedestrian Dead Reckoning (PDR).

Such sensors, measuring quantities with respect to the inertial frame, are acceleration and angular rate sensors. Next to that, additional sensors measuring quantities with respect to constant and natural present signals on earth, in a self-contained manner, such as magnetometers or barometers, are sometimes wrongly named inertial sensors. However, within the discussion of algorithms and approaches in this work, where magnetic sensors are directly used in line with inertial sensors, magnetometers will be discussed along with accelerometers and gyroscopes. The main difference is, that when subtracting the earth acceleration and rotation, the acceleration and angular rates of the subject with respect to the inertial frame can be measured, where the earth magnetic field still needs to be measured.

Inertial sensors are accelerometers as well as gyroscopes, measuring the accelerations and angular rates of the body attached. Historically, accelerometers were based on measuring the deflection of a certain proof mass, which underlies the laws of gravity. Since the mass itself is known, based on the design of the sensor this deflection can be measured. Mechanical gyroscopes were based on the principle, that rotating bodies try to keep their momentum and axis of rotation. The concept is, to measure the deflection of the gyroscope axis with reference to the inertial frame. The deflection occurs if one ore more axis are held steady. The concepts of accelerometer and gyroscope sensors are extensively presented in Titterton and Weston (2004) as well as Jekeli (2000) and Lawrence (1998). These concepts of accelerometers and

gyroscopes are working accurately, however, the implementation is vastly limited for the sake of miniaturization.

The concept of miniaturization started with the advancements in Micro-Electrical-Mechanical Systems (MEMS) technologies in the last decade. This basically describes, the development of constructing miniaturized sensors. For accelerometers, it is possible to create sensors which are called solid-state accelerometers, so accelerometers without a proof mass. Multiple approaches exist, based on measuring the deflections in the domain of vibration, acoustical, optical or electrical capacity effects. The most widespread MEMS accelerometer is the silicon-based solid-state accelerometer, measuring deflection in electrical capacity if one partly connected part of the sensor is deflecting, see Titterton and Weston (2004). For gyroscopes, the concept of vibrating parts, either in the form of wine glasses or tuning forks is wide spread. Based on the angular rate present at the sensor axis, the vibration orthogonal to the axis indicates the angular rate, see Titterton and Weston (2004). This concept is not as accurate as the multiple optical approaches for measuring angular rates based on the Sagnac effect, though vibrating gyroscopes can be constructed small enough as a MEMS architecture.

Sensor errors are present within all sensors, and can be divided into systematic and random or stochastic errors. There is always the question, how detailed ones current understanding of the sensor and its errors is. If a sensor error model is 100% true, every possible sensor error is accounted for, and the remaining noise is purely random noise. However, this is not possible, and not every possible sensor error fraction will be accounted for. Therefore, sensor errors which are not accounted for, are usually estimated along with other errors or within the random noise. Additionally, only sensor errors which can be estimated correctly should be modeled in the observation model.

Typical systematic errors are the offset or bias and the scale factor, and for gyroscopes the g-sensitivity. Additional errors comprise thermal errors, hysteresis, quantification errors and non-orthogonality. All possible types of sensor errors are explained in detail in Aggarwal et al. (2010) and Lawrence (1998).

The acceleration observation model, with the basic and observable sensor errors can be noted as

$$\tilde{f} = S_f \cdot f + b_f + n_f, \tag{2.12}$$

with $\tilde{f}$ being the true accelerations, S_f being the scale factor, b_f the bias, and n_f the accelerations noise. The gyroscope observation model, with the basic and observable sensors errors, can be noted as

$$\tilde{\omega} = S_g \cdot \omega + b_g + G \cdot f + n_g, \tag{2.13}$$

with $\tilde{\omega}$ being the true angular rates, S_g being the scale factor, b_g the bias, G the g-sensitivity, and n_g the angular rate noise.

The random or stochastic error is usually assumed to be Gaussian, although this can be modeled more complex using Gaus-Markov models and random walks of first or second order. Within this work, and in the lack of a better understanding of MEMS inertial sensor errors, n_f and n_g are assumed to be Gaussian.

Strictly, the basic concept of inertial navigation of integrating the measured acceleration twice over time, leading to a relative position estimation, is done with the so called coarse alignment and strapdown approach, explained later. However, such a rigorous strapdown approach is only meaningful, if the sensors used are of a good enough quality. As discussed further, the inertial sensors implemented in todays smartphones are not able to provide meaningful sensor reading for computing rigorous strapdown solutions. However, with the help of assumptions, e.g., the sensor being sufficiently stationary for at least shorter sections of time, the basic attitude angles can still be computed sufficiently accurate. Roll r and pitch p may be computed out of the acceleration measurements, called leveling, where yaw y may be computed out of the gyroscope measurements, called gyrocompassing, or magnetometers, called magnetic compassing.

Leveling:

Leveling describes the process of leveling a sensor corresponding to the local level, and therefore the local gravity vector. Assuming no other accelerations are influencing the sensor or smartphone, which practically means, that the sensor is sufficiently stationary, the following applies:

$$f^b = \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix} = R_l^b g_l \tag{2.14}$$

Assuming stationarity with $\ddot{x}^b = 0$, the specific force measurements are in level if

$$f^b = g_l = \begin{pmatrix} 0 \\ 0 \\ -g \end{pmatrix}, \tag{2.15}$$

with g being earth's gravity.

Otherwise, substituting equation 2.15 into equation 2.14, using 2.9, leads to

$$f^b = R_l^b g_l = \begin{bmatrix} sin(p) \\ -sin(r)cos(p) \\ -cos(r)cos(p) \end{bmatrix} g. \tag{2.16}$$

Solving for roll r and pitch p, leads to a relation purely based on the measurements of the specific force, without having to estimate g at the current location:

$$r = \arctan(-f_y, -f_z) \tag{2.17}$$

$$p = \arctan(f_x, \sqrt{f_y^2 + f_z^2}) \tag{2.18}$$

Magnetic compassing:

Magnetometer compassing describes the basic process of measuring the earth's magnetic field, and orientating sensor measurements accordingly to it, similar as done with an analogue compass. The magnetometers measure the magnetic flux density in three orthogonal axis, noted as m. Assuming that solely the earth magnetic field is present and measurable, which is not true due to magnetic disturbances and will be discussed later, the magnetic measurements in the local level frame can be represented as a spherical function of the magnetic declination α_l and the magnetic inclination β_l of the total flux density, see Groves (2013) for details:

$$m^b = \begin{pmatrix} m_x \\ m_y \\ m_z \end{pmatrix} = R_l^b \begin{pmatrix} cos(\alpha_l)cos(\beta_l) \\ sin(\alpha_l)cos(\beta_l) \\ sin(\beta_l) \end{pmatrix} \tag{2.19}$$

The magnetic heading ψ_b of the forward or along axis of a body or object with respect to magnetic north can be noted as $\psi_b = \psi_l b - \alpha_l$. Assuming that $\psi_l b = 0$, the declination can be expressed with $\alpha_l = -\psi_b$. Substituting this into equation 2.19 leads to

$$m^b = R_l^b \begin{pmatrix} cos(\psi_b)cos(\beta_l) \\ -sin(\psi_b)cos(\beta_l) \\ sin(\beta_l) \end{pmatrix} = R_l^b \varphi. \tag{2.20}$$

Reorganizing $m^b = R_l^b \cdot \psi$ to $\psi = (R_l^b)^T \cdot m^b$ and solving this equation, using $(R_l^b)^T = (R_l^b)^{-1}$, leads to

$$\begin{pmatrix} cos(\psi_b)cos(\beta_l) \\ sin(\psi_b)cos(\beta_l) \\ sin(\beta_l) \end{pmatrix} = \begin{pmatrix} m_x cos(p) + m_y sin(r)sin(p) + m_z cos(r)sin(p) \\ -m_y cos(p) + m_z sin(r) \\ -m_x sin(p) + m_y sin(r)cos(p) + m_z cos(r)cos(p) \end{pmatrix}. \tag{2.21}$$

Solving 2.21 for the magnetic heading ψ consequently leads to

$$tan(\psi_b) = \frac{-m_y cos(p) + m_z sin(r)}{m_x cos(p) + m_y sin(r)sin(p) + m_z cos(r)sin(p)}, \tag{2.22}$$

or, when the sensor or body is in a level and $r = p = 0$,

$$\psi = \arctan(-m_y, m_x). \tag{2.23}$$

The biases within magnetometer sensors are global influences on the signal, local magnetic deviations as well as influences from the model itself. The global magnetic field is not constant but time variable, therefore the difference between magnetic north and true north has to be taken into account. This difference is tracked and modeled by the World Magnetic

Model. Future differences between grid north and true north, as explained in Wieser et al. (2007), are not discussed within this thesis. Note, that when using simple projections for visualizing results, like the Mercator projection utilized by *Google Maps*, that grid north equals true north. Additionally the geophysical space weather can vary, and events such as magnetic storms and solar variation can lead to effects of up to 1 ° in observed magnetic heading. However, local magnetic deviations are amongst the strongest influence for pedestrian navigation.

Finally, the model itself, assuming that roll and pitch are known for using equation 2.22, which is not entirely accurate. Depending on the qualities of the used inertial sensors, the errors introduced corresponding to the magnetic heading may be up to 0.5 °.

Coarse alignment:

A coarse alignment is the combined process of leveling and compassing, only using inertial measurements. It describes the analytical solution for a start orientation at a known point, and is used as a starting point and orientation for a strapdown computation or as additional update in the attitude domain in between. For computing a coarse alignment, stationarity of the sensor is assumed, and usually the measurements have to be conducted for a long enough time to gain sufficiently accurate attitude estimations. Especially in the domain of angular rate measurements, the actual signal of the earth rotation is small compared to the existing sensor biases.

When the inertial sensor is stationary, the accelerometers only measure earth's gravity, see equation 2.15. Additionally, when stationary, the angular rates only measure parts of the earth rotation, namely

$$\omega = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} \omega_E cos(\varphi) \\ 0 \\ -\omega_E sin(\varphi) \end{bmatrix} \quad . \tag{2.24}$$

This is called gyrocompassing, where it is possible to align sensor measurements to the known earth rotation in static cases.

Using these relations between the inertial measurements in the body frame corresponding to the local level frame, R_l^b, the following equations are suffice:

$$-f^b = R_l^b g_l \tag{2.25}$$

$$\omega_{ib}^b = R_l^b \omega_{ie}^l \tag{2.26}$$

Adding a third artificial relation, by simply using the cross products

$$C^b = -f^b \times \omega_{ib}^b \quad , \quad C^l = g^l \times \omega_{ie}^l \quad , \quad C^b = R_l^b C^l \quad , \tag{2.27}$$

these equations can be framed as matrices using

$$\left[-f^b, \omega_{ib}^b, C^b\right] = R_l^b \left[g_l, \omega_{ie}^l, C^l\right], \tag{2.28}$$

and can be solved for R_l^b, what is described as coarse alignment.

Basic strapdown:

The strapdown algorithm consists of integrating the measured accelerations in a known direction, starting with a known starting location and known starting alignment. Based on the known starting alignment, the measured angular rates, present in R_b^l, get integrated with

$$\dot{R}_b^l = R_b^l \Omega_{ib}^b - \Omega_{il}^l R_b^l. \tag{2.29}$$

Where Ω_{ib}^b contains the gyroscope measurements and Ω_{il}^l contains the earth rotation. Measuring the specific force f^b in the body frame, this then can get transformed to the local level using $f^l = R_b^l f^b$. Establishing $g^l = \left[0, 0, \gamma\right]^T$ and $f_{Coriolis}^l = -(\omega_{il}^l + \omega_{ie}^l) \times v_e^l$, the velocity v_e^l in the local level frame can be received by integrating

$$\dot{v}_e^l = f^l + g^l + f_{Coriolis}^l. \tag{2.30}$$

And finally, using the integrated velocity v_e^l in the local level frame, rotated into the terrestrial frame, the position can by received by integrating

$$\dot{x}^e = R_l^e v_e^l \quad . \tag{2.31}$$

Practically, this leads to a two times integration of the accelerations or a three times integration of the angular rate measurements. Consequently, sensor errors in the angular rate measurements are the most severe when computing a strapdown, where sensor errors in the acceleration measurements are still severe for a useful strapdown solution.

Updates in inertial navigation:

Since inertial navigation is practically a two times integration of the accelerations in a known direction, the error in the position domain accumulates directly correlated with the integration time. Therefore, the time span, where inertial navigation is used without integration of other sensors, has to be minimized. This can be achieved by integrating additional sensors when possible, or to make use of so called *updates*.

Such updates are updating the algorithm that is processing the inertial measurements, e.g., a basic strapdown approach, making use of the fact that certain movements or states can be assumed to be true. Updates can be divided into updates relative to the observed measurement frame, or as absolute updates with known, additional information. Updates relative to the observed measurement frame are mostly making use of the information of:

- zero velocity
- zero angular rotation

A zero velocity update (ZUPT) makes use of the fact, that a phase of zero velocity in the measurement frame is detected. Phases of zero velocity are, based on the mounting of the inertial sensor, easily detectable, and therefore used widely. ZUPT is a frequently used update within vehicle navigation, but is also widely used in pedestrian navigation, especially together with foot-mounted inertial navigation. Within foot-mounted inertial navigation, the static phase of the foot each step can be detected, and the integration of the measured accelerations can be processed from one ZUPT to the next ZUPT only.

A zero angular rotation update (ZARU) makes use of the fact, that a phase of zero angular rotation in the measurement frame is detected. For example, this is certainly true for a non-moving vehicle, but not necessarily true for a non-moving foot.

ZUPT and ZARU are used on the one hand to simply not integrate the accelerations or angular rates during the updates with a known static phase, but additionally it is possible to apply sensor bias estimations during the updates, see e.g. Jekeli (2000) for details on updates and fine alignment.

Additionally, updates can be applied using additional information, like known coordinates, a known velocity or a known attitude, e.g. using approaches which transfer information:

- coordinate update
- velocity update
- attitude update

2.3 Least squares and optimal filtering

In this section, the basics of least squares adjustment and parameter filtering is described for future applications within pedestrian navigation.

Least squares parameter estimation:

The basic parameter estimation, based on minimizing the least squares of the residuals of the estimated measurements to the actual measurements, is called least squares parameter estimation. This approach assumes Gaussian distribution of the measurements, and is frequently used as basic approach to estimate parameters out of sets of measurements. Given

the relation between measurements or observations and certain parameters, the basic relation between observations and parameters can be described with

$$z = H \cdot x \quad . \tag{2.32}$$

z is representing the observations, x the parameters and H includes the derivations of z to x, if the relation are linear. If more measurements are present than needed to solve for x, the residual vector r is introduced and added to z, leading to

$$z + r = H \cdot x \quad . \tag{2.33}$$

Assuming Gaussian distribution of the residuals, with the goal of minimizing the residuals in the sense of $r^T r \rightarrow min$, according to Niemeier (2008) this leads to

$$\hat{x} = (H^T H)^{-1} H^T z \quad . \tag{2.34}$$

The presented approach is true for unimodal measurements. This can be extended to observations, where the stochastic informations are present for the observations, so Σ_{ll} exists and suffices $\Sigma_{ll} = \sigma_0^2 Q_{ll}$, with Q_{ll} including the stochastic information of the observations. According to Niemeier (2008), where $P = \Sigma_{ll}^{-1}$, with $r^T P r \rightarrow min$, the following set of equations is true for a least squares adjustment, based on Gaussian distribution and with known stochastics, where Q_{xx} includes the stochastic information of the estimated parameters and Q_{rr} the stochastic information of the computed residuals:

$$\hat{x} = (H^T P H)^{-1} H^T P z \tag{2.35}$$

$$Q_{xx} = (H^T P H)^{-1} \tag{2.36}$$

$$Q_{rr} = Q_{ll} - H Q_{xx} H^T \tag{2.37}$$

Kalman Filter:

Generally, Bayesian filtering describes the filter process of probabilities of the actual state, corresponding to a conditional probability that relates to the state. For positioning, a Markov process of the consecutive states is assumed, see Thrun et al. (2006). The Kalman filter is the basic, most straight forward parametric approach of Bayesian filtering.

The Kalman filter concept can either be derived out of the Bayesian theorem of a conditional probability, using the current epoch as condition for the next. Or the Kalman filter can be derived out of the basic least squares parameter estimation, grouped into multiple sets of observations, see Niemeier (2008) under gradual parameter adjustment.

The basic concept of a Kalman filter is, that a so called dynamical model Φ is used to propagate the parameters from one epoch to the next. This dynamical model has to be defined based on the actual motion, according to the application used. Usually this motion

will be predefined as either static or uniform motion, but more complex dynamical models might be used.

The basic phases of a Kalman filter are:

- initialization
- computation of Kalman gain
- measurement update
- observation update

The observation relation has to be defined, assuming Gaussian distribution and using a measurement noise of $v_k \sim N(0, R_k)$:

$$z_k = H_k x_k + v_k \tag{2.38}$$

Additionally, the dynamical model for state propagation has to be defined, again assuming Gaussian distribution and using a system noise of $w_k \sim N(0, Q_k)$:

$$x_{k+1} = \Phi_k x_k + w_k \tag{2.39}$$

The computation of Kalman gain is done using

$$K_k = \tilde{P}_k H_k^T (H_k \tilde{P}_k H_k^T + R_k)^{-1}. \tag{2.40}$$

The measurement update is computed using:

$$\hat{x}_k = \tilde{x}_k + K_k(z_k - H_k \tilde{x}_k) \tag{2.41}$$

$$P_k = (I - K_k H_k)\tilde{P}_k \tag{2.42}$$

And the observation update, also noted as time update or prediction phase, is done using:

$$\tilde{x}_{k+1} = \Phi_k \hat{x}_k \tag{2.43}$$

$$\tilde{P}_{k+1} = \Phi_k P_k \Phi_k^T + Q_k \tag{2.44}$$

Using the basic relation between the derivation of the parameters and the parameters $\dot{x} = Fx$, the dynamic model Φ can be derived through $\phi = e^{F\Delta t}$, see Zarchan and Mussoff (2013). Developing this relation into a Taylor series until the first order leads to

$$\Phi = 1 + F\Delta t. \tag{2.45}$$

The basic design of a Kalman filter can be executed as an absolute state or error state filter, see Aboelmagd et al. (2013) for details. Absolute state means, that the state parameters are

estimated in their absolute domain, where error state means, that the state parameters will be designed as the error portion of the predicted to the measured state.

When designing the filter, all observation type measurements can either all be used within the observation vector z, or relative measurements can be used directly within the dynamic model, as suggested from e.g. Thrun et al. (2006). The argument for using an actual dynamical model is, that the stochastics of Φ can be set according to the application, where the advantage of using actual measurements in the prediction phase is, that the prediction is more accurate, and therefore the computed residuals are smaller.

Additionally, based on the information, whether the functional relation between the measurements and the parameters is linear or not, either a basic Kalman filter, an extended Kalman filter or an unscented Kalman filter has to be used, see Wieser et al. (2007) for more details on filter design.

Particle filter:

A particle filter is an alternative, non-parametric implementation of the Bayes filter, where the samples of a distribution are called particles. Basically, the approximate belief of a set of particles x is estimated, compared to parameters. The particles are defined with a certain number, e.g. 500 or 1000 particles, and initialized and therewith sampled. Per filter iteration, these particles are then propagated, either using a model or using observations, and then resampled after a measurement update has be performed. So the basic phases of a particle filter are:

- initialization
- propagation of particles
- importance weight computation
- resampling of particles

These phases are executed for every particle x, which is represented by the probability p with

$$x \sim p(x|z,u), \tag{2.46}$$

using the measurements z and the control u.

After initializing and sampling, the so called importance weight ω for each particle is computed, using

$$\omega = p(z|x). \tag{2.47}$$

This weight is set for each particle, and is directly dependent on the used measurements and their stochastic distribution.

The propagation phase propagates every particle according to the design of the particle filter. Within this phase, it is relatively easy to use additional observations, like maps within pedestrian positioning. For example, maps may be used to evaluate the correct propagation, and to eliminate particles with an impossible propagation.

The probability of each particle is given by its importance weight. The resampling or importance sampling transforms the current set of particles into a new set of particles, taking the current weight into account. In this process, the original number of x is regained, so that no degeneration of particles is possible.

Thrun et al. (2006) describes the mathematical derivation of the particle filter concept in detail. Every particle filter is designed for the measurements available. The specific particle filter used within this work, for the positioning approaches presented in chapter 4, will be described while describing the specific applications.

2.4 Map-based filtering

Maps are used within positioning historically since the beginning of navigation itself. More specifically, digital maps are used in positioning and navigation today.

Generally, digital maps can be divided based on their form of representation, storage or visualization, as raster maps or vector maps, also denoted as graphs. Raster maps use a pixel by pixel representation of the map, e.g. gained directly from digital plans. Every pixel may have additional information assigned, e.g. color coded. The most basic color coding is the coding of space available for moving on the map as white, and of space non-available for moving on the maps as black. Contrary, vector maps or graphs are represented using a node-to-edge structure, where it is possible to navigate on the edges which are linked through defined nodes. As example, the same real case situation, represented as raster map on the left as well as vector map on the right is shown in figure 2.5. Details about digital maps, their modeling and storage as well as techniques for navigation like map matching and map aiding are thoroughly discussed in Wieser et al. (2007).

Both representations, raster as well as vector maps, may be used to enhance pedestrian positioning approaches. Raster maps can be used e.g. for visualization or planning purposes, but can also be used when processing a particle filter for pedestrian navigation. During initialization and resampling of the particles, the occupancy of certain coordinates in a raster map can be checked. Based on the state of the pixel in the raster map, the particle can be identified as valid or invalid on the map, and therefore eliminated. Additionally, during propagating, the validity of the current path of the particles can be checked based on the raster map representation. This concept of using raster maps for pedestrian navigation is shown within Hafner (2015) and is applied within the application in chapter 6.2.

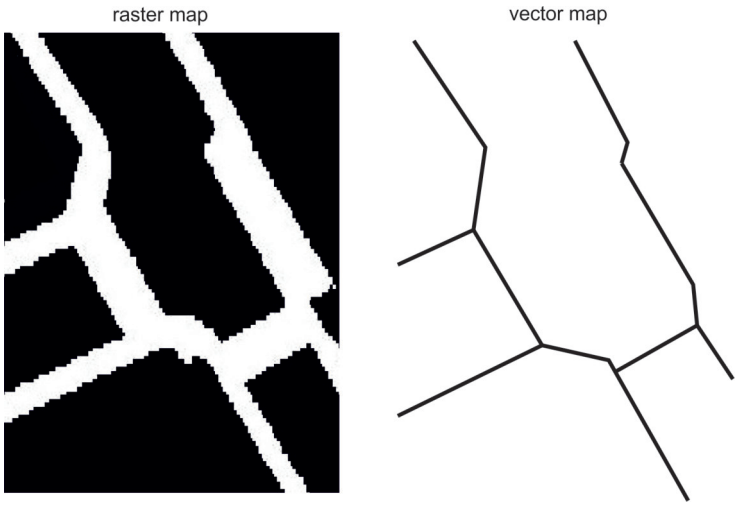

Figure 2.5: Difference of representation of raster and vector maps

Vector maps or graphs are widely used within navigation, e.g. car navigation. Also, graph-based filtering is specifically useful for pedestrian navigation. The approach of using approximations based on routing graphs for pedestrian positioning was first shown by Willemsen and Keller (2015), adapted within Reitbauer (2017a) and applied within the application presented in chapter 6.3 and chapter 7. The main idea, using a routing graph within pedestrian positioning, is to specifically use it to enhance relative positioning like PDR. Approximating the available space with a routing graph, the movement of the pedestrian gets matched onto this graph. Therewith, relative position techniques can be examined per edge, rather than for the whole processing time. For every edge, the change in direction can be accumulated, and reset at the start of the next edge. Likewise, the estimation of the covered distance can be accumulated for the current edge only. The prerequisite is, that the graph is structured using edges preferably not longer than 10 m, and that huge open spaces have to be approximated, using the main paths along those spaces. The approach presented by Willemsen and Keller (2015) can be broken down to the following cases:

- walking straight along
- turning to another edge on a node
- making an U-turn

When walking straight, the accumulated change in direction on the current edge is not exceeding a set threshold, and therefore the walked trajectory is assumed to be straight ahead. Similarly, if the accumulated change in direction when reaching a node, is within the thresholds of a edge in another direction, it is assumed that the pedestrian is taking the indicated turn. Also, when the accumulated change in direction indicates an U-turn, it is possible that the pedestrian changed the direction on the graph. If none of the presented

cases is true, the graph-based filtering has failed for the current scenario, and an alternative matching approach has to be executed.

2.5 Pattern recognition and machine learning

Pattern recognition and machine learning are two sides of the same coin, that is automated learning of thresholds or algorithms to classify certain patterns.

Pattern recognition is considered to be the engineering type of approach to classify a signal, where machine learning being more the mathematical approach, and deep learning or it's derivations being developments based on machine learning techniques, see Bishop (2006). Starting, the signal used within the approach as well as the wanted detectors have to be defined. Based on them, an approach to set thresholds or to train the algorithms has to be established, and training data has to be defined. Considering the signal and training data, multiple types to classify the recognition or learning approach are available, and discussed in more detail in chapter 4. In pedestrian navigation, these signals are directly correlated to the gait pattern, for example the gait cycle itself, or the types of locomotions the pedestrian is carrying out.

The general problem, more specifically discussed for human activities, is described thoroughly by Labrador and Yejas (2014). Firstly, the activities or motions have to be defined. Descending from the general problem of pattern recognition, this may be called activity or motion recognition. Such activities can and have been defined as e.g. activities within the house, like sleeping, vacuuming, cooking and so on, as activities of modality like walking, riding a bike, driving a car, taking a train, or as the different locomotions, like standing, walking normal, walking fast. Secondly, the sensor used for detecting the activity or motion has to be defined. This can either be done using external sensors like vision-based sensors, or using body fixed sensors like accelerometers and gyroscopes.

The basic locomotions used in pedestrian navigation have been defined by the community using them, see Nguyen et al. (2015) or Bobokov et al. (2015). Since 2018, the basic parameters for pedestrian navigation are included in a standard, namely the ISO/IEC 18305 standard, where also the possible locomotions are defined. Based on this, the locomotions

- standing
- walking normal
- walking fast
- going upstairs
- going downstairs

are defined to be used within the applications presented, see figure 2.6.

Following, discussing the positioning approaches in chapter 4 and the applications of pedestrian navigation in chapter 6, only motion recognition will be examined.

2.6 Basics of pedestrian positioning and the gait pattern

Navigation and positioning systems have historically been designed and developed for ships, airplanes and land-based vehicles. The evolution of navigation systems for pedestrians is, relatively speaking, new, in that global solutions only do exist since roughly a decade. However, specialized solutions for pedestrians do exist since multiple decades. The evolvement of the smartphone can be directly correlated to the evolvement of pedestrian navigation and it's accessibility.

According to a national human activity pattern survey, Klepeis et al. (2001), we are indoors for more than 80% of our time. Additionally, there are some specific criteria when it comes to pedestrian navigation. The main assumption with pedestrians is, that they move forward along their main axis, with steps on their two legs. This assumption is true for the majority of pedestrians, however one might argue that, e.g., users of a wheelchairs might be classified as pedestrians as well.

The most characteristic specifications of pedestrians for navigation is the movement with a typical gait pattern, forward with steps from two legs, see figure 2.7 for the relation of steps and stride. Next to the gait, the environment pedestrians usually navigate in, which is indoors, as well as the way pedestrians navigate and need their guidance instructions are typical characteristics.

Historically, in case of navigation systems developed for ships, airplanes or vehicles, the design of our globally existing navigation systems did not take the possibility into account of being mostly indoors. Therefore, globally existing positioning solutions, like the Global Navigation Satellite System (GNSS) mostly only work outdoors.

Figure 2.6: Locomotion modes used within pedestrian positioning

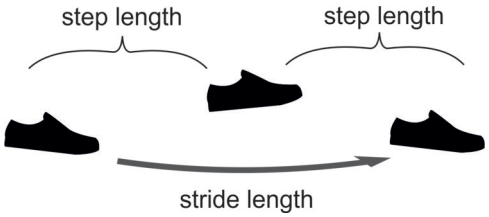

Figure 2.7: Gait cycle with steps and stride length

The specific restrictions of pedestrian positioning are, that pedestrians navigation systems have to work indoors, that pedestrians usually move with a typical gait pattern and that pedestrians have more specific needs when it comes to guidance. The specifics of positioning systems used indoors are elaborated in chapter 4, where the specific characteristics of the gait cycle are analyzed in this section.

The study of Öberg et al. (1993) elaborates the gait pattern and gait speeds based on 233 users, comprising men and women of all age groups, which is the most comprehensive study known. Following, an analysis of this study defines slow, normal and fast walking based on gait speed and frequency. This analysis is divided into gait speed, step frequency and step length. These parameters are extracted for men, woman as well as their intersection, which is defined as mean over all age groups and genders.

In table 2.1, the **gait speed** as a mean of all users is shown.

Table 2.1: Gait speed, mean of different gaits of men, woman and their intersection, $[cm/s]$

	Men			Women			Intersection		
	low	high	mean	low	high	mean	low	high	mean
slow	79.5	93.5	86.5	70.1	91.3	78.9	70.0	94.0	82.7
normal	118.2	132.8	128.2	108.6	128.5	118.4	109.0	133.0	123.3
fast	158.6	184.3	168.7	141.8	172.1	157.8	142.0	184.0	163.3

Based on the simplified intersection of age and sex, groups out of the study of Öberg et al. (1993) following typical gait speeds can get roughly identified:

- Slow walking: 70 to 94 cm/s, mean of 83 cm/s
- Normal walking: 109 to 133 cm/s, mean of 123 cm/s
- Fast walking: 142 to 184 cm/s, mean 163 of cm/s

Following, the border between slow walking and normal walking, based on the low, high and mean gait speeds, is defined with 100 cm/s and the border between normal and fast walking with 140 cm/s.

In table 2.2, the **step frequency** as a mean of all users is shown.

Table 2.2: Step frequency, mean of different gaits of men, woman and their intersection, [$steps/s$]

	Men			Women			Intersection		
	low	high	mean	low	high	mean	low	high	mean
slow	1.49	1.68	1.56	1.48	1.68	1.57	1.48	1.68	1.56
normal	1.91	2.14	2.00	1.97	2.16	2.07	1.91	2.16	2.03
fast	2.27	2.51	2.37	2.40	2.61	2.52	2.27	2.61	2.45

The typical stride and step frequencies are:

- Slow walking: 1.48 to 1.68 $steps/s$, mean of 1.56 $steps/s$ and 0.78 $strides/s$
- Normal walking: 1.91 to 2.16 $steps/s$, mean of 2.03 $steps/s$ and 1.015 $strides/s$
- Fast walking: 2.27 to 2.61 $steps/s$, mean of 2.45 $steps/s$ and 1.225 $strides/s$

Following, the border between slow walking and normal walking is defined with 1.8 $steps/s$ or 0.9 $strides/s$ (0.56 Hz for steps or 1.11 Hz for strides) and the border between normal and fast walking with 2.2 $steps/s$ or 1.15 $strides$/s (0.43 Hz for steps or 0.87 Hz for strides). These borders are chosen as an empirical mean of the difference between the typical stride and step frequencies between slow walking, normal walking and fast walking. Step frequency thresholds for distinguishing walking slow to walking normal and walking normal to walking fast are 0.56 Hz and 0.43 Hz. Basically, 0.5 Hz is assumed to be walking normal.

In table 2.3, the **step length** as a mean of all users is shown. The typical step lengths

Table 2.3: Step length, mean of different gaits of men, woman and their intersection, [cm]

	Men			Women			Intersection		
	low	high	mean	low	high	mean	low	high	mean
slow	51.7	56.2	54.2	46.6	51.8	49.0	46.6	56.2	51.6
normal	61.5	66.0	63.6	53.5	59.7	56.6	53.5	66.0	60.1
fast	68.7	78.7	73.2	60.3	68.6	64.3	60.3	78.7	68.7

are:

- Slow walking: 52 cm
- Normal walking: 60 cm
- Fast walking: 69 cm

Following, borders between walking slow and walking normal might be defined at 55 cm and between walking normal and walking fast at 65 cm, however distinction between paces in the step length domain is much more ambiguous than in the step frequency domain. A typical stride at normal walking would then be around 1.2 m. The presented quantities are used within this thesis as a starting point to define walking slow, normal and fast within the step length and locomotion estimations.

2.7 Pedestrian guidance

Generally, route finding for pedestrians is similar to routing processes for, e.g. vehicles. The main difference is the variability in movement pedestrians possess, as well as the possibility to navigate by sight. Therefore, multiple concepts for something that is today called *natural guidance* has been adapted for pedestrians. Two main keywords for pedestrian guidance are universal navigation as well as natural navigation.

Universal navigation, e.g. discussed within Karimi (2011), expresses the need of a universal positioning and navigation solution. In detail, the concepts of a seamless transition between outdoor and indoor positioning and navigation is a prerequisite for a universal navigation solution. Universal positioning solutions do exist, however the available accuracy of the position as well as guidance varies strongly depending on the environment.

Natural navigation or natural guidance describes guidance instructions based on natural indications relative to naturally visible objects compared to metrical indication. For example turning left after passing a certain landmark or sign compared to turning left in 30 meters. This is a problem in the domain of pedestrian guidance compared to e.g. guidance for cars, where roads are usually at least spaced a couple of tens of meters apart. Natural guidance is common since some years, and for example an option within the *Nokia Here* maps. However, the approach of natural guidance doesn't work with e.g. blind people, and is not discussed in more detail.

Within this work, the process of navigation will mainly be discussed as a problem of positioning, where the subproblems of routing and guidance will be assumed to be sufficiently known or solved for pedestrian guidance. Note, that the routing and guidance approaches should not be considered as completely solved within pedestrian navigation, but the concepts for these do already exist and have been discussed thoroughly.

3 Smartphone as device for pedestrian navigation

This chapter discusses smartphones as device for pedestrian positioning. Starting with a description of the main types of smartphones and currently available types of operating systems, the built-in sensors within smartphones are outlined. The discussed smartphone sensors are therewith grouped and named accordingly to the used operating system, not in alignment with a proper, scientific description of these sensors. The main part of chapter 3, comprises the presentation and description of all developed smartphone applications used within this research. Closing, a brief discussion of the possible options carrying a smartphone of a pedestrian navigating is given.

The evolution of computers is happening since decades, and basically taking place in the domain of processing power as well as miniaturization. Where Moore's law states, that processing power doubles every two years, which was true for a while, there is no such law for miniaturization. However, whenever possible, engineers tried to develop smaller and lighter units of computers. First, computers that were portable were developed, evolving to laptops, evolving to so called personal digital assistant devices.

Next to the development of computing technology, cellular phones evolved over the last decades. With the first consumer grade mobile phone becoming available in 1983, the development of the corresponding cellular networks started from the first generation up to the 5th generation of cellular networks within two decades, based on the country of origin, see Cudjoe (2014). With the 4th generation cellular network being a standard today, the 5th generation network is being established right now in most parts of the world. From the 3rd generation on, the possibility of large data packet transfer became possible, Cudjoe (2014). This enabled the use of the Internet and other communication technologies, next to cellular communication.

Now, the idea of smartphones is to combine a mobile phone with the basic functionalities of personal digital assistants and Internet-based functionalities. This was done most prominently, by *Apple* with the introduction of the first *iPhone* in 2007, arguably the first smartphone. This is related to the fact, that by 2007 the 4G standard already existed and started to be established. There is no global definition of what defines a smartphone. However, usually a smartphone is defined as a mobile phone with extended functionalities. So,

smartphones do have mobile phone capabilities as well as the possibility to connect to the Internet using data packages. From that point on, smartphones developed to the point where there are today.

According to Statista (2018b), in 2018 35% of the world population is using a smartphone. Accordingly, out of all mobile phone users worldwide, 57% are using smartphones today. This is estimated to rise even further, while technically most mobile phones today are produced as smartphones. More specifically, it is estimated that 65% of all people in Germany are using a smartphone, which is representable for Austria and Europe.

When smartphones started to gain market shares amongst mobile phones, multiple operating systems where available. Although even within the first years, mainly *Apple's iOS* and *Google's Android* where leading in terms of market share, operating systems from *Blackberry*, *Windows*, *Samsung* or *Firefox* were available. Today, virtually only *iOS* and *Android* are remaining, with *Android* having roughly 88% and *iOS* roughly 12% market share amongst the current existing smartphone models, see Statista (2018b) and Statista (2018a). Smartphones evolved to be our everyday companion within the last years. Today, in Europe, Canada or the United States of America, between 60% and 80% of the population own a smartphone and take it along for about 80% of the time, Statista (2015). From all available smartphone applications, the majority today is classified as location-based service, i.e. the application would like to use the location of the smartphone, GSA (2015).

With smartphones evolving, more and more sensors were embedded into them. Where some sensors were originally built in for certain purposes, they soon became important for other applications as well. The most known example might be the acceleration sensor, which initially was built as a two-axis acceleration sensor to measure the screen orientation of the phone. Nowadays, the acceleration sensor is built as three-axis sensor and used for multiple applications, e.g. in the scope of this work for step detection. Additionally, the processing power was adapted accordingly, on order to have strong enough processors to process all evolved algorithms and sensors. Originally implemented for certain applications, today's smartphones possess a multitude of sensors which make it possible to compute the position of a pedestrian carrying the smartphone. Next to obvious sensors for a mobile phone, like radio transceivers or mobile communication sensors, sensors which can be used for obtaining the current position or for alternative positioning approaches are present. Among these sensors are a Global Navigation Satellite Systems (GNSS) sensor, accelerometer, gyroscope, magnetometer, pressure sensors etc., see figure 3.1.

The most directly usable sensor for pedestrian positioning is the GNSS sensor chip. However, in urban areas the position estimation based on low-elevation satellite signals is subject to signal shadowing or multipath effects, which worsens the position solution dramatically up to the point, where the position estimation is not solvable anymore. Next to GNSS, alternative approaches for putting a constraint on position estimations for pedestrians are dead reckoning approaches specifically for pedestrians, the identification of the human motion

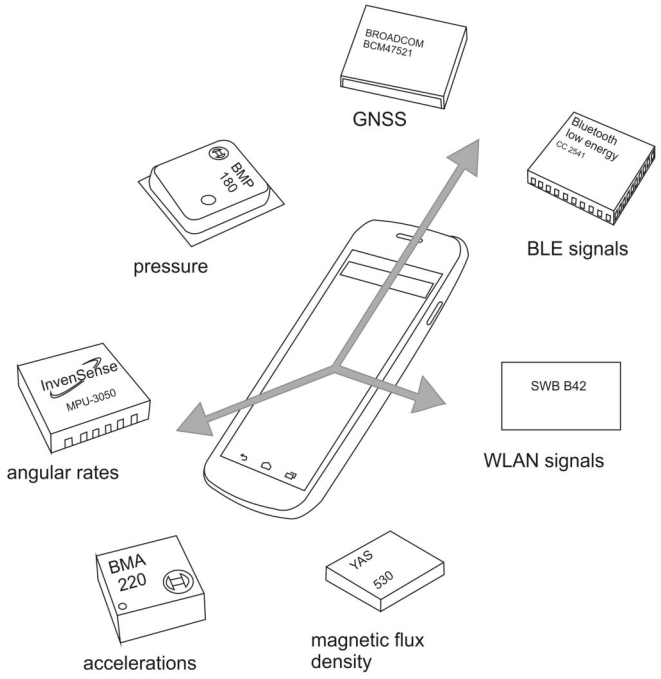

Figure 3.1: Smartphone sensors usable for positioning

of the pedestrian actually carried out, and the incorporation of urban maps within a filtering of these positioning approaches.

