Master's Thesis

Sensor Fused Indoor Positioning Using Dual Band WiFi Signal Measurements

Fredrik Karlsson
Martin Karlsson



Department of Electrical and Information Technology,
Faculty of Engineering, LTH, Lund University, January 2014.
In cooperation with Sony Mobile and dept. of Automatic Control.

Sensor Fused Indoor Positioning Using Dual Band WiFi Signal Measurements

Fredrik Karlsson & Martin Karlsson



LUNDS UNIVERSITET

Department of Electrical and Information Technology

Abstract

A ubiquitous and accurate positioning system for mobile devices is of great importance both to business and research due to the large number of applications and services it enables. In most outdoor environments this problem was solved by the introduction of the *Global Positioning System* (GPS). In indoor or suburban areas however, the GPS signals are often too weak to enable a reliable position estimate. Instead, other techniques must be utilized to provide accurate positioning. One of these is trilateration based on WiFi signal strengths. This is an auspicious technology to use partly because of the large number of access points (APs) in our everyday environment, and partly due to the possibility of measuring signal strength with a normal smartphone. The technique is further enabled by the move to include transmitters at 2.4 as well as 5 GHz in modern APs, providing a better basis for accurate position estimations. Furthermore, the motion sensors present in today's smartphones are accurate enough to provide a short-time estimate of the user's movement with high accuracy. In this thesis, both of these technologies are used to develop an accurate method for indoor positioning, and the contributions can be summed up into two points.

The first contribution is an investigation of the behavior of two WiFi frequencies, 2.4 and 5 GHz, where their time dependent noise is proven to be almost uncorrelated with each other. This is then exploited to develop a WiFi-only trilateration algorithm by the use of a particle filter (PF), where the only restriction is that the locations of the APs need to be known.

The second contribution is adding an accelerometer and a gyroscope to the algorithm, to provide a more accurate estimation. A step counter is developed using the accelerometer, and the gyroscope detects changes in heading while the WiFi signal strengths give information about the position. This makes it possible to alongside the position also estimate both heading and step length, while still keeping the only restriction of knowing the AP locations. The resulting algorithm produces position estimates with a mean error less than two meters for a specific use case, and around three meters when a more lenient user behavior is allowed.

Acknowledgements

Several people have contributed with technical expertise, new and alternative ideas, practical solutions and great opinions. Without these, this thesis would never have been created.

First of all, we would like to show appreciation to our advisor at Sony Mobile, Magnus Persson. If not for your help, excellent ideas and feedback along the way, this thesis would not exist. Secondly, our advisors at Lund University, Assoc. Prof. Fredrik Tufvesson and Prof. Bo Bernhardsson, have contributed with great ideas on how to improve our project and advise on topics of a more technical nature. Your help has been most appreciated. Further, Oskar Grönqvist has helped to solve practical problems that have arisen along the way, and supplied improvements to the developed solutions. Finally, Dr. Peter Karlsson deserves gratitude for providing great technical insight, especially concerning radio signal propagation.

To all, thank you!

Contents

1. Introduction **11**
- 1.1 Background . . . 11
- 1.2 Previous Work . . . 12
- 1.3 Goal . . . 13
- 1.4 Limitations . . . 13
- 1.5 Platform . . . 14
- 1.6 Outline . . . 14

2. Background on Navigation **16**
- 2.1 Brief History . . . 16
- 2.2 Coordinate Systems . . . 17
- 2.3 Sensors . . . 20

3. Radio Signal Propagation **23**
- 3.1 Free Space Propagation Model . . . 23
- 3.2 Path Loss . . . 24
- 3.3 Basic Mechanisms of Signal Propagation . . . 25
- 3.4 Indoor Propagation Models . . . 26
- 3.5 Small-Scale Fading . . . 32
- 3.6 Aspects for Positioning Applications . . . 33

4. The Particle Filter **34**
- 4.1 Non-Linear Filtering Problem . . . 34
- 4.2 The Particle Filter Process . . . 35
- 4.3 Computational Aspects . . . 40
- 4.4 Particle Filter for Positioning Applications . . . 41
- 4.5 Other Estimation Techniques . . . 42

5. Introduction to WiFi **43**
- 5.1 The 802.11 Standard . . . 43
- 5.2 WiFi and Positioning . . . 44
- 5.3 Performance of 2.4 GHz Versus 5 GHz . . . 46

6. Indoor Positioning - RSSI Approach **50**
6.1 On Parameter Choices . . . 52
6.2 On Model Choice . . . 56
6.3 Test Setup . . . 60
6.4 Results . . . 60
6.5 Concluding Remarks . . . 68
7. Adapting the Model Parameters **70**
7.1 Estimation of Position Error . . . 70
7.2 Updating the Parameters . . . 72
7.3 Compensating for Signal Blocking . . . 75
7.4 Conclusion . . . 79
8. Modeling the Kinematics **81**
8.1 Random Walk . . . 81
8.2 Quaternions . . . 82
8.3 Characterization of Steps . . . 85
8.4 Dead Reckoning . . . 89
8.5 Dealing With a Non-Constant Sampling Rate . . . 91
8.6 Direction of Movement . . . 93
9. Indoor Positioning - Sensor Fused Approach **98**
9.1 Extension of Particle State Space . . . 98
9.2 Test Setup . . . 99
9.3 Results . . . 100
9.4 Limit the Impact of ”False” Turns . . . 102
9.5 Concluding Remarks . . . 105
10. Algorithm Implementation in Android **106**
10.1 Java Implementation . . . 106
11. Conclusion **112**
11.1 Contributions . . . 112
11.2 Future Work . . . 113
Bibliography **117**
Appendix:
A. **120**
B. **121**

1

Introduction

This Master's thesis investigates possible algorithms for indoor positioning with smartphones. The project is a cooperation between Sony Mobile Communications, the Department of Automatic Control, Lund University, and the Department of Electrical and Information Technology, Lund University. The major part of the work was performed at Sony Mobile, Lund.

1.1 Background

The research area concerning techniques for positioning of handheld devices is of great importance, both from a user and commercial standpoint. Many different services and applications depend on, or could be improved by, a ubiquitous localization system. In addition, the user experience of applications could be aided by including information based on the user's current position.

In most outdoor environments the *Global Navigation Satellite System* (GNSS), of which the *Global Positioning System* (GPS) is a part, provides locations with an accuracy around 15 meters. These types of systems require signal connections to satellites to function, and in obstructed outdoor and indoor areas these signals are often too weak to enable accurate positioning.

Fueled by the lack of accurate positioning in the indoor environment, a multitude of techniques that solve the problem under different constraints has been developed [Mautz, 2012]. Attempts have been made using WiFi received signal strength, inertial sensors, ultrasound etc. requiring different infrastructure and properties of the environment.

The advantages of using WiFi access points (APs) are several. The technique is well established and APs exist in almost every indoor environment, making no further infrastructure investments necessary. Another interesting development is the move to include transmitters at both 2.4 and 5 GHz in modern APs, in best case resulting in two independent measurements from each AP. Further, there is potential for

techniques to locate APs with reasonable accuracy, such as the one presented in e.g. [Hansson and Tufvesson, 2011].

However, WiFi signals are fickle and for an accurate positioning aid from complementary methods are needed. Here, different approaches could be used, and an especially enticing one is inertial sensors, like accelerometers and gyroscopes. Most modern smartphones contain both sensor types along with the possibility to measure WiFi signals at both 2.4 and 5 GHz. Additionally, the market penetration of smartphones is large, making it a suitable platform for a large scale roll out of an indoor positioning application.

1.2 Previous Work

Indoor positioning is a large research area, and various techniques have been proposed. They range from using only one portable device, to attaching several sensors on different parts of the user's body [Morrison et al., 2012].

There are numerous examples where WiFi APs with known locations are used to trilaterate the position of the user. The distance to each AP is often estimated using the signal strength. This can also be done by measuring the time it takes for a signal to travel between the device and an AP, see e.g [Li et al., 2007].

Another method is *pedestrian dead reckoning* (PDR), where sensors such as accelerometer and gyroscope are used to determine the motion of the user. An example of this is presented in [Hansson and Tufvesson, 2011]. Here, the accelerometer was used for counting steps, and the gyroscope was used to determine the change of heading. The step length, and the initial position and heading, were assumed to be known. The device used was a sensor equipped smartphone. Further attempts have been made, where the sensors are attached to certain parts of the user's body. In [Godha and Lachapelle, 2008], a method using a single set of foot mounted sensors is described. This gives the advantage that, since the foot has zero velocity once every step, the sensors can be calibrated, which mitigates drifts. This idea is extended in [Morrison et al., 2012], where arrays of sensors are attached on various body parts, allowing even better error mitigation.

Dead reckoning has also been combined with e.g. WiFi signals into sensor fused approaches. In [Retscher and Kealy, 2005; Retscher and Thienelt, 2004], two different approaches to this are presented using wide ranges of sensors including accelerometer, gyroscope, WiFi, magnetometer, GPS, barometer etc. Both methods are based on an extended Kalman filter to perform the position estimation. The first technique is developed for pedestrians, whereas the other one is geared towards vehicles. Another contribution to this area is presented in [Faragher and Harle, 2013]. Here, PDR is combined with GPS for an initial position estimation, and with magnetic field and radio signal strength fingerprinting. One important feature of this procedure is that

the algorithm is learning the radiation environment with time, and this is used to detect when the user returns to a previously visited area. Then, errors from the PDR are corrected. This method is an a example of *Simultaneous Localization and Mapping* (SLAM), where a map of, in this case, the radiation environment is developed during the positioning [Gustafsson, 2012]. The positioning is in turn aided by the map.