Next to urban areas and limited GNSS availability, pedestrians are indoors about 80% of the time. Despite advances in high sensitivity GNSS, where it is possible to receive and keep up GNSS signals within buildings in specific situations, it is usually impossible to gain GNSS solutions indoors using smartphones.

With all the presented sensors, processing power, but most importantly, the ubiquity of the smartphone being present with us virtually all the time, it is obvious to use it as navigation device. This is why, numerous studies have been carried out to exploit the current available smartphone sensors for positioning purposes, in order to run navigation algorithms, specifically designed for smartphones and pedestrian navigation.

3.1 Smartphone operating systems

As laid out, smartphone operating systems are dominated by two major companies, *Google* and *Apple* respectively. Amongst those, *Google's* operating system *Android* has the major market share by far, and is, contrary to *Apple's* operating system, promoted to be open source. Due to the fact, that *Android* is generally speaking open source, and it is possible for developers to access most available sensor and location data, and that *Android* has the

highest global market share amongst operating systems, this work focuses solely on *Android* when discussing smartphones.

Although *Android* as operating system is being promoted as open source, this is technically not true. Multiple instances are not open source, and the source code is not visible to the developers. This leads to the fact, that it is unknown how some artificial sensors work exactly, and how the *Android positioning service* estimates its position exactly. For example, it is unknown how the *step detection sensor* works exactly. Still, *Android* tries to be open access in the sense, that the raw data of every sensor possible is made available for the developers. This is why *Android* is used extensively within the research community, and why certain test setups are possible using smartphone sensor data.

The operating system *Android* is under constant development since 2008, with one new update at least every year. The major *Android* system updates, which are important for the developers use of sensor data, are listed in table 3.1. Every major new release of the operating system is labeled with a software number or so called *API level*, starting with 1 at the first *Android* release. The *API level* may not be confused with the official *Android* numbers for the main releases, where e.g. *Android 4.0* equals a software number of 14, *Android 7.0* equals a software number of 24 and *Android 9.0* equals a software number of 28.

Table 3.1: *Android* versions and important updates

Android software number	Added possibility	year
1	Wifi Manager	2008
9	Inertial Sensors	2010
14	Activity	2011
18	Bluetooth Manager	2013
19	Step detector	2013
26	GNSS	2016

Very important for developers were the first version, where it was already possible to access Wifi or Wide Area Local Network (WLAN) signals. Next, the 9th version marked a milestone for developers, where it was possible to access the inertial sensor data. Thereafter, version 18 made it possible to access Bluetooth Low Energy (BLE) signals. Finally, the last important versions for developers in the positioning domain, was version 26 with the possibility to theoretically access raw GNSS measurements.

3.2 Smartphone sensor data

Android subdivides the available data, usable for navigation, into actual sensors and artificial sensors, where actual sensors refer to real sensors, and artificial sensors are sensor readings computed out of actual sensors. Note, that a computational process and sometimes data filtering is performed on the raw data in order to gain these artificial sensors. Therefore, in this work the raw sensor data is used when available. Artificial sensor types, such as step detector or activity, will only be used as global standard to compare solutions to each other. Amongst the sensors, Android currently divides them into motion sensors, position sensors, environment sensors and raw GNSS measurements. Within this section, the sensors, being actual sensors, artificial sensors and classification or location algorithms, will be described with the terms used within the operating system itself. All presented sensors are event-based, meaning that within the software, their variable will have a new value whenever the sensor has a new measurement. Within the context of *Android*, the present sensors usable for positioning are divided into the groups of sensors and user location. Additionally, the sensors are grouped to motion sensors, position sensors, environment sensors as well as the raw GNSS measurement sensor.

In the group of **motion sensors**, all types of inertial sensors as well as their artificial derivatives are present. Note, that most of these so called sensors, are not sensors in the technical sense, but most often quantities computed or derived out of actual sensor readings. However, since they are described as sensors within *Android*, this is the descriptive word used in this section. The motion sensor group comprises the following sensors:

- accelerometer
- gravity
- gyroscope
- linear acceleration
- rotation vector
- significant motion
- step counter
- step detector

The accelerometer, gravity, gyroscope as well as linear acceleration sensor all consist of three sensor values corresponding to the three orthogonal sensor axis. The accelerometers, gravity and linear acceleration are given in $[m/s^2]$, the gyroscope is given in $[rad/s]$ and the other sensors being unit less. The acceleration sensor delivers the actual measured accelerations along the sensor axis, the gravity sensor delivers the estimated force of gravity occurring

on the sensor axis and the linear acceleration sensor represents the estimated acceleration without the gravity vector. Obviously, the gravity and linear acceleration sensor are artificial sensors estimated out of the acceleration sensor. The gyroscope delivers the actual measured angular rates, with the rotation vector being an artificial sensor delivering the rotation vector component along the sensor axis. The significant motion and step detector sensor are artificial sensors, which estimate if a significant motion is acted on the smartphone and if a step is detected of the user carrying the smartphone. The step counter is accumulating the detected steps since the last sensor reboot.

The **position sensors** comprise, amongst others:

- geomagnetic rotation vector
- magnetic field
- orientation
- proximity

With the position sensors, the geomagnetic rotation vector is unit less and similar to the rotation vector, but based on the magnetometer readings and not the gyroscope readings. The magnetic field is measured in $[\mu T]$, with Tesla being the SI unit for the magnetic flux density, the orientation is given in [°] and the proximity is given in $[cm]$. The geomagnetic rotation vector is delivering the rotation vector component along the sensor axis, where the magnetic field sensor is delivering the actual measurements of the magnetic flux density at the sensor axis. The orientation sensor is an artificial sensor, which delivers the sensor axis orientation in yaw, pitch and roll, and the proximity sensor delivers the estimated proximity of the smartphone to another object, indicating if the smartphone is faced to an object or e.g. hand held.

Finally the group of **environment sensors** comprises:

- temperature
- light
- pressure
- humidity

The temperature is given in $[°C]$, the light in $[lux]$, the pressure in $[mbar]$ and the humidity as [%], all parameters as single event values.

Finally, as latest update in the sensor domain usable for positioning, is the implementation of the **GNSS raw data sensor**. The concept of GNSS as a positioning system is described thoroughly in e.g. Hofmann-Wellenhof et al. (2008). Basically, the GNSS satellites send a coded message within the ultra high frequency band, which can get correlated and measured

using a receiver on earth. The measured signal correlates directly to the signal propagation time, which is then called a pseudorange. With the known satellite positions, using at least four such pseudoranges, a position can be estimated. The satellite positions are either estimated using data about their positions transmitted through the Internet, or data coded onto these pseudoranges, also known as ephemeris data. This basic concept works great, with the major downside being the satellite visibility. In order to have access to at least four visible satellites, the GNSS receiver has to have line-of-sight view to the open sky, and this concept is not working well indoors or in urban canyons. GNSS position estimates, known as position fixes, were available even before smartphones in high-priced mobile phones, and are today virtually present in every sold smartphone. The accessibility of GNSS raw pseudorange data is only available since the latest smartphone generation. The pseudorange data is available in $[m]$ as a function of the signal propagation time, and the ephemeris data can also be polled. From a software side, there is the possibility to access all available GNSS systems, with pseudorange and code range and multiple frequencies in the ultra high frequency band. However, the actual available measurements are dependent on the present GNSS chip in the used smartphone. Currently, there is only one *Broadcom* chip present, that delivers multi frequency data, and most GNSS chips only deliver one or two GNSS systems simultaneously.

Next to the group of sensors, the *Android* **user location** programming interfaces comprise, amongst others, mainly:

- activity
- location

The activity sensor estimates the currently acted activity as well as activity transitions with the smartphone. Where activity transitions means, the identification of the transition process between distinct activities. It currently comprises of the five activities *in vehicle*, *on bike*, *running*, *still*, and *walking*. The activity recognition is probably based on a pattern recognition approach, and used within applications for e.g. multi-modality in navigation.

The location sensor is implemented within the so called *Google Play Services*, and therefore called *Google Play Service location*. The location estimate is an estimate out of the most probable positioning approaches available to the smartphone. If a GNSS fix is available it will most likely be used, if other radio signals such as WLAN or BLE are present, they will most likely be used within the *Google Play Service location*. The resulting location for the application developer is a location information, as well as a standard deviation to that location. However, how this location is being processed, is unknown to the developer.

Next to the activity and location sensor, additional functionalities such as geofencing or optimization strategies for battery management when using location sensors are present, which will not be discussed within the scope of this work.

3.3 Smartphone applications

Smartphone applications are standalone, executable applications, which can be run on smartphone operating systems. While the basic concept of the existing smartphone operating systems is similar, for the reasons mentioned before, only *Android* as operating system will be described in detail. For creating an application, coding environments for e.g. *Java* exist, where the operating system specific commands are implemented. Additionally, simulation environments exist, where debugging and the execution of applications can be tested.

However, it has to be noted, that the main purpose of a smartphone is still being a cellular phone. Therefore, the main priority are still tasks like staying assigned within a cellular network, or, if there are receiving phone calls or texts, such tasks will be prioritized. This means, that other tasks don't get the highest priority. One of the main example necessary for positioning would be data sampling. Take, as example, the inertial sensor chip implemented in a smartphone. One of the main sensors used in the last years is the *Invensense MPU 9250*, being analyzed in e.g. Nilsson et al. (2014b). Where such sensors are capable of more than 200 Hz data sampling, with a stable sampling rate, using the *Android* sensor class, only 100 Hz data with occasional data gaps can be accessed.

The basic structure of a smartphone applications is lifecycle-based. This means, that certain lifecycle activities will get called if the application is started, if the application is paused, closed, or if interactions are made within the application. Using actual or artificial sensors can be done either event-based or sample-based. This means, that sensor readings are delivered when certain events, e.g. a significant motion was conducted, are present, or at every sensor sample available. For the sake of analyzing all data, a sample-based sensor reading was utilized for every sensor possible within this research.

Within an smartphone application, permissions to use certain sensors or location information have to be acknowledged by the user. Permissions set and used within the developed applications are mostly the permission to access sensors, to access the coarse and fine location and to access the activity recognition. When using certain sensors, at the start or creation of an application a manager that listens to the type of sensor has to be set, in order to be able to receive sensor- or sample-based measurements. Amongst the managers used in the developed applications are the sensor manager, the location manager and the so called wifi manager and bluetooth manager.

The developed algorithms and applications were analyzed and tested using multiple mobile phones of multiple generations, mainly a *Samsung Galaxy Nexus* produced starting from 2011, a *Samsung Galaxy S4* produced starting from 2013 and a *ZTE Blade V8* produced starting from 2017.

Within carrying out the research question, multiple smartphone applications were developed as tools. Either to gather and log sensor data for evaluating algorithms, for taking reference measurements, or for proofing concepts of applied navigation applications. The main applications, developed and used within the presented research, are listed in the following:

- Smartphone data logger
- Step detection logger
- GNSS logger
- PDR logger
- Application for graph-based PDR within INK project

Most of these applications were developed with the only purpose of logging time-synchronized sensor data, saved to a text file. Those sensor data files were evaluated and analyzed in post processing, in order to have accessible and reusable data. However, all algorithms described and used, are designed for and have the possibility to be coded for real time applications and implemented within a mobile operating system.

Smartphone data logger

Multiple applications for logging smartphone sensor data were developed and tested within the scope of this work. Mostly, the main approach is to log the needed data measured with the smartphone, and process that data later. This allowed for convenient adaption of the tested algorithms and repeatability of conducted tests. However, all used algorithms and approaches are constructed in a manner, so that they can be implemented in real time if needed.

The main functionalities are logging the inertial sensor data, consisting out of the accelerometer and the gyroscope data, and next to them the magnetometer data. Additionally, the barometer data, light sensor data, proximity data and temperature data can be logged.

Further, all possible WLAN and Bluetooth data can be logged, consisting out of all available radio readings of the WLAN or Bluetooth front end of the current scan. Contrary to the other sensor data, for the WLAN and Bluetooth information, a radio scan has to be conducted to gain the information of the current WLAN routers or Bluetooth beacons as well as their signal strength.

Finally, usually the GNSS fix, computed internally by the GNSS chip, will be logged. This is mostly done in order to synchronize the test measurements to an absolute time stamp, using the internal processor time stamp in $[ns]$, stored simultaneously with the GNSS fix, which has an absolute time stamp. Therewith, the logged measurements can be synchronized later, using multiple smartphones or other, external reference sensors.

As exemplary applications, the view of an inertial sensor logging application as well the view of an WLAN logging application is shown in figure 3.2.

For the inertial sensors and magnetometers, the sensor readings of every sample can be logged. Similarly, the barometric and temperature readings can be logged at every sample. Using the radio sensors, namely the WLAN and Bluetooth front ends, the name, address and received signal strength for each router or beacon can be received and logged.

Step detection logger

For reference purposes, the *google step detection* can be logged. The *google step detection* function outputs an event every time it estimates a step acted by the user of the smartphone. Tests show, that this step detection has a time lag of a couple of seconds, therefore it is hard to use it in real time for certain applications. However, for comparing multiple algorithms after logging sensor data, it is a useful tool to compare results to each other. The reason for utilizing the *google step detection* is, that it is available to all developers, and therefore can be used within applications as an inbuilt function and is directly comparable to other applications. For all sensor readings, the internal time stamp will be recorded in order to synchronize the gained measurements.

GNSS data logger

The *GNSS Logger* is a simple application, developed and provided by *Google*. It became available, when the possibility to gather GNSS raw data within smartphones became possible within *Android*. The basic concept is to provide users a quick tool to access the GNSS raw data, to be able to process that data with further software or look at the raw data. Most currently available GNSS chips, which support raw data output, are capable of providing the pseudorange measurement as well as the navigation message. The application has the possibility to show and/or log the computed location, the actual pseudorange measurements, the navigation message, the individual GNSS status indicators, as well as the standardized Radio Technical Commission for Maritime Services (RTCM) messages. The application uses a proprietary format to log the data, which can be converted to standardized formats if needed for further computations.

A sample screenshot of the basic application is shown in figure 3.3, and it is used to log GNSS measurements for the analysis presented in chapter 4.

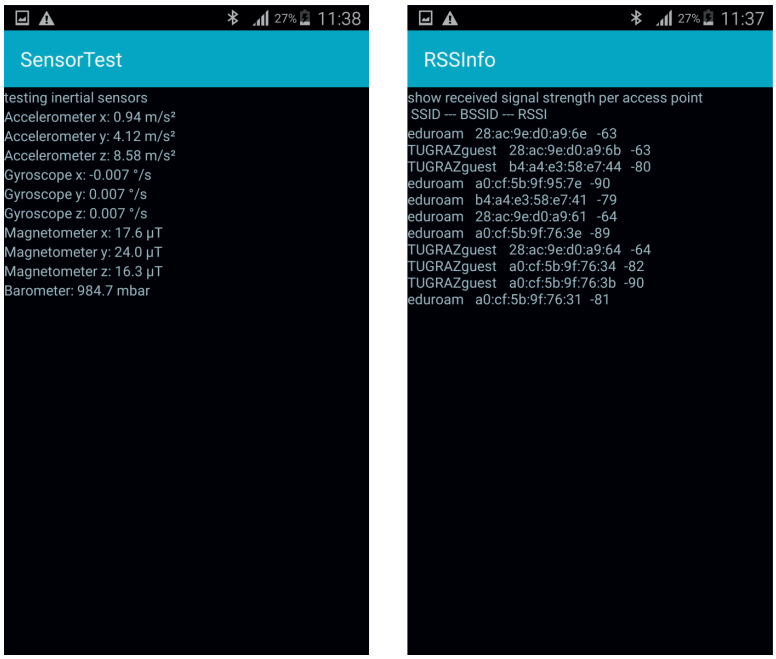

Figure 3.2: Sample screenshots of sensor logging applications

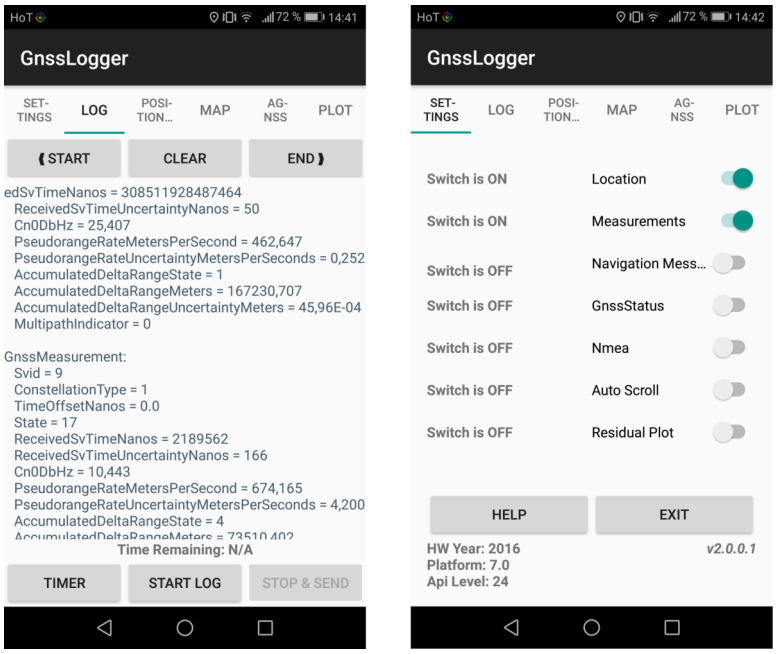

Figure 3.3: Sample screenshot of GNSS logging application

PDR Logger

One application, used as tool for reference, calibration measurements, and automated approaches within this work, is the so called *PDR Logger* application. This application was developed in cooperation of Thomas Moder and Dr. John-Olof Nilsson, research fellow at the KTH Stockholm. It is an application, that logs step events gathered in real time by a foot-mounted module and the smartphone sensor data simultaneously in a time synchronized manner.

The foot-mounted module, is an open source module based on Multiple Inertial Measurement Units (MIMU), and therefore called MIMU module or foot-mounted module. The foot-mounted module and the basic concept of foot-mounted navigation based on inertial measurements is presented in e.g. Nilsson et al. (2014a) in detail. The main view of the application is shown in figure 3.4. There is the possibility to connect up to two foot-mounted modules via Bluetooth. The Bluetooth address has to be input into the application, and when the *Start* button is pressed, the application starts connecting to the module, and logging the received step events as well as the smartphone sensor data, synchronized using the internal system time of the smartphone.

The mounting of the foot-mounted module has to be done somewhere at the foot, where the zero velocity update (ZUPT) of the foot can be observed. Within the conducted tests, the module was usually temporarily fixed externally on the shoe of the subject conducting the tests, using electrical tape.

The computation of the foot-mounted inertial navigation is carried out on the foot mounted module directly, and the processed and sampled step events are send to the application at every detected event. Since the used foot-mounted module is open source, provided by the so called *open shoe project*, the development of the PDR logger application was possible. The micro controller on the module can be controlled directly using a micro-controller software. Within the processing of the module, the processing itself as well as certain thresholds can be varied and set, depending on the usage.

The foot-mounted module is used with a one-sided sampling window of 0.25 s, meaning that 0.25 seconds after every ZUPT detected by the module, a step event is generated in the data. This sampling window can be adjusted, only if it is set too low, certain events may not be detected anymore. This means, that if events happen too fast, the output of those step events may be neglected some times. However, the cumulated step lengths are still accounted for. Since $0.25s$ proved to be stable within the module for the application used, this sampling window has to be taken into account when time-synchronizing the sensor data from the smartphone and the detected step events. Additionally, the information of this sampling window has to be taken into account when thinking about real time applications, as well as the calibration algorithms where for example only directly available, feasibility-checked step events may be taken into account.

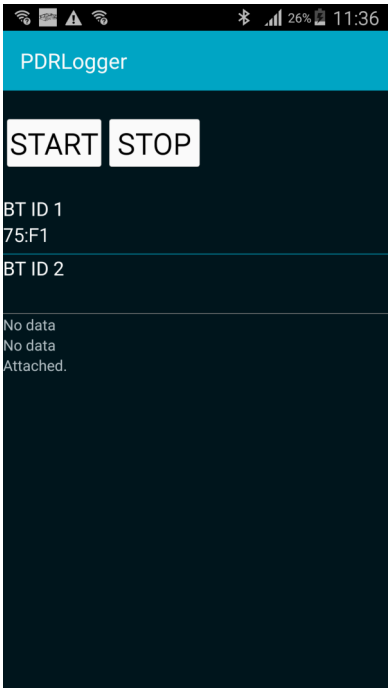

Figure 3.4: Sample screenshot of PDR Logger application

INK application

INK is the acronym of *Indoor Navigation und Kommunikation*, indoor navigation and communication, a research project for navigating people with visual impairments within public transport. INK was carried out and lead by the Working Group Navigation of the Institute of Geodesy, Graz University of Technology, from October 2016 until April 2018. The research project was performed with support from the the Austrian research program *Mobilität der Zukunft* and received funding from the Federal Ministry for Transport, Innovation and Technology in Austria. The shown application was developed within this research project from the whole project team, and parts of it will be used to demonstrate the benefits of pedestrian dead reckoning within applications. A more detailed description and results, especially of the positioning, will be given in chapter 7, in the following an overview of the application itself is given for the sake of completeness.

Because of the German speaking project team, and German being the mother tongue of all involved test users, the user interface of the application was developed in German, see figure 3.5 for an overview of the main application views.

The application itself has four distinct modules, namely route planning, navigation, vehicle communication as well as audiovisual aiding. The route planning is visible at the left side of figure 3.5, part of the navigation is visible on the middle view of figure 3.5, and the audiovisual aiding is visible on the right side of figure 3.5. The computation of the combined

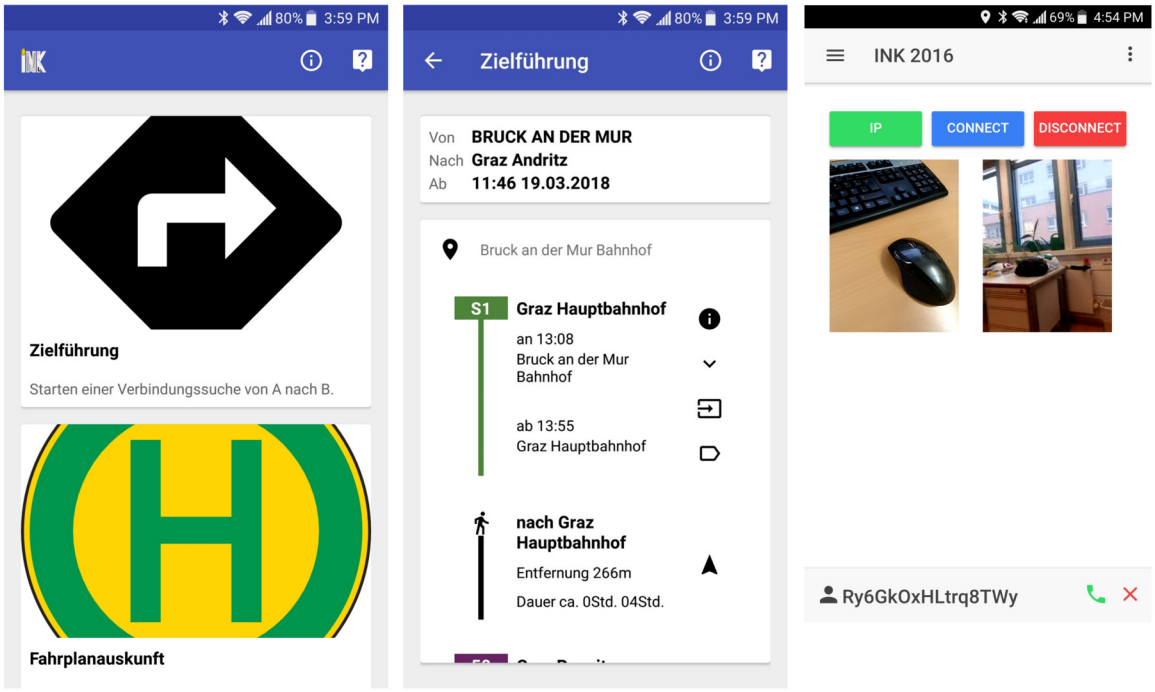

Figure 3.5: User interface of INK2016 application in German

position estimation and the navigation as well the vehicle communication is not visible in the application views, but happens at the back end of the application.

The route planning offers the option to choose a start and end location for computing an optimal route using public transport. The transit between different modes of public transport is usually done by walking within a transport hub. This transition between different modes of public transport can be guided by the application, specifically designed for people with visual impairments. The guidance is executed on present, tactile paths at the transport hub. Based on the current location and the planned route, the vehicles of public transport can then be informed, using vehicle communication, if a person wants to board or exit the vehicle. Finally, the option of audiovisual aiding is implemented using a help button within the navigation. Activating the audiovisual aiding, another user, or professional operator, can access the camera view of the smartphone running the application, as well as the planned route and current location. Using this information, the user or operator can then instruct the user of the application, providing navigation input for the public transport hub.

In chapter 7, mainly the positioning service, which serves as input for all modules, will be presented in more detail. Especially the navigation is used to show the benefits of currently available pedestrian dead reckoning approaches, tailored for visually impaired users.

3.4 Smartphone position within pedestrian navigation

Concluding chapter 3, a short discussion of the smartphone position relative to the user's body, also called smartphone mounting in some publications, is given. The smartphone is carried or mounted in different positions, with reference to the user. Naturally, a couple of smartphone positions occur, when a pedestrian is carrying or using a smartphone while walking. Multiple studies define multiple possibilities of mounting the smartphone, see figure 3.6, including:

- hand held
- swinging
- carried in a pocket
- carried loosely attached in a bag

Holding the smarthpone in the hand can mean holding it steady in front of the body, while swinging means holding it while naturally swinging the hand while walking, texting with the smartphone or e.g. phoning. Carried in a pocket comprises the modes of carrying the smartphone in multiple possible pockets, attached to the waist or torso of the user. And carried loosely attached comprises usually carrying the smartphone in a shoulder bag, but again, multiple options of carrying are available.

There are multiple studies, which aim to identify the smartphone position relative to the user, mainly using forms of motion recognition. For example, Susi et al. (2013) presents an algorithm using threshold parameters in the frequency and signal variance domain, which differs between the user being static, walking while swinging the smartphone in one hand and walking while texting, phoning or having the smartphone in a bag.

Usually, different smartphone positions relative to the body frame of the pedestrian, yield to different effects of the gait pattern on the smartphone sensor signals. Therefore, adjustments may have to be made for pedestrian dead reckoning or activity recognition approaches, depending on the used algorithms. Within the presented applications, the mountings hand

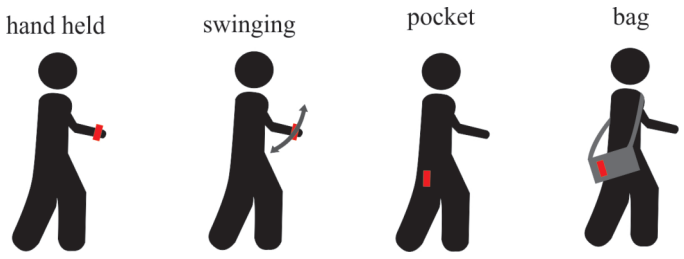

Figure 3.6: Smartphone mounting options

held, swinging and pocket are used, where the device position is assumed to be stable in their mounting. Usually, the mounting option *hand held in front of body* was used, although when not specifically mentioned otherwise, for the step detection process the approach of step segmentation is used, which is not effected strongly using different smartphone positions. Regarding the orientation estimation, only the relative change in orientation is taken into account, and therefore the orientation angle between smartphone's sensor and the user's body frame, which is dependent on the smartphone position, is not of interest for the presented tests and applications.

4 Positioning approaches for pedestrian navigation using smartphones

This chapter discusses the currently available technologies for positioning using smartphones. In an introduction, the generally available technologies are outlined, where further on the actually applied techniques are discussed in detail.

Positioning technologies can either be classified based on their measurement principle, or on their mode of positioning. Based on their measurement principle, techniques based on totally self-sustained measurements or measuring naturally available signals, and techniques based on measuring artificially created signals exist.

Techniques based on self-sustained measurements or measuring naturally available signals comprise:

- inertial navigation, based on inertial signals like accelerations or angular rates
- observing techniques, using vision sensors like cameras based on light or infrared measurements
- navigation based on the earth magnetic field

Techniques based on measuring artificially created signals, whether created and installed for the purpose of positioning or not, comprise:

- Global Navigation Satellite Systems (GNSS), high sensitivity GNSS approaches, pseudolites
- (ultra)sound-based
- radio-based (other than GNSS), which includes Wide Area Local Network (WLAN), Bluetooth Low Energy (BLE), Radio Frequency Identification (RFID) as well as Ultra Wide Band (UWB) radio signals and their variations

Contrary, positioning technologies can be classified based on their positioning mode into absolute and relative techniques. For the sake of this manuscript, the classification will be made between absolute and relative positioning technologies in the following sections.

As the most advanced as well as widespread technology, GNSS is an absolute positioning technique, which is also available within smartphones. Next to GNSS, since high sensitivity GNSS is not working properly with smartphones indoors today, alternative positioning techniques for absolute indoor positioning are discussed. Motion recognition, not being a positioning technique, but used as input for positioning techniques within this research, will be discussed within this chapter. And lastly, positioning techniques for relative indoor positioning, especially for pedestrian navigation are discussed.

4.1 GNSS positioning using smartphones

Smartphones exist since many years, where the first mobile phone incorporating a Global Positioning System (GPS) sensor chip appeared 1999. However, the first mass market smartphone, using GPS as positioning sensor, became available in 2008. Today, multi-GNSS sensors are present within all available smartphones.

GNSS chips for smartphones are currently developed and produced by, e.g., companies like *SiRF*, *u-blox* or *Broadcom*. When starting with the introduction of smartphones in 2008, chips that focused on GPS only were available, soon thereafter chips combining GPS and the Russian GNSS, GLONASS, became popular. Today, state-of-the-art GNSS chips combine up to three GNSS like the Broadcom BCM4773, which uses next to GPS and GLONASS the Chinese BeiDou or the Broadcom BCM4774 which additionally incorporates the European Galileo system. It is predicted, that future GNSS chips will be able to support all four worldwide available GNSS. The use of multiple GNSS is especially useful in urban areas. Where pedestrian walkways are often close to multi-story buildings, shading effects may be apparent and satellite visibility disturbed, or, e.g. in urban canyons, insufficient for a position computation.

Before the release of *Android* software number 26, and the first implementation of GNSS chips in smartphones capable of raw data, there was no option for retrieving raw data information such as pseudorange, Doppler or phase measurements, so only final position fixes were received. Also, currently used GNSS chips offer single-point-positioning based on pseudoranges only. For evaluating smartphone GNSS performance, static and kinematic tests with different smartphones were conducted.

Such GNSS single-point-positioning chips deliver a position information and an estimated accuracy information. However, through the smartphone application programming interface, it is not possible to identify the mode of position solution. It may be the actual computed position fix without any further filtering, estimated out of an pseudorange adjustment, or, additionally, filter processes and further computations with these positions may have been carried out. However, in our analysis, the assumption is that all tested smartphones actively used position filtering.

The three main different modes currently available and known are position estimations that get rastered, position estimations that get filtered and unfiltered position fixes. For static tests with three different smartphones, incorporating three different GNSS sensor modes, see figure 4.1. The different behavior is visible in the presented static test series of ten minutes. The test was carried out on the roof of a building on static pillars.

The first GNSS sensor chip from the *Motorola Defy* is quantifying its position solutions within a 1.5 m raster. The second GNSS sensor chip from the *Samsung Galaxy Nexus* is strongly filtering its final position estimation. The position solution is only changing, when the acceleration and gyroscope sensors of the smartphone are not stationary enough anymore. The start and end time provide the very same position solution, where in the middle of the time series, a couple of other position solutions occur in these ten minutes. The third GNSS sensor chip from the *Samsung Galaxy S2* is providing the output of the pseudorange adjustment, visible randomly distributed position estimations.

In the non-static case, position solutions based on smartphone measurements offer accuracies within a couple of m, with possible outliers of up to 30 m present in urban canyon scenarios. The overall accuracy mainly depends on the geometrical satellite constellation as well as the satellite visibility from the smartphone point of view, see Dabove (2014).

Since *Android* version 26, GNSS raw data is available within smartphones. However, the smartphone used has to feature one of the few chipsets available, which is able to output pseudoranges. The smartphone tested according to GNSS raw data, is a *Huawei P10*. The test data, shown in figure 4.2, shows the evaluation of statically recorded pseudoranges during a timespan of 10 minutes. The pseudoranges were evaluated using the open source software *RTKlib*, with a singlepoint solution only and without the use of any positioning filter. The data shown is centered on the actual position center and has a two-dimensional standard deviation of 5.4 m within this timespan of ten minutes, using six visible satellites.

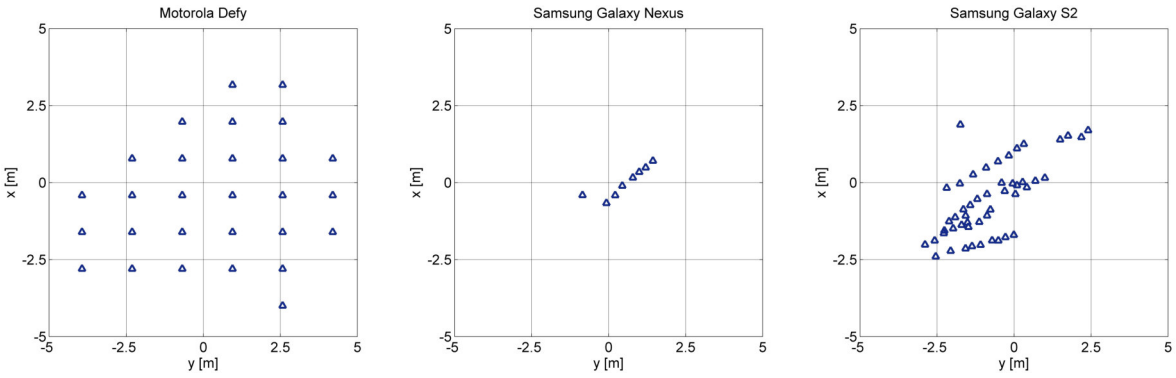

Figure 4.1: Static GNSS position estimations, using on-chip computed solution

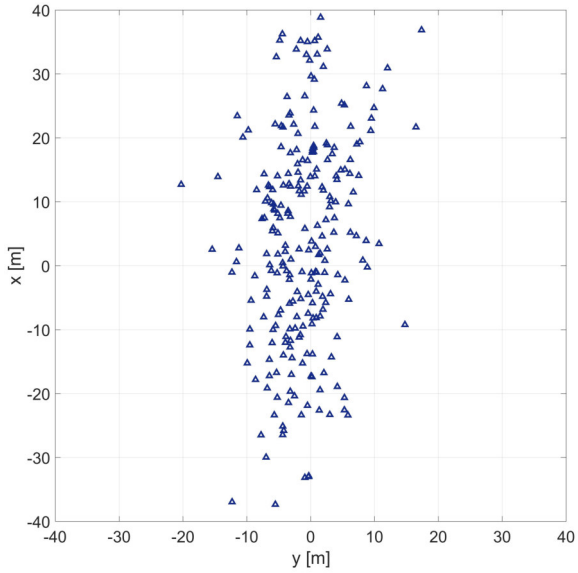

Figure 4.2: Static GNSS position estimations, computed using raw data measurements recorded with a smartphone

Comparing these position estimates, on the one hand out of final position fixes with an involved filtering process, and on the other hand an independent position computation, estimated out of raw pseudoranges, it is visible that there is a significant data filter process apparent within since GNSS position fixes.

Since the introduction of the Broadcom BCM47755 chip for smartphones in 2019, it is possible to retrieve dual-frequency GNSS measurements. At the time of writing this manuscript, only a couple of smartphones from Xiaomi, Huawei, Oppo, Honor or OnePlus had dual-frequency GNSS enabled. However, the availability of dual-frequency support is expected to rise to an industry standard in the next years. Additionally, even phase measurements should be possible since the possibility of activating the state *code lock* in *Android* is available. *Code lock* is a GNSS tracking state, which is locking the code, and therefore does get rid of cycle slips when activated. However, the usefulness of phase measurements using smartphones, without the use of more precise antennas, has to be proven.

Out of the advanced GNSS processing strategies, the two basic concepts are relative positioning and Precise Point Positioning (PPP). PPP is an ideal application for smartphones, since the downlink for the necessary PPP corrections is already present within the smartphone's Internet connection. The computation of PPP is shown to deliver position solutions within a couple of 0.1 m. However, currently there are still lag times of a couple of minutes present at the start of a PPP computation, see Huber (2015) for details. When working more robustly, smartphone-based PPP could be of major importance in big cities, with multipath errors at single point solutions being critical for smartphone GNSS.

4.2 Absolute indoor positioning

Generally, all absolute positioning techniques, developed in the last decades, are based on stable, available and measurable signals, mostly signals in the radio band. This comprises long-range radio navigation, GNSS, as well as techniques based on other forms of radio patterns. An extensive overview of possible techniques is given in e.g. Mautz (2012). The most common methods for absolute indoor positioning solutions comprise positioning based on:

- radio signals
- magnetic signals
- vision
- pseudolites
- high sensitivity GNSS
- mobile network positioning

Since this section discusses absolute indoor positioning techniques, especially usable for pedestrians and therefore smartphones, GNSS and its complementary solutions like high-sensitivity GNSS, repeaters or pseudolites are not taken into account. Out of the remaining possibilities, techniques, so far demonstrated to be usable for absolute indoor positioning using smartphones, are either based on radio signals or magnetic signals measured with the smartphone, on ultrasonic signals from the smartphone, on mobile network positioning or based on vision.

Radio-based indoor positioning include the use of all measurable radio signals. This comprises e.g. WLAN, BLE, UWB and RFID. Usable within smartphones today are WLAN and BLE, basically available on the same frequency band, as well as RFID, which is also known as Near Field Communication (NFC) within *Android*. Received radio signals can, based on the received signal strength indicators and path loss models or on round trip measurement approaches, be directly correlated to direction estimations from the radio sources. This is the basic concept of all existing radio positioning approaches, including long-range radio navigation as well as satellite positioning. Therefore, received radio signals can be used for positioning using cell identification approaches, lateration techniques or so called location fingerprinting approaches.

Cell identification is the most basic approach, and simply identifies the most likely visible radio beacon, e.g. based on the received signal strength indicator, and then assumes the same location for the device than for that beacon. The accuracy of a cell identification is directly correlated to the spacing between the used radio beacons. Therefore, this approach

is most suited to low-cost, highly available beacons, which is true for e.g. BLE beacons or RFID tags.

One of the simplest practical approaches for indoor positioning using hand-held devices, is based on using RFID tags. RFID tags are simple, small and monetarily efficient. However, they are only detectable within a couple of dm up to some m, strongly dependent on the utilized frequency-band and whether an active or passive mode is used, see Chen (2012) chapter 4. Therefore, RFID tags are widely used for identification of e.g. books in libraries or logistical solutions in warehouses. Equally, they can be used for estimating the position of a device, if sufficiently enough RFID tags are distributed within a building, e.g. at every door, every office and every couple of meters along the hallways. Chen (2012) chapter 4 thoroughly investigates the possibilities of lateration and location fingerprinting next to cell identification using RFID tags.

Lateration using radio signals is an approach, where a position is computed based on two or more distance measurements from sources with known coordinates. The position accuracy is dependent on the number of distance measurements as well as the measurement setup, e.g. the spatial distributions of the signal sources. Using smartphones, the only possibility to retrieve distance measurements is based on assigning an distance operator to the received signal strength measurement. Since these operators usually rely on the assumption of a free air propagation, they fail when the measurement path is not free air and e.g. a wall or other sort of building structure. Since the present building infrastructure is very complex to model for the path loss, this usually results in erroneous distance relations. Approaches to enhance the lateration technique are known, e.g. Retscher and Tatschl (2016) successfully suggest the use of differential lateration. However, for cell identification as well as lateration, the radio beacons have to be known coordinate-wise. Due to the mentioned restrictions of cell identification and lateration, the concept most utilized within radio-based positioning is still location fingerprinting, see sections 4.2.1 and 4.2.2.

Absolute indoor positioning approaches using the **magnetic** field have evolved in the last years. Either the earth magnetic field can be used as signal source, for e.g. estimating absolute directions from multiple positions. However, most often the local magnetic deviations, induced through building infrastructure, are utilized as spatial information to estimate positions. Most approaches presented are multimodal stochastically, and therefore need additional information, e.g. distance traveled between two measurement points. On its own, the magnetic field can be used similar to a location fingerprinting, see section 4.2.1. There, instead of radio patterns, the leveled magnetic vector is used as a spatial pattern. Robertson et al. (2013) presents a working algorithm, which simultaneously localizes a device and matches the magnetic surrounding, based solely on the magnetic sensor. Also, the magnetic field can be used as additional parameter for combined approaches, see e.g. Ettlinger and Retscher (2016) for a combination of the ambient magnetic field as well as WLAN and RFID. Ettlinger and Retscher (2016) show, that the overall positioning error is the smallest when

all ambient radio sources as well as the magnetic field are incorporated into the position estimation.

Vision is one of the major self-reliant positioning techniques. Either vision from the smartphone camera itself can be utilized, or approaches using external vision signals like external cameras can be used. One major concept for visual navigation using smartphones is presented by Ruotsalainen et al. (2013). Here, the concept of using the camera data to process a visual gyroscope and a visual odometry solution is presented. Therewith, a dead reckoning can be computed, and the camera can be used instead of inertial sensors. Next to the concept of visual-based dead reckoning, the approach is to match an image with an image stored in a spatially referenced database, and therefore yield an absolute position information if an image is matched, see Chen and Guiness (2014). If using landmarks within the images, the camera and therefore the smartphone orientation can be estimated. Additionally, if using a times series of images, the change in orientation of the used camera or smartphone can be estimated robustly and drift-free. Note, that the obvious disadvantage for all vision-based concepts, is the dependency on available light.

Lastly, the most direct position estimation of a smartphone can be done using information of the currently assigned **mobile network** cell. The basic information when using the mobile network as position information, is the currently used cell of the smartphone. In urban environments, the cell spacing can be as closely as 30 m, where in rural areas, the cell spacing can be up to some km. More sophisticated position estimations can be made using angle of arrival information. Details on accuracy and advanced cellular network positioning are presented in e.g. Groves (2013).

Alternative approaches, using e.g. acoustic signals or visible light measured by the smartphone, are recently being discussed. However, all known approaches have underlying restrictions, for example corresponding to acoustic and light matching, additional speaker and microphone infrastructure, or holding the smartphone with the ambient light sensor strictly facing upwards. The concept of combining multiple approaches, whether it being two or more absolute positioning techniques or an absolute as well as a relative positioning technique, is usually called hybrid positioning. Examples are, e.g., combining either GNSS or an absolute indoor position technique with a relative position approach like Pedestrian Dead Reckoning (PDR), combining multiple forms of radio-based positioning, or combining radio-based positioning with magnetic positioning approaches as well as vision-based techniques.

Within the presented techniques, the most stable positioning modes for smartphones are currently based on measuring radio signals. Most of the described techniques for absolute positioning either require specific and additional equipment, or they use computationally expensive approaches as well as huge data sets to work properly. As main technique, usable without additional equipment and working using multiple radio signals, location fingerprinting is utilized by many applications.

4.2.1 Smartphone location fingerprinting

Consequently, location fingerprinting is used as absolute position approach for smartphones within the presented applications. Location fingerprinting does not require the installation of additional infrastructure, and the location distribution is distinct contrary to, e.g., magnetic fingerprinting. It is an approach, usable without additional infrastructure, and unimodal location distribution. However, the establishment of the fingerprinting data base might be time consuming using traditional approaches.

The basics of location fingerprinting are shown in more detail in Hafner et al. (2013), as well as in e.g. Chen et al. (2012). Location fingerprinting consists of an offline phase, as well as an online phase. During the offline phase, a data base of radio signal measurements is established, also called radio map. During the online phase, the currently measured radio signals are used to estimate the current position. For establishing a fingerprinting database, the Received Signal Strength Indicator (RSSI) measurements at all defined reference points are obtained and stored during the offline phase. In the online phase, the currently measured RSS indicators $RSSI_i$ are compared to the previously measured $RSSI_R$. This is best done using a stable quantity like the L_2 norm:

$$D_i = ||RSSI_R - RSSI_i||_{L_2} \tag{4.1}$$

The most likely reference position can then be obtained using

$$\hat{p}_i = min(D_i). \tag{4.2}$$

With the fingerprinting concept established, based on the most likely reference positions, a blunder detection or further enhancements to improve the position estimation may be computed. One often referred concept is the k nearest neighbor approach, where the k most likely reference positions are taken into account for a final position estimation, e.g. using

$$\hat{p}_{KNN} = \frac{1}{k} \sum_{i=1}^{k} (\hat{p}_i). \tag{4.3}$$

A comprehensive overview of currently used RSSI-based fingerprinting is given in Kushki et al. (2012), and the most comprehensive literature study about WLAN fingerprinting known is presented in Khalajmehrabadi et al. (2017). Additionally, further approaches, considering other quantities next to the L_2 norm, like probabilistic methods, are presented in Sanford et al. (2012).

Following, the focus lies on the explanation of the practical implementation of location fingerprinting, using radio signals measured by smartphones. The tool for computing and analyzing fingerprinting, developed during this research, is shown in figure 4.3. The main options within

the fingerprinting tool, are *database*, *input*, *output*, *process* and *plot analysis*. Within *database*, a database can be created out of multiple radio measurements at known coordinates. Additionally, a created database can be saved or loaded for processing. As *input*, a radio measurement file, and optionally a *GPS file* containing GPS fixes with absolute timestamps for time synchronizing the radio data, can be loaded. Accordingly, as *output*, the processed fingerprinting solution can be exported. Within *plot analysis*, the processed positions solutions can be visualized and analyzed within the available test bed.

The option *processing* contains multiple options for processing the fingerprinting estimation. Mainly, a basic location fingerprinting is computed. A default minimum signal strength, for non-simultaneously measurable signal sources can be set. Additionally, a time filtering of the radio signals in the received signal strength domain can be computed. The final location fingerprinting can be estimated using the most likely fingerprint, a nearest neighbor approach as well as a weighted nearest neighbor approach.

Location fingerprinting can be processed using all sorts of distance indicators and radio signals. However, the most prominent as well as available radio signals within smartphones are the WLAN as well as BLE radio band. The details of the differences of those radio bands are discussed in detail in e.g. Wilfinger (2015) and Reitbauer (2017b). The main difference between WLAN and BLE is their emitted signal strength, and therefore received signal strength. The theoretical path loss, based on the distance of a smartphone to either a WLAN as well as a BLE beacon, show the same decay over distance. However, the received signal strength of a BLE beacon starts with approximately 20 db less at $d = 0$ then the WLAN signal strength.

Location fingerprinting with smartphones, based on either WLAN or BLE as radio measurements, is comparable and equally multimodal in it's stochastic behavior. The main difference is, that due to the lower signal strength threshold of BLE, more beacons have to be utilized for a similar position accuracy then when using WLAN as radio source. Studies exist that show, that using BLE might be more accurate, e.g. see Faragher and Harle (2015). However, using an appropriate number of beacons for a fair comparison, no significant difference in the final position accuracy can be gained compared to WLAN fingerprinting, using the test bed shown in figure 4.4 and 4.5.

Following, two examples of location fingerprinting using a smartphone as measurement device are shown. In the building *Steyrergasse 30, Graz, Austria*, at the 4th floor, measurements were taken. Starting at the north-east corner of the floor, walking along the floor to the south-west corner of the floor, turning around and walking back half of the distance to the stairways. During this walk, a smartphone was hand held, recording WLAN as well as BLE measurements with 1 Hz. While on this floor only 2 WLAN routers are naturally present, up to 10 WLAN signals from other floors and the other wing of the building can be observed. At the time of the tests, no BLE beacons were already present, although computers on every office could function as beacons. Therefore, up to 20 BLE beacons were placed along the floor

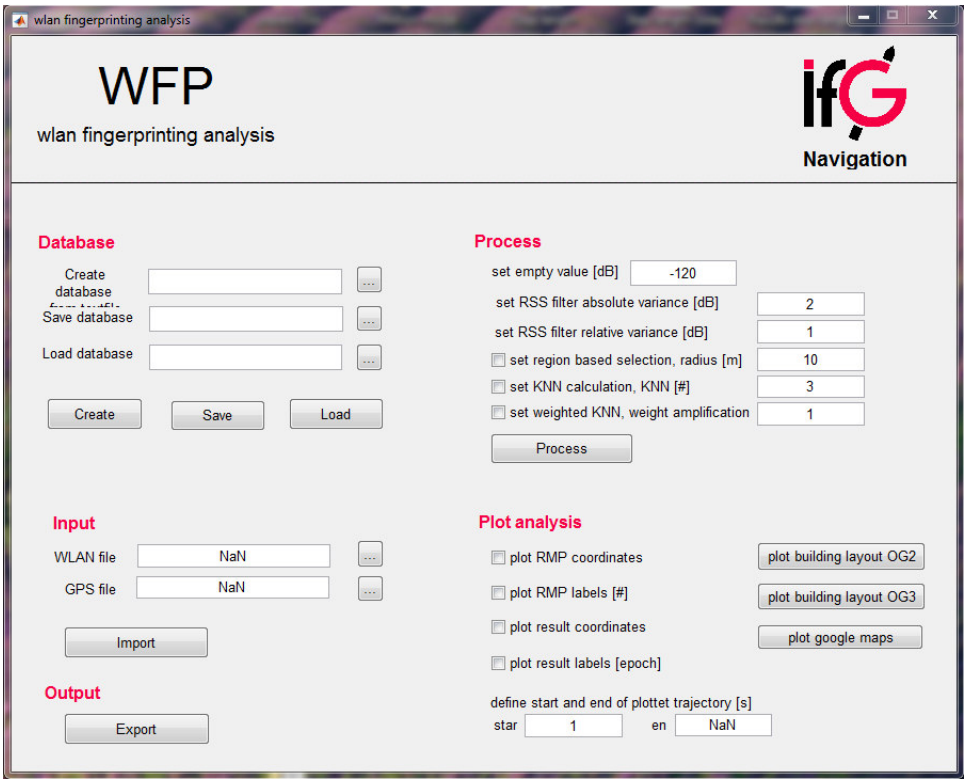

Figure 4.3: Fingerprinting tool

as well as inside the rooms for the presented test bed on each floor. The results of the WLAN fingerprinting are shown in 4.4, where the results of the BLE fingerprinting are shown in 4.5. Both presented trajectories show the direct estimation out of the fingerprinting, without any additional filtering process.

Without a dedicated reference system for this test setup, no stochastic evaluation can be presented. However, room level accuracy is achievable robustly using either WLAN or BLE as radio source. The final position accuracy is strongly dependent on the fact, of how many usable BLE or WLAN beacons are available. Although the number of available radio signals as well as their absolute signal strength are indicators of the absolute position accuracy, no overall accuracy measure is directly obtainable from location fingerprinting.

4.2.2 Advanced location fingerprinting

Where section 4.2.1 describes the basic concept of location fingerprinting and the implementation specifically for smartphones, this section describes the possibilities for advanced methods using location fingerprinting.

There are multiple approaches present to estimate an absolute position using smartphones. However, out of the concepts shown, only a few do not rely on either additional infrastructure,

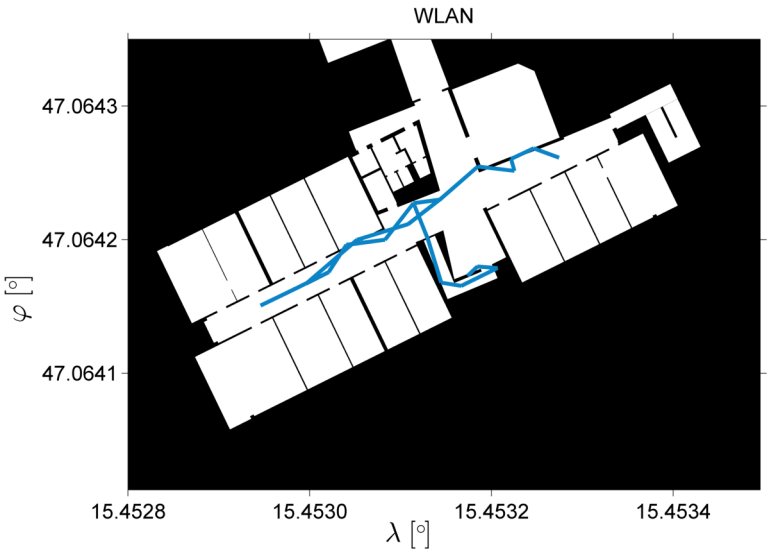

Figure 4.4: Example of location fingerprinting using smartphone data and WLAN signals

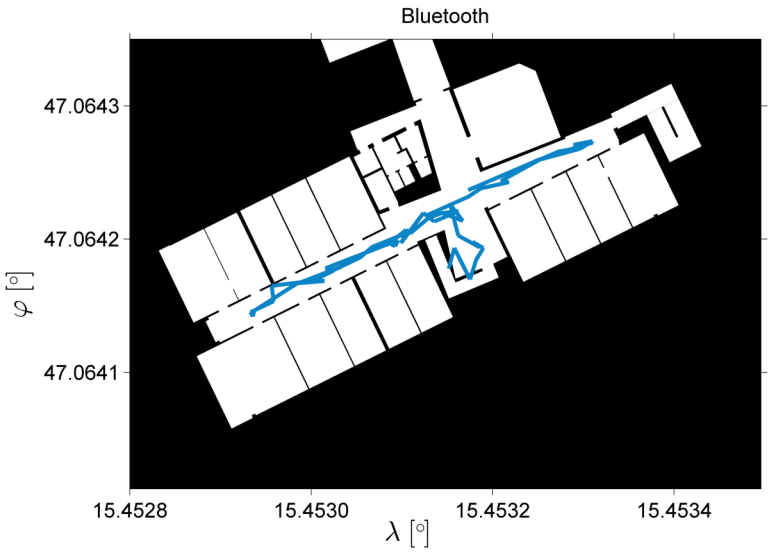

Figure 4.5: Example of location fingerprinting using smartphone data and BLE signals

or on sensors which are currently not implemented within available smartphones. Therefore, location fingerprinting based on already available radio signals, is still the most common method for estimating absolute position solutions for smartphones. This is why there is currently research performed on advancing and improving the offline as well as online phase of location fingerprinting.

For improving the offline phase, multiple options exist, e.g. radio map interpolation, the automation of the radio map creation or the reduction of the possible radio map points. Radio map interpolation is an efficient and easy method of extending existing radio maps. The concept is to, e.g., use a simple path loss model, and interpolate the existing radio map into a more detailed database. The main problem is the assumption of the path loss model, which assumes free air or line of sight between all interpolation points. Another approach is to automate the process of the radio map generation itself. Multiple crowd-sourcing approaches for automating this process have been presented, see e.g. Chintalapudi et al. (2010). The concept is, to measure the radio pattern while estimating the own position, and therefore creating the radio map. Since this usually happens in spaces without other possibilities of positioning, the main form of position estimation used is relative positioning. Due to the drift in the position domain when using relative positioning, this concept only works when sufficiently enough data is utilized. For a more detailed overview of advanced approaches for radio map creation, see Reitbauer (2017b). An approach for a sparse radio map, or reduction of the possible radio map points, is given by Retscher and Hofer (2017) with the concept of an intelligent checkpoint sequence. Retscher and Hofer (2017) presents, that when using a meaningful selection of radio map points in an logical order, the temporal effort of the radio map creation as well as the matching process can be reduced significantly.

For improving the online phase of location fingerprinting, approaches to use different metrics or different averaging of the most likely radio map points exist. One method is to use a probabilistic fingerprinting, where the likelihood distribution is expressed through the probability of a certain signal strength pattern. Reitbauer (2017b) thoroughly studies the impact of different distance metrics for a weighted nearest neighbor approach, although with no significant differences between different metrics for the final position accuracy.

4.3 Motion recognition

Motion recognition, by its name, deals with the identification and recognition of motions. It is seen as a subtask of pattern recognition or machine learning, dependent on which approach is applied to the problem. While the differences between machine learning and pattern recognition are discussed in chapter 1, the difference in their application with motion recognition are mainly the personalized data. Machine learning doesn't need the feature engineering, a major part of the classification process, because it learns the features itself. Also, it outperforms feature classification with any given dataset, that is used to train the

learning algorithm. However, the disadvantage is, that in order to function robustly, a large dataset of reference data is needed, which is problematic when personalized reference data is needed. In the future, maybe transfer learning could be a possibility for human locomotion recognition using smartphone data to create a huge amount of reference data. However, within the scope of this work, classification approaches are utilized for locomotion recognition.

Indoor positioning based on smartphones is desirable for many applications, for example for location-based services. In order to achieve a robust and ubiquitous position solution for a smartphone the motion of the pedestrian is considered to be very important. A correctly detected motion of the pedestrian may be used to support relative positioning solutions or Bayesian filtering approaches tremendously, Lee et al. (2014). Motion recognition can either support absolute and relative positioning approaches, or can be used to directly estimate velocity information of the pedestrian carrying the smartphone to calculate a relative position, Labrador and Yejas (2014).

For the task of motion recognition using pattern recognition or machine learning, the specific motions have to be defined. Multiple sets of motions are useful for different applications, where the two main sets of motions for navigation might be mode of motion as well as locomotion.

Mode of motion, in navigation, usually means the currently used mode of transportation or human motion. This comprises walking, biking, taking a train, riding a car and so on, as well as alternations of these modes. Locomotions strictly mean the mode of a human, that is standing, walking, running and so on. Where mode of motions as well as locomotions are being discussed within motion recognition for a while, especially with smartphones no commercial available locomotion detection exists. Contrary, a mode of motion recognition is already implemented within *Android*.

Mainly, mode of motion identification can be used for routing as well as positioning, where locomotions are mostly useful as input for positioning. Especially for pedestrian positioning using smartphones, it is suggested that locomotions are used to enhance positioning.

4.3.1 Locomotion Recognition

Human motion recognition describes the process of detecting the actual carried out motion of a person. This is generally possible using either external sensors, mostly visual sensor systems, or wearable sensors fixed to the body of the person. Possible sensors include location sensors, accelerometers or physiological sensors, mainly dependent on the definition of the motions itself. For an extensive survey on human motion recognition using wearable sensors, see Lara and Labrador (2013).

For pedestrian positioning using smartphone data, it is possible to incorporate human motion recognition directly into the PDR approach. Therefore, the basic locomotions are defined as motions and are the most helpful input for pedestrian positioning approaches. Other possible motions might include daily motions, activities, multimodal motions and many more. The basic locomotions used within this work, chosen accordingly to Bobokov et al. (2015), Moder et al. (2016), Nguyen et al. (2015) and the ISO/IEC 18305 standard, comprise

- standing,
- walking normal,
- walking fast,
- walking upstairs and
- walking downstairs.

Machine learning comprises the fitting of measurements to prior trained measurement instances. The main concept of machine learning, or pattern recognition, is similar regarding all possible approaches. However, for creating training data, extracting features and generating models for classification, multiple approaches exist, see e.g. Bishop (2006) for details. The basic workflow of a motion recognition based on machine learning may be given with:

1. feature creation
2. classification computation
3. estimation of motions

With smartphones, sensors like the acceleration, gyroscope and pressure sensor may be used as input for detecting motions. Since human motions, and especially locomotions, happen over seconds rather than samples of 100 Hz, a feature extraction out of these sensors is necessary. This accords to the transition from a sample-based approach into a window-based approach, so features are computed within a certain temporal window. Features may be computed out of the mentioned sensors in the frequency or time domain using the norm or gradient measure or being based on deviation and correlation. See Labrador and Yejas (2014) for details on feature selection for specific motions. In figure 4.6, the principle of motion clustering within a plot of three features is shown. When combining certain features, some motions will still be confused but others may be distinct. For example, walking upstairs compared to downstairs is easily distinguished by a function of the pressure gradient, where standing compared to other locomotions is distinguishable using a function of the acceleration norm.

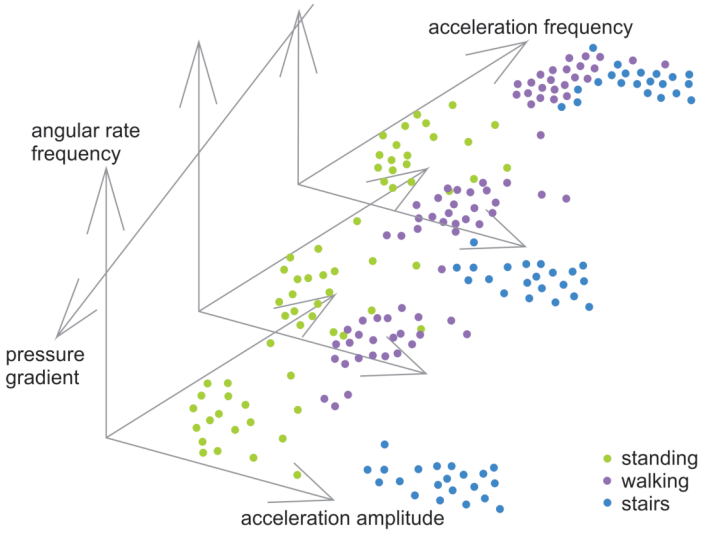

Figure 4.6: Example of multi-dimensional feature plot for locomotion recognition

For classification, different approaches exist, see figure 4.7 for examples. The most straightforward approaches use linear models for classification and classify the input vector to one class, so the decision is linear. This corresponds for e.g. to a nearest neighbor or decision tree classification. More sophisticated, support vector machines define certain basis functions based on the training data and are therefore able to be more precise in the classification. Additionally, neuronal or Bayesian networks may be used for classification. These use a fixed number of basis functions but allow for adaptation of their feature input, Bishop (2006). Within this work, a classification tree approach is applied.

The generation of training data and the following creation of the classification model is of high importance for the performance of the motion recognition. Where Bobokov et al. (2015) proposes a fixed model for locomotion recognition, best results may be achieved using personal parameters. However, the creation of extensive training data, which is necessary for establishing a proper personal model, is time intensive. Consequently, approaches to automatize these processes are proposed in chapter 5.

Based on features of the accelerometers, gyroscopes and pressure sensors, it is possible to distinguish the defined locomotions. Theoretically, all other available sensors, like the magnetometer, ambient light sensors etc. can be utilized for the creation of features. However, the most stable results are achieved using mainly the accelerometers as well as pressure sensors. Features are then engineered out of the sensor measurements over the period of a selected time range, which is obviously defined in the time span of a step for the sake of locomotion recognition. Features can consist of, for example:

- activity unit
- maximum

- mean
- standard deviation
- root mean square
- inter quantile range
- mean frequency component
- gradient

These features are then computed for the selected time range. Most approaches for feature engineering are straight forward, where only the activity unit is a specially designed parameter within motion recognition, see Moder et al. (2014) for details.

For the classification process, a machine learning software package called *Weka* is utilized. *Weka* is a software package developed at the Waikato University, New Zealand, and is published under a general public license for use. It is possible to call *Weka* from external scripts. Therefore it is implemented in the tools presented. The use of *Weka* for locomotion recognition using smartphone sensors is first shown in Wisiol (2014), and is used as starting point for the results presented. Weka provides tools for data preparation, data clustering, classification and visualization. Within this work, the classification tool is utilized based on specifically chosen and computed features. The classification is chosen to be computed using *Weka*, because this is an computationally expensive approach, which is already implemented successfully and is open source in this toolbox.

For the presented analysis, a decision tree classification, corresponding to a C4.5 classification approach, see Bishop (2006), is performed. This approach of utilizing a classification tree is used because of its low complexity. Additionally, a tree classification is chosen because of the usefulness in the motion transition approach, where it is possible to estimate a probability for every motion at every epoch.

Through correlation-based feature selection, it is possible to choose the most essential features for creating the classification model. Theoretically, all possible features computed out of all possible sensor measurements will deliver the best results. It is advised to focus on the most essential features in order to control computational efficiency. For the proposed locomotion recognition, exemplary results from Moder et al. (2015) show recall values of about 97.7 %, see table 4.1. The defined locomotions comprise standing, walking normal, walking fast, walking upstairs and walking downstairs. The classification is computed using a manually created set of training data.

The incorporation of motion recognition into PDR is advised and shown in Moder et al. (2015) and Bobokov et al. (2015), and discussed in chapter 5. If correctly identified, the current locomotion is a useful parameter within the step length estimation approach. Especially when walking on stairs, where the usual length of a stair is in the range of 30 cm and the

Table 4.1: Cross validation of locomotion recognition

Weighted Average	**Recall values [%]**				
	Downstairs	Standing	Upstairs	Walking fast	Walking normal
97.7	92.3	99.9	93.6	96.5	98.3

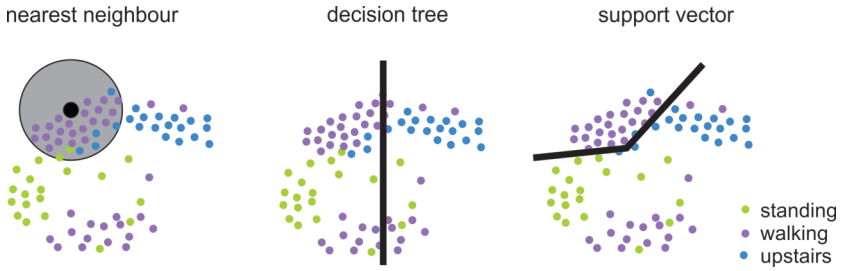

Figure 4.7: Examples of most common classification approaches

mean step length is in the range of 60 *cm* to 80 *cm*, common PDR algorithms tend to overestimate the step length at stairs dramatically. Also, one parameter set of the whole step length relation might be used per locomotion. Stair lengths are usually standardized and strictly 30 *cm* in length, however on shorter flight of stairs or winding stairs, this is not necessarily true. Therefore, stair lengths may get coded within a geographical information system, so the step length algorithm can pick the actual stair length according to the coarse position. For more details on locomotion recognition using smartphone sensor data, see Wisiol (2014).

4.3.2 Locomotion transition

Locomotion transition is dealt with in Moder et al. (2015), where you can find the following introduction: "Since a tree classifier is used for the final classification, the resulting probabilities are the product of multiple hypothesis tests within the classification tree. This leads to the calculated probabilities for each motion, more specifically one locomotion, being either close to 0 or 1 for each epoch. Therefore, a filter process on the motion probabilities is introduced within this paper. In human locomotion it is unusual for the motion state to change erratically between epochs. For example, it is unlikely that a user is changing from standing to walking upstairs to walking fast to walking downstairs in consecutive epochs. Therefore, a filtering in the domain of the motion probabilities may be carried out. Since the motions from the classification are given as probability between 0 and 1, a transition parameter, which defines the probability of the transition between two motions, is introduced. Therefore, the probability of the current motion P_{motion} is modeled as product of the motion probability $P_{classification}$ from the classification algorithm with the introduced transition

probability $P_{transition}$:

$$P_{motion} = P_{classification} P_{transition} \tag{4.4}$$

This corresponds to a filtering of the motions in the motion domain. The transition probability $P_{transition}$ defines the transition from epoch k-1 to epoch k.

However, this correction for the motion transition is only applied, when the classification shows a change in the current motion. Since no mathematical model describing the probability of a pedestrian motion transition exist, this transition has to be defined empirically. However, this transition probability has to be directly correlated to the numbers of its consecutive appearance of a motion transition. The transition probability is therefore modeled as $P_{transition} = f(x)$, with x being the number of the consecutive appearances of the particular motion transition:"

$$P_{transition} = 1 - e^{-ax} \tag{4.5}$$

The transition probability is calculated with the empirically defined parameter a, where $a = 2$ for a very likely transition, $a = 1$ for a likely transition, $a = 0.5$ for a unlikely transition and $a = 0.3$ for a very unlikely transition. One the one hand, this transition parameter has to be defined empirically, and on the other hand the transition itself has to be defined empirically, depending on their transition likelihood.

The transition probability modeled as $\alpha - \beta$ filter was first introduced in Moder et al. (2015), and explained with: "For example, standing to walking or walking normal to walking fast were modeled as very likely. Contrary, walking upstairs to walking downstairs or walking fast to standing were modeled as unlikely. This motion probability, corrected by the transition parameter, is then filtered in the time domain. Based on the assumption of a consistent motion, a filter process of the motions by a simple $\alpha - \beta$ filter is carried out. Such a $\alpha - \beta$ filter is described through two variables, herewith through the motion and its variation. Analog to Painter et al. (1990), the $\alpha - \beta$ filter for motions is given in matrix format. The motion in the parameter state corresponds to the motion probability P_{motion} in this time filtering. The whole process of a $\alpha - \beta$ filter corresponds to a damping in the time domain and is a computational inexpensive method of filtering. The parameters α and β may be chosen empirically but have to be adjusted to the underlying time process. However, as with all best-fitting linear estimations, the signal is assumed to be Gaussian with zero mean."

In figure 4.8 the estimated locomotions are shown for each epoch. Standing is noted as ST, walking normal as WN, walking fast as WF, walking upstairs as US and walking downstairs as DS. In the presented time series, the test started with the user standing, walking normal, walking fast, walking normal, followed by walking downstairs and walking normal, fast and normal again, finishing the test with standing. The estimation of the classification is directly the output of the classification tree prediction. The filtering of the classification is the

locomotion recognition after applying the transition probability and time filtering the locomotions. As visible, blunders are present in the locomotion domain, which can be overcome with introducing the transition probability. The main known drawback of the filtering in the motion domain is, that short-term motions, like walking upstairs or downstairs on a short flight of stairs with only two or three stairs, or walking fast for only two or three strides, may not be detected anymore.

4.4 Pedestrian dead reckoning

Pedestrian Dead Reckoning (PDR) describes the approach of consecutive computation and summation of the relative changes in position. Especially for pedestrians, this may be achieved using the typical movements and signals pedestrians introduce while walking on their feet, the gait pattern. Possible approaches comprise sensors mounted to the foot of pedestrians for rigorous inertial navigation, using visual odometry for identifying steps or, most usable with smartphones, using sensors loosely attached to the pedestrian for detecting step patterns. Sensors incorporated into smartphones today, usable for PDR, are accerelometer and gyroscope sensors. The basic phases of PDR are:

1. detection of step event
2. estimation of step length
3. computation of step orientation

Steps can be detected with the acceleration or angular rate measurements, strongly dependent on the mounting of the used sensor. When mounted rigidly to the body, the detection of zero velocity updates and zero angular rotation updates can directly lead to the detection of steps. If the sensor is loosely attached to the body, approaches like acceleration zero crossing, principal component analysis or step segmentation can be used.

The step detection can be done integrating the accelerations over time, $\iint a_{total}\,dt\,dt$, if the sensor is rigidly mounted to the body, e.g. the foot, where it can observe zero velocity updates. If the used sensor is loosely attached to the body, empirical step length formulas might be used.

Finally the step direction estimation can either be done in the absolute domain, the relative domain or using supported approaches like map matching where the detected steps are matched on edges. In the following sections, the approaches for smartphone-based PDR will be described and elaborated.

The whole process of a PDR implementation is given in figure 4.9 as flow chart. The main event in the code is the step detector itself, running in real time, with a defined moving window function. Whenever a step is detected, the step length estimation, the heading estimation as

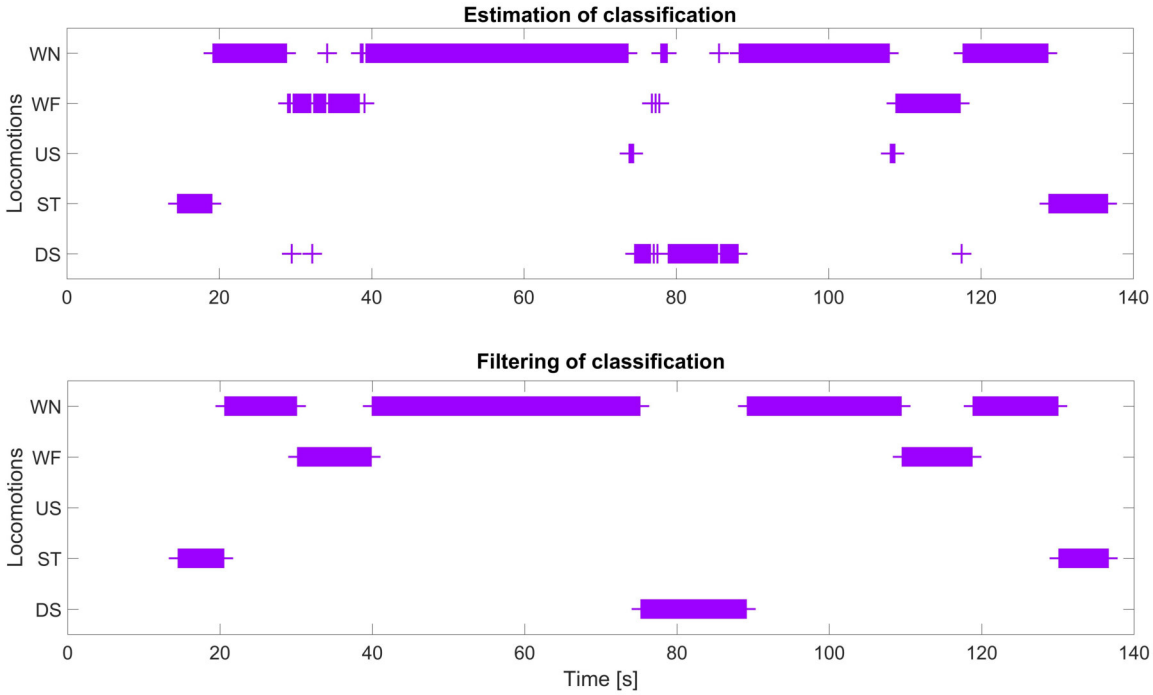

Figure 4.8: Classification before and after filtering and applying a transition probability

well as, dependent on the application, the height estimation and floor identification is called. After going through all functions of the PDR implementation, the data may get sampled, e.g. by a sampling function in the time domain, every 1 s, or in the step domain, every step.

4.4.1 Step detection

The step detection is the most important task within PDR, because the tasks of step length estimation as well as step direction estimation are based on accurately detected steps. The methods usable for step detection comprise:

1. zero crossing
2. frequency-based method
3. segmentation
4. principal component analysis
5. machine learning

The detection of a step using a smartphone as sensor, is usually done using either the smartphone acceleration or angular rate measurements, or a combination of both. Alternative

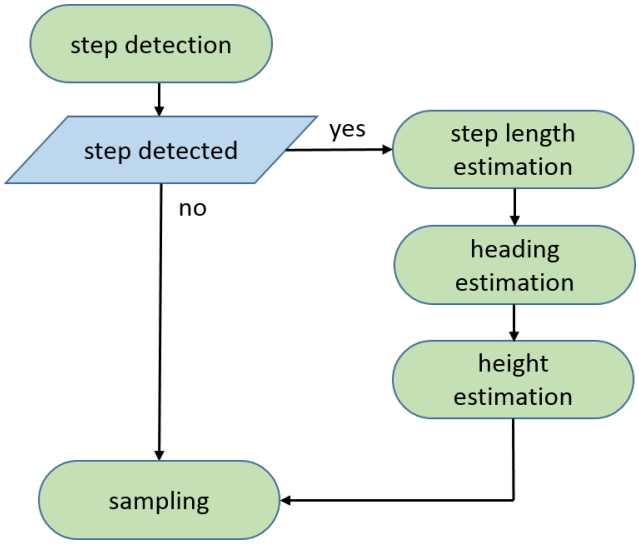

Figure 4.9: Pedestrian dead reckoning processing

approaches, e.g. step detection using the smartphone camera monitoring the feet while walking, exist, but are only usable with restrictions.

The approach **zero crossing**, e.g. uses the total acceleration $acc_{total} = \sqrt{acc_x^2 + acc_y^2 + acc_z^2} - g$ of the smartphone data as a time series of measurements. Assuming that only the gait pattern is influencing the acceleration signal, the total acceleration is undergoing a zero crossing at every step. The same result can equally be achieved, when using basic peak detection functions, where e.g. the main peak in a certain time window of a signal is detected. This approach is computationally effective, and widely used for test purposes, but not stable against other influences onto the measured signals, e.g. moving or shaking the smartphone in a certain manner. However, a surprisingly large number of studies claim to use this approach for testing and implementing step detectors.

Using a **frequency-based** approach, the signal, e.g. the total acceleration, is analyzed using it's main frequencies. Using a Fourier transformation, a signal in the time domain can be transformed into the frequency domain, where the main frequencies can be detected. Assuming that the gait pattern is the main influence on the measured signal, the main frequency or main component is assumed to be directly correlated to the gait. These approaches are, if the assumption that the gait pattern is the main influence on the signal is true, stable, but it is not strictly possible to design them as capable for real-time applications. E.g. Kang et al. (2018) shows the approach of step detection of smartphones using a frequency-based approach. The step counter implemented within *Android* is assumed to be mainly based on a frequency detection approach.