1.3 Goal

The goal of this thesis is to develop, implement and evaluate an algorithm for indoor positioning. The thesis describes the result of using both 2.4 and 5 GHz WiFi signals together with inertial sensors.

In the first part, positioning is done using information from WiFi signals only. This will be combined with a particle filter (PF) to estimate the true position, and this method will be compared to the more conventional method of least squares (LS).

In the second part, the goal is to improve the performance by adding information from the sensors, in order to receive a more accurate model of the user's movement.

When the algorithm is developed, the aim is to implement it as an Android *application programming interface* (API), for use in e.g. SonyMap, an indoor positioning application developed by Sony Mobile.

1.4 Limitations

The positioning application developed requires the positions of the APs used to be known. Furthermore, sufficiently good models of the indoor path loss are needed, which in turn requires information about the power with which the APs transmit, and how dense the environment is with obstacles such as walls and furniture.

In order to include all parts of the algorithm, it is necessary to use a device that can receive both 2.4 and 5 GHz WiFi signals. Moreover, it is assumed to be equipped with both accelerometer and gyroscope.

For optimal performance, the smartphone should be reasonably fixed in relation to the user while the application is used, e.g. held in a fixed orientation or placed in a pocket. However, the algorithm still produces fairly accurate results, even if this condition is not met.

1.5 Platform

In this thesis the following smartphones are used: *Sony Xperia Z Ultra*, *Sony Xperia Z1* and *Sony Xperia LT29i*. Hereafter, they are referred to as *the phone* if nothing else is specified. Each of these phones are running an *Android* operating system and is equipped with an accelerometer, a gyroscope and a WiFi chip capable of communication over both 2.4 and 5 GHz.

Android

Android is an open source operating system used mostly by different handheld devices. In the smartphone segment, Android is used by around 80% of the units in use today (2013) [IDC, 2013]. The Android standard development kit contains the necessary tools and API:s to develop applications for the Android platform using the Java programming language. Using the standard APIs, it is possible to access the device's sensors, take measurements of the WiFi environment etc. All data collection performed during this thesis has been done through Java applications developed by the authors running on the phones mentioned in Section 1.5.

1.6 Outline

This outline presents the large scale structure of this thesis.

Chapter 2 gives a brief introduction to the history of navigation and positioning, and an overview of the coordinate systems used.

Chapter 3 describes the theory of radio signal propagation and how to obtain a good model of the radio channel.

Chapter 4 introduces the particle filter (PF) and gives a description of its process. Further, a brief description of the least squares (LS) estimation is given.

Chapter 5 covers some of the different technologies concerning WiFi and its uses for positioning application. Additionally, the differences between the two separate WiFi frequency bands used are investigated.

Chapter 6 presents a method to obtain a position from WiFi RSSI measurements, and this method is compared with the least squares estimation.

Chapter 7 discusses how the models developed in Chapter 3 could be adapted using received RSSI measurements and an estimated location.

Chapter 8 models the kinematics of walking using different sensors. These sensors are described and a step detection algorithm is presented.

Chapter 9 merges the algorithm introduced in Chapter 6 with sensor data to a sensor fused indoor positioning algorithm using dual band WiFi measurements.

Chapter 10 is dedicated to the Java implementation of the algorithms presented in Chapter 6 and 9 and the specific considerations that was made.

Chapter 11 evaluates the results of the thesis and presents possible future work on the subject.

In accordance with Lund University policy the contribution made by each author must be clear. During the process of creating this thesis, all work has been done in collaboration between the authors, both concerning the process of developing and implementing the algorithms, as well as condensing the work into this report.

2

Background on Navigation

This chapter gives a brief introduction to the background and history of navigation. First, there is an overview of some of the different stages that navigational technology has conquered over the course of history. Then, an introduction to the different coordinate systems used in this thesis will be presented and special emphasis will be put on the system used when evaluating the positioning algorithm. In connection with this, a brief presentation of the *longitude and latitude* coordinate system will be given. Further, a few sensors are presented, including the accelerometer, the gyroscope and the magnetometer.

2.1 Brief History

Mankind has always had a need to navigate in its surroundings. At an early stage these surroundings were fairly small and visual references to known landmarks were good enough to find one's way. As the distances traveled grew, so did the need for a more well defined frame of reference, so mankind turned to celestial bodies. Using the sun and the stars, with their relatively fixed positions in reference to the earth, gave the ability to keep a desired heading over a long distance. This ability was then refined over the centuries and triangulation was discovered as a mean to find one's position. The invention of the compass further improved the navigation. With a known orientation, only the relative angles to two known landmarks are needed to find the position [Hofmann-Wellenhof et al., 2003].

In the 20th century the improvements in measuring accelerations and angular velocities promoted a technology called *dead reckoning* (DR) [Gustafsson, 2012; Figueiras and Frattasi, 2010]. From a known position and heading, it is possible, using an accelerometer and a gyroscope, to estimate current speed and heading to find the current position. With occasional fixes of the position and heading, the current position can be estimated for all times. However, the size and cost of the devices involved make the technology suitable only for larger vehicles, like ships or airplanes. If no heading and position fixes are available, the error grows over time.

Another technological leap was taken with the introduction of the *Global Positioning System* (GPS) which became fully operational in 1995 [Parkinson and Spilker, 1996]. This system uses the time difference between signals received from satellites with known locations, which made it possible for almost anyone to find their position with an accuracy of around 15 meters. Today, GPS receivers are readily available both as standalone devices and integrated in cars or smartphones.

A problem, however, exists with the GPS system. To obtain a good position, signals from at least three (or four, if the altitude should be determined along with the longitude and latitude) satellites must be available. Preferably these satellites should be in line-of-sight (LOS) from the GPS receiver. Putting more satellites in orbit increases the probability that enough satellites are visible, but this only partly solves the problem. In some areas, like a suburban environment or indoors, the paths between the GPS receiver and the satellites are obstructed either by the building it is in or buildings in its surroundings. An obstructed path results in that the GPS signal has to travel a longer distance, i.e. by reflection, diffraction or scattering, to reach the receiver. Because the distance is estimated by the difference in time between when the signal was sent and received, an obstruction increases the estimated distance between receiver and satellite. Further, as the GPS signal is sent from space, it is weak when it reaches the Earth's surface and an obstruction may even make it too weak to be detected by the receiver [Grimes, 2008].

Altogether, this makes GPS a good positioning technique in open outdoor areas, but in a suburban or indoor environment other techniques must be used to obtain accurate positioning. A multitude of techniques that aim to solve this problem with different accuracies and constraints exists. There is no single technology that these attempts are based on. Instead, they are spread over a wide range of areas, including inertial sensors, signal strengths, magnetic fields, cameras etc. [Mautz, 2012].

2.2 Coordinate Systems

It should be apparent to the reader that a well defined coordinate system is of great importance for positioning applications. Further, to be able to receive a meaningful position, the coordinate system needs to be fixed in reference to the Earth. In the sections below, two such coordinate systems are presented along with the coordinate system of the phone.

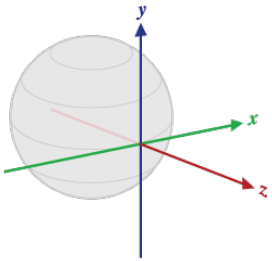


Figure 2.1 World coordinate system.[1]

World Frame

The *world frame* or the global coordinate system is commonly used for navigation, at least in a global perspective. The definition is, in accordance with Figure 2.1 as follows:

x_w**:** Tangential to the ground at the current location, defined as $y_w \times z_w$. Points approximately east.

y_w**:** Tangential to the ground at the current location, pointing towards the magnetic North Pole.

z_w**:** Perpendicular to the ground, pointing skywards.

Noteworthy is that this system is not fixed in space, and different positions on the Earth's surface will result in a differently oriented coordinate systems. Further, as the world rotates around its own axis so does the coordinate system (approximately 15° per hour). In the sense of indoor positioning however, both of these effects can be neglected.

Local Frame

Using the world frame might in some cases be impractical or cumbersome for a certain location. As an example, consider a building with most corridors oriented with an angular offset of ϕ in reference to the global frame. For this location it would be considerably more practical to orient the coordinate system so that most movement is done in the $x-$ and $y-$ directions.

[1] This image is reproduced from work created and shared by the *Android Open Source Community* [Google, 2013] and used according to the terms described in the *Creative Commons 2.5 Attribution License* [Creative Commons, 2013].

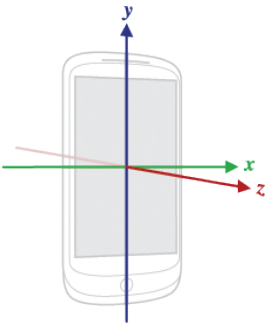


Figure 2.2 Phone coordinate system.[2]

Such a coordinate system is defined as:

x_l**:** Located in the plane defined by x_w and y_w. Pointing as x_w with an angular offset of ϕ.

y_l**:** Located in the plane defined by x_w and y_w. Pointing as y_w with an angular offset of ϕ.

z_l**:** Equivalent to z_w, pointing skywards.