The used approach within the applications presented, is called **step segmentation** or **signal segmentation**. Through segmentation of either the accelerations or the angular rates, according to steps, each segment can be evaluated. If all segments are occurring in the

right order, a step is detected, see figure 4.10 for a schematic representation. This approach is stable against other influences on the measured signal, as well as real-time capable.

The process of a step detection implementation based on signal segmentation is shown in figure 4.11 as flow chart. The implementation is designed to have a minimal lag time in order to be usable in real-time for navigation applications. The signal segmentation is based on accelerations only, and starts with a fitted low pass filter applied to the raw data as well as an exponential moving average, using roughly two strides as averaging time. Using the exponential moving average of the acceleration signal, a part of the signal has to be chosen which is used within the computation. This is done using a simple moving window, with the minimal window size of one step. Based on the time series of data from the moving window operator, a moving variance as well as maximum and minimum parameters are estimated. Remember, that the used signal is already strongly filtered, starting with a low pass and the exponential moving average, yielding to actual minimum and maximum values in the acceleration domain without blunders. Using those estimated variance, maximum and minimum parameters of the time series of filtered accelerations, the segmentation of a step is computed, based on the step pattern shown in figure 4.10. If the step segmentation is true, a step is detected.

Additionally, approaches using **principal component analysis** or **machine learning** might be utilized for step detection. The basic idea of using a principle component analysis for step detection is, to transform the acceleration signal from the time domain and detect its principal components, which then theoretically should be correlated to the induced gait cycle on the signal. The basic idea of machine learning for step detection is, that the step event in, e.g., the accelerations measurements, can be learned using reference data with state of the art machine learning approaches. Park et al. (2017) shows first results for step detection using machine learning, utilizing a support vector machine classifier, and gaining 98% accuracy for the step detection using a hand-held smartphone mounting.

The detection of the step events is considered more important than the step length estimation

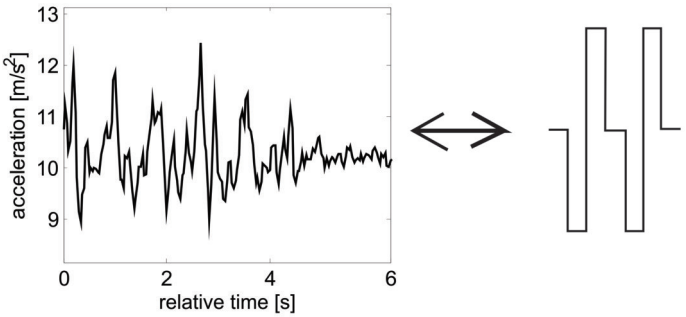

Figure 4.10: Signal segmentation approach for step detection

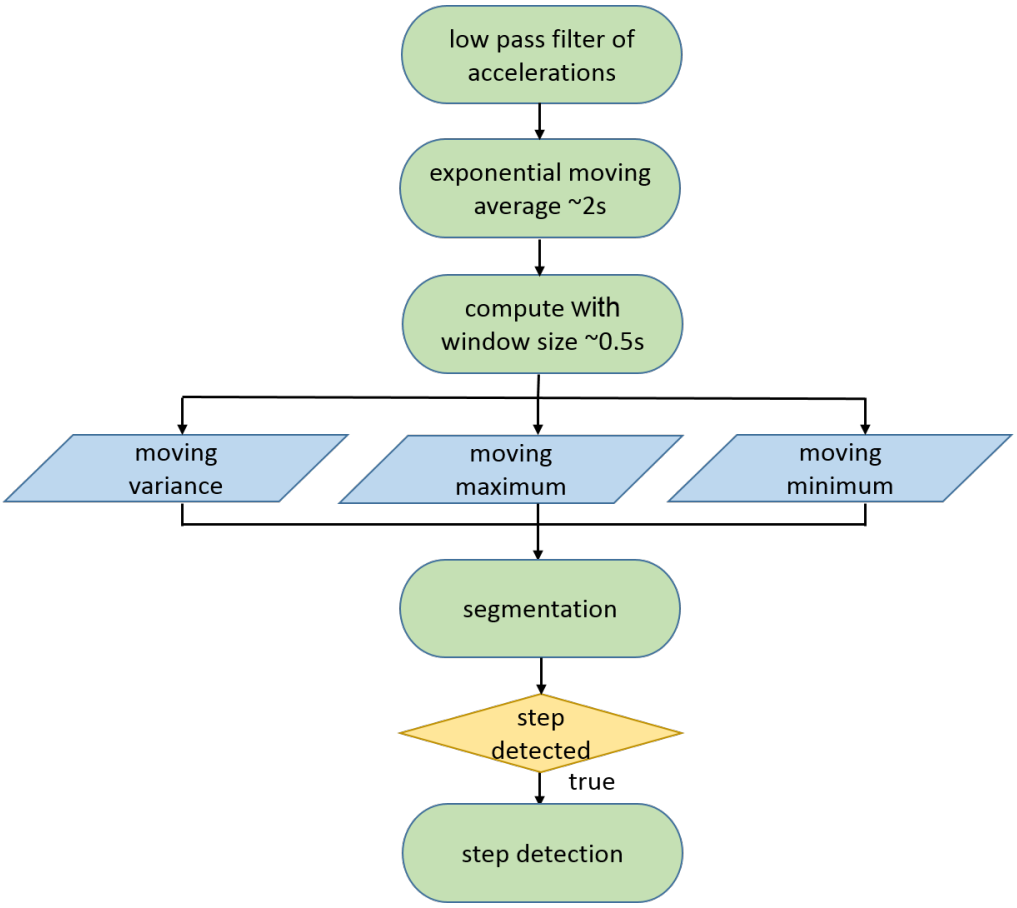

Figure 4.11: Step detection processing

itself, arguably because a missed or unjustly counted step introduces bigger errors than a flawed step length estimation. Therefore, it is of high importance to detect the actual steps and avoid step-like patterns in the sensor signals. These approaches, however, also depend on the device position of the smartphone relative to the pedestrian, e.g. hand-held, carried in a pocket or stored in a shoulder bag. Exemplary results of a step estimation are shown in figure 4.12. The steps are detected after the occurrence of the step itself in the acceleration signal, based on the proposed segmentation of the signal.

4.4.2 Step length estimation

For step length estimation, a multitude of empirical formulas exist, see Chen (2012) for common examples. All of these formulas are usually tailored for calm walking, and do not take specific human motions into account, see section 4.3.1 for an example of motion incorporation into positioning.

The presented approaches estimate the step length SL usually as a combination of model parameters and functions of the step frequency f_{step}, the acceleration amplitude $\sqrt{acc_{max} - acc_{min}}$ as well as the user- or stand-over height h of the pedestrian, see equation (4.6). In equation (4.6), the absolute parameter a and the directly proportional parameters b and c have to be defined, or existent fixed values have to be used. However, such empirical formulas work best when these parameters have been calibrated directly for the pedestrian using the PDR approach.

$$SL = a + b \cdot (f_{step}) + c \cdot (h) \tag{4.6}$$

Following, the most commonly used examples of step length formulas are shown. The most basic, but very robust step length estimation is directly defined as a linear function of the user height:

$$SL = a \cdot h, K = \{a\} \tag{4.7}$$

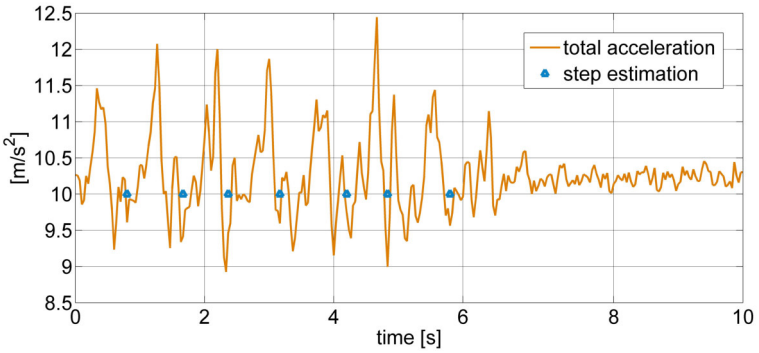

Figure 4.12: Example of step estimation based on signal segmentation of the total acceleration signal

A basic linear relationship, additionally using the step frequency, is given with:

$$SL = a \cdot f_{step} + b, K = \{a, b\} \tag{4.8}$$

An advanced step length estimation, using the step frequency relation, and introduced by Susi et al. (2013), is:

$$SL = h(a \cdot f_{step} + b) + c, K = \{a, b, c\} \tag{4.9}$$

An advanced step length relation, estimated using multiple data sets of test subjects and using the step frequency as well the user height was first introduced by Chen (2012) with:

$$SL = (0.7 + a(h - 1.75) + b(f_{step} - 1.79)h/1.75)c, K = \{a, b, c\} \tag{4.10}$$

Finally, the basic non-linear approach for step length estimation, based on the deflection of the acceleration introduced by the step event, first presented by Weinberg (2002), is:

$$SL = a\sqrt[4]{acc_{max} - acc_{min}}, K = \{a\} \tag{4.11}$$

Generally speaking, all investigated step length estimations, which use the step frequency or empirically estimated parameters, work sufficiently within the limits of smartphone based PDR, if calibrated correctly, except for the models using the acceleration amplitude. The stage of calibrating the step length parameters to the user is generally of higher importance than the chosen step length formula itself.

4.4.3 Step orientation estimation

Finally, the step direction of the detected steps is of importance for the dead reckoning approach. The estimation of an absolute orientation of the smartphone is theoretically possible, using e.g. the magnetometers, however the estimation of the orientation angle between the walking direction of the pedestrian and the smartphone orientation is possible but introduces errors in the range of 15 to 30 °, see e.g. the study of Combettes and Renaudin (2015).

Therefore, it is advised, depending on the application, to use the relative heading information of the pedestrian only. The three-axis gyroscope measurements can easily be rotated into the local-frame of the pedestrian, where the angular rate in the z-axis of the pedestrian is assumed to be directly the relative change in heading of the pedestrian, using equation (4.12).

$$\dot{R}_b^l \approx R_b^l \cdot \Omega_{ib}^b \ , \tag{4.12}$$

where equation (4.12) is true if the influence of the earth rotation is neglected. This is eligible for smartphone-grade gyroscopes, because their sensor characteristics introduce bias errors noticeably higher than the earth rotation itself. This relative heading may be used if the device position, relative to the pedestrian is sufficiently stable. During device transition phases, this relative heading information may not be used.

The process of a heading estimation implementation, using gyroscopes as well as magnetometers, is shown in figure 4.13 as a flow chart. If both heading solutions, an absolute heading estimation, computed out of the magnetic measurements and a relative heading estimation, computed out of the rotated angular rates according to equation (4.12), are performed the following explanation is true. If only the angular rates are used, simply the rotated $yaw_{gyroscope}$ estimation gets filtered in the attitude domain. If both, the angular rates as well as magnetometer sensors are used, the difference between the yaw rate and the magnetic rate is observed. If this difference exceeds a certain threshold, the magnetometer is assumed to be underlying a magnetic deviation. This assumption is true for short term observations, if the angular rate biases are estimated correctly. If the magnetometers are assumed to be deviated, by e.g. the current building infrastructure, the standard deviation of the magnetic parameter for the filtering stages is set high compared to the yaw rate. Contrary, if the magnetic sensor is assumed to be true, the standard deviation of the magnetic heading estimation is set low compared to the yaw rate. Finally, a filtering approach in the heading domain, using the magnetic heading estimation as well as the yaw rate computed out of the leveled angular rates is conducted. The heading estimation, similarly to the step length estimation, is computed and sampled once a step is detected, for the detected step.

The resulting accuracy of the step direction estimation is dependent on the performance of the on-route gyroscope bias estimation and the device position itself. Certain device positions deliver more stable heading estimations, where, e.g. smartphones carried in a shoulder bag, are additionally subject to a change in the orientation angle between device and pedestrian. Also a representative wobble of each step may be observed, which can theoretically be time filtered. A typical example of the accumulated, relative heading estimation of a pedestrian walking inside a building on hallways and holding a smartphone in front of his body is given in figure 4.14. The bias estimation is visually working well, where no obvious drift in the accumulated solution is shown. Additionally, the typical wobbling of each step is visible and the changes along the hallway of 180 ° and 90 ° multiple times are visible.

4.4.4 PDR computation

For computing and visualizing PDR solutions for the presented applications, a basic tool is developed within this work, see figure 4.15.

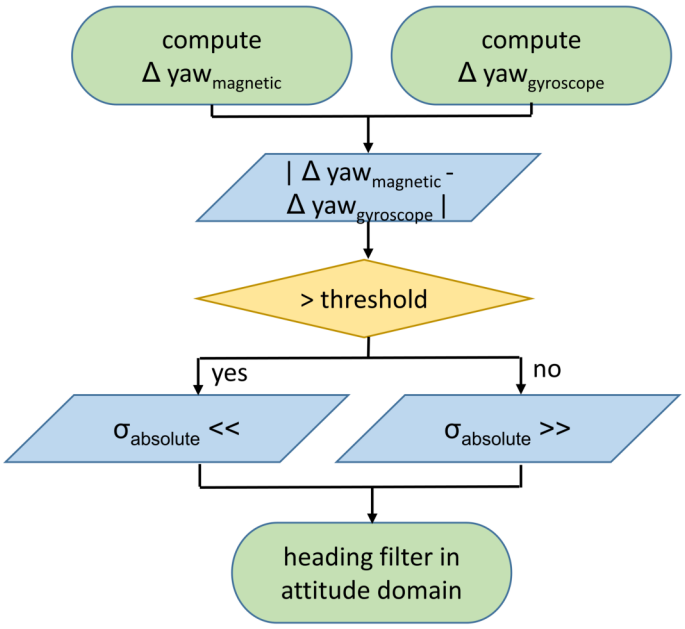

Figure 4.13: Heading estimation processing

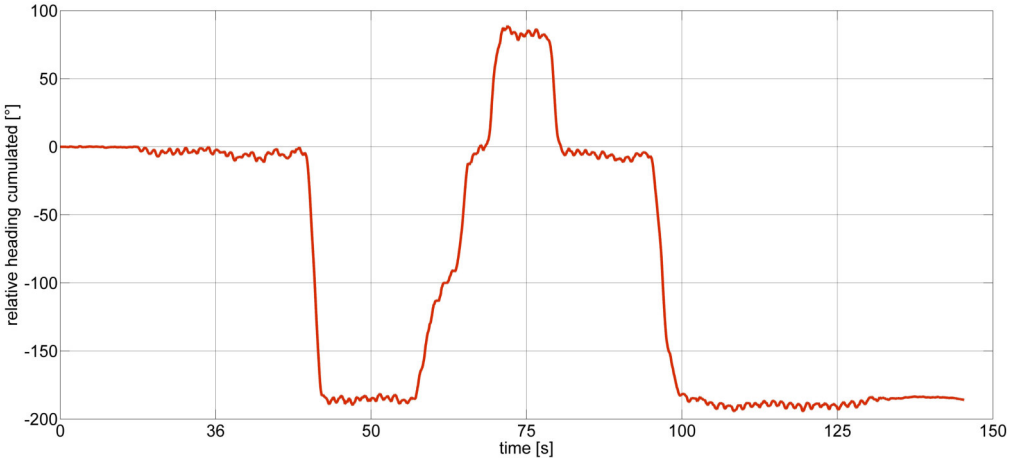

Figure 4.14: Relative step direction estimation

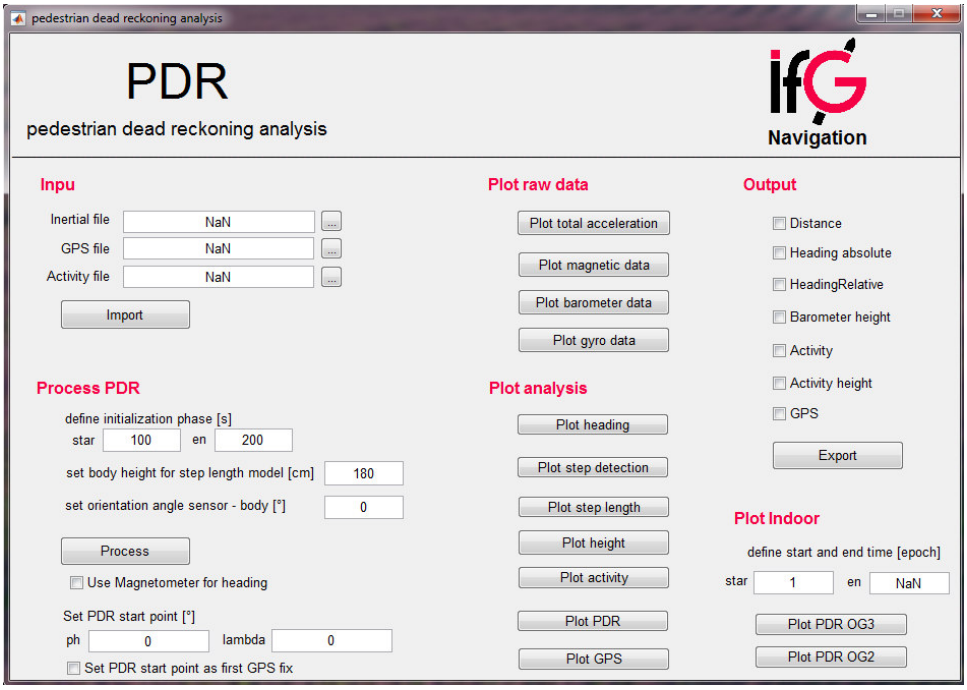

Figure 4.15: PDR tool

As input, it is possible to input files containing the inertial data, GNSS data as well as the estimated locomotion data, which is named as activity data. For processing the basic PDR phases, only the inertial data is necessarily needed. If the GNSS data is available, an absolute time stamp can be assigned to the inertial data, and the GNSS information can be used for plotting, e.g. using the first GNSS fix as starting point for the stand-alone PDR solution. If the activity file is present, it will be used within the PDR computation.

For processing the PDR, the initialization phase can be defined externally, and the user height as well as a possible orientation angle between the sensor and body frame are used. Additionally, the information if the magnetic sensors should be taken into account, can be set. Based on the magnetic sensors, the decision whether to compute an absolute or relative heading only can be made. All other details for processing the PDR solution, which are presented within chapters 5 and 6, are tackled and adjusted within the scripts itself. This includes the details of the step detection, the details of the step length estimation and whether calibrated parameters are used, as well as the step direction estimation, and the implemented form of bias estimation for the specific application.

After processing, the results can be plotted and visualized. The raw inertial data as well as the derived PDR solutions and modes of motions can be plotted as time series as well as spatial data. Additionally, specifically for the test bed Steyrergasse 30, the coordinates of the PDR solution can be shown. As output, files including the solution of the PDR process can be exported for future use within, e.g., a particle filter.

4.4.5 Incorporation of motion recognition into pedestrian dead reckoning

The incorporation of locomotion or multimodality recognition into positioning approaches is not a novel idea, although it is not often applied within actual positioning approaches. More often, the information of the current mode of motion is used as input within the navigation process, but not into positioning algorithms.

Within absolute positioning, the current locomotion can, for example, be used within the particle filter. When the current mode of motion is defined as walking on a level or walking upstairs or downstairs, this information can be used within the particle filter phases: either, within the importance weight computation, where more likely spatial areas based on the locomotion are weighted higher, or within the propagation and resampling phases, where the according spatial areas are taken into account during those phases.

Within PDR computation, the current locomotion can be used in the step detection as well as in the step length estimation phases. In a step detection process, the used parameters can be adjusted based on the current locomotion. Obviously, the main step frequency is different between walking on stairs, walking normal or walking fast, respectively running. Therefore, the thresholds within a step segmentation, or the main frequency within a frequency-based approach or principle component analysis, can be adjusted, based on the identified locomotion.

Similarly, the step length estimation is directly correlated to the current locomotion. The most direct example is walking upstairs or downstairs compared to walking normal. As shown in chapter 2, a mean step length is in the range of 60 cm. Contrary, a standardized length of a stair is 30 cm, leading to a mean step length in that range. Consequently, the step length of most people is only half the length when walking on stairs, which can lead to accumulated errors and filtering failures within PDR. A more indirect approach of using locomotion recognition within PDR is the calibration of step length parameters per locomotion and the application of these parameters.

4.5 Conclusion of positioning approaches usable with smartphones

Closing chapter 4, a short conclusion on the shown positioning approaches is given. Presented are the major positioning approaches known, which are executable with smartphone sensors today.

Starting with GNSS, which is used as the main positioning service today, GNSS is a very robust positioning system outdoors, if position accuracies within a couple of m are sufficient

for the used application. Additionally, even with cold starts, the initial computation of a position fix using GNSS, is possibly achieved within a couple of seconds with state of the art algorithms. However, if GNSS is not available to the user, e.g. indoors, there is a lack of standardized positioning services for smartphones. Multiple approaches exist and are listed, although no service is accepted thoroughly as of now. The main crux is, that most of the discussed approaches rely on some sort of additional infrastructure, which has to be installed, serviced, and payed for. Most prominently, approaches based on signals of opportunity, i.e. signals which are already present and setup for different purposes, are used. The main example are radio signals, and therefore WLAN or BLE for smartphones are used frequently. However, since WLAN or BLE beacons are setup for communication, and not for estimating ranges, the usual indicator, i.e. the relation between received signal strength of the radio beacon and the distance to the beacon, is not especially accurate. The most common method known therefore still is location fingerprinting, making use of the radio patterns within buildings.

Then, although not directly a positioning technique, the possibility of detecting and using locomotions with smartphone data is presented. Especially the recognition of locomotions, as well as the transition of the detected locomotions in the time domain, is discussed. Correctly detected motions are useful for multiple applications, for example the guidance process. More specific, correctly detected locomotions will be shown to be useful for the positioning phase within navigation, for example when used within PDR.

Finally, the focus is on the major, self-contained algorithm for smartphone positioning, which is PDR. Current PDR models and possible techniques for step estimation, step length estimation as well as step orientation estimation are given and discussed exactly as they are applied within the applications presented in chapters 6 and 7.

5 Calibration of sensors and algorithms usable for pedestrian dead reckoning

This chapter describes the calibration approaches for sensors and algorithms. The focus is herewith solely on sensors and algorithms, which are used within the shown Pedestrian Dead Reckoning (PDR) phases. Starting with a smartphone sensor analysis, the focus is on the inertial sensors of the smartphone. Following, the calibration of the inertial and magnetic sensors of a smartphone is shown and discussed. Regarding algorithms, the approach for calibrating the motion model as well as calibrating the step length parameter estimation is explained. The focus herewith lies on the developed approach for an automated motion model and step length parameter calibration. Closing, a conclusion and brief discussion on time variability regarding the presented calibration procedures, as well as techniques for how and when to perform calibration is given.

5.1 Smartphone sensor calibration

The discussion of smartphone sensor calibration is divided into a general analysis of the stochastical behavior of the inertial sensors, using an Allan variance analysis. Following, an analysis using different, independent time series of data from different, consecutive days is used for a specific sensor error analysis.

The Allan variance σ_τ analysis is conducted using tests with two independent, static measurements over a three and a five hour period. Utilizing the Allan variance for analyzing inertial sensors is a common and advised method. It is derived grouping the measurements into bins in the time domain, averaging the measurements in these time bins and computing a variance for each time bin, based on the averaging time τ with:

$$\sigma_\tau^2 = \frac{1}{2 \cdot (n-1)} \cdot \sum_{i=1}^{n-1} (y(\tau)_{i+1} - y(\tau)_i)^2 \tag{5.1}$$

A variance series can then be computed out of these bins, using at least nine samples. Therewith, the Allan variance can describe the theoretical stability and bias behavior of the observed sensors, and shows the sensor stability in one run, with a controlled ambient temperature and no other external signal influences on the sensors. However, this can only be

conducted with continuous measurements of one static time series, and has therefore obvious drawbacks for observing dynamic sensor errors. Therefore, long term effects like climatic changes and turn-on effects of the smartphone itself cannot be taken under account using an Allan variance only.

The Allan variance is analyzed using the mentioned time series with a 100 Hz sampling rate. The data was recorded using a Samsung Galaxy S4 smarphone. For the three hour period, 16 bins are used, where the largest bin has 11 samples. For the five hour period, 17 bins are used. In figures 5.1 and 5.2 the inertial data of the measured time series is given. Shown is the total acceleration as well as the total angular rate, due to no effect appearing on a single signal axis being present for this static test setup. The tests were conducted in an office environment after office hours, where no external acceleration or angular rate influences are present on the sensors.

No significant difference between the independent time series can be observed, therefore the analysis is assumed to be correct. Analyzing the inertial data, the fundamental sensor noise as well as the sensor warm up effect is visible, especially present with the accelerometers. This results from the fact, that the smartphone was turned on directly before the test measurements, and following the data logging application uses the operating system and induces an increase in the smartphone temperature.

The warm up bias, neglecting an absolute bias in the accelerations, can be in the range of up to 0.1 m/s^2. The visible noise is in the range of up to $3 \cdot 10^{-2°}/s$ in the angular rate domain, and up to 0.1 m/s^2 in the acceleration domain.

The results for the computed Allan variance are shown in the figures 5.3 and 5.4. The Allan variance can be seen as a starting point for the minimal observable bias error and its correlation time of a sensor. However, in reality, multiple other sensor errors contribute on top of the minimal bias error. Both independent time series yield similar results in the Allan variance. In the angular rate domain, a stability of $10^{-4°}/s$ at a correlation time of 30 s can be observed as the low point of the Allan variance. In the acceleration domain, a minimal bias in the range of 0.010 m/s^2 to 0.007 m/s^2, with a mean of 0.008 m/s^2 and a correlation time of 10 s can be observed as the low point of the Allan variance. These sensor specifications are the maximum controllable bias errors in the corresponding correlation time. However, more importantly, the long term stability of the analyzed minimal biases can be observed. Here it is visible, that the gyroscopes are not stable in the long term regarding their bias behavior. However, the accelerometers might be stable up to some 0.01 m/s^2. Dependent on the used application, this might be enough to calibrate the accelerometers.

Contrary to the static, long term analysis, an analysis on multiple, consecutive days was conducted. Such an analysis shows, e.g., turn-on effects of the observed sensors. The presented tests on multiple, consecutive days, were combined with a sensor calibration approach. On each of the days, a test and calibration series was performed using the same smartphone for

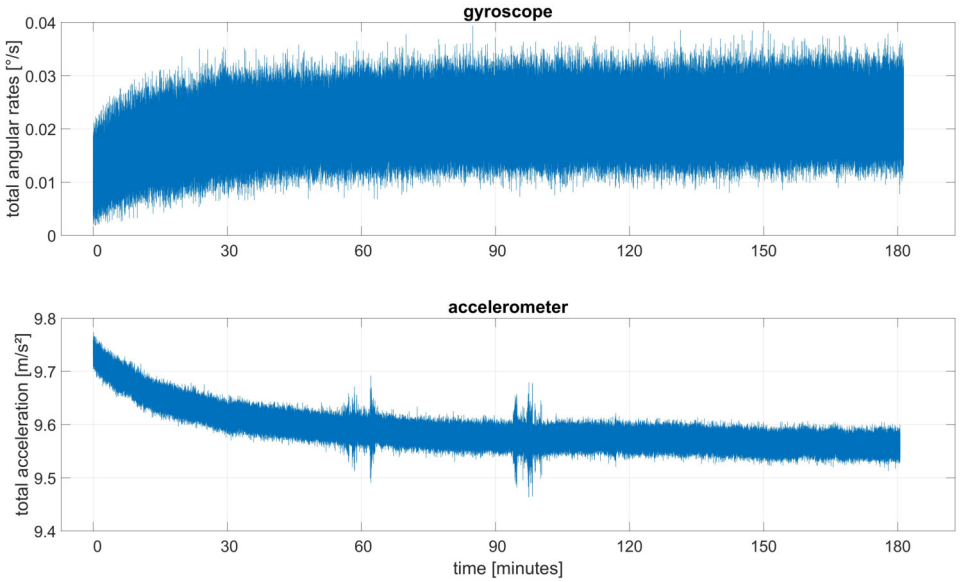

Figure 5.1: Inertial data of three hour static time series

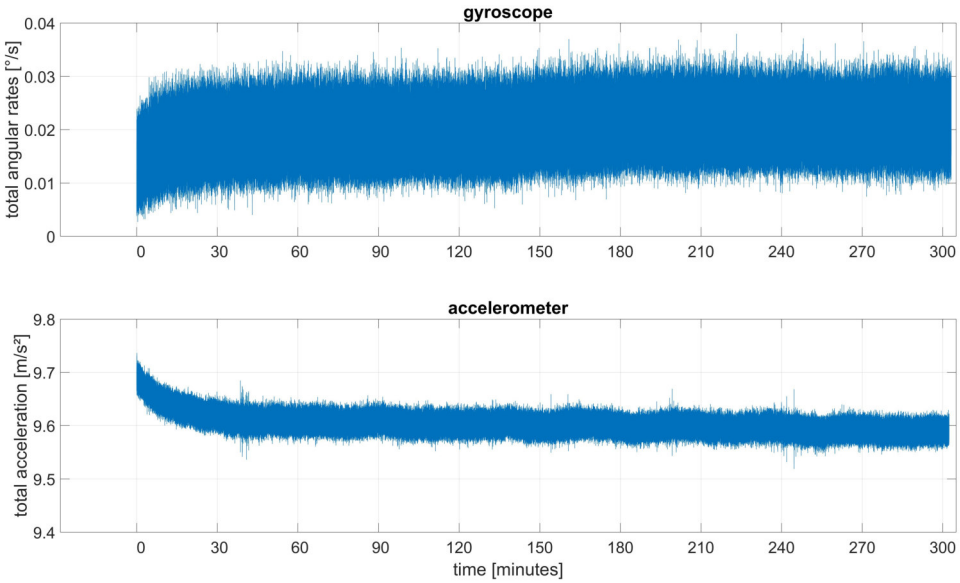

Figure 5.2: Inertial data of five hour static time series

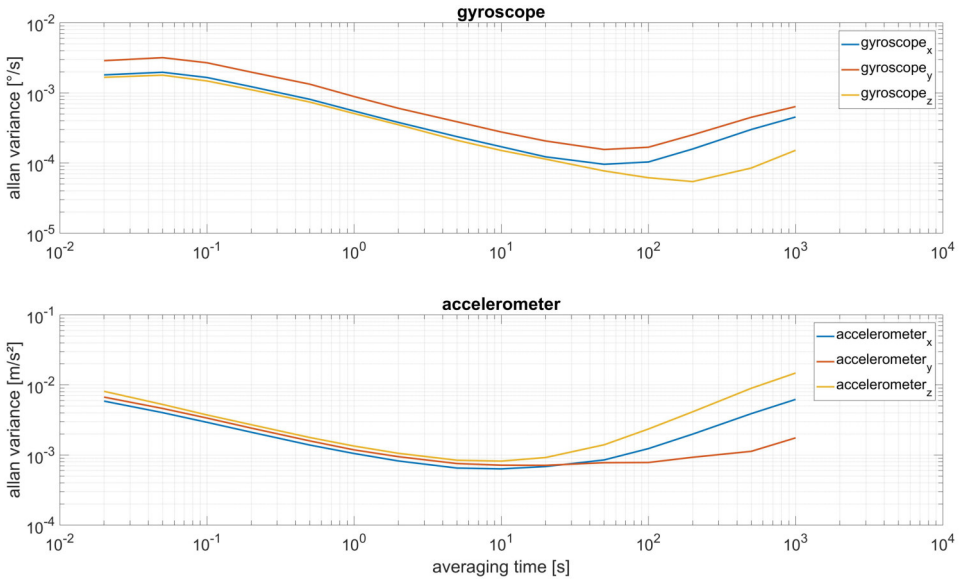

Figure 5.3: Allan variance of three hour static time series

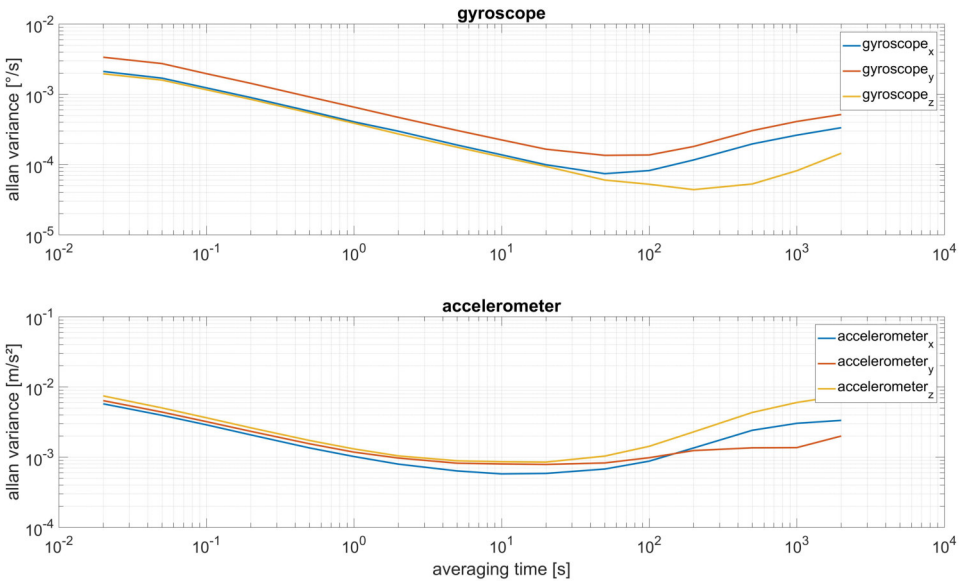

Figure 5.4: Allan variance of five hour static time series

data logging. This enables practical investigation of bias stability from day to day, including turn-on effects of the smartphone sensors itself.

The general necessity of calibrating inertial sensors is tackled in Moder et al. (2017) and described with: "Calibration of sensor errors is most easily accomplished using a reliable ground truth, which is significantly more accurate than the sensor. For acceleration and angular rate sensors, a ground truth might be accomplished using turning platforms or rate tables, see e.g. Beravs et al. (2012) for low-cost accelerometer calibration using a robotic arm. However, such equipment is costly and the corresponding measurement procedures are temporally extensive. When stationary, inertial sensors located on the earth measure the earth's gravity and rotation and, if enough observations are present, this is sufficient to compute inertial sensor errors."

This concept of independent measurements in multiple, static phases, compared with external, known reference measurements, is utilized for the time series of ten consecutive days. When the acceleration and angular rate sensors measure the earth's gravity, and respectively the earth rotation, this can be used within a parameter estimation to estimate a sensor model as complex as needed. However, since the measured quantities are, compared to the sensor errors, low, only low-complexity sensor error models are estimated. This means, for the accelerometers bias and scale factors as well as for the gyroscope the bias might be defined within the sensor model. These sensor biases then can be estimated using basic least squares estimation, assuming independent measurements. In order to have a sufficiently working condition of the least squares adjustment, at least as many measurements as unknown parameters have to be made. Therefore, an object with as many independent positions as possible was chosen for this test setup. A rhombicuboctahedron, a 26-sided object, was used, inspired by the research of Nilsson et al. (2014b). The 26-sided object was constructed using wood, and is referred to as wooden structure, see figure 5.5. Wood is used, because of it's invariability to magnetic influences. Using only wood and plastic screws, a magnetic invariable platform is created, which can be used to test the inertial sensors equally as well as the magnetic sensors. The wooden structure is constructed out of two parts, with the possibility to place the smartphone in the center of it, see figure 5.6. The place for being able to insert the smartphone is big enough to accommodate most smartphones, and the part closing the front insertion can be adjusted.

The wooden structure is designed, so that all flat surfaces are equally distributed, and that all 26 possible measurement phases can be conducted using it. If the calibration is conducted, the sensor respectively the smartphone is inserted into the wooden structure. Then it is moved to each of the 26 sides, staying static for a certain time on every of these 26 sides. For every static period, one measurement value as mean out of the accelerations or angular rates can be estimated, using a stationary detection and a basic blunder detection in the sensor domain.

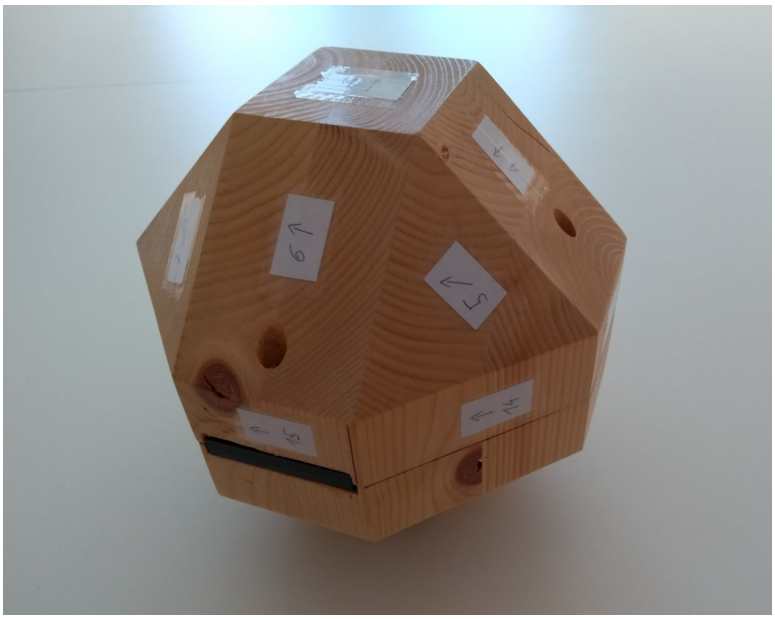

Figure 5.5: Wooden structure for smartphone sensor calibration purposes, closed

Figure 5.6: Wooden structure, open with visible smartphone insertion

The significance of error types for smartphones is explained in Moder et al. (2017) with: "Generally, the bias errors are by far the biggest error source for all presented sensors, followed by the scale factors. With the sensor quality of smartphones and the application background of PDR, the bias and scale factor of the accelerometers and the bias of gyroscopes and magnetometers are of practical interest. Additionally, the gyroscope and magnetometer scale factors can not be estimated directly with the proposed calibration method due to numerical reasons. If the bias values are estimated and the measurements corrected for them, then it is theoretically possible to estimate the main scales in a second estimation phase. However, with the proposed calibration method, it is not feasible to calibrate gyroscope and magnetometer scale factors in a statistically significant manner for current smartphone sensors."

Following, the sensor calibration for each of the accelerometers and of the gyroscopes, as well as for the magnetometer is presented in the sections 5.1.1, 5.1.2 and 5.1.3. The calibration delivers the sensor errors, modeled for the specific sensors. Additionally, the tests were performed on ten consecutive days, in order to show possible turn-on effects and day-to-day changes in the sensor stochastics. However, other possible influences, e.g. temperature variations, were not modeled within this analysis. These tests and results were already presented in Moder et al. (2017), and therefore the specific results are according to the cited publication.

Corresponding to the observation model presented in chapter 2 with equations 2.33 to 2.37, the calibration parameters are defined as part of x, where the observations are defined as l. A typical parameter estimation has more observations than parameters, which is definitely true for the accelerometers, gyroscopes as well as the magnetometers. For all three sensors, the observations are the mean of the sensor measurements per position of the wooden structur, and used as input for l. The parameters x in the case of the accelerations are the main scales as well as the bias errors, in case of the gyroscopes the bias errors, and in case of the magnetometers the radius as well as the offset parameters.

5.1.1 Accelerometer

The measured accelerations f^b in the smartphone body frame are given by

$$f^b = \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix} . \tag{5.2}$$

The acceleration vector g_l in the local level frame when the sensor is stationary can be written as

$$g_l = \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} . \tag{5.3}$$

g is computed as normal gravity γ, which can be estimated as $\gamma = f(\varphi, h)$ and even further approximated as $\gamma_\varphi \cdot f(h)$. If the smartphone is assumed to be stationary, satisfying $f^b|_{\dot{\varphi}=\dot{\lambda}=0}$, the relation between the inertial measurements in the body frame and the accelerations in the local level frame is given with

$$-f^b = R_l^b g^l. \tag{5.4}$$

As presented in chapter 2, R_l^b is the rotation matrix from the local level to the body frame, and contains the rotations roll r, pitch p and yaw y around three axes. At stationarity, equation 5.4 leads to

$$f_x^2 + f_y^2 + f_z^2 = g^2 \ . \tag{5.5}$$

In the acceleration domain, equation 5.5 is used as an observation for estimating the parameters of the sensor model. The observation model for the accelerations, can be noted as

$$\tilde{f} = S_f \cdot f + b_f + n_f. \tag{5.6}$$

Note that equation 5.6 is a simple observation model, specifically used for low-cost smartphone accelerations sensors.

The acceleration bias b_f and scale factors S_f can be calibrated using g as measurement quantity. $\tilde{f}$ corresponds to the actual measured accelerations, f to the true accelerations and n_f to the Gaussian distributed measurement noise. The acceleration bias and scale factors can be estimated with a sufficiently conditioned parameter estimation.

The results of the sensor model calibration for accelerations at ten consecutive days are given in figure 5.7 for the acceleration bias and in figure 5.8 for the acceleration scale factors.

The analysis of the accelerometer sensor errors is given in Moder et al. (2017) with: "Acceleration biases generally can be up to a couple of 0.1 m/s^2, the highest value obtained is in the range of 0.34 m/s^2. Regarding bias variation, the acceleration biases vary in the range of their standard deviation over the test cycle of ten days, which is 0.008 m/s^2 for all estimated accelerometer biases. The mean of the biases in all three axes are: $b_x = -0.17$ m/s^2, $b_y = 0.22$ m/s^2, $b_z = 0.34$ m/s^2. All resulting accelerometer biases are significant with respect to their standard deviation as well as stable over time (over the course of these ten days the test cycle took place). Therefore, calibrating the accelerometer biases seems useful, however, the actual benefit will depend on the needed measurement accuracy for the particular application."

In figure 5.8, the acceleration scale factor calibration is given. The main scales are stable over the observed samples, according to their calibration deviation. The calibration results

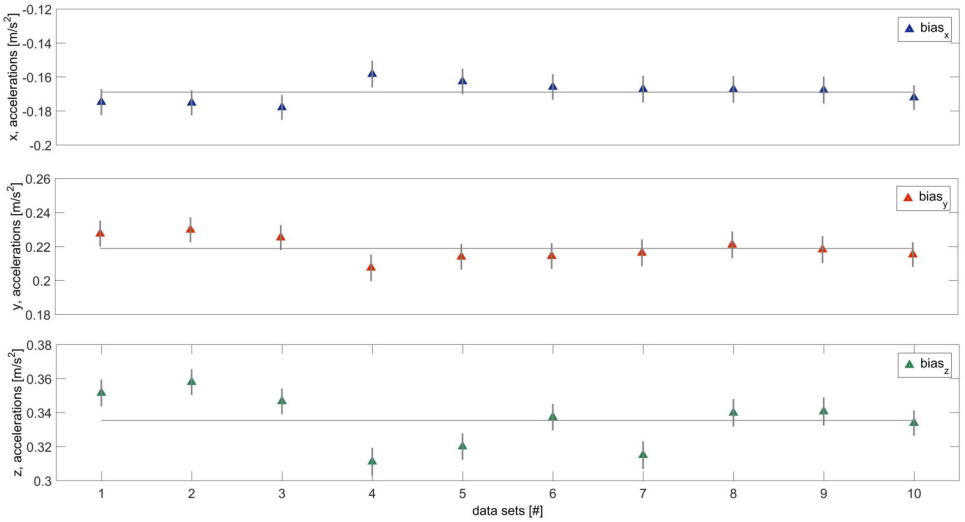

Figure 5.7: Accelerometer biases calibration with mean and standard deviation

are given in Moder et al. (2017) as: "The mean standard deviation of all estimated scales are in the range of 0.001. The actual estimated scales, s_x, s_y and s_z being the main diagonal elements of S_f, are $s_x = 0.999$, $s_y = 0.995$, $s_z = 0.975$." Therefore, a calibration seems useful for the main scales.

5.1.2 Gyroscopes

The measured angular rates ω_{ib}^b in the smartphone body frame are given by

$$\omega_{ib}^b = \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix}. \tag{5.7}$$

The angular rates vector ω_{il}^l in the local level frame can be written as

$$\omega_{il}^l = \begin{pmatrix} (\dot{\lambda} + \omega_E) cos\varphi \\ -\dot{\varphi} \\ -(\dot{\lambda} + \omega_E) sin\varphi \end{pmatrix}. \tag{5.8}$$

At stationarity of the sensor, satisfying $\omega_{il}^l|_{\dot{\varphi}=\dot{\lambda}=0}$, ω_{il}^l can be reduced to

$$\omega_{il}^l = \begin{pmatrix} \omega_E \ cos\varphi \\ 0 \\ -\omega_E \ sin\varphi \end{pmatrix}. \tag{5.9}$$

The relation between the inertial measurements in the body frame and the angular rates in the local level frame can be written as

$$\omega_{ib}^b = R_l^b \omega_{il}^l. \tag{5.10}$$

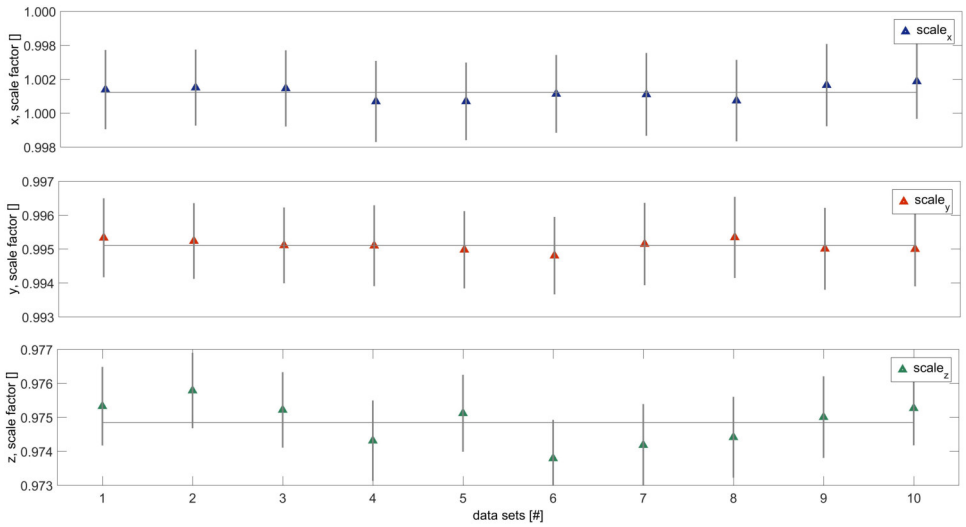

Figure 5.8: Accelerometer scale factors calibration with mean and standard deviation

Utilizing equation 5.10, the earth rotation ω_E can be given with

$$(\omega_E \; cos\varphi)^2 + (-\omega_E \; sin\varphi)^2 = \omega_E^2 = \omega_x^2 + \omega_y^2 + \omega_z^2. \tag{5.11}$$

In the angular rate domain, equation 5.11 is used as observation for the parameter estimation of the sensor model. As presented in chapter 2, the error model for the angular rates is given as

$$\tilde{\omega} = S_g \cdot \omega + b_g + G \cdot f + n_g \; , \tag{5.12}$$

$\tilde{\omega}$ corresponds to the actual measured angular rates, ω to the true angular rates and n_g to the Gaussian distributed measurement noise in the presented error model. The angular rate bias b_g can be calibrated using ω_E as measurement quantity. The measured quantity, ω_E, is in the range of 0.003 $^\circ/s$ for the presented tests, which is small compared to the expected calibration parameters. Therefore, only the gyroscope bias was calibrated in order to still have a well conditioned least squares adjustment approach. The angular rate error model was reduced to

$$\tilde{\omega} = \omega + b_g + n_g \; . \tag{5.13}$$

The angular rate main scales S_g as well as the gravity sensitivity G are not further taken into account.

It is theoretically possible to estimate the bias error, then iterate the parameter adjustment and estimate e.g. the angular rate scale factors. However, practical investigations show no useful results in dealing with the angular rate scale factors with smartphone gyroscopes.

In figure 5.9, the presented gyroscope bias estimation is not stable from day to day. Additionally the estimated mean of the bias error per sensor axis is not within the range of the estimated standard deviation from the bias error estimation.

The analysis of the gyroscope sensor errors is given in Moder et al. (2017) with: "The mean standard deviation of all estimated gyroscope biases is 0.02 $°/s$. In equation 5.11, ω_E is the earth rotation with 0.004167 $°/s$. More precisely, the influence of ω_E on a single axis at the latitude of Graz, the location where the presented tests were conducted, is at most 0.0028 $°/s$. The mean standard deviation of all estimated biases with respect to the range of the earth rotation delivers a difference with the factor of 7 of measured deviation to the estimated parameter. This clearly indicates that the consideration of the earth's rotation may be neglected and that the parts of the systematic error from equation 5.11, the components b_{g_x}, b_{g_y} and b_{g_z}, can be set to $b_{g_x} = b_{g_y} = b_{g_z} = 0$ during stationary phases. The variation of the gyroscope biases with respect to their mean bias is in the range of 0.05 $°/s$ from sensor axis to sensor axis and day to day. This, with respect to the mean standard deviation of all estimated biases with 0.02 $°/s$, clearly indicates that a calibration of gyroscope biases for longer than one measurement cycle, i.e. longer than a day, is not sensible. Therefore, gyroscope biases, with smartphone-grade inertial sensors, have to be estimated every measurement cycle during stationary phases."

5.1.3 Magnetometers

The magnetic sensor errors cannot be modeled as easily, because no direct measurement quantity or reference is available. However, the magnetometers are implemented in order to measure the magnetic field of the earth, which can assumed to be constant over the short term. The assumption, that only the magnetic field of the earth is observed is true, i.e. no magnetic deviations on the smartphone are present, has to be taken into account when calibrating the magnetometers.

If no external magnetic deviation is affecting the sensor, the magnetic vector has the same length in every orientation, measured at one spatial point. Therefore, if measuring the magnetic field strength uniformly in every magnetometer orientation at one point, the magnetic vector yields a perfect sphere in the body frame. Multiple approaches to estimate the magnetic error model exist, however, e.g. Kok and Schoen (2016) show that the calibration of a unit sphere to the magnetic vectors is sufficient when using magnetometers from smartphones.

The basic sphere equation, given with 5.14, can therefore be utilized to model the magnetometers. The sphere is modeled with its offsets x_0, y_0 and z_0 in every axis, with its main scales S_{11}, S_{22} and S_{33} in every axis as well as its radius r.

$$S_{11}^2 \cdot (x - x_0)^2 + S_{22}^2 \cdot (y - y_0)^2 + S_{33}^2 \cdot (z - z_0)^2 - r^2 = 0 \tag{5.14}$$

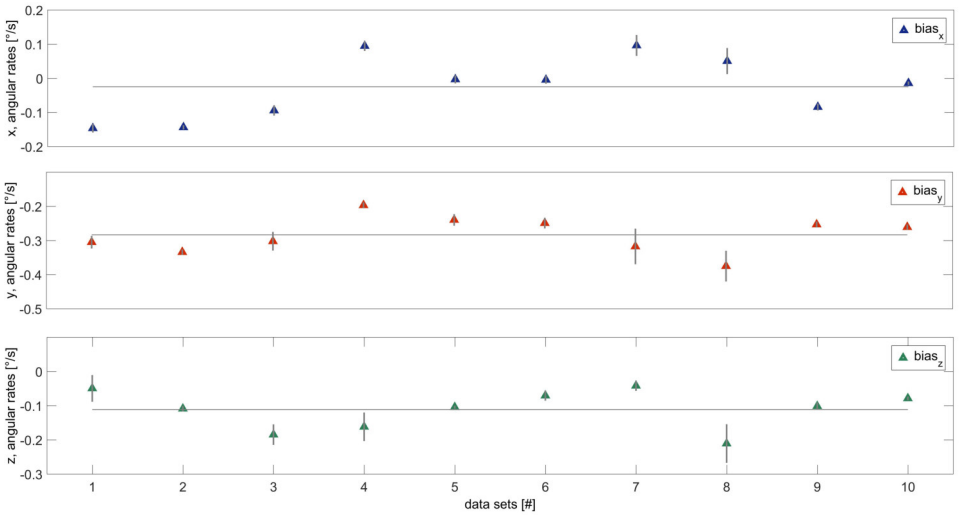

Figure 5.9: Gyroscope biases calibration with mean and standard deviation

As presented in chapter 3, the magnetic field is measured in $[\mu T]$. Due to the present hard iron calibration, performed by *Android* on the smartphone sensor, the actual units are arbitrary units $[aU]$. Since no actual reference observation is modeled, but a geometric constraint is adjusted to the measurements, the fact that arbitrary units are used for the magnetometer calibration has no further influence. Figure 5.10 shows the magnetic vector, measured out of the 26 static orientation states, in the body frame. It is visible, that the estimation of a sphere is converging the magnetic vector measurements.

In figure 5.11, the calibration of the bias and radius parameters according to equation 5.14 are shown. Since practical investigations show no major influence of the scale factors, due to a more practical conditioning of the parameter adjustment, the main scales are assumed to be $S_{11} = S_{22} = S_{33} = 1$. This leads to

$$(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 - r^2 = 0, \tag{5.15}$$

and only the offsets x_0, y_0, z_0 and the radius r are estimated out of the calibration process. As seen in figure 5.11, neither the bias offsets, nor the radius is stable from day to day or within the standard deviation from the parameter adjustment. Consequently, the magnetometers have to be calibrated every day, if a correct sensor accuracy is needed for the used application. If only a course direction estimation is needed, no specific magnetometer calibration may be carried out, especially if no high magnetic deviations are present.

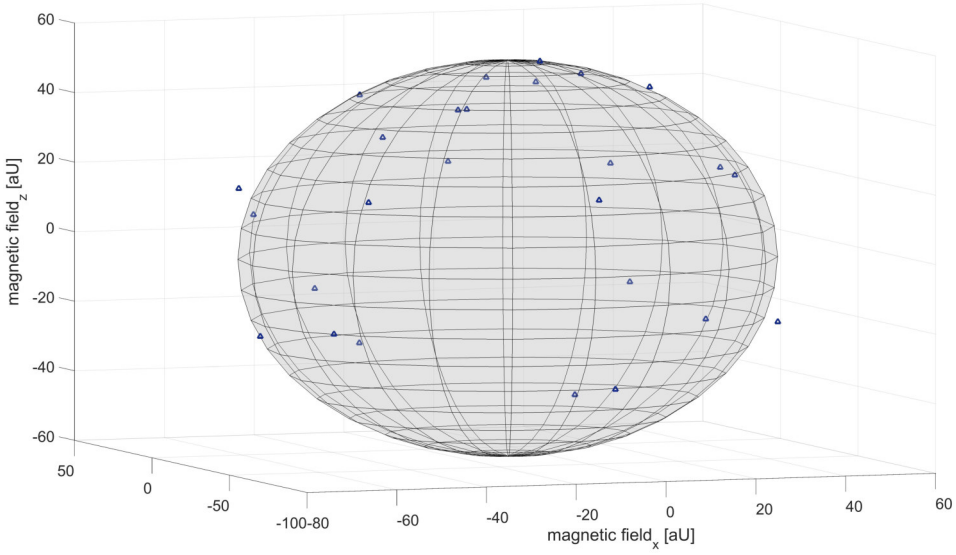

Figure 5.10: Estimation of a sphere to 26 static, magnetic vector measurements

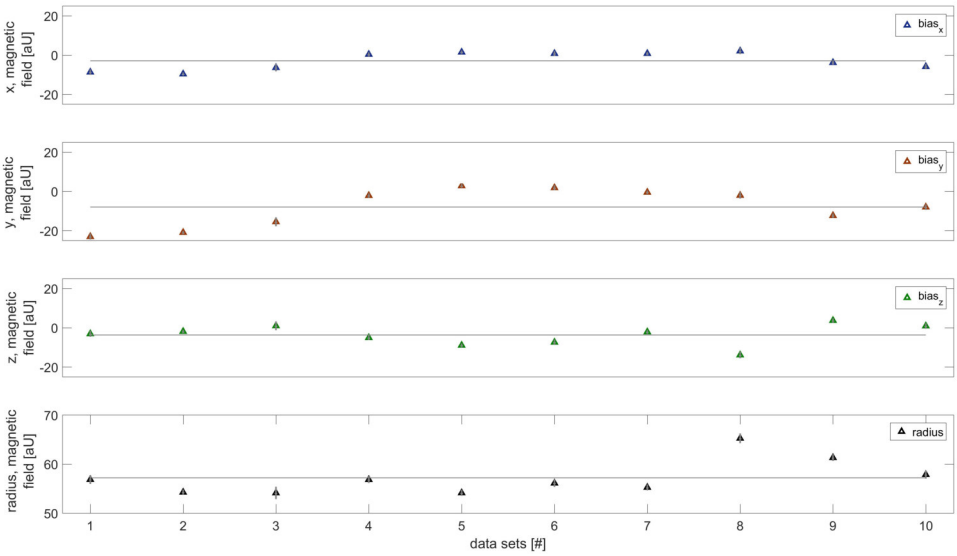

Figure 5.11: Magnetometer biases and radius calibration with mean and standard deviation

5.2 Motion model calibration for locomotion detection

Next to the calibration of the sensors used in PDR, it is equally important to calibrate the presented algorithms used within PDR. Namely, the locomotion recognition as well as the step length estimation.

Focusing on the locomotion recognition in this section, the calibration of the motion model is discussed. As presented in chapter 4, the motion model can effectively be calibrated using recorded reference data. As shown, this calibration based on using manually noted references of which locomotion was acted out, can accurately detect the current locomotion with recall values of a cross validation up to 98 %.

However, the presented approach of manually referencing the training data for computing a motion model is not sustainable for common users. Therefore, an approach of an automated motion model calibration is presented. Gaining the model parameters in an automated fashion, is more efficient for the user than previously published methods. The main approach is, to classify the current locomotion based on independent measurements. Since no complicated, external equipment may be used, the presented *PDR Logger* application is utilized. The *PDR Logger* logs the smartphone sensor data synchronized to a high precise step detection of the foot-mounted module. Because of the high precision available with the foot-mounted solution, it is possible to classify the current locomotions based on these step events.

The classification follows mainly the flow chart presented in figure 5.12. The defined locomotions standing, walking, walking fast and walking upstairs or downstairs are classified based on their corresponding vertical and horizontal step length. The foot-mounted step events are accurate in the range of $< 1\ cm$ for every step event, and especially stable within a detected zero velocity update. Therefore, classifying between standing and all other locomotions is very accurate. To avoid blunders, further a threshold is defined between standing and the other locomotions, ruling out minuscule steps. Then, the classification is made between walking on a level and walking on stairs. Herewith, the threshold is set corresponding to an incline of 25 % for a typical stride length. Following, for walking on a level the distinction between walking normal and walking fast is made based on the step frequency, again, gained accurately out of the foot-mounted step events. And the distinction between walking upstairs and downstairs can easily be achieved, using the vertical step length. The used thresholds for differing between the locomotions are based on the gait pattern, and are defined for the use of one foot-mounted module, so for a stride. The thresholds are chosen according to chapter 2, and adapted empirically.

Based on the described classification, a time filter in the motion domain, and a blunder detection is applied to the detected locomotions. Since locomotions are usually acted errati-

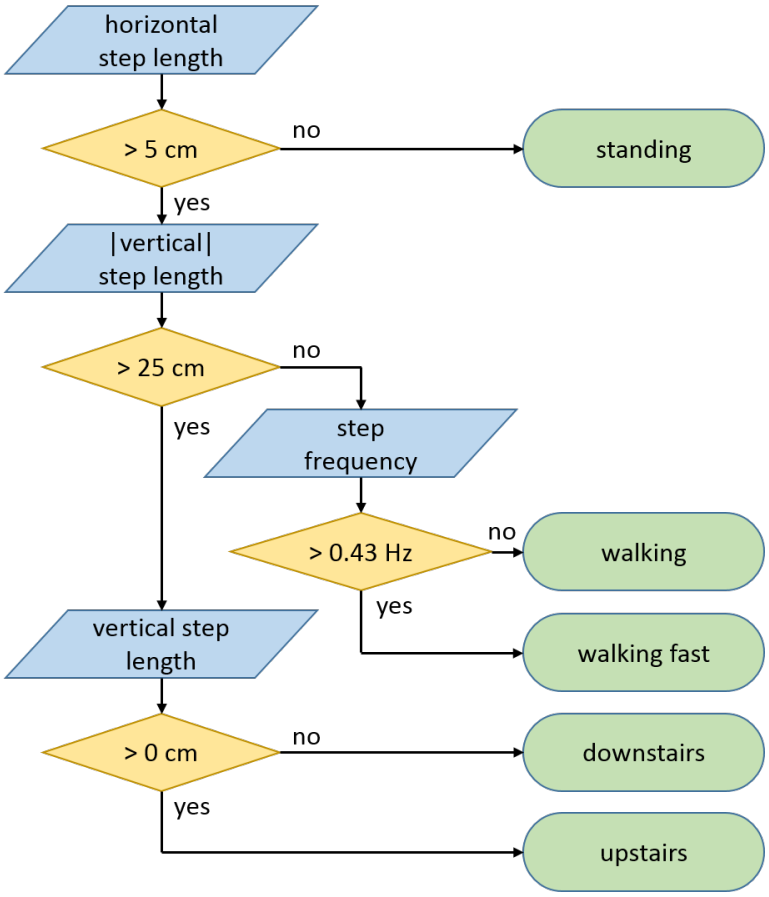

Figure 5.12: Automated classification for locomotion calibration

cally, the final motion detection starts and ends after three consecutive motions are observed, and can therefore be assumed to be correctly identified.

In figure 5.13, an example of an automated classification based on the presented approach is shown. As shown, the defined locomotions were executed within the time series of a couple of minutes. Starting with standing, walking normal, intermittent by walking fast, and finally walking three flights of stairs upwards and downwards. The classification works stable, however, due to the filtering described, not every step event is necessarily classified with a locomotion. This can be overcome by collecting sufficiently enough classification data.

In table 5.1, the recall values from the cross validation of an automated classification of such a data set is shown compared to a cross validation computed with personal recorded reference data. The data was recorded using the *PDR Logger* application, and contains of more than one hour of data, more precisely 77 minutes. This data set is used for the automated motion model creation, walking normal for about half of the time, with the remaining activities about being equally split for the other half of the time. The classification was computed using Weka, utilizing six features using the accelerometers and barometer, resulting in a C4.5 decision tree with 28 rules in the classification tree, see Moder et al. (2015) for details.

Table 5.1: Comparison of cross validations of locomotion recognition approaches

Approach	**Recall values [%]**					
	Weighted Average	Downstairs	Standing	Upstairs	Walking fast	Walking normal
Personal	97.7	92.3	99.9	93.6	96.5	98.3
Automated	94.7	72.1	99.5	85.7	87.8	95.5

The automated locomotion calibration is overall slightly worse than the approach using personal referenced training data. The main reason for this fact is, that the automated classification consists out of a larger data set then the personal training data. While recording personal training data, usually only a couple of instances per locomotion are trained. Additionally, the automated classification excludes certain data in its workflow, in order to function stable without any blunders in the motion domain. Especially the locomotion detection on stairs is worse using the automated classification compared to the personal classification. This is because multiple short flight of stairs with only two or three stairs were used in the presented test data. Therefore, the cross validation might be worse for the instance of stairs, however, if an external motion recognition system, e.g. based on vision might be used, it is assumed that the automated classification functions more accurately.

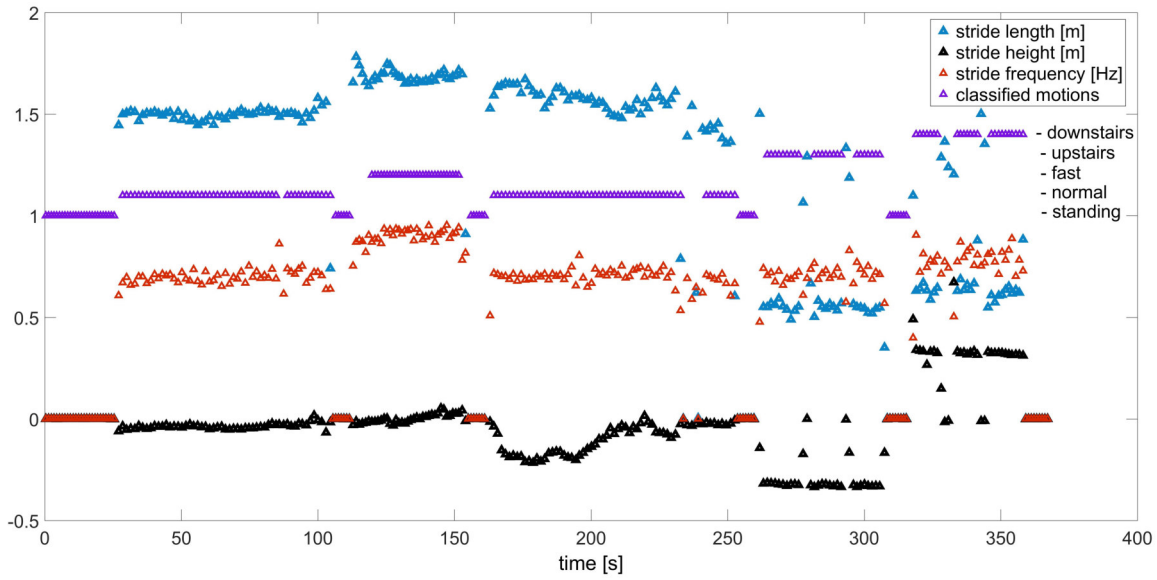

Figure 5.13: Exemplary test data for automated motion model calibration

5.3 Step length parameter calibration

Next to the motion model calibration, the step length parameters have to be defined and calibrated. It is possible to either use predefined classificators and parameters for the step length estimation, e.g. the use of the predefined parameters presented by Chen and Guiness (2014) for a frequency-based step length model.

The approach of calibrating the step length parameters is common and done whenever possible. However, usually only basic tools are available for estimating these parameters, and additionally these tools have to be operated by the user. The concept of using foot-mounted sensors, as well as additional sensors like cameras and light sensors for gait analysis was first presented by Tran and Young (2013). The presented setup is used for gait analysis within laboratories, and it is possible to calibrate the step length parameters for a person robustly with this setup. More approachable, the step length estimation within a Kalman filter, fusing absolute and relative positions on-route, is shown by Ivanon et al. (2018) for a two dimensional space. Recently, Diez et al. (2018) show the concept of estimating step length parameters within PDR approaches using Ultra Wide Band (UWB) signals as reference. Herewith, positions computed out of UWB signals as ground truth for estimating step length parameters are used. The average step length error presented, using a calibration with UWB as a reference, is in the range of 10 cm. Also, this approach needs additional infrastructure as it uses UWB beacons as well as UWB chips, which are not present in smartphones as of today. Summarizing, there are approaches for gait analysis and step length calibration, using external sensors like cameras within motion labs, or wearable sensors like inertial sensors. However, all of these approaches need additional infrastructure or have to be operated by

the user.

The approach for calibrating the step length parameters used in the following sections, is based on the presented *PDR Logger* application. Therewith it is possible, to log reference as well as smartphone data usable for PDR, and calibrate the step length parameters in an automated fashion with the gathered data. In figure 5.14, the estimated step length out of a basic PDR approach compared to the actual step length, computed out of the foot-mounted module is shown. The user was walking in a level at a normal pace, then walking fast between seconds 100 and 150 and, between seconds 250 and 350, walking on stairs. The step length differs throughout the time series, but mostly between seconds 250 and 350, where the user was walking on stairs. Analyzing this figure, the necessity of calibrating the step length parameters, and especially incorporating a locomotion detection is obvious.

The following sections are divided into an analysis of different step length algorithms and an analysis of the difference of the step length estimation during different locomotions.

5.3.1 Analysis of parameter estimation of different step length algorithms

Based on the idea of an automated calibration of the step length parameters, the estimation out of data gathered using the *PDR Logger* application is analyzed. As base, the different step length algorithms presented in chapter 4 are investigated and compared.

The main concept is, to log smartphone data as well as the step events out of the foot-mounted module in a time-synchronized manner. The step events are assumed to be a true reference, where the first and last step events of consecutive executed strides are not taken into account. With the smartphone sensor data, a step detection based on the presented step

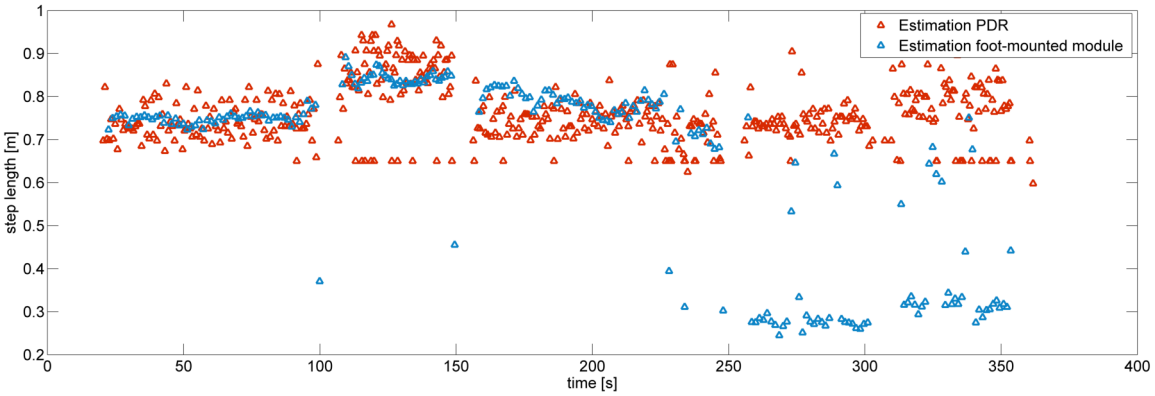

Figure 5.14: Step length estimation using a frequency-based model with ground truth out of foot-mounted module

segmentation is performed. Each detected step can then be correlated to a step event from the foot-mounted module. Here, the moving window size of the step events has to be accounted for in order to correlate the same steps out of both data logs.

Following, based on the step length algorithm and its according parameter set K, see section 4.4.2, the step length parameters can be determined by means of a recursive least squares parameter adjustment. Using a classic least squares parameter estimation as presented in chapter 2, estimating the K parameters, using every correctly identified step event as observation and iterating, using the unit weight of the adjustment, is performed.

In the following figures, the estimated step length using the different step length algorithms are shown compared to their actual step length and step frequency. In figure 5.15 a direct step length algorithm, in figure 5.16 a frequency-based algorithm, in figure 5.17 an advanced frequency based step length algorithm and in figure 5.18 an acceleration amplitude step length algorithm is shown.

Analyzing the presented figures for the main step length algorithms, the differences between the approaches are observable. Using a direct step length algorithm, or a step length estimation based on the body height or step over height, which corresponds to the groin height of a person, a constant step length per step is estimated. No matter if walking with a normal, slow or fast gait, the step length estimation is the same per estimated step. The approaches using the user height or step over height can deliver more accurate results if the algorithm is not calibrated. However, all these approaches deliver the same results if a calibration is done. Although, per single step, this approach is not very accurate, in terms of traveled distance, the direct step length algorithm can be stable enough for many applications. Using a frequency-based or advanced frequency-based approach, the estimated step length directly follows the frequency pattern of the estimated steps. Dependent on the complexity of the algorithm, one or more additional parameters next to the frequency dependency may be calibrated, and therefore the estimated step lengths are more or less dependent on the correct estimation of the step frequency. Using an approach based on the acceleration amplitude, the results estimated when using smartphone acceleration sensors are not satisfactory. This is mostly due to the fact, that the acceleration amplitude may not be estimated sufficiently enough for the purpose of step length estimation. Therefore, it is advised not to use such an algorithm within pedestrian navigation using smartphones.

In table 5.2, the involved users, which the analysis for the step length parameter estimations is based on, are listed. The involved users are listed including their gender, body height as well as mean step length. The body height is correlated to the step over height of a person and is used as input within some step length algorithms. The mean step length is estimated out of a mean of all investigated tests per user and is given as indicator. Regarding gender and body height, the users are viewed to be sufficiently distributed for the presented analysis. As seen, the mean step length is not necessarily directly correlated to the body height of the

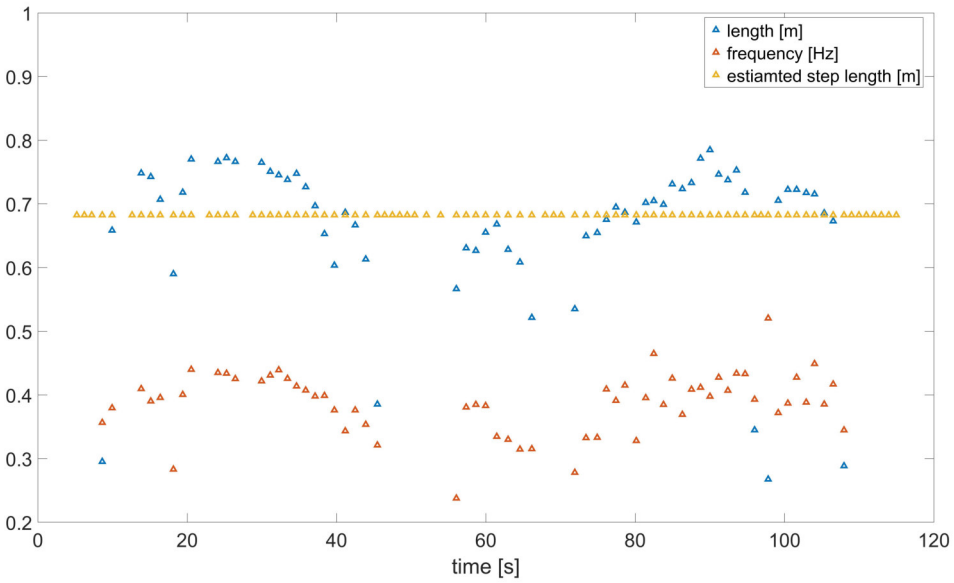

Figure 5.15: Estimated step lengths, direct algorithm

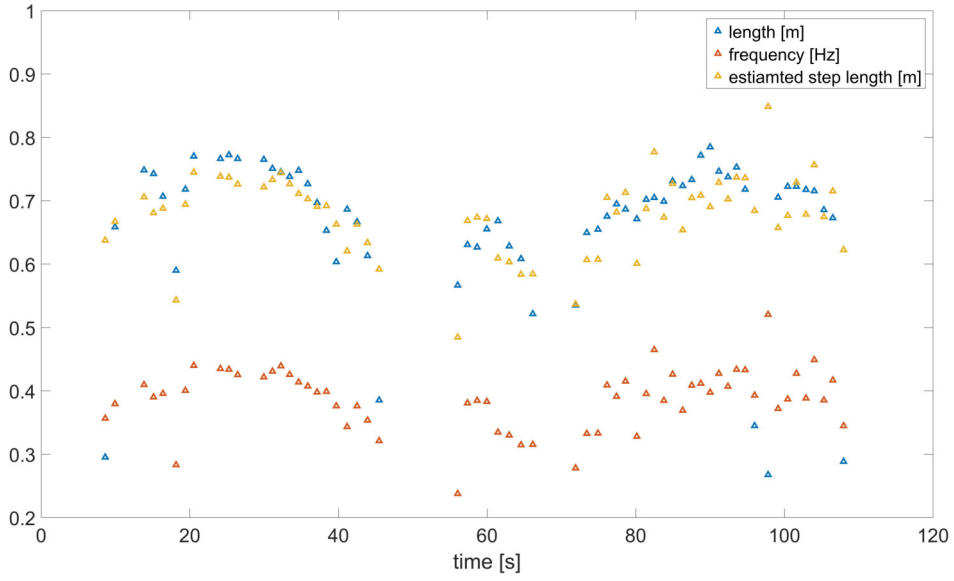

Figure 5.16: Estimated step lengths, frequency algorithm

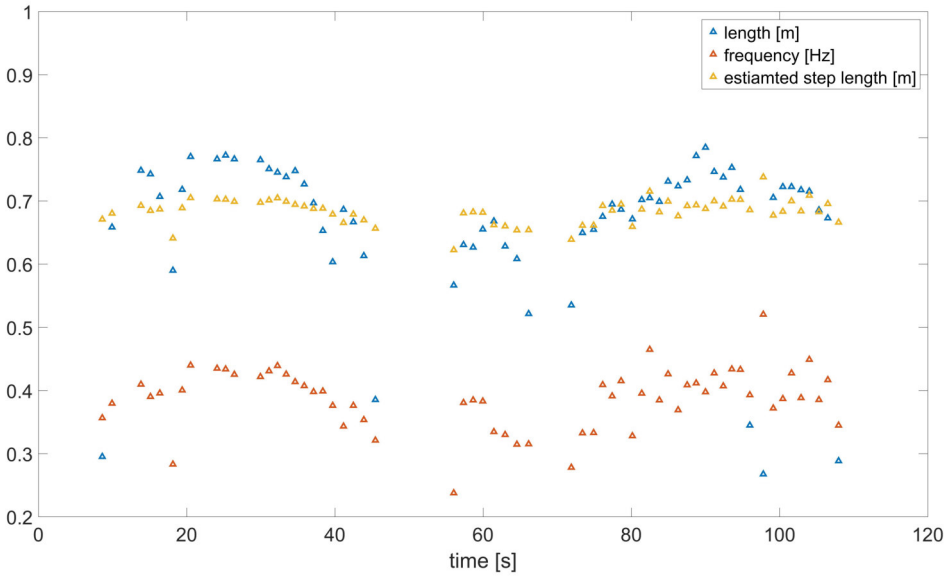

Figure 5.17: Estimated step lengths, advanced frequency algorithm

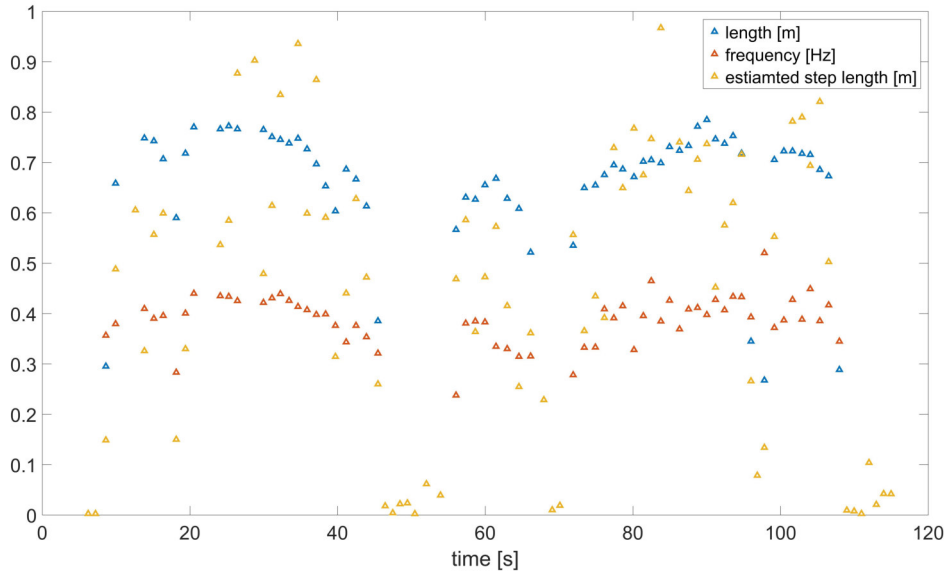

Figure 5.18: Estimated step lengths, acceleration amplitude algorithm

user. For example, user 2 has a slightly shorter mean step length compared to user 1, even though the test user is 5 *cm* taller than the other involved user.