Phone Frame

An unambiguous definition of a coordinate system with a fixed orientation in reference to the phone is crucial. Sensors report their measurements in coordinates that are fixed in reference to the phone.

In Android, the phone coordinate system is defined in the following way (see also Figure 2.2):

x_p**:** Pointing out of the side of the phone, defined as $y_p \times z_p$.

y_p**:** Pointing upward through the top of the screen.

z_p**:** Orthogonal against the phone's screen, pointing up if the phone is placed back down on a table.

Further, the phone's orientation may be expressed as three angles, $(\phi_x \ \ \phi_y \ \ \phi_z)$, representing the difference in orientation between each of the phone's axis and those of the world frame.

[2] This image is reproduced from work created and shared by the *Android Open Source Community* [Google, 2013] and used according to the terms described in the *Creative Commons 2.5 Attribution License* [Creative Commons, 2013].

Coordinate System Used for Measurements

As the measurements in this thesis are taken over a limited area, the *Glasgow* building at Sony Mobile in Lund, a local frame is a convenient choice. The chosen coordinate system is aligned with the building the measurements are taken in, allowing most of the movement to occur parallel to either the x- or y-axis. For convenience, a local frame with an angular offset of roughly 180° is used, giving the system:

x_t**:** Pointing roughly west.

y_t**:** Pointing roughly towards magnetic south.

z_t**:** Equivalent to z_w, pointing skywards.

In Figure A.1 in Appendix A, the chosen coordinate system used is shown along with a floor plan of the building and the different paths used to evaluate the positioning algorithms.

2.3 Sensors

Here follows a brief description of three sensors commonly available in modern smartphones, *the accelerometer, the gyroscope* and *the magnetometer*. A short description of their inner workings, advantages and disadvantages is given along with a description of their use in position estimation. For a deeper description of the sensors and a more extensive discussion about biases and drifts references are given in [Bentley, 1988] and [Morris, 1996].

The Accelerometer

An accelerometer is a sensor measuring acceleration (m/s^2) along one or several axes. The accelerometer measures the acceleration the sensor is subjected to. To exemplify, an accelerometer at rest on the Earths surface measures an acceleration of $g = 9.81 m/s^2$ and an accelerometer in free fall in space will measure zero acceleration. Today, most accelerometers measure acceleration along 3 orthogonal axes, thus giving information about the three dimensional acceleration.

The fact that every object at rest at the Earth experiences the acceleration associated with gravity poses a problem when using the accelerometer, since the gravity always will be present in the measurements. This needs to be managed in some fashion. The simplest is to keep the accelerometer fixed in reference to the Earth's coordinate system introducing gravity as a constant bias along a certain direction, which is easy to subtract. A more challenging case is when the accelerometer changes its orientation compared to the Earth's coordinate system. Here, an algorithm to keep track of the accelerometer's orientation is needed if the effect of gravity is going to be deductible.

Another interesting aspect of gravity in relation to the accelerometer is the ability of finding out the accelerometer's orientation in reference to the Earth's coordinate system. This is possible if the accelerometer does not experience any significant acceleration except for gravity. Then the accelerometer's orientation may be determined by the component of gravity present on each of its axis.

In principle the accelerometer could be used to estimate a position by integrating its output twice in respect to time. In practice, however, this is hard as the gravity needs to be accurately subtracted from the measurements if not to introduce a large positioning error. If just 1 % of gravity is left in the measurements it will introduce an error of 500 meters over 100 seconds. Further, the accelerometers present in today's smartphones suffer from biases and drifts which need to be accurately evaluated if not to further degrade the position estimate.

The Gyroscope

A gyroscope measures the *angular velocities* with which it is turning around each of its axes. Most gyroscopes use three orthogonal axes giving measurements of the angular velocities on the form $\vec{\omega} = (\omega_x \;\; \omega_y \;\; \omega_z)$. Mechanically, a gyroscope consists of a spinning wheel or disc mounted in a gimbal, a pivoted support that allows the disc to rotate freely about all three axes. This construction allows the disc to keep an almost fixed position in reference to the mounting platform's motion. However, this is a fairly large device not well suited for modern electronic devices. These instead rely on *MEMS gyroscopes* which can be fashioned into a more suitable size.

As the MEMS gyroscope measures angular velocities and, in this thesis, the absolute change in angle is of interest. A method to compute the absolute change from the velocities are needed. Given two sets of angular velocities measured with a time difference of Δt, the angular change, θ, may be computed using numerical integration according to the *trapezoidal rule* [Adams and Essex, 2010],

$$\theta = \Delta t \frac{\vec{\omega}_t + \vec{\omega}_{t+\Delta t}}{2}. \tag{2.1}$$

This can, of course, be applied to each of the gyroscope's axis, resulting in an estimation of how the gyroscope is turning and in extension, its orientation, in three dimensions.

There are a few problems of using the gyroscope to determine the direction of movement. As it only measures how the angles change over time, the direction at the start of the navigation must be known and the orientation needs to be kept fixed in reference to the movement. For example, it is impossible do discern between a 90° turn of the user and a mere turn of the gyroscope. Furthermore, the gyroscope suffers from drifts and biases resulting in a growing heading error. Hence, if it is to be used for long stretches of time, the heading needs to be calibrated.

The Magnetometer

A way around the problems of a known initial orientation is the use of a magnetometer, which measures the *magnetic field strength* in each of the phone's three coordinate directions. Using the known electromagnetic field produced by the Earth's core, it is possible to find the phone's heading.

There are a multitude of ways to measure the magnetic field strength. In most modern smartphones a measurement of the resistivity of a thin strip of magnetic film is used. The sensor contains a thin strip on permalloy (NiFe magnetic film) whose resistance is proportional to the magnetic field strength.

In theory the magnetometer along with a known phone orientation in reference to the Earth's coordinate system could be used to give an accurate estimation of the heading of the user. This is possible in environments where the electromagnetic fields, apart from the Earth's own, are weak, e. g. outdoors. In most indoor environments however, fields from electronic devices and structural elements (metal beams, pipes etc.) produce their own magnetic fields. These fields are in many cases strong enough to interfere or even overpower the Earth's, causing the estimated heading to be wrong.

The *fingerprinting* strategy for signal strengths described in Section 3.6, may also be utilized to characterize the magnetic field strengths in an indoor environment. These measurements can then be used to find the true heading, but this approach lacks generality as measurements at each location are needed.

3

Radio Signal Propagation

The performance of any wireless communication system is fundamentally limited by the properties of the radio signals. In the simplest case, these properties and hence the signal intensities would depend only on the distance between transmitter and receiver. However, they vary greatly depending on the environment, from a simple line-of-sight (LOS) case to a severely obstructed one, where walls, windows, furniture etc. distort the signal between transmitter and receiver. Furthermore, the signal is affected by a large number of small-scale effects. Among them are reflections from various surfaces, diffraction and Doppler shift due to a difference in speed between the receiver in reference to the transmitter [Rappaport, 1996]. In this thesis the focus is put on obtaining good models for the indoor propagation of radio signals in the range $[2.4;5]$ GHz for application in indoor positioning.

This chapter consists of a recapitulation of free space signal propagation and path loss, together with different ways of modeling these. Different sources of disturbances and their influences are discussed.

3.1 Free Space Propagation Model

How the magnitude of radio signals decay with increasing transmitter-receiver (T-R) separation is of interest in many applications, especially if a position is to be obtained from the received signal strength. The free space propagation model is a relatively simple model of the received signal power given a certain T-R separation and a LOS path in between. This model focuses on the large-scale propagation features, thus predicting the average signal strength received without small scale effects taken into account. As in most propagation models a power law function of how the signal intensity decays by distance is assumed $\left(\sim 1/d^2\right)$ [Rappaport, 1996]. The power received at a distance d from a radiating transmitter is given by the Friis free space equation,

$$P_r(d) = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^2 L} \tag{3.1}$$

where $P_r(d)$ is the received power in dBm at a T-R separation of d meters, P_t is the transmitted power in dBm, G_t and G_r are the gains of the transmitter and receiver antennas respectively, $L \geq 1$ is the system loss factor which is not related to propagation, i.e hardware losses, and λ is the wavelength of the transmitted signal. The gains (G_t and G_r) are dimensionless constants related to the antennas' effectiveness of receiving and transmitting a signal which is related to their construction and physical properties. The wavelength λ is related to the frequency of the signal by

$$\lambda = \frac{c}{f} = \frac{2\pi c}{\omega_c}, \tag{3.2}$$

where f is the signal frequency in Hertz, ω_c is the frequency in radians per second and c is the speed of light in meters per second.

The Friis equation can only be used as a predictor for P_r when d is in the far-field, or *Fraunhofer region*, of the transmitting antenna [Rappaport, 1996]. The far-field region of a transmitter is defined as $d > d_f$ where d_f is the *Fraunhofer* distance defined as

$$d_f = \frac{2D^2}{\lambda} \tag{3.3}$$

where D is the largest physical dimension of the transmitting antenna.