Table 5.2: Users involved testing step length parameter estimation

User	Gender	Body height [m]	Mean overall step length [m]
1	Male	1.80	0.70
2	Male	1.85	0.69
3	Female	1.63	0.68
4	Male	1.80	0.67
5	Female	1.65	0.61

A detailed analysis of each user, regarding the chosen step length algorithms are presented as result. In the tables 5.3, 5.4, 5.5, 5.6 and 5.7, the detailed results of the step length algorithm parameter estimation with respect to the specific smartphone mountings hand held, swinging, and carried in a pocket are shown. Since multiple smartphone mountings have no significant influence on the step estimation based on segmentation, the research question of the influence on the step length estimation is raised. The following results per user are each computed out of more than 100 step events per user and smartphone mounting, and are given as mean of the difference of the estimated stride length to the reference stride length solutions.

Table 5.3: User 1, mean deviation of stride length difference vector [m]

	Step length algorithm used:			
Mounting	Direct	Frequency	Advanced frequency	Acceleration amplitude
Hand held	0.1202	0.1158	0.1152	0.2373
Swinging	0.1180	0.1174	0.1147	0.2100
Pocket	0.1124	0.0827	0.0975	0.1451
Mean	0.117	0.105	0.109	0.197

The general differences between the users, throughout all conducted tests, are:

- the more stable and continuous a person's walk, the more accurate the step length estimation
- the more stable the smartphone mounting, the more stable the step length estimation
- the more intense the acceleration influences, the worse the step length estimation

Table 5.4: User 2, mean deviation of stride length difference vector [m]

	Step length algorithm used:			
Mounting	Direct	Frequency	Advanced frequency	Acceleration amplitude
Hand held	0.0694	0.0681	0.0744	0.2111
Swinging	0.0729	0.0693	0.0699	0.1742
Pocket	0.0679	0.0520	0.0577	0.1963
Mean	0.070	0.063	0.067	0.194

Table 5.5: User 3, mean deviation of stride length difference vector [m]

	Step length algorithm used:			
Mounting	Direct	Frequency	Advanced frequency	Acceleration amplitude
Hand held	0.0541	0.0383	0.0448	0.2017
Swinging	0.0557	0.0494	0.0505	0.1702
Pocket	0.0759	0.0720	0.0727	0.1016
Mean	0.062	0.053	0.033	0.158

The best working step length algorithms, are the ones utilizing the step frequency, e.g. standard or extended step-frequency-based algorithms. The most prominent representative of such an algorithm seems to be that one presented by Chen et al. (2011). The worst working model is the one using the acceleration amplitude, where the smartphone acceleration amplitude is used. Note, that this observation is only true for smartphone-based PDR, and not necessarily true for other sensors, especially if the sensor is mounted rigidly to the pedestrian, e.g. hip- or foot-mounted. Although, it has to be mentioned that the algorithm was tested using unfiltered accelerations. The second worst working models are the ones using direct step length calibration, although they are stable and obviously no blunders with direct step length estimation can be observed. However, blunders in the step length estimation can be present in more complicated approaches using the step frequency or acceleration amplitude.

Summarizing, the step length approaches based on acceleration amplitude does not work sufficiently, whereas approaches based on the frequency work well. Note that the presented results are based on a step detection computed out of a step segmentation. Therefore, the results are true, if the step detection is working well, and therefore the frequency can be estimated correctly. If the step detection is not working sufficiently, the step frequency cannot be estimated accordingly, and step length estimations based on the step frequency will deliver worse results than direct step length algorithms.

The analysis of the step length calibration was presented using the mean deviation out of a validation of the algorithms for single step events. However, more common and more ap-

Table 5.6: User 4, mean deviation of stride length difference vector [m]

	Step length algorithm used:			
Mounting	Direct	Frequency	Advanced frequency	Acceleration amplitude
Hand held	0.1465	0.1036	0.1356	0.1189
Swinging	0.0505	0.0502	0.2263	0.3927
Pocket	0.0930	0.0882	0.0904	0.1548
Mean	0.097	0.081	0.151	0.224

Table 5.7: User 5, mean deviation of stride length difference vector [m]

	Step length algorithm used:			
Mounting	Direct	Frequency	Advanced frequency	Acceleration amplitude
Hand held	0.0490	0.0482	0.0482	0.1664
Swinging	0.0470	0.0466	0.0585	0.1440
Pocket	0.0450	0.0453	0.0456	0.1396
Mean	0.047	0.047	0.051	0.150

plicable for applications, is the consideration of traveled distance. According to Chen and Guiness (2014), current step length models, working with predefined step length parameters, achieve up to 5% of the traveled distance. It is also stated, that the model shown from Renaudin et al. (2012) works within a range of 2.5% to 5% of traveled distance. Contrary, the presented approach using calibrated step length parameters, yield a mean of 1.6% of the traveled distance as average for all users, smartphone mountings and step length algorithms, with the exception of the acceleration amplitude approach. All presented step length algorithms, the direct correlated, the height correlated or the frequency-based approach, are in a similar range regarding traveled distance. The exception is the acceleration amplitude approach, which may not be used with smartphone sensors mounted loosely to the pedestrian.

In table 5.8, the results of traveled distance of the test data, using one step length algorithm, a standard frequency approach, is given. The results regarding traveled distance vary with 0.1% to 4.1% dependent between the users and smartphone mountings. As a mean over all users and all mountings, an offset of 1.6% of the traveled distance, for a standard frequency-based approach, can be observed. The same results for an approach directly based on the body height, as mean of all users and mountings, yields 1.5% of traveled distance. An advanced frequency-based approach, as mean of all users and mountings, yields 1.9% of traveled distance. Overall, between all users, smartphone mountings and step length algorithms, the traveled distance is within 1.6%. Which is a significant improvement compared to 5% for approaches not being calibrated to the user specifically. The results from the analysis regar-

Table 5.8: Results for calibrated step length estimation as difference in traveled distance to a reference in [%], using a frequency-based step length estimation

User	Hand held	Swinging	Pocket	Mean
1	2.3	3.3	1.8	2.5
2	0.9	1.7	1.4	1.3
3	0.1	0.8	1.8	0.9
4	4.1	1.2	2.7	2.6
5	1.0	0.9	0.9	0.9

ding the different step length algorithms, excepting the acceleration amplitude approaches, are all similar in terms of traveled distance. Therefore, it is assumed that they are all equally usable for pedestrian navigation, if calibrated correctly.

5.3.2 Analysis of parameter differences with respect to motions

The approaches for collecting the necessary data, and then estimating the locomotion as well as the step length calibration parameters, were presented so far. While the smartphone mounting has no major influence on the step length estimation, the current locomotion has. The detected locomotion can, e.g., be used as input for the position estimation using a map-supported particle filter, or in the guidance process of a pedestrian navigation. Additionally, the current locomotion has a direct correlation on the step length estimation. As already shown in figure 5.14, especially when walking on stairs, even calibrated step length models do not estimate the step lengths accordingly. Not taking the current locomotion into account is assumed to be one major reason, why global PDR step length models do not work sufficiently for more accurate applications.

Therefore, it is advised, when using a real time algorithm, that the calibrated step length parameters are estimated based on the estimated step detection as well as the estimated locomotion. If a step event is triggered, the step length is estimated based on the calibration parameters of the used step length model of the current locomotion. In table 5.9, the mean step length estimation per locomotion for one user and one set of data is shown. As assumed, the mean step length is nearly 10 cm longer when walking fast on a level compared to when walking normal. More important, the mean step length when walking on stairs is close to the assumed 30 cm, which is the standardized step length for stairs in public buildings in Austria.

More detailed, the standard deviation as difference between estimated step length out of calibrated algorithms and the true step lengths, is given for a all locomotions and the main step

Table 5.9: Step length parameter estimation for different locomotions, using one set of data and an advanced frequency estimator

Locomotion	Mean estimated step length [m]
Walking normal	0.753
Walking fast	0.842
Upstairs	0.273
Downstairs	0.326

length algorithms in table 5.10. The test data used for this analysis was recorded holding the smartphone in the hand while executing the different locomotions. The computed deviation may be seen as an indicator for the quality of the functionality of the used step length algorithms, while the used data mostly corresponds to walking normal, which yields similar results compared to the presented tests in section 5.3.1 for multiple users. Analyzing table

Table 5.10: Stochastic analysis of step length parameter estimation corresponding to the reference solution for different locomotions

	Mean deviation of estimated step length [m], algorihtm used:			
Locomotion	Direct	Frequency	Advanced frequency	Mean step length [m]
Walking normal	0.0743	0.0680	0.0715	0.7531
Walking fast	0.0123	0.0120	0.0124	0.8421
Upstairs	0.0115	0.0113	0.0121	0.2733
Downstairs	0.0373	0.0372	0.0381	0.3265

5.10, taking the information of the current locomotion into account seems to be necessary for an accurate PDR. The locomotion differentiation within the step length estimation may be a valid approach, where at least walking on a level and walking on stairs may be distinguished within a step length estimation approach.

5.4 Conclusion and time variability within the presented calibration approaches

Closing, a brief conclusion and discussion on time variability regarding the presented calibration procedures, as well as techniques for how and when to perform calibration, is given.

Concluding the smartphone sensor calibration, it is shown that all investigated sensors have significant sensor errors. However, only the accelerometers are stable enough over multiple days and turn-on processes, that a long term calibration proves useful. There, one obvious result is the calibration of the acceleration main scales and bias parameters. In figure 5.19, the total acceleration is computed using uncalibrated as well as calibrated acceleration measurements, for a data set of static measurements for 26 orientations.

The time series shows the total acceleration $acc_{total} = \sqrt{acc_x^2 + acc_y^2 + acc_z^2}$ for the calibration test setup, where the smartphone gets moved into multiple independent orientations. In these orientations, the wooden structure, see figure 5.5, is static for a sufficiently long time, usually at least 20 s. The difference between uncalibrated and calibrated measurements is visible. Where ideally, the total acceleration should be in the range of 9.8 m/s^2 at the tested location, the uncalibrated measurements are in the range of 9.5 m/s^2 up to 10.4 m/s^2, and therefore vary about nearly 1 m/s^2. Contrary, the calibrated measurements vary in the range of about 0.1 m/s^2. Therefore, the calibration of the acceleration scales and bias parameters can reduce the measurement errors on the total acceleration by a factor of 10.

The approach of calibrating the smartphone sensors can be seen similarly to a coarse and fine alignment, present within high precise inertial sensors. The acceleration sensors may get calibrated with static tests in a low vibration environment for multiple days or even weeks. The magnetometer sensors may get calibrated every day for the magnetic environment or building infrastructure used, but may then be possibly used for extended time periods. Finally, the gyroscopes are best calibrated in a kinematic fashion on-route. Ideally, utilizing a Kalman filter, which estimates the angular rate biases continually.

For the accelerometers, an approach similar to the one presented may be used. While it is not necessarily needed to calibrate the main scales for the purpose of step detection, the accelerometer biases are. Therefore, a test setup using at least three independent orientations is sufficient to compute the bias errors. However, a setup using more orientations is advised. For the magnetometers, the basic sphere adjustment proves to be functional for smartphone-grade sensors, even if coarse magnetic deviations are present in the environment. For the gyroscopes, the bias errors have to be estimated. The gyroscope bias estimation uses ω_E as measurement quantity, which is in the range of 0.003 $^\circ/s$. Since the bias error we are dealing with smartphone gyroscopes are in the range of up to some 0.1 $^\circ/s$, it is possible to neglect ω_E and estimate the gyroscope bias, assuming $\omega_{ib}^b = 0$. In figure 5.20, the approach for an on-route heading filter for estimating the gyroscope bias vector b_g is shown. Basically, the suggestion is to run a gyroscope bias estimation in a Kalman filter process, whenever the smartphone is stationary. The detection of stationarity can be done, using a combination of the acceleration and angular rate norm. Further, it is suggested to only estimate the relative heading when the device position is stable, and omit the heading estimation when the smartphone mounting is unstable.

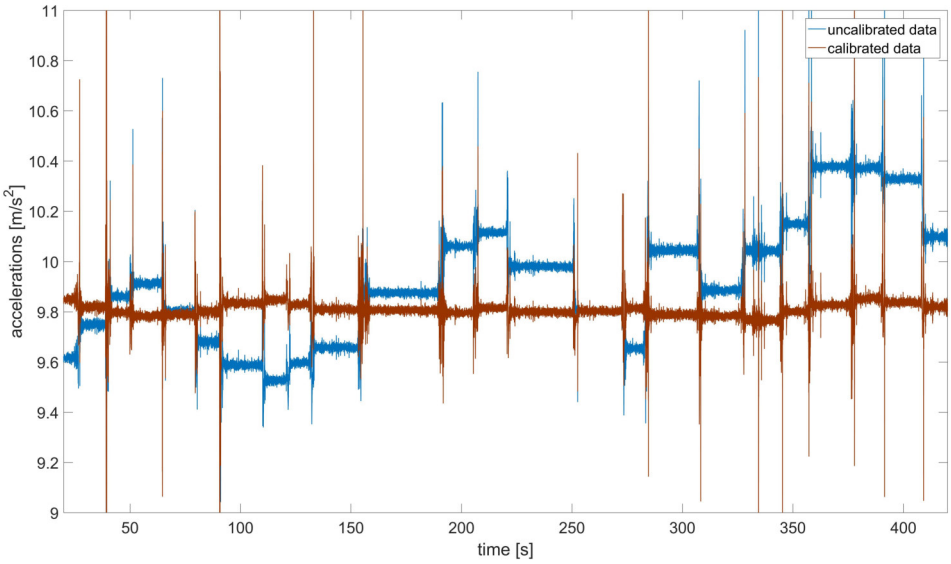

Figure 5.19: Total acceleration, computed using uncalibrated and calibrated accelerations measurements

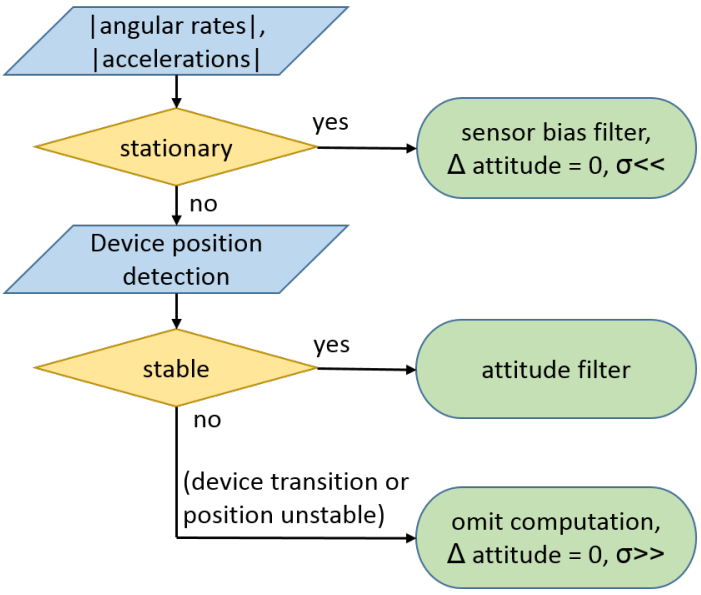

Figure 5.20: Gyroscope bias on-route estimation

Concerning the algorithm calibration, it is necessary to have personal calibrated parameters for the used locomotion and step length approaches. The main advantage in case of locomotion recognition is the easier access of the automated classification. However, for the step length estimation, the main locomotions necessary for real time estimation are walking on a level or walking on stairs. Dependent on the accuracy needed for the specific application, the difference between walking normal and walking fast can also be accounted for, to a lesser extend, using a frequency-based step length algorithm. For the calibration of the difference step detection models, the results out of the available approaches are not significantly different to each other. This is true for the traveled distance, which is of interest for the pedestrian navigation application. Out of the most prominent step length algorithms, only the calibration of the acceleration amplitude approach delivers no usable results using smartphone sensor data. The main conclusion for the step length estimation is, that the calibration of the algorithm parameters is useful, independent of the specific model used. Additionally, the presented, novel automated approaches work, and are more efficient for the user than previously published methods.

Generally, the presented tests were observed over multiple months and even years, involving multiple users. Between the same user and data gathered at different points in time, multiple months apart from each other, no significant time variability can be observed for the presented approaches. However, the step detection and step length estimation is mostly and strongly dependent on the human gait cycle. Consequently, everything that changes the gait cycle will affect the step detection and step length estimations. Possible influences include wearing different kind of shoes, possible injuries which affect the gait cycle, as well as long term aging of the user, which will influence the calibrated approaches.

6 Application of calibration approaches within positioning for pedestrian navigation

Within this chapter, the application of the discussed calibration approaches for Pedestrian Dead Reckoning (PDR) as positioning solution for pedestrian navigation is shown. Starting, the effects of sensor and parameter calibration with an unsupported PDR solution are presented. The effects of sensor and parameter calibration are analyzed for the components of step detection, step length estimation, step orientation estimation as well as for the incorporation of locomotion recognition into PDR, using examples.

Following, the application of PDR within a position filtering is presented, using a particle filter and raster maps. The usefulness of an accurate PDR as input for a position filtering is shown in general in an outdoor setting. Afterwards, especially the incorporation of a locomotion recognition within position filtering indoors is presented.

Finally, the application of PDR within a graph-based positioning approach is presented, using vector maps. The concept of graph-based positioning eliminates the major error source in a stand-alone PDR solution, because the angular rate integration is reset on every edge of the graph. Therefore it is possible, to compute positions based on a PDR only solution for extended periods, e.g. up to ten minutes or more. For the graph-based positioning, the usefulness of smartphone sensor calibration is presented.

Not all presented PDR innovations are used within each application. However, the main impact of the presented innovations is outlined directly in the section showing the effects of calibration on unsupported PDR solutions.

6.1 Analysis of calibration on unsupported pedestrian dead reckoning

In chapter 5, the calibration of the sensors as well as model parameters, usable for PDR using smartphones, are presented. Specifically for a PDR using smartphones, the calibration

approach for the inertial and magnetic sensors is discussed. For the algorithms, the calibration of the step length model parameters, as well as locomotion recognition parameters is shown.

Discussing the application of the calibration approaches, the effects directly observable of calibrating the smartphone sensors and algorithms on unsupported PDR solutions are shown. Then the effects of the calibration approaches presented are discussed, in a separated manner for the domains of step detection, step length estimation, step orientation and locomotion recognition, using examples. The analysis of calibration for the step estimation and step orientation uses the smartphone sensor calibration. Within analyzing the step length estimation models, the estimation of the step length model parameters are discussed. Here the analysis of the locomotion recognition uses the detected locomotions, for incorporating them into the step detection of PDR.

6.1.1 Analysis of calibration for step estimation

The step estimation is computed using a signal segmentation, see section 4.4.1. The effect of using correctly calibrated acceleration measurements, compared to uncalibrated acceleration measurements, on the step detection, is shown in figure 6.1. The detected steps, based on a signal segmentation of the total acceleration, are shown next to the calibrated and uncalibrated measurements. For this specific example of ten strides, all steps can be detected using calibrated accelerations, where two steps are not detected when using uncalibrated measurements. The magnitude of the direct effect of the acceleration calibration varies with the magnitude of the acceleration bias currently present in the measurements. However, in the tests executed, using calibrated measurements always resulted in a more robust step estimation.

The actual influence of the smartphone sensor calibration on the step estimation, is strongly dependent on the used step estimation approach. Using the presented signal segmentation with absolute thresholds, the correct calibration of the acceleration measurements is of importance. However, with different approaches, e.g. a step estimation based on detecting the main frequency component, a bias error on the acceleration measurements is not as critical, because it has no major influence on the frequency analysis. However, as mentioned, step detection based on detecting the main frequency, and correlating this to the gait pattern, is not strictly real-time capable, and therefore not ideal for some applications. In figure 6.2, the difference of the developed signal segmentation compared to a frequency-based step detection, in this case the *Google step detector*, is shown, next to the true step events estimated with the foot-mounted solution out of the *PDR Logger* application. The main effect visible is, that the frequency-based step estimation will always fail to detect the first two or three steps for a walk consisting of consecutive steps. Contrary, frequency-based step estimations are stable against other signal influences on the device used for step estimation. This is true

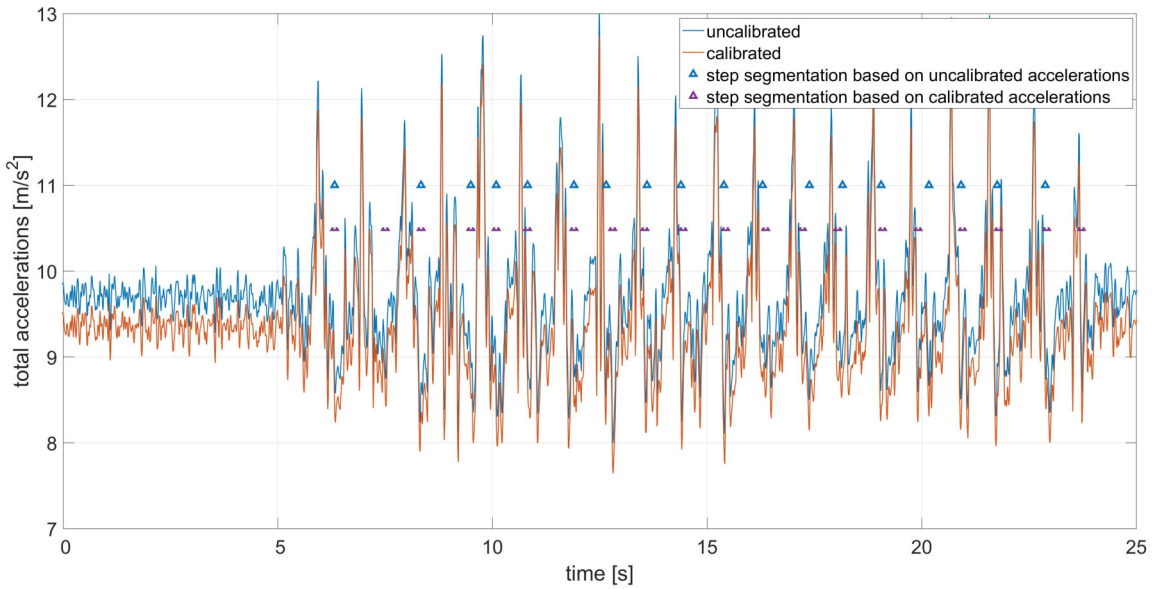

Figure 6.1: Step detection using uncalibrated and calibrated acceleration measurements

for every segment of consecutive steps. Imagine walking along a hallway, stopping to open a door, then starting walking again, or similarly stopping to wait for an elevator. All these sections of walking can be viewed as stand-alone segments of consecutive steps, where the presented lag time in frequency-based step detections will occur.

6.1.2 Analysis of calibrated step length estimation

Detailed analysis for the results regarding step length parameter estimation are presented in section 5.3.1. There, it is shown, that the analysis of the mean traveled distances of a calibrated step length estimations achieve up to 1.6%, compared to about 5% when using step length models with universally calibrated parameters. These detailed analysis are based on the test data from all users, testing multiple smartphone mountings and estimating the step length model parameters. The estimated step lengths are then computed out of these calibrated step length models, and compared to the actual distance walked, based on the foot-mounted solution. The then estimated deviations from these residuals result in a mean deviation per step of 7.0 cm for frequency-based step detections, and a mean deviation of 8.2 cm for advanced frequency-based approach. Computed as a mean of all users and step length models, 7.6 cm as deviation per single step are achieved.

An accuracy analysis for the step events of one trajectory, where the user was walking normal, is shown in figure 6.3. The standard deviation of the data presented is 5 cm with a mean of 0 cm, meaning that $N \sim (0, \sigma)$ is correctly assumed.

Note that the final results of traveled distance combine the results of the step detection as well as the step length estimation. For the presented step length analysis, where the smartphone

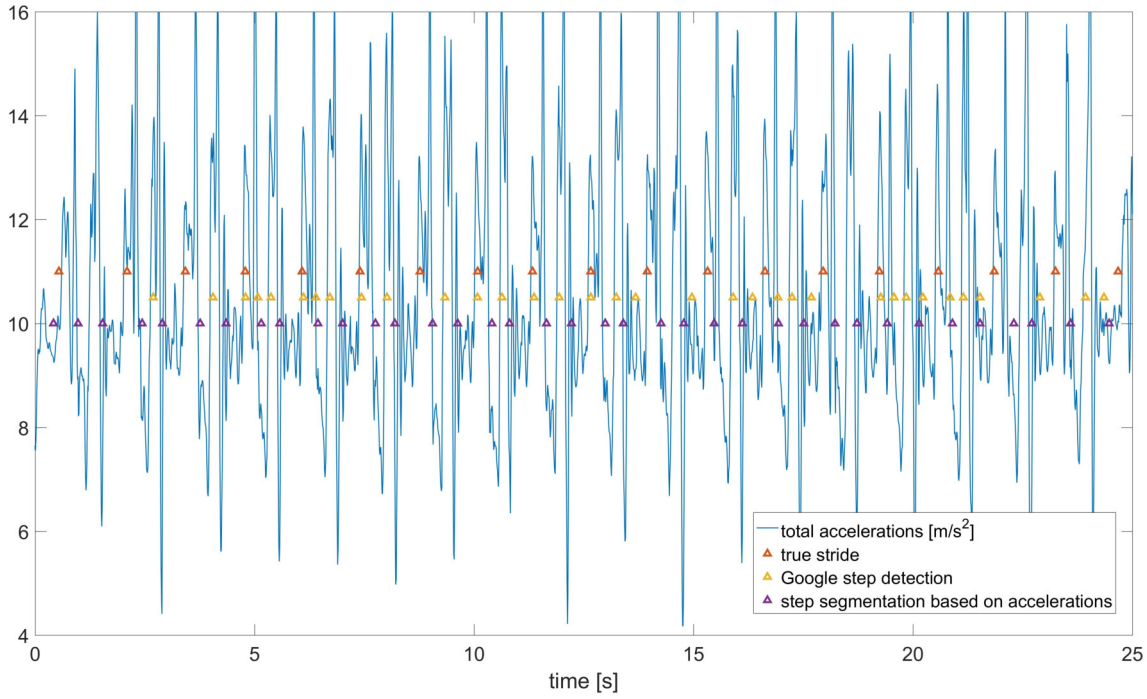

Figure 6.2: Step detection using uncalibrated and calibrated acceleration measurements

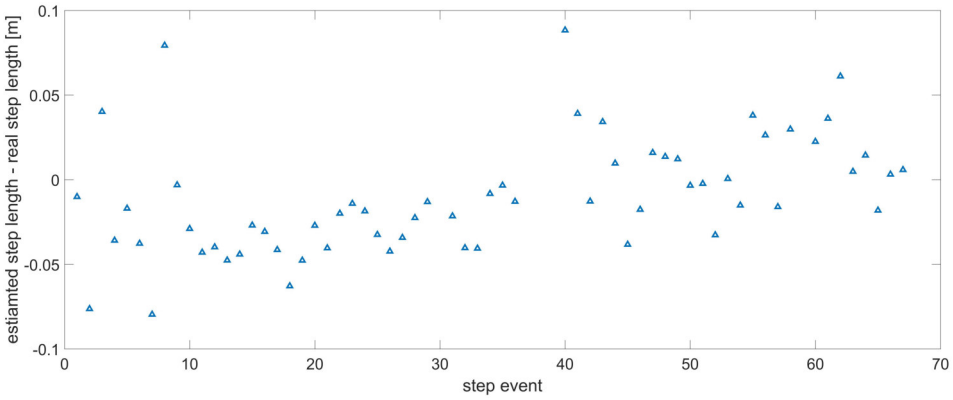

Figure 6.3: Accuracy analysis of step length estimation

mounting was assumed to be stable, the signal segmentation has proven to work and all steps were identified within the test setups. However, if the smartphone mounting is unstable, or unusual accelerations are present on the smartphone, the signal segmentation and therefore the step detection can fail.

6.1.3 Calibration of step orientation in pedestrian dead reckoning

For the step orientation estimation within PDR, two methods exist:

- using an absolute heading
- using a relative heading

When using an absolute heading solution, the absolute orientation of each step is estimated. This is a straight forward approach, which allows for a self-contained, relative position solution without any additional input, and can be useful for some applications. The main drawback of using an absolute heading is, that the orientation angle between the sensor and the body frame, so the smartphone and the user, has to be known. Approaches to estimate this orientation angle exist, e.g. estimated out of the magnetic heading of the smartphone compared to an absolute heading, out of a time series of absolute position solutions. However, such estimations of the orientation angle are computationally intensive, and not stable. They can work, if the accuracy of the absolute position solution is sufficiently high, and no transition of the device regarding its mounting in reference to the user takes place.

The absolute heading of the smartphone can be assumed to be equal to the heading computed out of the magnetic measurements only, and therefore follows the concept of magnetic compassing presented in chapter 2. Or, the absolute heading can be estimated using a filter based on the magnetic measurements and the inertial measurements, as presented in section 4.4.3.

Regarding the sensor calibration when using an absolute heading estimation, the magnetic measurements have to be accounted for. For smartphone sensors, usually the offset and radius of the basic sphere equation, see section 5.1.3, are calibrated and then applied to the magnetometer measurements. Mathematically, the accelerations measurements are involved in leveling the magnetometers, but the acceleration biases only have neglectable influences on the estimation of a magnetic orientation, and are therefore not discussed further.

The effect of a calibration of the magnetometers, dependent on the magnetic deviation present, can yield in up to some 10 ° of orientation error, compared to a reference direction. In figure 6.4, the estimation of an absolute heading of a smartphone while walking along a hallway, computed using calibrated as well as uncalibrated magnetic measurements, is shown

next to the reference direction of the hallway. At first, the user was standing still, and then started to walk along the hallway for 20 s. An offset in the heading estimation in the range of 10 ° is visible. However, if no especially high magnetic deviations are present in the current surrounding of the smartphone, the absolute direction can usually be gained within some 10 °, which might be sufficient for certain applications. Note, that when using devices different to smartphones, where a more severe magnetic deviation is present, a magnetometer calibration may be strictly necessary in order to derive meaningful direction estimations.

Due to the mentioned problems, introduced when using an absolute heading, if possible, it is advised to use a relative heading solution. Relative heading estimations can be useful inputs for multiple filter approaches, and may be computed using the approach presented in 4.4.3. An example of a relative heading estimation, in reference to the true step orientation computed out of a foot-mounted solution, is shown in figure 6.5.

The standard deviation of the data shown is 4.5 °, with a mean of 1.8 °, which means, that there is a drift in the presented relative heading estimation compared to the reference heading. The present drift is due to a bias in the angular rate measurements, which could not have been accounted for in the bias estimation in this example.

Similar to a traveled distance analysis, for the relative heading an accumulated orientation error can be computed. Out of the test series, using data of all users, accumulated heading errors in the range of $2.4 - 2.7$ ° over the time span of five minutes are achieved. As an average result, 2.5 ° heading error in the accumulated, relative heading estimation is achieved.

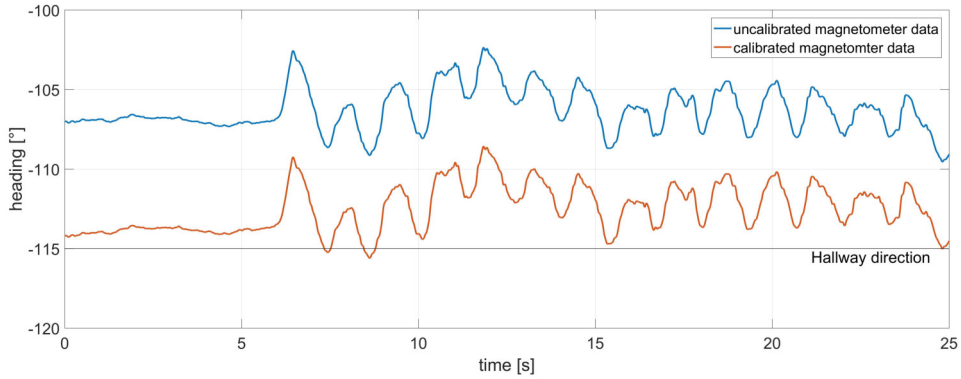

Figure 6.4: Estimation of magnetic heading while walking along a hallway, time-filtered using a Kalman filter

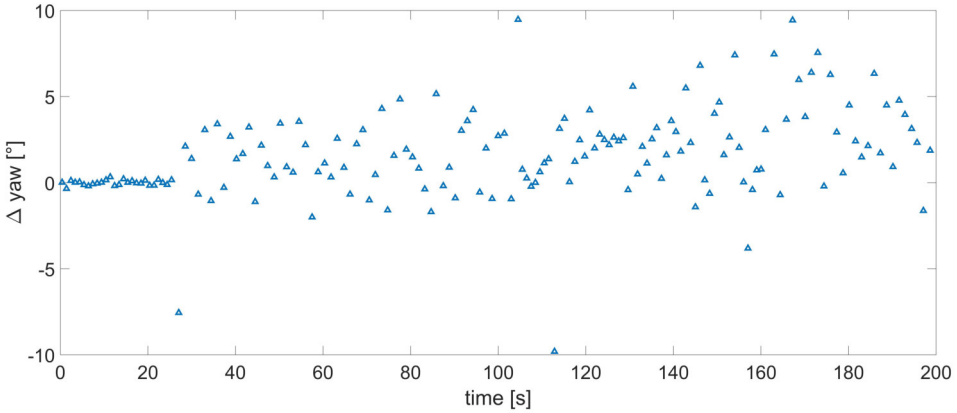

Figure 6.5: Accuracy analysis of relative heading estimation

6.1.4 Analysis of incorporation of locomotion recognition

Finally, the analysis of the calibration of an unsupported PDR, the incorporation of a locomotion recognition is discussed. The importance of a locomotion recognition is already discussed within section 4.4.2, with the conclusion that at least on stairs, the information of the current locomotion is of major importance. Next to being used as input for the step length estimation, the current locomotion can be used as information for the step detection estimation. Additionally, it can be used within position filtering approaches, e.g. as an additional parameter in the update process of a particle filter.

As example, the step length estimation of a test trajectory is given in figure 6.6. The step lengths of the same trajectory are estimated utilizing three methods, using an uncalibrated step length model, using a calibrated step length model without the input of the current locomotion as well as using a calibrated step length model for each locomotion, and with the input of the current locomotion. The time series of the test trajectory starts with the user standing, then walking straight along a hallway, walking on stairs between seconds 50 and 70, and then walking straight along a hallway again. As visible, the difference between using a calibrated and uncalibrated step length model is within a few *cm* for each step length estimation. The main difference visible is during the phase of walking on stairs, where only the approach using the information of the current locomotion is yielding useful results. The approaches not using the information of the current locomotion estimate steps in a similar range when walking normal, because the step frequency is in a similar range.

The effect of an incorporation of a locomotion recognition is directly visible. It is especially useful for the applications shown in the following section, e.g. an position filtering using a particle filter indoors. If the particle prediction is on stairs, using a wrong step length estimation compared to the assumed accuracy of the PDR model, a particle filter may fail and ignore all current particles. Therefore, dependent on the particle filter design, it may be

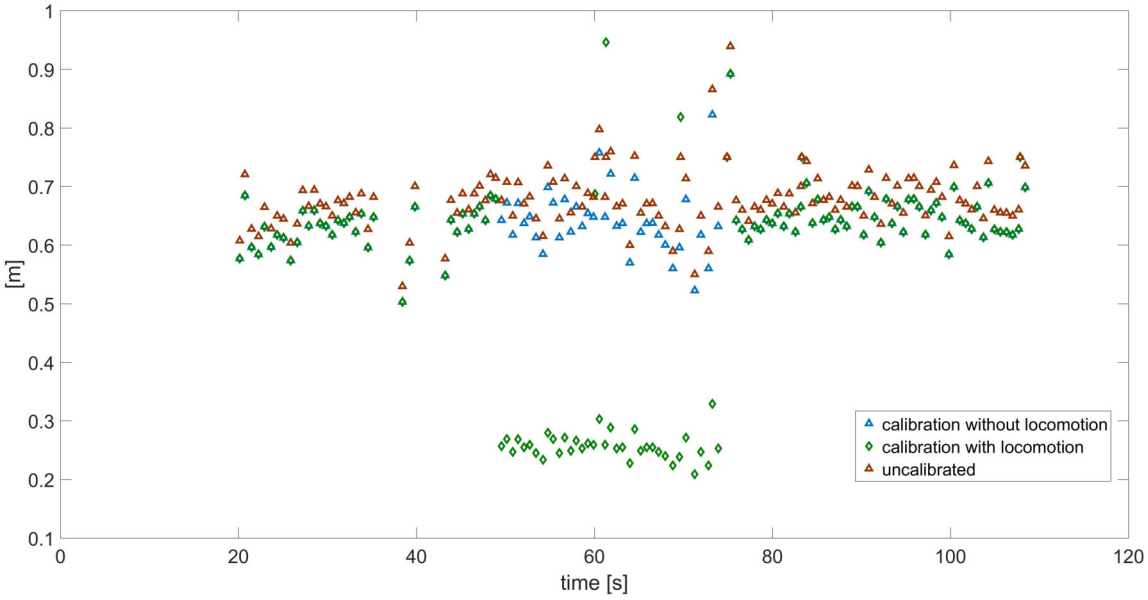

Figure 6.6: Example of step length estimation incorporating locomotion recognition

possible that a particle filter does not function properly because of the unrealistic estimation.