Further, to be in the far-field region

$$d \gg D \text{ and } d \gg \lambda \tag{3.4}$$

musty hold.

For a radio signal with frequency ranging from 2.4 to 5 GHz (WiFi) and a largest antenna dimension D of 0.1 m, d_f is in the range of $[0.36, 0.75]$ and a $d > 1$ m fulfills both (3.3) and the additional requirements from (3.4).

3.2 Path Loss

The *path loss* is defined as the difference in dB between the transmitted and received power and represents the signal attenuation, measured in a positive quantity of dB. If the antennas are assumed to have unity gain, i.e they are *isotropic* radiators, the path loss (PL) can be obtained from (3.1) as [Rappaport, 1996],

$$PL(\text{dB}) = 10\log_{10}\frac{P_t}{P_r} = -10\log_{10}\left[\frac{\lambda^2}{(4\pi)^2 d^2}\right]. \tag{3.5}$$

It is obvious that (3.1), and thus (3.5), does not hold for $d = 0$. To get around this, large-scale fading models use a fixed distance d_0 with known power [Rappaport, 1996]. At d_0 the power can either be measured or estimated from (3.1). The distance d_0 needs to be in the far-field region of the transmitter, as defined in (3.3). Furthermore, if d_0 is chosen to be smaller than any distance d used in the application, the power at an arbitrary distance may be related to the received power at d_0 by,

$$P_r(d) = P_r(d_o)\left(\frac{d_0}{d}\right)^2 \qquad d \geq d_0 \geq d_f. \tag{3.6}$$

For practical reasons, as most formulas express the power in dB, d_0 is chosen to result in a power of 10 dB. In outdoor environments a common value is 100 meters, and in indoor environments, 1 meter is suitable. Using the latter, the power can be expressed as

$$10\log_{10} P_r(d) = 10\log_{10} P_r(d_0) - 20\log_{10} d \qquad d \geq d_0 \geq d_f, \tag{3.7}$$

where $P_r(d_0)$ can be determined by a simple measurement at 1 meter from the transmitter. The resulting model represents the indoor LOS path loss case.

3.3 Basic Mechanisms of Signal Propagation

In this section the three basic mechanisms for signal propagation, reflection, diffraction and scattering, are presented together with their respective impact on the propagation model. The presentation is a rather brief recap of Sections 3.5 through 3.8 in [Rappaport, 1996], and interested readers are referred there for a more in-depth description.

Reflection

When a radio wave propagating through one medium encounters a medium with a different set of electrical properties, the wave is partially transmitted and partially reflected. The intensities of the transmitted and reflected waves may be related to each other through the *Fresnel reflection coefficient* (Γ). This coefficient is dependent on the material properties, wave polarization, angle of incident and the frequency of the wave [Rappaport, 1996].

The requirement of a surface to be considered as a possible source of reflection is that its dimensions are much larger than the wavelength of the incident wave and typical indoor sources include walls, floor, ceiling etc. The reflections cause the signal strength to be larger in some areas of the room and smaller in others than the free space model predicts.

Diffraction

Diffraction is the property that allows waves to propagate around different obstructions, e.g. propagate around corners, travel beyond the horizon or behind obstructions. Huygens principle states

> *All points on a wavefront can be considered as point sources for the new wavelet* [Rappaport, 1996].

Diffraction is caused by the propagation of a secondary wave into a shadowed region, and it can be viewed as the wave ”bending” around the edge of objects. Because only a fraction of the wave propagates into the shadowed region, the signal strength decays rapidly when moving further into it [Rappaport, 1996].

Scattering

Scattering occurs when a wave travels through an environment which has a large density of objects with small dimensions compared to the wavelength. The energy of a scattered wave is spread out in all directions from the scattering surface. Even surfaces considered flat usually possess some roughness and thus some scattering properties.

3.4 Indoor Propagation Models

The Friis free space equation together with the reflection, diffraction and scattering have spawned many different models for indoor signal propagation of a wide range of complexity.

Log-Normal Shadowing

One of the more simple models is the log-distance path loss model

$$\log_{10}(P_r(d)) = \log_{10}(P_r(d_0)) - 10n\log_{10}\left[\frac{d}{d_0}\right] + X_\sigma \tag{3.8}$$

where n is the path loss exponent and X_σ is a zero-mean Gaussian distributed random variable with standard deviation σ (both in dB). For some typical values of n for different types of environments, see Table 3.1. This model allows some tuning for a specific environment using different values of n. In Figure 3.1 and 3.2, measured values of the signal strength are compared to predicted ones using the log-normal shadowing model with $n = 2$ and $n = 3.2$ respectively, using a measured value of $\log_{10}(P_r(d_0)) = -30$.The measurements were taken while following the distance trajectory to the transmitting antenna shown in Figure 3.3. The environment in which the measurements were taken is an open office space with some obstructions.

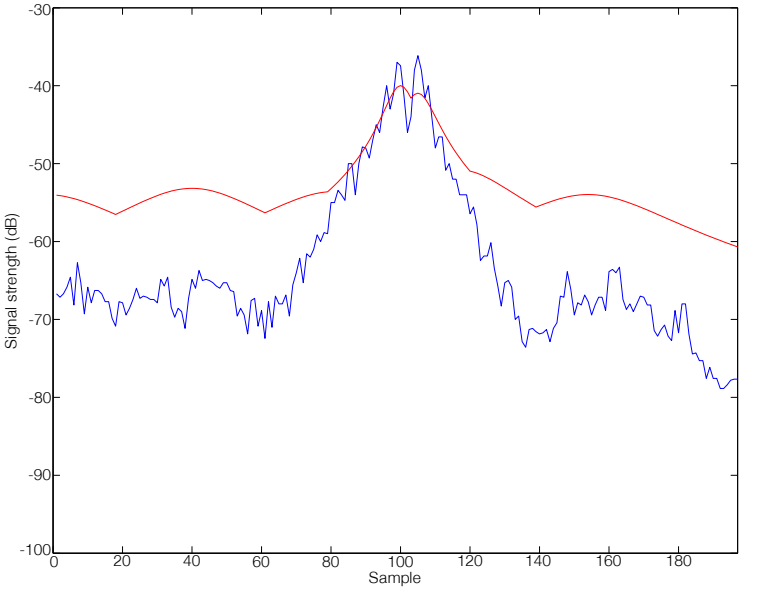


Figure 3.1 Measured signal strength (blue) versus signal strength modeled using the log-normal shadowing model with $n = 2$ (red). The model provides good estimation of the signal power close to the AP i.e. when it is in LOS. For greater distances to the AP, the model predicts higher powers than the ones received. At larger distances the signal is often further attenuated by objects in the surroundings and the model does not take that into account.

Signal environment	Path loss exponent, n
Free space	2
Indoor LOS	1.6-1.8
Indoor low number of obstructions	2-3
Indoor high number of obstructions	4-6

Table 3.1 Path loss exponent n for different signal environments [Rappaport, 1996].

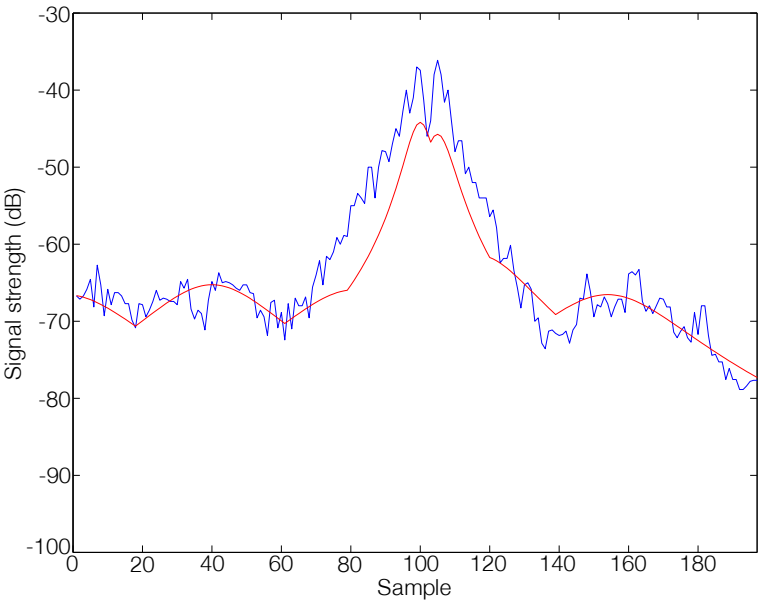


Figure 3.2 Measured signal strength (blue) versus signal strength modeled using the same log-normal shadowing model as Figure 3.1 but with $n = 3.2$ (red). This model performs good when the separation between receiver and transmitter is large. However, it predicts too low powers when the distance is short, due to the large path loss constant even when the AP is in LOS.

It is obvious that this model has its flaws. A low value of n fails to predict an accurate signal strength when the distance to the transmitter is fairly large, and the high value does not perform satisfactory close to the transmitter. The measurement taken is similar to a real world use-case where one moves through an office space, sometimes passing under a transmitter in LOS and sometimes far away in an obstructed area. The flaws promote the search for a model with better performance.

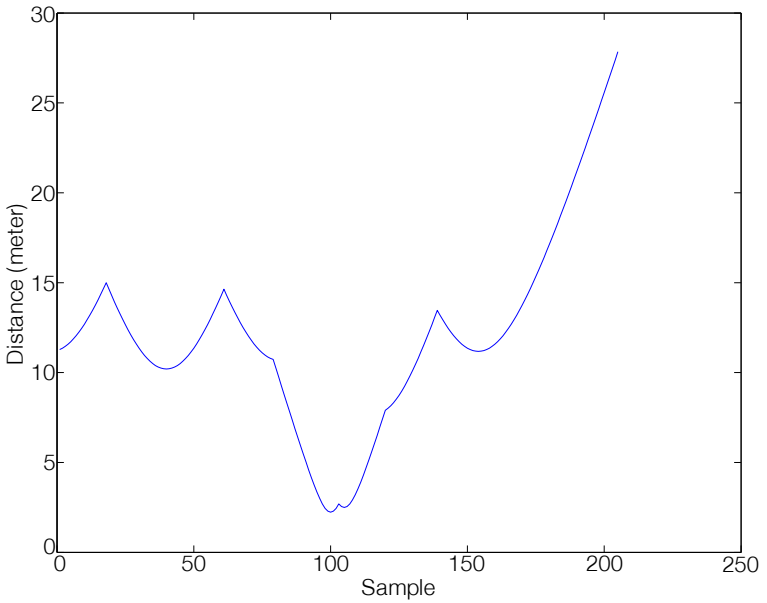


Figure 3.3 The distance to the transmitting antenna in meters used in the measurements of the signal strength. Measurements are taken both right underneath the AP and at distances of almost 30 meters. Further, the number of objects between the transmitter and receiver range from none to several.

Double Slope Model

A simple extension of the log-normal shadowing model is combining different values of n for use in various intervals of the distance to the transmitter. The most simple of these models is the double slope model where two different path loss exponents, n_1 and n_2, are used together with a single break distance, d_1, where the model changes from one to the other. The model then becomes,

$$\log_{10}(P_r(d)) = \log_{10}(P_r(d_0)) - 10n_1 \log_{10}\left[\frac{d}{d_0}\right] + X_{\sigma_1}, \qquad 0 < d < d_1 \tag{3.9a}$$

$$\log_{10}(P_r(d)) = \log_{10}(P_r(d_0)) - 10n_2 \log_{10}\left[\frac{d}{d_0}\right] + X_{\sigma_2}, \qquad d > d_1 \tag{3.9b}$$

where X_{σ_1} and X_{σ_2} are random Gaussian distributed variables with zero mean and standard deviation σ_1 and σ_2 respectively.

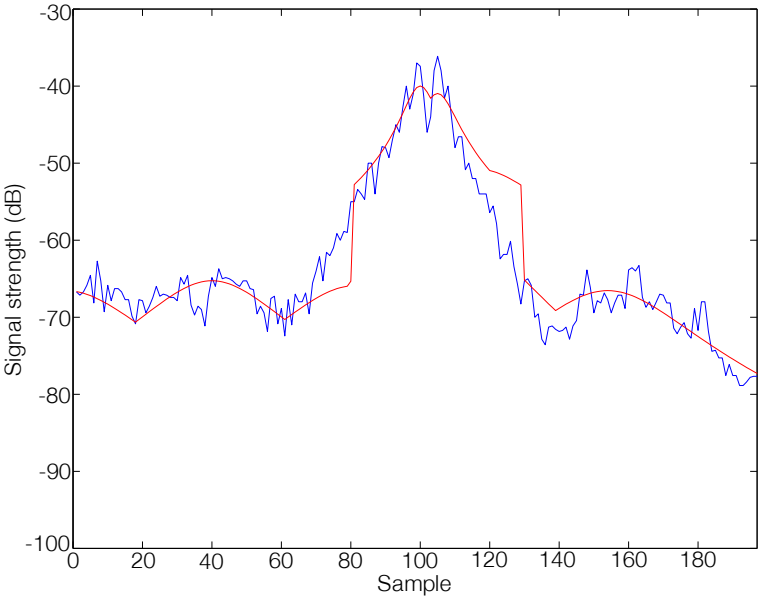


Figure 3.4 Signal strength measurements (blue) versus double slope model predictions (red) with $n_1 = 2$, $n_2 = 3.2$ and $d_1 = 10$ m. Here the two models from Figure 3.1 and 3.2 are combined. This gives good estimations of the power when the distance is short as well as long, but when the switch between the models occurs a large jump is present in the estimations.

In Figure 3.4, the double slope model for $n_1 = 2$, $n_2 = 3.2$ and $d_1 = 10$ m is displayed together with measurements using the distance trajectory in Figure 3.3. The value $n_1 = 2$, was chosen because at a short distance (0 to 5 meters) the path to an AP is assumed to be unobstructed. The other value, $n_2 = 3$, was obtained by performing measurements of the received power from APs at distances ranging from 15 to 30 meters and using these measurements to find the average path loss exponent. The distance at which the parameter is switched, d_1, is then chosen as the middle point between the distance ranges of n_1 and n_2.

This model combines the good traits from the two log-normal shadowing models. However, at $d = d_1$ the model presents an undesirable ”jump”, and the predictions around d_1 are rather poor. Furthermore, a single d_1 might be hard to find for a set of transmitters, and the different values of n may change between the transmitters.

α-Model

A third attempt to model the signal strength is to introduce a parameter α multiplied with the distance d in the log-normal shadowing model to account for the extra decrease in signal strength [Karlsson, 1992]. The model then becomes

$$\log_{10}(P_r(d)) = \log_{10}(P_r(d_0)) - 10n\log_{10}\left[\frac{d}{d_0}\right] - \alpha d + X_\sigma. \tag{3.10}$$

where X_σ is a zero-mean Gaussian distributed random variable with standard deviation σ.

A typical choice of n for indoor conditions is 2 or slightly less, to account for the LOS case when d is small, see Table 3.1. The range of α is around $[0.3, 1.5]$, depending on the building.

This model possesses the desirable feature of having only one parameter, α, to tune after the initial n is chosen. However, it tends to underestimate the signal strength at large distances d, as αd grows linearly with d while all other parts grow as the logarithm of d. Two simple ways to correct this easily comes to mind. Either the model can be changed to a log-normal shadowing when d is large, or a maximum of αd may be imposed. The first of these becomes

$$\log_{10}(P_r(d)) = \log_{10}(P_r(d_0)) - 10n_1\log_{10}\left[\frac{d}{d_0}\right] - \alpha d + X_{\sigma_1},\ d_0 < d < d_1 \tag{3.11a}$$

$$\log_{10}(P_r(d)) = \log_{10}(P_r(d_0)) - 10n_2\log_{10}\left[\frac{d}{d_0}\right] + X_{\sigma_2},\ d > d_1 \tag{3.11b}$$

and the second

$$\log_{10}(P_r(d)) = \log_{10}(P_r(d_0)) - 10n\log_{10}\left[\frac{d}{d_0}\right] - \alpha \cdot \min(d, d_{\max}) + X_\sigma. \tag{3.12}$$

Using one of these approaches, one loses some of the simplicity of having only one tunable parameter. However, these choices are simpler than for the double slope model, as there is no need to determine when the change from LOS occurs. The predicted signal strengths for $n = 2$, $\alpha = 0.9$ and $d_{\max} = 20$ using Equation (3.12) are shown together with measurements in Figure 3.5.

This model predicts the signal strength satisfactory both for small and large values of d, while removing the ”jump” present in the double slope model.

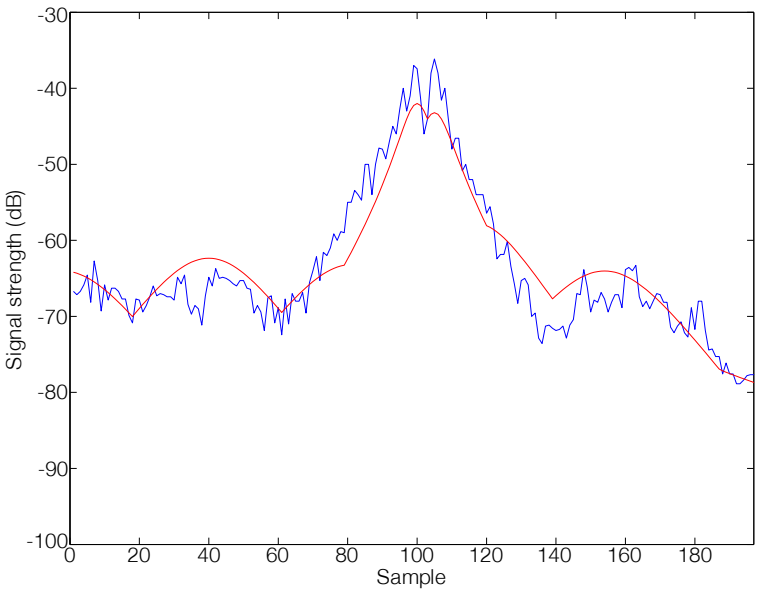


Figure 3.5 Signal strength measurements (blue) versus fixed α model predictions (red) with $n = 2$, $\alpha = 0.9$ and $d_{\max} = 20$ m. This type of model gives satisfactory estimations for both short and long distances. Further, it does not have the jump in predicted signal power present in the previous method.