6.2 Position filtering utilizing pedestrian dead reckoning

The approach, best used for filtering positions, usually depends on the available position solutions, the kinematics of the expected positions, as well as possible additional information, e.g. map information. The basic concept is to use a filter based on Bayesian estimation, either a Kalman filter representation or a particle filter representation, see chapter 2.

In the position domain, instance-based or continuous approaches may be used. Where in the continuous domain the Kalman Filter is used most often, in the instance-based domain the particle filter is becoming more and more important. For positioning, the particles represent the defined state in N instances, where for every particle the whole state is estimated. The number of particles N may vary, and generally the higher the number the more accurate the estimation. However, for the applications shown, 500 particles are used, chosen due to the empirical analysis in Hafner (2015) for positioning using PDR as input.

For pedestrian navigation, using the positioning approaches presented in chapter 4, a Kalman filter might be chosen if no additional information is present, and a particle filter might be utilized if additional information, like an occupancy map, is present.

In this section, the position estimation for pedestrians, using smartphone sensor data and an occupancy map is shown for an outdoor setting as well as an indoor setting, using a particle filter. The position filtering outdoor is using PDR as well as Global Navigation Satellite System (GNSS). And the position filtering indoors is using PDR, incorporating a locomotion recognition, as well as a Wide Area Local Network (WLAN) fingerprinting solution. When using an occupancy map as a raster map, the representation of the position estimation as particles is ideal.

In the position domain, the particles comprise a state with $(\varphi, \lambda, \theta)$ for the latitude, longitude and the heading. Based on initial information on the starting position and it's stochastics, the particles get distributed around that position. After an initialization, the main stages of a particle filter in the position domain comprise:

1. checking the occupancy of the particles
2. propagation of the particles and computation of the importance weight
3. resampling of the particles around the most likely estimation

The particle propagation within the positioning approach is done using data computed from the implemented PDR solution, namely velocity v and the change in heading $d\theta$, see equations (6.1) and (6.2).

$$v'_t = v_t + N(0, \sigma_v) \tag{6.1}$$

$$d\theta'_t = d\theta_t + N(0, \sigma_{d\theta}) \tag{6.2}$$

And the importance weight of these propagated particles is computed using an absolute position estimation and the map information, see equation (6.3).

$$\omega^i_t =< \omega^i_{absolute_t}, \omega^i_{map_t} > \tag{6.3}$$

The value $\omega_{absolute_t}$ is computed out of GNSS for the outdoor tests, and out of a WLAN fingerprinting for the shown indoor tests.

Within the importance weight computation, the particles can get spatially distributed corresponding to their position, which enables the use of map data. In this phase, occupancy checks of the particles corresponding to the map as well as wall collision checks along the path in the particle propagation are possible. If, due to the map information, the occupancy or propagation is not possible, the particle will be ignored and not used within the importance weight computation. The system design of particles being checked according to their occupancy, their propagation path and their final resampling is visualized in figure 6.7. At first, the occupancy of the current particle distribution is checked. Then, the particles are propagated, in this example from south to north, and again checked for a wall collision or

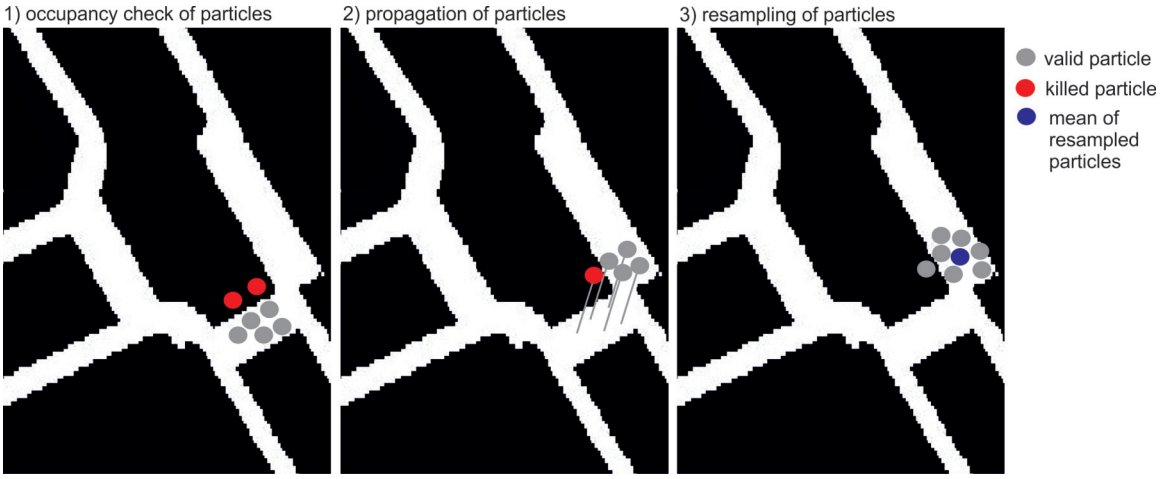

Figure 6.7: System design of particle filter incorporating occupancy maps

occupancy. Finally, the particles are resampled, and the mean of all resampled particles is computed.

The mean or most likely of the current particles, ω_t^i, can be estimated based on the current and posterior probability, and the position is then determined either as the particle with the highest probability or out of the weighted mean of all particles. The resampling of the N new particles around the most likely particle is then conducted randomly, based on the stochastics in proportion to ω_t^i with a systematic resampling, see Thrun et al. (2006). Therewith, all resampled particles get the same weight.

A basic occupancy map consists of a grid map, where the grid point has the value of 1 when the particle being there is an option and has the value of 0 when the particle cannot physically be at that position. Such occupancy maps can get generated out of city maps or open data, depending on the application. In the case of outdoor positioning comprising GNSS, the assumption is that the pedestrian is moving outdoors. Following, all accessible outdoors areas will be set 1 and all occupied areas will be set 0, like constructions or dense vegetation. In the case of indoor positioning, the occupancy map can usually be retrieved out of already present building plans. Again, the most straight forward approach is to set the grid values of a raster map 1 when the area is accessible for a pedestrian, and 0 when the area is occupied through the building itself, or simply not accessible.

One problem of a particle filter using an occupancy map, eliminating particles if they are estimated in or propagated into occupied areas, is the possible degradation of all particles. This is an extreme example, although not uncommon, if the particle filter setup is not correctly defined. One example, directly applicable using PDR, is a vastly wrong propagation using PDR for this propagation. If the PDR is computed without using a locomotion recognition, and if walking on stairs, this can wrongly propagate the particles, leading to a full particle degradation, and therefore the filter can fail.

A detailed analysis of using particle filters for positioning is given in Hafner (2015). Additionally, details on the setup and the strategy of importance weight computation and the methods of resampling are shown. The filtering results presented in the sections 6.2.1 and 6.2.2 were evaluated with the tool implemented from Hafner (2015), in cooperation with the presented work, using the position and calibration approaches developed in chapter 4 and chapter 5.

6.2.1 Position filtering outdoors using smartphone sensor data

For the outdoor tests, an occupancy map for the area of the *new campus* at Graz University of Technology was used, see figure 6.8.

The propagation is done using the implemented PDR solution, and the position update is computed using a GNSS position estimation as absolute position solution, using smartphone sensor data.

$$\omega_t^i =< \omega_{GNSS_t}^i, \omega_{map_t}^i > \tag{6.4}$$

The test setup, presented for the outdoor tests, is a pedestrian walking a predefined trajectory at the *new campus* at Graz University of Technology, purely using pedestrian walkways, see figure 6.9. The same test setup was already presented, for analyzing the performance of the particle filter, in Hafner (2015). The starting point is in the north-east of the shown figure, and the trajectory is executed counterclockwise. The user is holding the smartphone, logging all sensor data, in his/her hand in the front of his/her body, and walking at the trajectory at a normal pace. Additionally, all sorts of external influences, like other pedestrians, waiting at traffic lights and so on were executed as realistically as possible. As absolute position, the smartphone GNSS solution is used, where as relative position solution, a processed PDR algorithm based on the presented concept is computed. The results of this test trajectory are shown in figure 6.9. On the one hand, the GNSS position estimation is shown, which clearly can not stay at the pedestrian walkways consistently. Contrary, the trajectory estimated with the particle filter, incorporating the PDR and the occupancy map, manages to stay on the pedestrian walkway throughout the whole trajectory.

The evaluation of this trajectory was done using 40 established reference points along the trajectory, which are equally distributed, see figure 6.10 for residuals of the GNSS solution and the results from the particle filter. At these reference points, a synchronized time stamp was triggered from an observing person. Therewith, these distinct reference points can be used as ground truth throughout the trajectory. As a mean over all reference points, the GNSS position has horizontal position residuals in the range of 9.1 m, which is expected. The particle filter solution, using GNSS as well as PDR, results in 2.7 m. Where the residuals

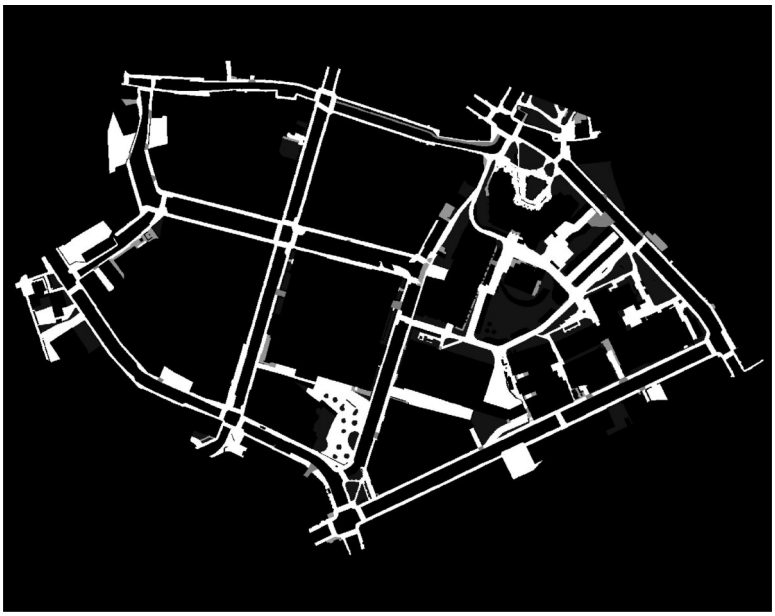

Figure 6.8: Occupancy map for outdoor area

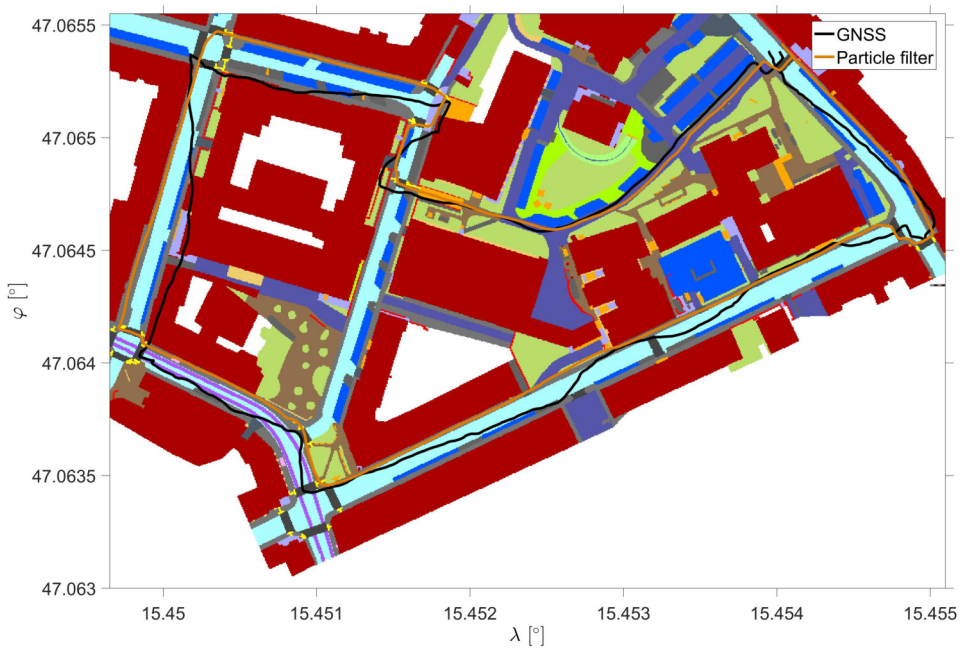

Figure 6.9: Results of test trajectory outdoor

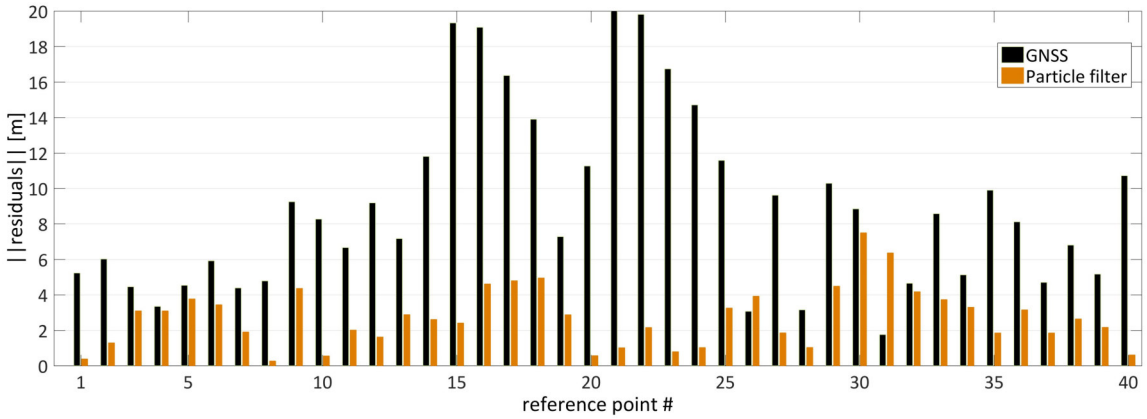

Figure 6.10: Residuals at reference points for test trajectory

of the GNSS only solution are equally distributed in the along- and across-component, the estimations based on the particle filter using the occupancy map performs better in the across-component of the residuals. This mainly results from the restrictions, that the occupancy map from figure 6.10 induces into the particle filter. Therefore, the mean residuals of 2.7 m are sufficient enough for the proposed particle filter solution, in order to have a position estimation on the correct pedestrian walkway. Contrary, a normal distributed standard deviation of 2.7 m, without the spatial restriction of the implemented particle filter, would not be sufficient to constantly stay on the correct walkway.

For computing positions, using smartphone sensor data, the particle propagation and occupancy map information enable the particle filter solution to stay on the walkway for a correct position estimation. This might be critical for guidance within pedestrian navigation, dependent on the application used.

6.2.2 Position filtering indoors incorporating motion recognition

The indoor tests were conducted at the test bed, set up in the building *Steyrergasse 30* in Graz. The occupancy map used, shown for one floor, is given in figure 6.11.

For the indoor setting, the implemented PDR solution was used for propagating the particles, and the absolute position is estimated out of a WLAN fingerprinting, see section 4.2.1. The integration of the PDR and WLAN data is done by a Rao-Blackwell particle filter. A Rao-Blackwell filter means that a two-stage approach is used, where in the first stage the height and floor are estimated, and in the second stage the position estimation can be computed, using the current floor plan. Besides the PDR and the WLAN data, the current locomotion and the floor maps of the building are used for the estimation of the three-dimensional coordinates and the heading. For the incorporation of a floor map, the actual floor of a

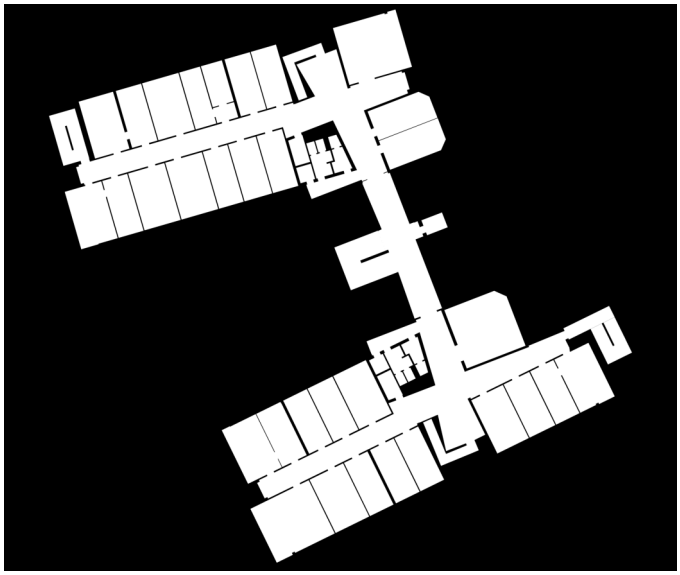

Figure 6.11: Occupancy map for indoor area

building has to be known. Therefore, the height estimation is realized by a Kalman filter, which provides as a result the correct floor map for the two-dimensional position estimation within the particle filter.

Within the particle filter, the appropriate floor map and the determined locomotion provide geometric constraints to reduce the drift in orientation from the propagation and to restrict the invalid variations in the absolute positioning solution.

The particle propagation for the indoor positioning approach is done in an equal fashion as for the outdoor approach, using the PDR data, see equation 6.5. Instead of using GNSS for the absolute position solution, indoors a radio fingerprinting based on WLAN is used. Therefore, the importance weight of the propagated particles is computed using

$$\omega_t^i =< \omega_{WLAN_t}^i, \omega_{map_t}^i > . \tag{6.5}$$

The exemplary results for a trajectory are shown in figure 6.12. Starting on the third floor in the north east corner, the user is walking along the hallway on the third floor. After performing an U-turn, the user is walking on the stairs down to the second floor. On the second floor, the user is walking along the hallway to the west. After performing another U-turn, and walking along the hallway again, the test trajectory ends.

The evaluation of the indoor tests is done using 14 established reference points along the trajectory, which are equally distributed, see figure 6.13 for residuals of the WLAN fingerprinting solution and the results from the particle filter. As visible, the fingerprinting solution is roughly in the range of a room level accuracy. However, the solution is within rooms where the user did not walk. Contrary, the solution computed with the particle filter is semantically correct for the whole trajectory.

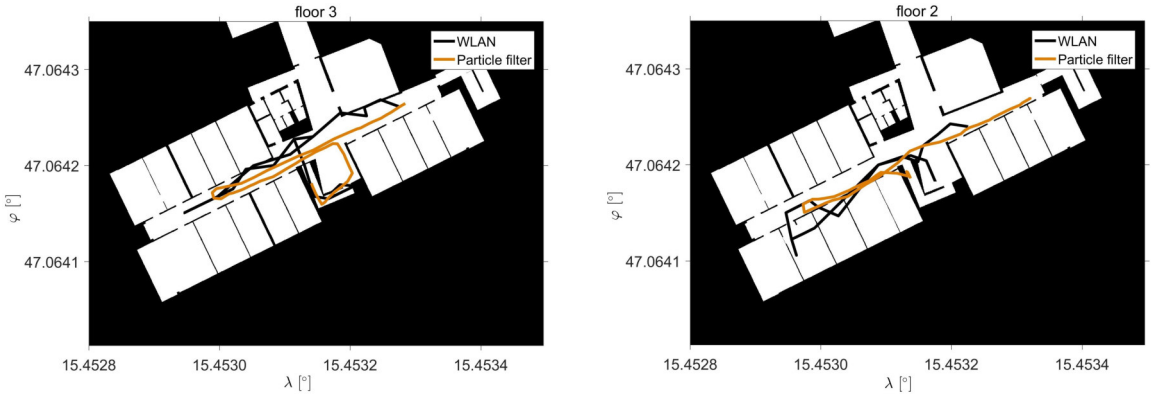

Figure 6.12: Results of test trajectory indoor

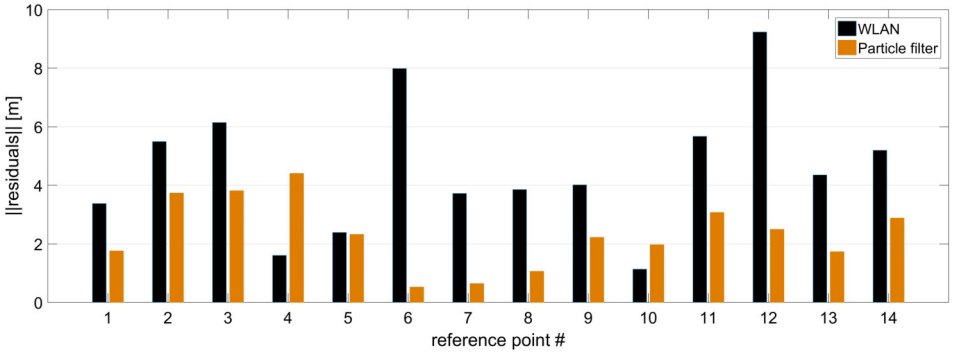

Figure 6.13: Residuals at reference points for test trajectory

As a mean of 5 trajectories of the same test set up, the WLAN fingerprinting-only position has residuals of 4.6 m. Contrary, as a mean, the particle filter solution incorporating PDR and an occupancy map has residuals of 2.3 m.

Note, that without the use of a PDR using locomotion recognition, the implemented particle filter would often fail when transitioning between floors. This is due to the vast overestimation of the propagation, when the step length is estimated wrongly on stairs. Therefore, using a PDR and locomotion recognition is considered to be a critical part of a functioning position filtering approach, for pedestrian positioning indoors.

For an indoor position solution, it is possible to achieve a course, room level position accuracy, using a WLAN or Bluetooth Low Energy (BLE) fingerprinting only. And if a consistent position solution, usable for pedestrian navigation, is needed, possibly a particle filtering of the fingerprinting estimation is sufficient. If more accurate position estimations are needed for a specific application indoors, using smartphone sensors for pedestrian positioning, the incorporation of PDR and locomotion recognition makes it possible to detect the current floor and estimate correct changes of position on stairs. Without the use of PDR and a locomotion information, the shown particle filter approach often fails to detect the current floor, and may not be usable in the presented setup.

6.3 Graph-based pedestrian dead reckoning

Similarly to a raster map, vector maps can be used for supporting a position estimation within pedestrian navigation. With the pixel information of the map, raster maps can easily be used to check the occupancy of an estimated position. Contrary, vector maps consist of node-to-edge relations. One specific vector map, also called routing graph or simply graph, within pedestrian navigation, is a graph which represents the walkable ways for pedestrians. Such graphs are possibly available, or can be created, for outdoor as well as indoor areas.

The main reason, why PDR is such a useful input for pedestrian navigation is, that it is a self-contained, relative positioning approach. However, as shown, for final position estimations, PDR is usually used as input within a filter process, and in combination with a complementary, absolute positioning system. This is due to the problem, that a stand-alone position solution, computed out of a PDR, shows a drift behavior in the position. Mainly, this drift originates in misalignments or unexplained errors within the orientation computation of PDR. One effective approach to overcome this drift in orientation, is to use a graph-based approach, or turn-by-turn navigation.

The main idea of a graph-based navigation is to restrict the estimated trajectory on the known graph network or routing graph. Therefore, dependent on the used approach, the drift in the orientation can be controlled. The obvious disadvantage is, that the estimated trajectory

will always be strictly on the given graph. This approach is already used for applications, where this restriction has no drawbacks, e.g. navigation applications for vehicles which are restricted to certain paths or rails by their nature. An example of a routing graph within an office building is given in figure 6.14. Following, an analysis of a graph-based positioning approach for pedestrian indoors is shown.

Karimi (2011) already presented the idea of utilizing a hallway plan for pedestrians, similar to the idea what we call a graph network or routing graph. In Karimi (2011), the concept of an indoor navigation, using such a hallway plan for map matching is presented. It is proposed, to use a hallway plan to support, e.g., a radio fingerprinting solution. This also enables the positioning approach, to only use relative positioning, without having to know the orientation angle between the device, e.g., the smartphone, and the user, e.g., the pedestrian. Then, Willemsen and Keller (2015) presented a PDR based on a routing graph, using a particle filter for the final implementation. Willemsen and Keller (2015) presented, that using a routing graph in combination with PDR works well in structured buildings, when they lack huge open spaces. Therewith, solutions based on PDR only, up to multiple minutes, can be achieved.

For the presented test site at *Steyrergasse 30* Graz, such a routing graph was modeled and implemented by Reitbauer (2017a). Following, this implementation is used to analyze PDR, and especially the calibration in the orientation estimation.

The routing graph needs, next to the graph network itself, an initial position and initial orientation information. Practically, this can be done using a starting node as well as the assignment of a starting edge. For the presented tests, the starting node is identified using a cell identification approach, utilizing a BLE beacon. For the test set up, only one beacon is installed. For more realistic test beds, BLE beacons can be installed, e.g., at every door frame. If the pedestrian is passing the door frame, the smartphone radio front end will identify a signal strength of this BLE beacon below a threshold, e.g. the typical threshold corresponding to 3 m in distance. Therefore, it can be assumed, that the pedestrian is precisely at the node at the current door frame. For the starting orientation, the smartphone magnetic compass can be used, even using uncalibrated magnetometers. Since in a typical node-to-edge structure of a graph network, differences in orientation of subsequent edges are usually at least 90 °. Therefore, some 10 ° error in orientation still allows for the correct identification of the starting edge.

Based on an identified starting node and starting edge, the graph-based PDR can be computed in a self-contained manner. The flow chart of the graph-based approach is shown in figure 6.15.

The shown processing of the graph-based PDR always refers to an edge. The graph-based PDR algorithms, gets executed with each step estimation. If the position is currently assumed

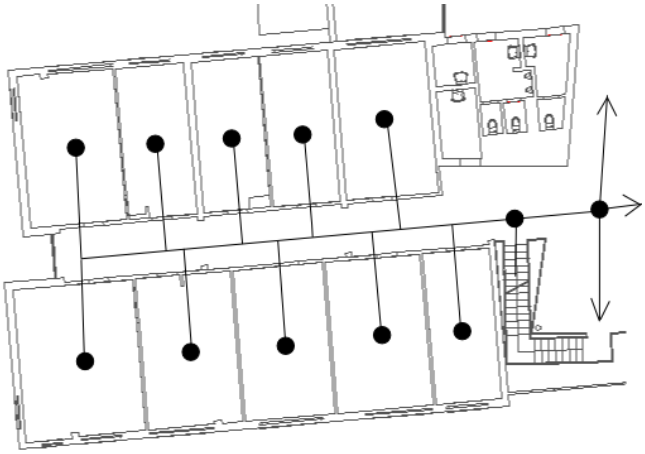

Figure 6.14: Hallway plan with graph network at test bed

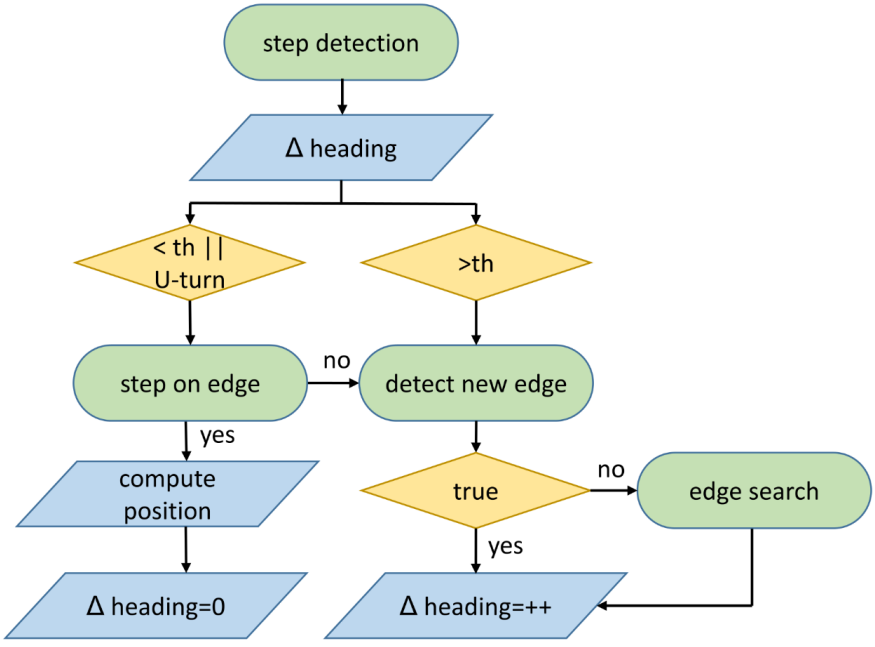

Figure 6.15: Processing of graph-based pedestrian dead reckoning

to be on a certain edge, and a new step is detected, the accumulated heading $\Delta heading$ for this edge is considered.

The main differentiation is, after each detected step, if the pedestrian is still walking along the current edge, or not. Therefore, the $\Delta heading$ is checked against a threshold th, if the movement of the step is still along the current edge, if within the last steps an U-turn was performed on the graph network, or if the current heading does not correspond to the currently assumed edge.

If the step is estimated to be still on the same edge and the orientation stays the same, the current position on the edge, based on the last position and the estimated step length, is computed. If the step is estimated to be on the same edge, but an U-turn was performed, the edge orientation is adapted and the current position on the edge is computed, based on the step length estimation. After computing the current position on the edge, the accumulated heading for the step is reset, with $\Delta heading = 0$.

If the step is estimated not to be on the same edge, or the accumulated heading is exceeding the set threshold, the assumption is, that the current position belongs to a new edge. Therefore, the estimation of the current edge is conducted. If this is true and yields the new edge as a result, the current position can be computed on the new edge, based on the step length estimation, and the current heading can be added to the accumulated heading with $heading++$. If this is not true, the current edge cannot be detected correctly. This may have multiple reasons, e.g. a wrong estimation of the PDR, a wrong estimation within the graph-based approach, or, most likely, that the generalization of the graph network is not corresponding to the actual walked path of the user. If the current edge cannot be detected, an edge search is started.

An edge search tries to identify the current edge, without the use of the past, identified edges. The concept of the edge search is to use the relative PDR, without the graph matching, and cross correlate it to a sequence of edges. This can work in the position domain, but is best done in the orientation domain, e.g. see Wieser et al. (2007). The edge search is started within a defined area around the last correct estimated position, comparing the orientation of the last x meters of the trajectory with the orientation of the last x meters of the possible edge combinations. Since a search usually happens at turns, x can be defined in the range of a couple of edges, so in the range of 10 to 30 meters. If such a search in a small space is not yielding results, a more broad search is conducted. A search within a bigger area of the graph network, or finally within the whole graph of the building is then executed.

How often such a search has to be called, is a quality parameter of how good the relation of the computed PDR solution to the graph-based approach and to the design and generalization of the graph network is. A test example of a short trajectory on a graph network at the test bed is shown in figures 6.16 and 6.17. The trajectory starts in the office in the south

east of the figure, then walks out of the office to the west. Making an U-turn, the user is then walking to the north and shortly to the west again. After another U-turn, the user is walking back to the office, in front of it turning to the east, and returning to the office after another U-turn. The test example is designed to have especially many turns and U-turns, and to cross in the middle next to the stairs, where there is a slightly larger, open space without a hallway restriction. Therefore, multiple edge detection failures should occur on this test. As a base, the implemented PDR is used. Critical for the graph-based approach is the correct accumulation of the heading, as shown in figure 6.15. Therefore, the PDR is computed once using uncalibrated angular rates, as well as using calibrated angular rates for the heading estimation. The results of the graph-based approach, using uncalibrated angular rates is given in figure 6.16, and the result using calibrated angular rates in figure 6.17.

The correctly identified edges, when executing the graph-based approach, are shown in green, where the missed edges are shown in red. When analyzing the solution using the uncalibrated angular rates, six edges are missed in the graph-based PDR. While using the calibrated angular rates, only two edges are missed, and these edges are in the slightly larger, open space, and therefore missed due to the graph generalization. This difference of four missed edges between using the uncalibrated and the calibrated inertial data obviously shows, why the sensor calibration presented in chapter 5 is useful for the application of graph-based approaches. Since the edges are, on average, only a couple of m in length, and thereafter the estimation is reset for the next edge, the calibration of the step length model estimation is not as critical using a graph-based approach.

An analysis of five, longer trajectories at the test bed, which show multiple edge searches, is given in table 6.1. The analyzed trajectories are on average over 500 m in length, and use over 200 edges. From these more than 200 edges, on average an edge search has to be executed 13 times when using uncalibrated inertial data for computing the graph-based PDR, and 10 times when using calibrated angular rates. This is an improvement of 3 edge searches per 500 m trajectory on average, and proves the usefulness of the smartphone sensor calibration.

Table 6.1: Analysis of graph-based pedestrian dead reckoning using sensor calibration

Values as mean of 5 test trajectories	
Detected turns	25
Length [m]	566
Used edges	227
Searches with uncalibrated angular rates	13
Searches with calibrated angular rates	10

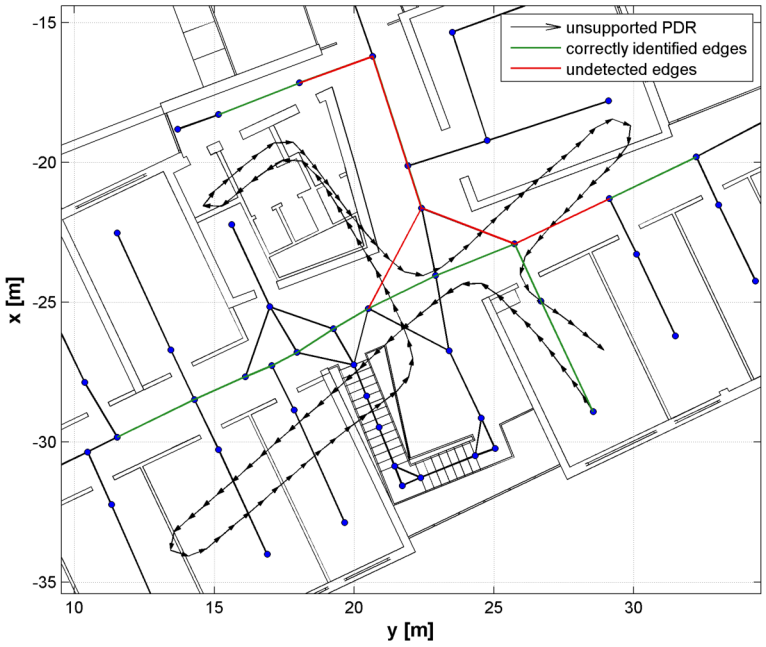

Figure 6.16: Correctly identified edges with uncalibrated angular rates, Moder et al. (2017)

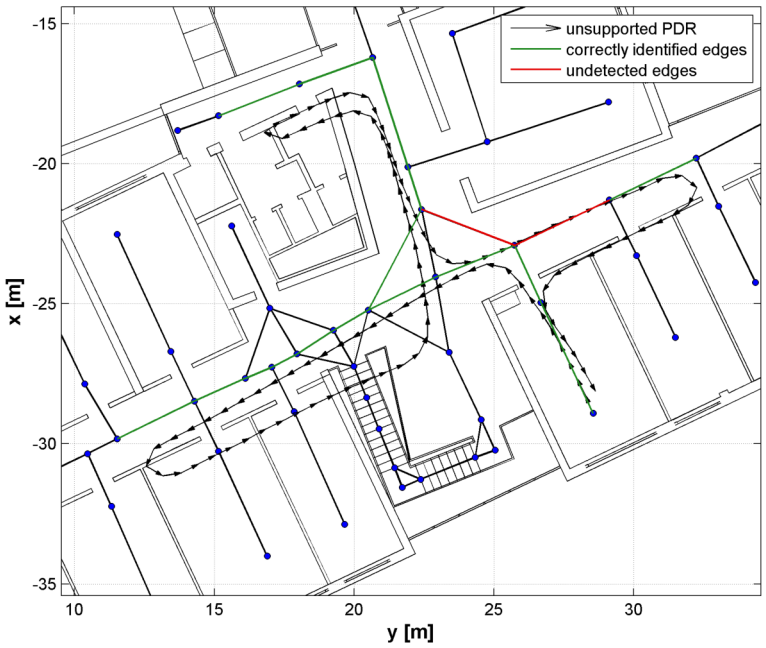

Figure 6.17: Correctly identified edges with calibrated angular rates, Moder et al. (2017)

For all shown trajectories of, on average, more than 500 m in length, the graph-based approach enabled the PDR to be used as a stand alone positioning system for pedestrians. However, the estimated position is strictly restricted to the routing graph used. Therefore, the movement of the pedestrian is strongly generalized. This is useful and works for large office buildings, with long hallways and multiple, smaller offices, like the tested university buildings. But contrary, this can induce further problems, e.g. with large open spaces, like present in public transport hubs. More details and analysis for graph-based filtering, using PDR results, are presented in Reitbauer (2017a).

6.4 Conclusion: pedestrian dead reckoning for navigation applications

The approaches for calibrating the sensors and models usable for PDR are shown in chapter 5. In chapter 6, the application of these calibration approaches is shown, and the better the calibration performances, the better the final position estimation can be. The usefulness is discussed in front of the background of the single phases of an unsupported PDR. Additionally, the usefulness of the calibrations, utilizing the two main approaches of enhancing a PDR solution, namely using a position filter as well as a graph-based approach, is shown.

Analyzing the calibration using a stand alone PDR solution, the improvements of the smartphone sensor calibration are shown for the step detection and step orientation. Further, the usefulness of calibrating the step length parameters and the locomotion recognition model are shown, comparing estimated step lengths for a trajectory walking normal and on stairs in between.

For the positioning approaches, using a particle filter for outdoor and indoor areas is shown. Incorporating an absolute position solution as well as a PDR, a locomotion recognition and a map information, the innovations in the position accuracy are shown. Additionally, especially indoors, where stairs are present continuously, the gain in robustness of a filter approach using PDR in combination with a locomotion recognition is discussed.

Finally, the graph-based positioning approach is shown. This is an approach, which strongly generalizes the walked path of the pedestrian, and therefore can be used with a PDR solution only, without the input of any additional position estimation. Doing so, resets errors in the step length estimation. However, the accumulation of errors in the step orientation can effect the quality of the graph-based approach. It is shown, that a calibration of the smartphone sensors can be used to improve such graph-based PDR approaches.

7 Specific application of pedestrian navigation for people with visual impairments

Pedestrian Dead Reckoning (PDR) can be a useful input in general for a position filter, as shown in chapter 6. More specific, smartphone-based PDR can be used for location-based services, where pedestrians use a smartphone for the service anyway. Usually, PDR is used within applications for pedestrians, and usually indoors due to the fact, that the achievable position accuracy outdoors is already at a high level with multiple Global Navigation Satellite Systems (GNSS) being available today.

Location-based services for pedestrians can be, for example, application using augmented reality, guidance within museums where the applications acts as presentation tool, or, for example, as a positioning and navigation tool at flight terminals or hospitals.

Specifically one group of people, where PDR might be useful for, are people with visual impairments. Being visually impaired, has no influence on the mobility of the user. However, due to the public transport system available, the mobility is restricted if a person is visually impaired. So, especially for people with visual impairments, the knowledge of the current position and navigation information can be a key to overcome current restrictions.