3.5 Small-Scale Fading

In addition to the large scale propagation effects described above, a radio signal usually displays a phenomenon called small-scale fading. The small-scale fading causes the signal strength to fluctuate rapidly over small distances or short time-spans [Rappaport, 1996]. This effect is caused by different waves of the transmitted signal, called *multipath waves*, arriving at the receiver with a phase difference and causing interference. It can be a reflected or scattered wave arriving right after the direct wave, or reflected/scattered waves from different sources reaching the receiver at slightly different times.

Another effect included in small-scale fading is a random frequency modulation caused by different Doppler shifts on different multipath signals. The source of this is a relative movement between transmitter and receiver or movement by objects in the vicinity of the signals path.

The description presented above is brief, and for a more complete description of the small-scale fading phenomenon readers are referred to chapter 4 in [Rappaport, 1996].

3.6 Aspects for Positioning Applications

For a signal strength model to be practical for positioning, it needs to possess two properties. It needs to predict the mean signal strength at a specific distance with satisfactory precision, and the variance around the mean should be sufficiently small. A few different approaches to reach this goal are available. One is to try to characterize the environment where the positioning is taking place, using measurements of the signal environment in numerous points. This is feasible only when the environment is fairly uniform, as the resulting model is an approximation for the entire environment.

Another approach is to create a signal strength map of the entire environment, by a large set of measurements. This is called *fingerprinting*, and overcomes the problem of having a changing environment [Taheri et al., 2004]. However, it is impractical if used in a large area.

A third way of finding a model is to try to estimate the model parameters while the positioning is ongoing. This requires that at some point during the positioning, the position error is known to be small. Using this knowledge suitable model parameters can be calculated. Estimating the model has the advantage of not being tied to a specific environment and whilst staying in the same environment the model will continue to improve. On the other hand it may not produce the best results, especially when moving between different environments or if small positioning errors are rare or hard to distinguish.

4

The Particle Filter

The *particle filter* (PF), or *sequential Monte Carlo method*, are sets of estimation algorithms for estimating the posterior density of the state-space in a non-linear filtering problem. The PF uses a set of particles distributed over the state space using a system model, and measurements of one or several states are used to determine each particle's probability to represent the "true" state of the system.

In this chapter the non-linear filtering problem will be explained along with its solution using the PF. Some computational aspects will be investigated and the PF used in positioning briefly discussed. The chapter ends with a section about other estimation techniques, where most focus is placed on the *least squares* (LS) estimation.

4.1 Non-Linear Filtering Problem

The non-linear filtering problem consists of estimating the states in a non-linear non-Gaussian model on the general form

$$x_{k+1} = f(x_k, u_k, v_k) \tag{4.1a}$$
$$y_k = h(x_k, u_k) + e_k \tag{4.1b}$$

where f and h are arbitrary non-linear functions of the states x_k, inputs u_k and process noise v_k. Furthermore y_k is the measurement at time k. The measurement noise e_k and process noise v_k are random processes with arbitrary probability density functions (PDFs) [Gustafsson, 2012].

There is a large collection of filtering methods solving this estimation problem in different ways with different restrictions. One of the most common is the *Kalman filter*. This method is intuitive, simple and computationally effective. On the other hand, it suffers from requiring the functions f and h to be linear, and the PDFs of the process v_k and measurement noise e_k to be zero-mean Gaussian. The *extended*

Kalman filter solves the problem of f and h being non-linear, but still assumes v_k and e_k to be zero-mean Gaussian noise. Further, both Kalman approaches require the posterior distribution of the states to be zero mean Gaussian.

Here we will instead focus on another filtering approach, the *particle filter* (PF), also called the *Sequential Monte Carlo method*. This is a simulation based approach for solving the estimation problem (4.1), only requiring the PDF of e_k and v_k to be known [Gustafsson, 2012; Figueiras and Frattasi, 2010].

The PF consists of a set of N particles, $\{x^i\}_{i=1}^N$, which represent different samples of states. These particles are used to create an approximation of the distribution $p(x_k|y_{1:k})$ of the states x_k given the set of measurements $y_{1:k}$. The strength of the PF is that the distribution $p(x_k|y_{1:k})$ can be arbitrary. On the other hand, to provide the same particle density the number of particles grows by the power of the number of states. Hence, if you have 10 particles and one state, you need 100 particles to provide the same density for two dimensions. For three dimensions, you then need 1000 particles. This makes the PF suitable only when the number of states is relatively small.

4.2 The Particle Filter Process

The process of the PF consists of three separate steps [Gustafsson, 2012],

1. *Weighting of particles*: using the measurements, each particle is assigned a weight, corresponding to the likelihood of its states being the true ones.

2. *Re-sampling*: From the existing N particles, create N new ones by a clever choice.

3. *State update*: Using some trajectory of the states, update the states of each particle.

In the following sections these steps will be explored more in-depth along with a simple one dimensional example using $N = 1000$ particles. We start by distributing the particles using a uniform distribution in the range $[-10, 10]$, using 0 as our true state and Gaussian distributed zero-mean random variable with standard deviation 1 as the measurement noise, e_k, from equation (4.1b). The initial particle distribution is displayed in Figure 4.1

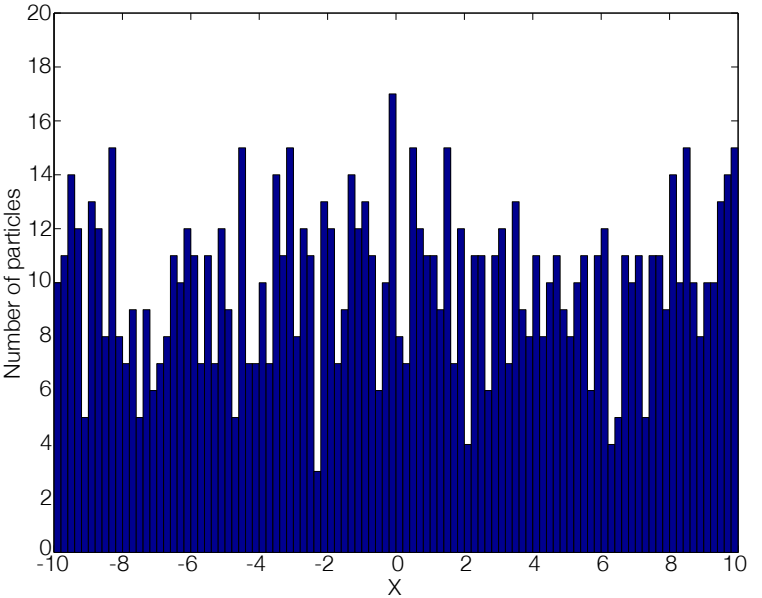


Figure 4.1 Histogram over the initial particle distribution in the range $[-10, 10]$ using 100 bins.

Computing the Weights

Given that the measurement noise e_k is known, the weighting of the particles is a straight forward process. For each measurement $y^l_{1:k}$ compute the probability $p^l(x^i_k|y^l_{1:k})$ that the particle x^i has the true set of states. Then the total probability for each particle is,

$$p(x^i_k|y_k) = \prod_{l=1}^{L} p^l(x^i_k|y^l_{1:k}), \tag{4.2}$$

where L is the number of measurements. It is also important to have the probabilities satisfying,

$$\sum_{i=1}^{N} p(x^i_k|y_k) \equiv 1 \tag{4.3}$$

as to have the probabilities normalized.

In Figure 4.2 the normalized weights assigned to each particle in our example are displayed. It comes as no surprise that the weights resemble the normally distributed measurement noise.

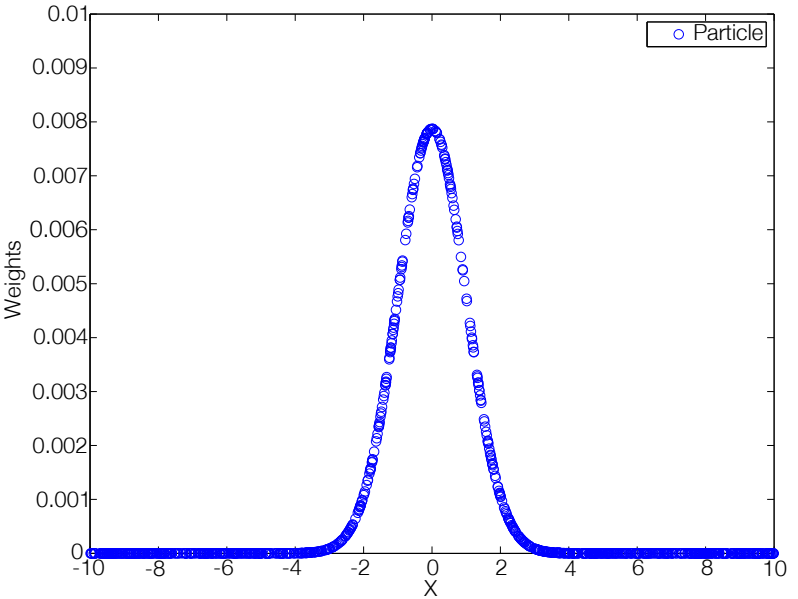


Figure 4.2 Particles versus their normalized weights. As expected this is a good estimation of the Gaussian normal distribution.