7.1 Application design and positioning results

The application presented is developed to be a positioning and navigation application, for use within transits in public transport hubs, specifically designed for the use of people with visual impairments. The main application is already shown in section 3.3, where the main use cases are outlined. The application was developed within the research project *INK*, lead by the Working Group Navigation of the Institute of Geodesy, Graz University of Technology. The shown overall application is developed with inputs from the whole project team within this research project, where the positioning service is solely developed from the Institute of Geodesy, Graz University of Technology. After outlining the use cases, the focus is to show the importance of PDR for the navigation of visually impaired people in public

transport hubs. An overview of the application, as well as other details next to the positioning, e.g. the navigation and guidance of visually impaired people, is shown in Moder et al. (2018).

As discussed in section 6.3, the graph-based PDR approach is getting rid of the most critical part of PDR, the drift in the heading domain. Note, that this statement is only true, when the step detection is performed robustly. Else, a failing step detection might be the most critical part in a PDR. However, the drawback of graph-based PDR is, that it forces the estimated trajectory on the defined graph. This assumption is true for some examples, e.g. buildings with long floors und multiple, small rooms, like office buildings. This is not true, for buildings with large, open spaces, like frequently common in public buildings, e.g. in public transport hubs.

When designing the application for navigating within a public transport hub, for visually impaired people, a survey amongst the users was conducted. This survey showed, that the tactile paths are used whenever possible and especially if the user is not familiar with the building. The main advantage of public buildings with large, open spaces, is the availability of such tactile paths, especially designed to be used for people with visually impairments using the tactile feedback through their shoes, or using white canes. The possible use of these tactile paths, eliminates two problems of using a graph-based PDR: first, the creation of the routing graph in large, open spaces, where the approximation of actual walkable ways is not obvious, and second, the forcing of the position estimation on these tactile paths, which if communicated to the user, has no disadvantage for people with visual impairments.

The main cases of the developed application are:

- time table request, routing and navigation
- audiovisual aiding
- vehicle communication

Developed as service, and running in the background, is the positioning service. The positioning service estimates the current position, outdoors as well as indoors, of the user at the public transport hub, and acts as input and base for all mentioned use cases. The test bed of the main railway station in Graz, where all presented tests were conducted, is shown in figure 7.1. The figure includes the main train platform and the two main test trajectories, leading either to the bus stations or the city trains in an underground tunnel, where a detailed description of the test bed is given later.

The **time table request and navigation** is the main view of the application. It is possible to view a possible connection between a starting point and end point, or to use the current position as starting point. The time table request then uses all available modes of public transport, e.g. trains, buses, city trains, to compute the optimal route. The route is presented

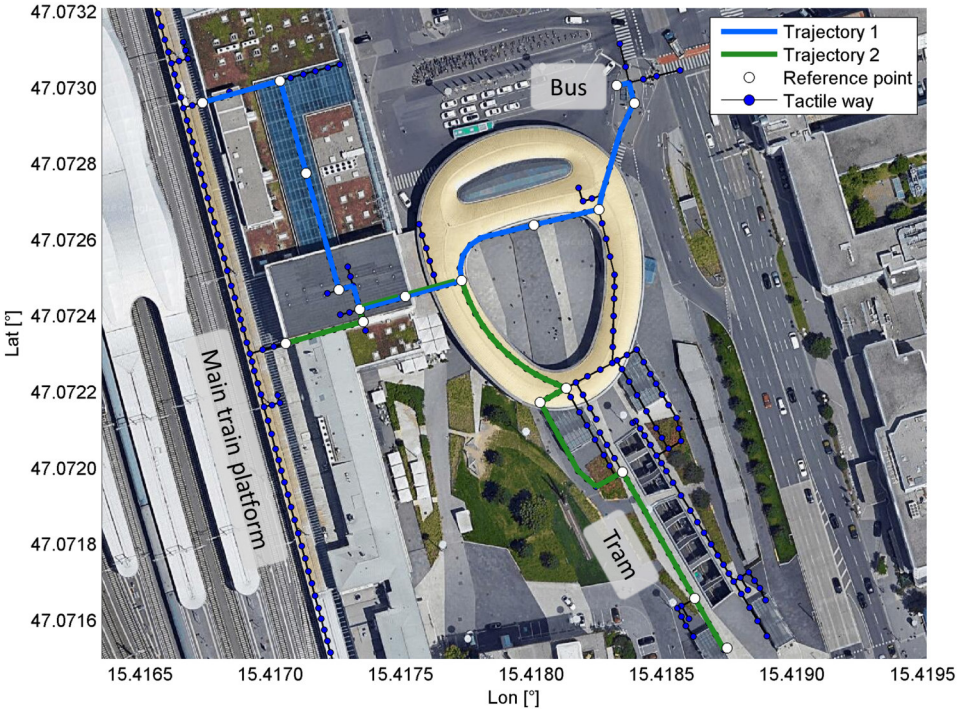

Figure 7.1: Test trajectories at the test bed main train station Graz, Moder et al. (2018)

to the user in a list, with each mode of transportation, and the transits between the modes of transportation, e.g. the transit from exiting a train to entering a bus or city train. While the routing and guidance of the navigation runs consistently, it is especially designed for these transits. During a transit between modes of transportation, the guidance uses the estimated position, and guides the user on the tactile paths from the exit of one mode of transportation, to the entrance of the other mode of transportation.

The **audiovisual aiding** is designed as backup option, if the developed navigation fails, or to help with neglected problems. The concept is, to have an operator, e.g. a relative, friend, or professional operator, who will be able to see the smartphone camera and hear the microphone of the person who activated the audiovisual aiding. Additionally, the planned route and current position can be viewed by the operator. Using this information, the operator is able to see the estimated position and planned route, but equally the camera view of the smartphone. If the user holds the smartphone camera into certain directions, the operator is able to identify the surroundings, and can identify the problem present within the automated navigation. Additionally, changes in the time table, acquiring tickets or similar tasks are solvable with this audiovisual aiding.

Within the **vehicle communication**, it is possible to communicate with the vehicles of transport through a defined interface. For example, if a user is navigated, the request to enter a certain vehicle can be set, which then is visible to the operator of the vehicle. Contrary, when the user is riding with a vehicle, the user can set the request to exit the vehicle at a

certain station, or request at which station the vehicle currently is. Doing so, enables the operator of the vehicle to have more information about the passenger, and, for example, stop exactly at the tactile path system with the front door of the vehicle.

For all these views, the current position, estimated out of the developed **position service** is used. The position service estimates the current position out of a combination of the position estimation implemented in *Android*, called *Location Service*, and a graph-based PDR, tailored to use tactile paths. The position estimation directly from the *Location Service*, uses GNSS if available, but also uses radio signals if possible. The practical implementation is not published by *Google*, however, the position estimation is similar to a GNSS only position outdoors. Therefore, the *Location Service* position estimation is used as a replacement for a GNSS estimation outdoors, which uses less battery. The graph-based PDR used, is essentially a graph-based PDR as shown in section 6.3, using the tactile path network as graph. At the test bed, no installation of additional infrastructure was allowed, however, the temporary placement of Bluetooth Low Energy (BLE) beacons was possible. So, overall about 20 BLE beacons were placed at door frames at the test bed. The BLE beacons were used in cell identification mode, and nodes of the graph were defined directly at the door frames, where beacons were present. Therefore, the position of the graph-based PDR can be enhanced, using the cell identification with the BLE beacons. The problem of restricting the estimated position to the graph is solved by introducing a plausibility test. With this plausibility test, the enhanced graph-based position is based on the estimated accuracy, compared to the position estimation from *Android*. If, based on the accuracy, the *Location Service* position is more likely, the restriction of the position estimation to the routing graph is canceled. Practically, this mostly leads to an distinction between indoor and outdoor. Where indoor, the graph-based estimation is robust, outdoors, the accuracy of the *Location Service* position estimation, based strongly on GNSS, is often preferred.

For a test bed, it was possible to use the main railway station in Graz. The main railway station consists of nine train platforms in the west, with one main train platform, a transit hall, a covered outdoor transit area, an area for local buses in the north east and an underground area for city trains in the south east, containing four stations inbound and outbound. The tactile path network at the whole area was mapped, and input into the open street map. From there, it is possible to call that graph network, and use it for the position service as well as for the navigation process. The position service was set up, using the mentioned BLE beacons for support, the graph-based PDR using the tactile paths as well as the *Android Location Service*. For distinct tests, two main trajectories are defined. Both trajectories start at the exit of the train platform, assuming that the user arrived with a train an the main station. Trajectory 1 then goes through the indoor area, the transit hall, into the covered outdoor area, and ends at the station for one specific, local bus. Trajectory 2 as well goes through the indoor area and transit hall, enters the underground tunnels, ending at the station for the city trains, bound to the inner city of Graz. For an overview of the trajectories see figure 7.1. The trajectories were walked by a blind user, using the application, which was

set up to log the data for processing in real time. Each trajectory was executed five times in the same fashion, and analyzed using the reference points shown in figure 7.1. The residuals, analyzed at 11 respectively 10 reference points, of the five test runs per trajectory are shown in figure 7.2, respectively figure 7.3.

Overall, position residuals in the range of 5 to 10 m are present when using a graph-based PDR. The advantage is, that the solution is, in itself, robust. However, in the results presented, some blunders are visible. This can occur, when a turn is either missed or estimated wrongly. Note, that such blunders occur most often with the presented approach, if the plausibility test suggests to use the *Android Location Service*, because it suggests a better accuracy compared to the graph-based PDR, but the actual position is still indoors. If this happens, the actual solution might be worse than the graph-based PDR, but will be used in the implemented position service. The residuals, as a mean over all all reference points, for the trajectory leading to the bus station as well as to the city trains and for the five executed runs, is given in table 7.1. Over all tests, using the graph-based PDR, a mean 7.7 m is observed at the reference points. However, more importantly, the trajectory always stays on the tactile paths, and is therefore always usable for guidance on these tactile paths, which is the desired result for this application.

Table 7.1: Analysis of trajectories to city train and bus station, residuals as mean of all reference points [m]

Run	Bus	City train
1	5.1	6.6
2	6.5	3.9
3	5.9	6.3
4	14.9	10.3
5	6.9	11.2
mean	7.8	7.6

7.2 Demonstration results and conclusion

In April 2018, there was the possibility to demonstrate the presented application in real time, at the test bed at the main train station in Graz. Multiple test users, from three different societies of visually impaired people in Austria, helped performing the tests.

The tests were structured into testing the time table request and receiving a transit route. These routes were tested from the users, if correct position estimations and guidance instructions were received. The audiovisual aiding was tested, by setting up an operator, and

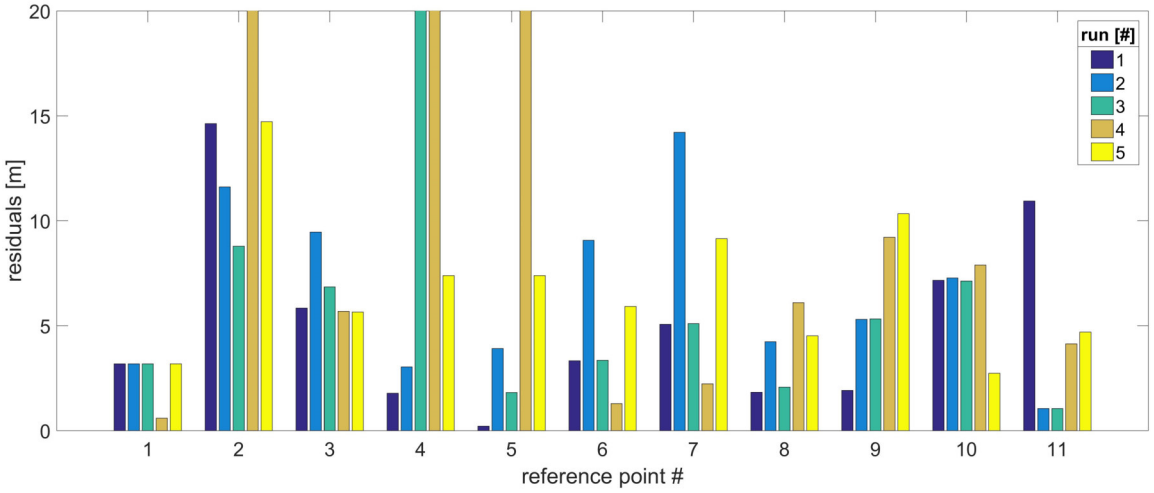

Figure 7.2: Analysis of test trajectory 1 to bus station

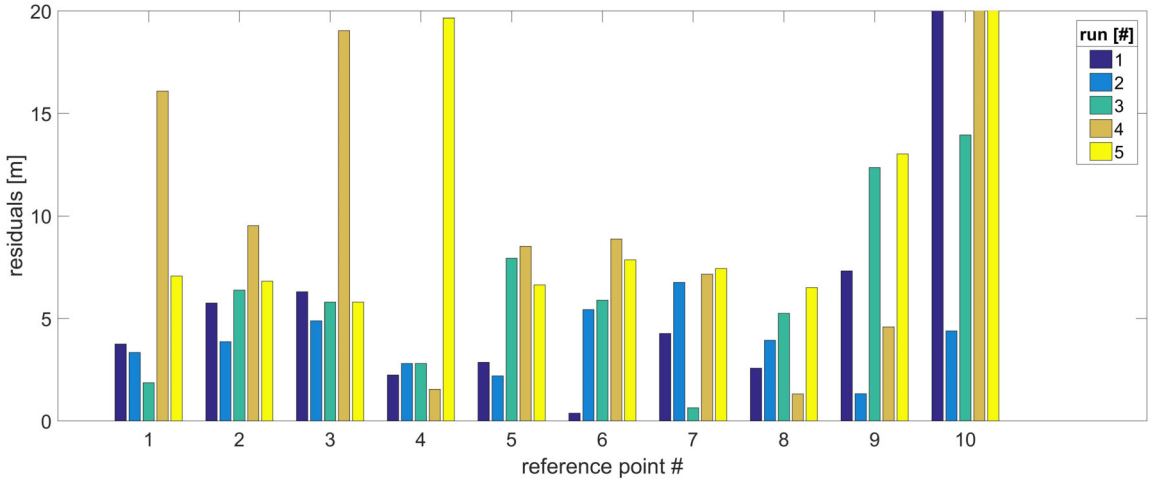

Figure 7.3: Analysis of test trajectory 2 to city train station

the vehicle communication was used, by using a local bus, running the implemented software needed at the board computer within the bus.

The user feedback regarding the overall application and the specific use cases was positive. However, the accessibility of some features is not ideal. Even though *screen reader* or *voice over* implementations help for accessibility, the feedback is to focus more on an easily accessible user design, when creating an update for the application.

The time table request and navigation view was tested, executing multiple requests, and testing the guidance from multiple requests, at the test bed. For visualizing and logging the positioning service, as well as the navigation, an additional test application was created for the demonstration, see figure 7.4 for the application view. This demonstration application shows the smartphone orientation, the estimated accuracy, the location provider as well as the guidance information. The location provider can be the graph-based PDR, a BLE cell identification update, or the GNSS-based *Location Service*. Using this demonstration application, allowed for testing the modes of positioning, as well as the guidance on the tactile path network. This was done thoroughly through the team of the Institute of Geodesy. Within the tests, the application was able to derive a position estimation for all the tested trajectories, with mean position residuals in the range of 3 m, and the guidance instructions on the tactile path network were always consistent and true. For more details on the navigation and guidance, see Moder et al. (2018). Within the tests with visually impaired users, the position estimation showed mean residuals in the range of 7 m, so similar to the rigorous tests from the blind user. The difference is due to the different gait patterns, the change of smartphone mounting from the users as well as the use of white canes of the test users. However, all guidance tests were executed consistent and true.

Two main scenarios were defined for testing the audiovisual aiding. On the one hand, a user pretended to be lost and had to receive navigational aid from the operator. The operator would tell the user to orientate the smartphone camera into a certain orientation, and based on the camera view and the map data, would identify the current position of the user and also give navigation inputs to the user. And on the other hand, a user wanted additional information. For example, acquiring a train ticket, or reading the local arrival and departure time table, including possible delays. Both scenarios were solvable by the operators, although all used smartphones had wide angle cameras as their main cameras, where it was difficult to identify features further away, e.g. the arrival and departure time table.

Testing the vehicle communication, a local bus was used. For the application, an interface containing multiple possibilities to communicate between the application and the board processor of the public vehicles was developed. For the demonstration, the setting of boarding and de-boarding requests between test users and the local bus was tested. First, a boarding request, when the user boarded the bus was set, and received by the bus driver. While driving, a

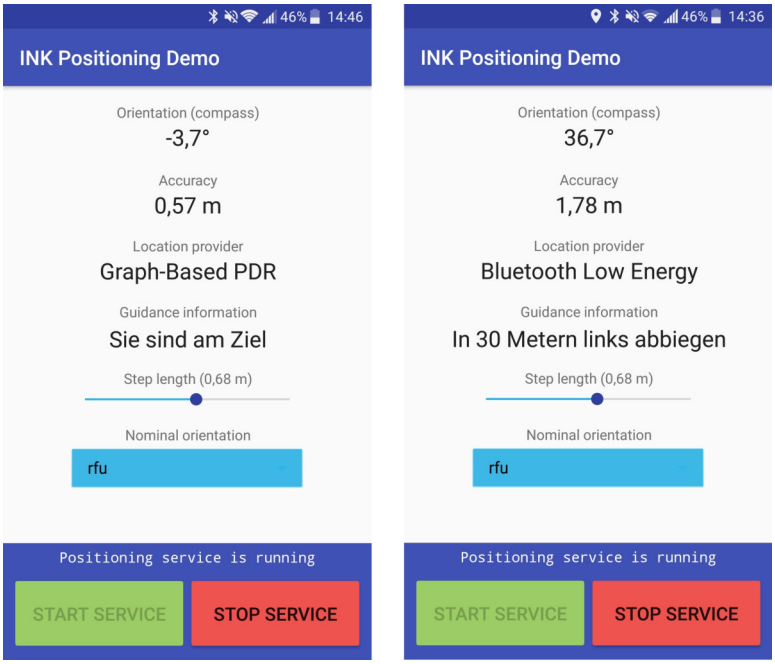

Figure 7.4: Screenshot of demonstration application views

de-boarding request from the user for a specific station was set, and again, received by the bus driver, including the profile of the user, e.g., being visually impaired.

The use of graph-based PDR on tactile paths using smartphones can enable positioning and navigation specifically for visually impaired users. The implemented modules could demonstrate to work overall, and worked very stable for the positioning as well as guidance, if the users applied the instruction of staying on the tactile path network. As of today, where external systems are available to act as navigation tool or communication tool to the vehicles of public transport, the presented application can be used for free with any commercial available smartphone. Therefore, the presented application can be used within the community as socially fair approach, to further enable mobility for visually impaired users.

8 Conclusion

Concluding, a short summary and conclusion of the presented chapters, outlining the basics of pedestrian positioning, discussing the developed calibration approaches, and showing their application, is given. Finalizing, an outlook on possible integration of the presented approaches, as well as further developments are discussed.

8.1 Summary and conclusion

The presented work shows the possibilities of positioning and navigation for pedestrians, using smartphones as device. Therefore, only sensors already presented within smartphones are used. Equally, only signals already available, signals of opportunity, are used for the shown positioning approaches. Starting, the basics of pedestrian navigation, including the basic principles of reference frames, the concepts of the used sensors as well as of the used filtering approaches are presented. Then, the smartphones and currently present and usable sensors are discussed. Following, the main chapters of this work discuss the positioning and calibration approaches for pedestrian navigation, using smartphones.

Within chapter 4, the positioning modes possible for estimating positions of the smartphone, are given. Here, Global Navigation Satellite Systems (GNSS) are the main positioning mode for smartphones. GNSS is worldwide available, free of charge, and accurate and robust enough for multiple use cases. However, GNSS is still subject to signal dilution or loss when indoors, which is true for a huge amount of use cases for pedestrians. Therefore, the possibilities of estimating positions indoors using smartphones are presented. The main possibilities used later are discussed in detail. These possibilities comprise radio location fingerprinting, for retrieving absolute position informations, mainly Pedestrian Dead Reckoning (PDR), for retrieving relative position informations, and additionally, identifying the current locomotion mode.

In chapter 5, the main innovations regarding calibrating the sensors and algorithmic models, used for PDR, are presented. The sensors and algorithms are analyzed scientifically, regarding their need for calibration, and concepts why and when to calibrate them are presented. Additionally, the concept for the automation of the process of calibration is evolved and presented as a novel approach. The focus here is on PDR, as the main, relative position solution, usable in a self-contained fashion utilizing smartphones.

Closing, the presented positioning and calibration approaches are applied in multiple scenarios for pedestrian navigation in chapter 6 and 7. Analyzing the effects of calibrating the sensors and algorithms is done for the distinct events of PDR. Then, the two main approaches of filtering positions utilizing maps are shown for pedestrian navigation. On the one hand, position filtering using a particle filter, using PDR, locomotion recognition and raster maps, is shown. And on the other hand, graph-based positioning, using PDR and a vector-based routing graph, is shown. Lastly, the specific application and usefulness of PDR and graph-based positioning, for supporting people with visual impairments, is shown in chapter 7. Here, an application, where people navigating on tactile paths in public transport hubs, is shown to work with minimal infrastructure, in a robust manner.

The main innovations presented in this work are the applications of the calibration methods for PDR, specifically for pedestrian positioning. The automation of these calibrations and classification are shown. The use of locomotion recognition, as input for PDR, as well as the introduction of a transition probability within a locomotion recognition is presented. And the application of the shown positioning and calibration approaches are shown.

The sensor calibration can efficiently estimate the accelerometer bias errors. This reduces the influence of the measurement errors on the total acceleration, which is used for the step detection, about ten times. Calibrating the magnetometers reduces offsets of the magnetic orientation of up to some 10 °. And the concept for estimating the bias errors in the angular rates, yields in improvements of 2.5 ° for the accumulated orientation. The shown approach of locomotion recognition is shown to work up to 98%, using manually classified data, and up to nearly 95%, using automated classified data. The automated calibration of the step length models, yields in an error of 1.6% of the traveled distance, compared to 5% when using uncalibrated step length models. The presented automated calibration is more efficient for the user than previously published methods. And finally, the incorporation of the detected locomotion into the step length estimation prevents vastly wrong step lengths estimations when walking especially fast, or when walking on stairs.

For pedestrian navigation applications, currently mostly GNSS is used. Mostly, because it is free of charge, available and a robust system. However, GNSS is still not usable, in a satisfying manner, indoors using smartphones. Also, GNSS spoofing and jamming is already and probably will be, a huge problem in the next decade, for smartphone positioning. Therefore, self-contained positioning will be mandatory for many applications. PDR is one of the few, possible approaches for self-contained positioning for pedestrian navigation, using smartphones. PDR, if performed as presented, is only reliant on the inertial sensors of the smartphone, and works outdoors as well as indoors. PDR using smartphones can be done in a very simple fashion, using the inbuilt *Google* step detector, a standardized step length and a magnetic heading estimation, which is possible to derive with an inbuilt function without using any specific calibration. Therefore, a basic PDR can be computed easily and used

within virtually every application for smartphones. There is no published information, of how the position estimation of the *Android Location Service* works exactly. It is only noted, that it uses GNSS as well as other possible solutions. However, based on the characteristics of the derived positions, it is highly assumed, that the *Location Service* already uses a basic version of an PDR approach as well.

In this work, the improvement of incorporating motion recognition, and sensor as well as algorithmic calibration, into the PDR process are shown, which can make the difference for some applications. Additionally, the concept of automation for these approaches is presented. The improvements shown will not be of interest for every scenario, or every application, and might be too complex, or too computationally expensive, for certain applications. But, for some aspects, and some specific applications, the presented approaches might be helpful to improve pedestrian navigation in the domains of accuracy and robustness in the future.

8.2 Outlook

All presented positioning and calibration approaches are designed and implemented in a manner, to work in real time. However, they were tested and analyzed using logged data, or a specific application, and then computed offline or within this application, for easier modification and the possibility of comparison of different approaches. Therefore, the calibration approaches have to be adapted for the implementation in an online service.

Regarding absolute position solutions, the techniques usable are strongly dependent on possible developments within smartphone chips. The main possible developments might be either the implementation of Wide Area Local Network (WLAN) round trip measurements within smartphones, or the development of Ultra Wide Band (UWB) chips for smartphones. Both possibilities would offer more accurate distance operators, leading to more precise absolute position estimations. However, the best method currently available, is still location fingerprinting using signals of opportunity.

Additionally, for all tests with reference measurements and classification measurements, the foot-mounted module was used. Currently, the module is mounted to the foot temporary, using electrical tape and simply taping the module to the foot for the duration of the tests. Again, this approach has to be adapted for an actual, usable online application.

Taping a module to your foot, starting a specific application, walking along while logging data, and then processing this data, seems unapproachable from a user perspective. However, the rise of smart shoes could practically solve this problem.

The rise of smart shoes started around the year 2017. Models with sensors in the heel or insole exist, although no standard is present as of now. The most popular companies known, producing smart shoes, are *Nike* and *Altra*. *Nike* produces and sells smart shoes under the name *Nike+*, which are shoes with inertial sensors, which can be connected to a smartphone. A comparison of the used foot-mounted module, taped to a shoe on the left, and an actual *Nike+* shoe with an implemented module in the insole on the right is shown in figure 8.1. The main advantage of smart shoes, is the possible use of zero velocity updates within the position algorithms.

Therefore, using a smart shoe, while automatically logging the reference events as well as the smartphone data while the user is walking, could be used to efficiently estimate the presented calibration parameters and algorithmic models. Such an application can either be running within a smartphone, or within a smartwatch. Therefore, if the approach is adapted, the automated, personal estimation of the parameters and models, may be done by simply walking with a smart shoe for a certain while. And the estimated parameters, can then be transfered and used within other applications, even when not wearing the smart shoe.

Possible further developments might be using big data and deep learning for the presented approaches. If enough data of gait patterns is present, concepts like transfer learning can be applied, using the base parameters of data sets similar to those of a user, as a first guess or starting point of the calibration. Additionally, machine learning, based on already present, huge data sets, may be more efficient.

Also, the step detection itself can be seen as a basic deep learning problem. The signal, e.g. the total acceleration, can be classified with step events out of a smart shoe, and a deep learning approach may find the optimal estimation for gait detection using smartphones, for the particular user. Therewith, the step detection itself may be personalized as well.

Figure 8.1: Left: Foot-mounted module used, Right: *Nike+* shoe and transmitter

List of Figures

List of Tables

Bibliography

Aboelmagd N., Tashfeen B. and Karamat J. G. (2013). *Fundamentals of Inertial Navigation, Satellite-based Positioning and their Integration*. Springer.

Aggarwal P., Syed Z., Noureldin A. and El-Sheimy N. (2010). *MEMS-Based Integrated Navigation*. Artech House.

Beravs T., Podobnik J. and Marko M. (2012). Three-axial accelerometer calibration using Kalman filter covariance matrix for online estimation of optimal sensor orientation. In *IEEE Transaction on Instrumentations and Measurement, Vol. 61, No. 9*.

Bishop C. (2006). *Pattern Recognition and Machine Learning*. Springer Science+Business Media.

Bobokov D., Grimm F., Steinbach E., Hilsenbeck S. and G. S. (2015). Activity recognition on hand-held devices for pedestrian indoor navigation. In *Proceedings of the 6th International Conference on Indoor Positioning and Indoor Navigation, Banff, Alberta, Canada*.

Chen R. (2012). Introduction to Smart Phone Positioning. In *Ubiquitous Positioning and Mobile Location-Based Services in Smart Phones*. IGI Global.

Chen R. and Guiness R. E. (2014). *Geospatial Computing*. Artech House.

Chen R., Ling P. and Chen Y. (2011). A smart phone based pdr solution for indoor navigation. In *Proceedings of ION GNSS 2011, Portlang, Oregon, USA*.

Chen R., Pei L., Liu J. and Leppokoski H. (2012). WLAN and Bluetooth Positioning in Smart Phones. In R. Chen (Ed.), *Ubiquitous Positioning and Mobile Location-Based Services in Smart Phones*. IGI Global.

Chintalapudi K., Padmanabha I. A. and Padmanabhan V. N. (2010). Indoor localization without the pain. In *Proceedings of the sixteenth annual international conference on Mobile computing and networking, Chicago, Illinois, USA*.

Combettes C. and Renaudin V. (2015). Comparison of Misalignment Estimation Techniques Between Handheld Device and Walking Directions. In *Proceedings of the 6th International Conference on Indoor Positioning and Indoor Navigation, Banff, Alberta, Canada*. Banff, Alberta, Canada. Banff, Alberta, Canada.

Cudjoe D. (2014). Review of generations and physics of cellphone technology. *Inernational Journal of Information Science*.

Dabove P. (2014). What are the actual performances of GNSS positioning using smartphone technology? *GNSS Solutions*, 34–37.

developers.android.com (2018). *Sensors Overview*. Online. Available: `https://developer.android.com/guide/topics/sensors/sensors_overview`.

Diez L. E., Bahillo A., Otim T. and Otequi J. (2018). Step Length Estimation using UWB Technology:

A Priliminary Evaluation. In *Proceedings of the 9th International Conference on Indoor Positioning and Indoor Navigation, Nantes, Frances*.

Ettlinger A. and Retscher G. (2016). Positioning using ambient magnetic field in combination with Wi-Fi and RFID. In *Proceedings of the 7th International Conference on Indoor Positioning and Indoor Navigation, Alcala, Spain*.

Faragher R. and Harle R. (2015). Location fingerprinting with bluetooth low energy beacons. In *IE-EE Journal on Selected Areas in Communications, vol. 33, no. 11*.

Groves P. D. (2013). *Principles of GNSS, Inertial, and Multisensor integrated Navigation Systems*. (2nd ed.). Artech House.

GSA (2015). *Location-Based Services. Excerpt from the GNSS Market Report, Issue 4, 2015.*. URL: `http://www.gsa.europa.eu/sites/default/files/LBS\_0.pdf`.

Hafner P. (2015). *Development of a Positioning Filter for the Smartphone-based Navigation of Visually Impaired People*. Ph.D. thesis, Graz University of Technology.

Hafner P., Moder T., Wieser M. and Bernoulli T. (2013). Evaluation of Smartphone-based Indoor Positioning Using Different Bayes Filters. In *Proceedings of the 4th International Conference on Indoor Positioning and Indoor Navigation, Montbeliard, France*.

Hofmann-Wellenhof B., Lichtenegger H. and Wasle E. (2008). *GNSS - Global Navigation Satellite Systems; GPS, GLONASS, Galileo & more*. Springer-Verlag New York.

Huber K. (2015). *Precise Point Positioning with Ambiguity Resolution for real-time applications*. PhD Thesis Graz University of Technology.

Ivanon P., Raitoharju M. and Piche R. (2018). Kalman-type filters and smoothers for pedestrian dead reckoning. In *Proceedings of the 9th International Conference of Indoor Positioning and Indoor Navigation, Nantes, France*.

Jekeli C. (2000). *Inertial Navigation System with Geodetic Applications*. de Gruyter.

Kang X., Huang B. and Qi G. (2018). A novel walking detection and step counting algorithm using unconstrained smartphones. *Sensors 2018, 18*.

Karimi H. A. (2011). *Universal Navigation on Smartphones*. Springer.

Khalajmehrabadi A., Gatsis N. and Akopian D. (2017). Modern WLAN fingerprinting indoor positioning methods and deployment challenges. *IEEE CommunicationsIEEE CommSurveys & Tutorials, Vol. 19, No. 3*.

Klepeis N., Nelson W., Ott W. R., Robinson J. P., Tsang A. M., Switzer P., Behar J. V., Hern S. C. and Engelmann W. H. (2001). The national human activity pattern survey. *Journal of Exposure Science & Environmental Epidemiology 11*.

Kok M. and Schoen T. (2016). Magnetometer calibration using inertial sensors. In *IEEE Sensors Journal, Vol. 16, No. 14*.

Kushki A., Plataniotis K. and Venetsanopoulos A. (2012). *WLAN Positioning Systems - Principles and Applications in Location-Based Services*. Cambridge University Press.

Labrador M. and Yejas O. (2014). *Human Acitivty Recognition*. CRC Press.

Lara D. and Labrador M. (2013). A Survey on Human Activity Recognition using Wearable Sensors. *IEEE Communications Surveys & Tutorials, Volume 15, Issue 3*.

Lawrence A. (1998). *Modern Inertial Technology - Navigation, Guidance, and Control*. Springer.

Lee J., Shin B., Lee S., Park J., Kim J., Kim C. and Lee T. (2014). A step length estimation based on motion recognition and adaptive gait cognition using as smartphone. In *International Technical Meeting of the ION Satellite Division, Tampa Florida, September*.

Mautz R. (2012). *Indoor Positioning Technologies*. ETH Zürich. Published: Habilitation Thesis.

Moder T., Hafner P., Wisiol K. and Wieser M. (2014). 3d Indoor Positioning with Pedestrian Dead Reckoning and Activity Recognition Based on Bayes Filtering. In *Proceedings of the 5th International Conference on Indoor Positioning and Indoor Navigation, Busan, Korea*.

Moder T., Nilsson J.-O. and Wieser M. (2016). Semi-automated calibration of pedestrian dead reckoning parameters. In *Proceedings of the 7th International Conference on Indoor Positioning and Indoor Navigation, Alcala, Spain*.

Moder T., Reitbauer C., Dorn M. and Wieser M. (2017). Calibration of smartphone sensor data usable for pedestrian dead reckoning. In *Proceedings of the 8th International Conference on Indoor Positioning and Indoor Navigation, Sapporo, Japan*.

Moder T., Reitbauer C., Wisiol K., Wilfinger R. and Wieser (2018). An indoor positioning and navigation application for visually impaired people using public transport. In *Proceedings of the 9th International Conference on Indoor Positioning and Indoor Navigation, Nantes, Frances*.

Moder T., Wisiol K., Hafner P. and Wieser M. (2015). Smartphone-based Indoor Positioning Utilizing Motion Recognition. In *Proceedings of the 6th International Conference on Indoor Positioning and Indoor Navigation, Banff, Alberta, Canada*.

Nguyen P., Akiyama T., Ohashi H., Nakahara G., Yamasaki K. and Hikaru S. (2015). User-friendly activity recognition using SVM classifier and information features. In *Proceedings of the 6th International Conference on Indoor Positioning and Indoor Navigation, Banff, Alberta, Canada*.

Niemeier W. (2008). *Ausgleichsrechnung - Statistische Auswertemethoden*. de Gruyter.

Nilsson J.-O., Gupta A. K. and Händel P. (2014a). Foot-mounted inertial navigation made easy. In *Proceedings of the 5th International Conference on Indoor Positioning and Indoor Navigation, Busan, Korea*.

Nilsson J.-O., Skog I. and P. H. (2014b). Aligning the forces - eliminating the misalignments in IMU arrays. In *IEEE Transaction on Instrumentations and Measurement, Vol. PP, No. 99*.

Öberg T., Karsznia A. and Öberg K. (1993). Basic gait parameters: Reference data for normal subjects, 10-79 years of age. *Journal of Rehabilitation Research and Development, Vol. 30 No. 2*.

Painter J., Kerstetter D. and Jowers S. (1990). Reconciling steadey-state Kalman and alpha-beta filter design. In *IEEE Transactions on aerospace and electronic systems, Vol. 26, No. 6*.

Park S., Heo S. and Park C. (2017). Accelerometer-based smartphone step detection using machine learning technique. In *International Electrical Engineering Congress*.

Reitbauer C. (2017a). *Graphen-basiertes Dead Reckoning für Fussgänger*. Master's thesis, Graz University of Technology.

Reitbauer E. (2017b). *Radio Fingerprinting Optimization Tailored to Vehicles in Parking Garages*. Master's thesis, Graz University of Technology.

Renaudin V., Susi M. and Lachapelle G. (2012). Step length estimation using handheld inertial sensors. *Sensors, Vol. 12, Issue 7*.

Retscher G. and Hofer H. (2017). Wi-fi location fingerprintng using an intelligent checkpoint sequence.

Journal of Applied Geodesy 2017; 11(3):197-205.

Retscher G. and Tatschl T. (2016). Positionierung in Gebäuden mit differenziellem WLAN. *Zeitschrift für Geodäsie, Geoinformation und Landmanagement, 2/2017.*

Robertson P., Frassl M., Angermann M., Doniec M., Julian B. J., Garcia P., Khider M., Lichtensteiner M. and Bruno L. (2013). Simultaneous localization and mapping for pedestrians using distortions of the local magnetic field intensity in large indoor environments. In *Proceedings of the 4th International Conference on Indoor Positioning and Indoor Navigation, Montbeliard, France.*

Ruotsalainen L., Kuusniemi H., Bhuiyan M., Chen L. and Chen R. (2013). A two-dimensional pedestrian navigation solution aided with a visual gyroscope and a visual odometer. *GPS Solutions, Volume 17, Issue 4.*

Sanford J., Potkonjak M. and Slijepcevic S. (2012). *Localization in Wireless Networks.* Springer.

Statista (2015). *Smartphones - Statista-Dossier.* URL: `http://de.statista.com/statistik/daten/studie/198959/umfragen/anzahl-der-smartphonenutzer-in-deutschland-seit-2010/`.

Statista (2018a). *Mobile market share held by the leading smartphone operating ssystem in sales to end users from 1st quarter 2009 to 2nd quarter 2018.* online. URL: `http://de.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/`.

Statista (2018b). *Statista Dossier zu Smartphones.* online. URL: `http://de.statista.com/statistik/studie/id/3179/dokument/smartphones-statista-dossier/`.

Susi M., Renaudin V. and Lachapelle G. (2013). Motion Mode Recognition and Step Detection Algorithms for Mobile Phone Users. *Sensors* 13, 1539 – 1562.

Thrun S., Burgard W. and Fox D. (2006). *Probabilistic Robotics.* MIT Press.

Titterton D. H. and Weston J. L. (2004). *Strapdown Inertial Navigation Technology.* (2nd ed.). MIT Lincoln Laboratory.

Tran N. H. and Young S. S. (2013). Inertial sensor-based two feet motion tracking for gait analysis. *Sensors 2013, 13.*

Weinberg H. (2002). Using the adxl202 in pedometer and personal navigation applications. *Analog devices AN-602 application note.*

Wieser M., Hofmann-Wellenhof B., Mayerhofer B. and Pressl B. (2007). A Navigation Concept for Visually Impaired Pedestrians in an Urban Environment. *Österreichische Zeitschrift für Vermessung & Geoinformation* 2007(2), 159 – 165.

Wilfinger R. (2015). *Absolute positioning of vehicles in indoor environments using Wireless LAN and Bluetooth LE.* Master's thesis, Graz University of Technology.

Willemsen T. and Keller F. (2015). A topological approach with mems in smartphones based on routing-graph. In *International Conference and Workshop on Computing and Communication, Vancouver, BC, Canada.*

Wisiol K. (2014). *Human-Activity Recognition.* Master's thesis, Graz University of Technology.

Zarchan P. and Mussoff H. (2013). *Fundamentals of Kalman Filter: A Practical Approach.* American Institute of Aeronautics, Inc.