Re-Sampling

The re-sampling step is introduced to eliminate the possibility of one or a few particles to be the only probable after a few iterations of the PF. If the re-sampling is omitted, the computing of weight and subsequent state update could dilute the particles in the state-space until only one particle is the probable one. This could be the case even if the most probable particle does not agree with the measurement. Thus, the purpose of the re-sampling is to use the weights to create N new particles from the N old ones. This can be done in many ways. One is to allow a number of the most probable particles to spawn new particles until N particles are obtained. A statistically more stringent way is to compute a uniformly distributed random number r between zero and one. Then, assuming the sum of all particle probabilities is one, start adding these until the sum is greater than r. Then, let the particle that caused the change from below to above r, spawn a new one. Using this method particles with large weights have a larger chance of spawning a new one.

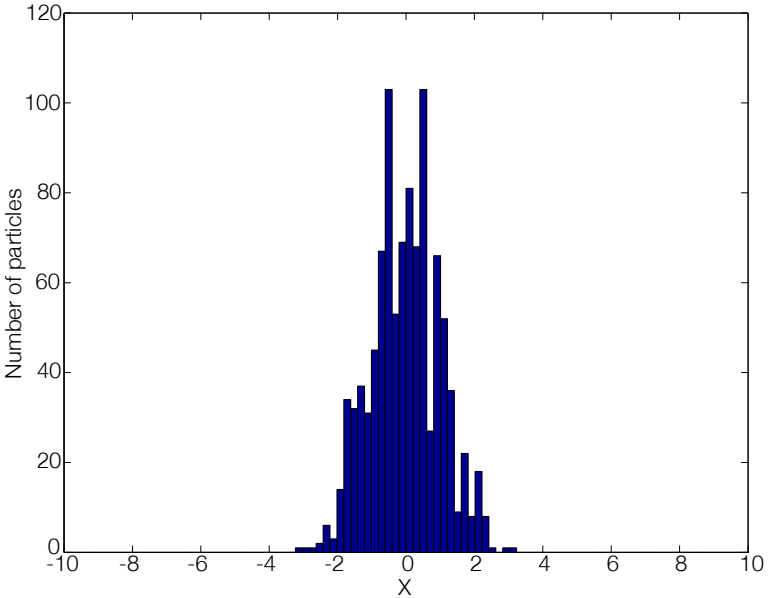


Figure 4.3 Histogram over the particle distribution, after one iteration of the PF, in the range $[-10, 10]$ using 100 bins. The particles tends more towards the true distribution.

After the re-sampling is done, it is crucial to give all particles the same weight, i.e $1/N$, as to avoid the dilution of probability.

After the re-sampling technique described above has been applied to the particles of our example, their distribution is displayed in Figure 4.3. The states of the particles still existing tend more towards the true one, $X = 0$. The mean of the states may now be used as an estimate of the true state.

State Update

In the state update step some model of the system behavior, $f(x_k, u_k, v_k)$ in equation (4.1a), is used to update the states of each particle in each time step, k. If the system behavior is well known, for one or all of the states, the use of an elaborate model improves the convergence rate of the particles and thus the estimation accuracy. This usually allows the use of fewer particles, since there is no need to move particles to improbable states.

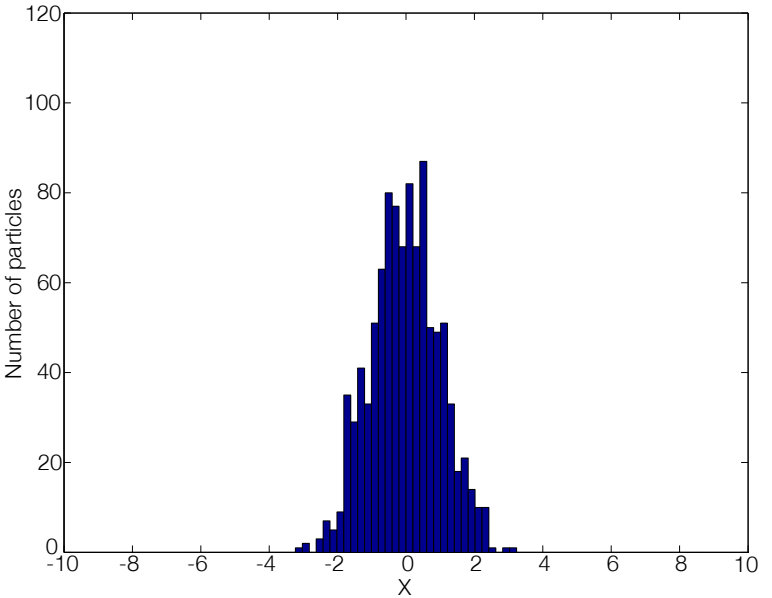


Figure 4.4 Histogram over the particle distribution, after one iteration of the PF and a state update, in the range $[-10, 10]$ using 100 bins.

However, if there is little or no knowledge of the states' behavior, some type of random walk update can be used, only trying to capture the variance of the state over a time step. If the variance is large, this calls for a large number of particles to keep the particle density for all probable states high.

In our example the "true" state is stationary, so the state update consists only of adding a Gaussian distributed random number with a small standard deviation (0.1) to each particle. This corresponds to letting each particle undergo a *Gaussian random walk*, and the resulting particle distribution may be viewed in Figure 4.4.

Finally, in Figure 4.5, the particle distribution after ten iterations is displayed, and the mean of the particle states is

$$\sum_{i=1}^{N} \frac{x_{10}^{i}}{N} = 0.04. \tag{4.4}$$

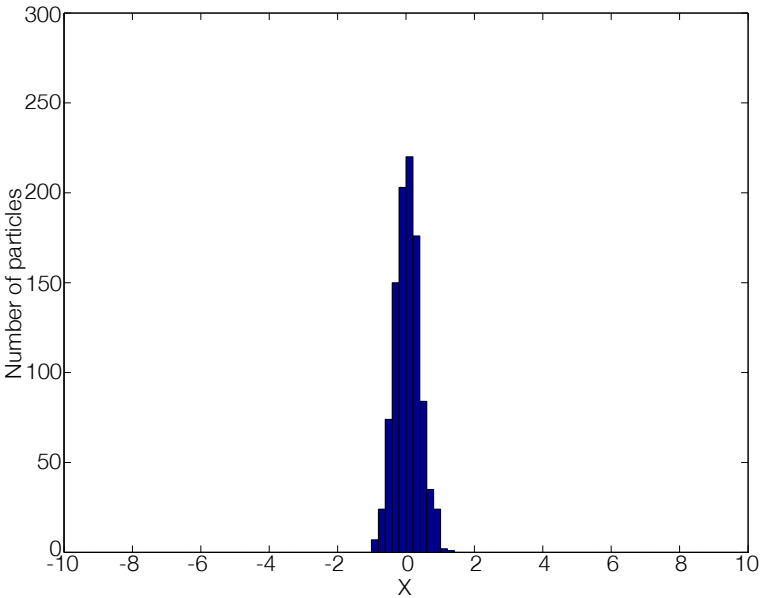


Figure 4.5 Histogram over the particle distribution, after ten iterations of the PF, in range $[-10, 10]$ using 100 bins. After this many iterations the mean of the particles are 0.04 compared to the true value, 0.

4.3 Computational Aspects

The PF, like most *Monte Carlo* based filters, suffers from high computational complexity. Especially as the number of particles necessary to keep a fixed particle density in the state space grows as the power of the number of states. A solution to this is proposed in [Gustafsson, 2012], where only states bearing non-linear distributions are passed throughout the PF and states bearing Gaussian distributions may be filtered by a different method. This is called the *marginalized particle filter* and aims to keep the number of states passed to the PF low.

The three steps discussed in Section 4.2 pose somewhat different computational difficulties. The weights need to be calculated for each particle and thus need to be computed N times, giving an $O(N)$ complexity. However, the calculation for each weight depends of the number of measurements M giving a total complexity of $O(M \cdot N)$. In most cases however, the number of measurements is small in comparison to the number of particles and may be neglected. Another pleasant feature of the

weight computation is their independence of each other, thus making parallelization of computations simple.

For the re-sampling step, assuming the number of samples before and after is constant, the algorithm described in Section 4.2 comprises two distinct steps. First the *cumulative sum* of the weights is computed. This is of complexity $O(N)$ but can, depending on the implementation, be computed alongside the weights. Next, for the N new particles, it is needed, for each, to determine which of the N old ones to inherit states from. This is done by, for each new particle, computing a uniformly distributed random number in the range [0,1]. This random number is then compared to the cumulative sum of weights to find the index where the cumulative sum for the first time is equal to or larger than the random number. This can be done using a *binary search* as the cumulative sum is sorted by its nature, giving the step complexity $O(N \cdot \log_2 N)$. Furthermore, the weights need to be computed before the re-sampling is initialized, making it impossible to parallelize the particle filter process from start to finish. However, when the weights have been computed, the particles, once again, are independent.

Updating the states is done for each particle and it depends on the number of states K along with the complexity of the state model, resulting in roughly $O(N \cdot K)$ complexity. During this step, the particles, once again, are entirely independent, which makes parallelization possible.

In all, the filter has approximately a complexity of $O(N \cdot \log_2 N) + O(N \cdot M) + O(N \cdot K)$.

4.4 Particle Filter for Positioning Applications

The PF may be successfully used in positioning applications, especially if the states and models used possess certain features. More specific, the PF may be used with good results if the models are non-linear and the posterior distributions are non-Gaussian while the number of states is small [Gustafsson, 2012]. If the number of states grows, the required number of particles makes the use of a PF infeasible.

A case where the PF is useful is the estimation of position or position and heading in a two-dimensional space, which is the case for most indoor positioning applications. Furthermore, using signal strengths from multiple sources as a measurement of position renders, both a non-linear model and non-Gaussian posterior distributions further promote the use of a PF. However, introducing more dynamic states, e.g. acceleration, unmeasured velocities or trying to filter sensor biases or drifts, introduces too many dimensions. Thus, such estimations require their own filter [Gustafsson, 2012].

4.5 Other Estimation Techniques

There are some different techniques for state estimation given a set of measurements, the *Kalman* methods mentioned earlier, a couple of *maximum-likelihood* based methods and the *least squares estimation* method [Gustafsson, 2012].

Least Squares Estimation

The method of *least squares* (LS) is a common estimation technique for overdetermined systems, and is based on minimizing the error between an observed value and a modeled one.

Given a series of n measurements, z_i and n models $f_i(x_k)$ of the m states x_k, the goal of the LS estimation is the find the minimum of

$$S = \sum_{i=1}^{n} (z_i - f_i(x_k))^2 \tag{4.5}$$

for $n > m$. In the case of $f_i(x_k)$ being linear for all i, the algorithm is called the *linear least squares* and S, in this case, has only one global minimum due to convexity. Also, an analytic solution is always present.

Generalizing the linear LS estimation to non-linear models is fairly simple. The sum in Equation 4.5 is still to be minimized, but in the non-linear case a closed-form analytical solution can not be found. Instead the minimum is found by numerical approximations. As S in the non-linear case is not, in general, convex, there exist multiple local solutions to the minimization problem. This requires a guess of the approximate states to ”guide” the algorithm to the correct solution, and if the guess is too far from the global minimum, a local minimum could be found instead.

The non-linear LS can be used to estimate a position in two dimensions from the models in Chapter 3, together with measurements of signal strengths from at least three APs.

5

Introduction to WiFi

The *WiFi Alliance* defines WiFi as any ”wireless local area network (WLAN) products that are based on the *Institute of Electrical and Electronics Engineers* (IEEE) 802.11 standards”. However, since most WLAN products use the 802.11 standards, in common tongue, WiFi and WLAN are interchangeable.

This chapter is intended to give an introduction to the WiFi technologies and why these are of interest for indoor positioning.

5.1 The 802.11 Standard

The IEEE 802.11 standard consists of a series of techniques for over-the-air radio modulation using the same basic protocol [IEEE Computer Society, 2013]. The standard allows different products, computers, smart-phones etc. to implement the protocol to communicate wirelessly. Historically the communication has taken place over the frequency 2.4 GHz, however, in later years devices implementing 5 GHz communication protocol have become increasingly common [Walrand, 2010].

The standard splits each frequency range into several channels over which the communication takes place. The frequency 2.4 GHz is split into 14 channels with 5 MHz spacing. For 5 GHz the situation is more complicated and will not be explained in detail. Local regulations affect which channels are allowed in a certain region.

Wireless technologies are evolving towards higher frequencies. The radio channel surrounding 2.4 GHz is well populated and a move to other frequency domains is necessary to continue the technology expansion. Further, a near to full frequency environment promotes high noise levels and dropped connections due to interference between devices. Another advantage with transmitting information at higher frequencies is that the signal can contain more information per time unit, resulting in higher bit rates.

The MAC-Address

Most devices using the IEEE 802.11 standard have a *Media Access Control* (MAC) address assigned as its network address [IEEE Computer Society, 2013]. The MAC address is usually assigned by the manufacturer and can be considered unique in a local area network. The address consists of twelve hexadecimal numbers. In its human readable form it is presented as six groups of two hexadecimal numbers, separated with either `:` or `-`, e.g. `0a:1b:2c:3d:4e:5f`. The address can be used to identify from which device a signal originates.

5.2 WiFi and Positioning

The WiFi standard provides enticing possibilities for indoor positioning, especially in public or corporate areas. Some of the advantages compared to other positioning techniques is that the range of typical wireless products, like WiFi APs, is limited ($\sim$ 30 meters), and thus they need to be placed with relatively short intervals. Moreover, WiFi infrastructure is ubiquitous and already deployed at many places, making WiFi based positioning an attractive method. Further, more and more devices become capable of communication over WiFi and platforms like Android make the deployment of applications easy and readily available for a large amount of users. One of the disadvantages is that indoor signal environments are often complex, posing a harder modeling task.

A few different techniques for indoor positioning are presented below.

- *MAC-Address*:

  Receiving a signal, and thereby a MAC-address from an AP, indicates that the receiving device is within approximately 30 meters, depending on building, transmitted power and effectivity of the receiving antenna. This is a rather crude positioning, but may be used to pinpoint a device to a certain area. Furthermore, receiving signals from multiple devices will improve the positioning, if the transmitter locations are different, i.e reducing the possible position to a subsection of the area.

- *Received Signal Strength (RSS)*:

  Measuring the signal strength from an AP puts the device not just in the vicinity, but gives a rough indication of the distance to the AP. Multiple APs may then be used to trilaterate a position, but noisy measurements give a rather large uncertainty.

- *Time of Arrival (ToA)*:

  If the APs and the measuring device are synchronized in time, the time difference between when a message was sent and received may be used to compute the distances between the device and the APs. However, the time differences involved are usually very short ($\sim 3 \cdot 10^{-9}$ seconds per meter), especially in indoor environments, requiring the time synchronization to be precise.

- *Time Difference of Arrival (TDoA)*:

  If only the APs are synchronized, the measurement of the time difference between signals arriving from different APs can be used to determine the position. This requires the same type of time synchronization as the ToA method.

- *Round Trip Time (RTT)*:

  A way to eliminate the requirement of synchronized APs, is to measure the time it takes for a message to travel to the AP and back again. This time is called the *round trip time*. As only one device performs the time measurement the need for synchronization is eliminated, but instead the time from when an AP receives a message until it sends the response has to be known.

The techniques presented here assume that the measuring device is also performing the positioning. Similar approaches are available using the APs, to position devices in the environment. Furthermore, the technologies above could be combined or used together with different technologies such as Bluetooth, NFC, RFID etc [Choo et al., 2012; Chon et al., 2005]. to increase the positioning accuracy. In this thesis, however, the focus will be placed on the received signal strength from WiFi.

Received Signal Strength Indication

The *Received Signal Strength Indication* (RSSI) is a value indicating the short-time average of the received signal power [Figueiras and Frattasi, 2010]. The RSSI is given in an arbitrary unit, usually dBm, and there is no standard of how RSSI should be related to the physical properties of the received signal. This means different manufacturers are able to choose the range in which the RSSI is presented.

For the RSSI to be a meaningful measurement of the received signal strength it must somehow be related to the physical properties of the signal. Usually, a measurement of the RSSI at a known distance is used together with an investigation of how the RSSI changes with the distance. The models discussed in Chapter 3 can then be applied. Different devices may posses different parameters, thus requiring modeling for a specific device or the use of an adaptive estimation scheme.

RSSI Measurements in Android

An RSSI measurement on a device running the Android operating system is initiated by an application requesting a scan of the signal environment. The operating system performs the scan when the necessary system resources are available. Each channel specified by the IEEE 802.11 standard and available in the current region is scanned. When a transmitter is found, a short-time average of the received signal power is measured and converted to an appropriate RSSI. The process is repeated until all reachable transmitters have been measured, and the results are returned along with information about each transmitter's MAC-address, its transmission frequency and a time-stamp.

To conserve battery life for the measuring device, a technique called *passive scanning* is used. Simply explained, each AP sends a message approximately every 100 ms telling devices in the vicinity it is active, its MAC-address and some additional information. The measuring device scans each channel for a little more than 100 ms to catch all APs on that channel, resulting in a quite long scan time ($\sim$ 3-5 s), depending on the number of channels to be scanned. Most commonly the channels related to 2.4 GHz will be scanned first and subsequently the 5 GHz channels. This will generally introduce a time difference between the two different frequency measurements from a single AP.

Which channels to scan could be specified to reduce the scan time, but some AP data might then be lost. Another way of performing the scans is to have the measuring device actively probing (”asking”) for APs in its surroundings, but this would require almost constant sending of messages and listening for replies, thus draining the battery.

5.3 Performance of 2.4 GHz Versus 5 GHz

The evolution of WiFi has prompted many APs to implement standards for both 2.4 and 5 GHz communications. Alongside the performance boost to devices able to use both frequencies, this poses an enticing possibility for higher performing indoor positioning. Different frequencies may, in the best case, provide two independent measurements of the distance to an AP. To investigate if this is the case, the correlation between the two frequencies must be determined.

Signal Correlation

Of course, some of the signal behavior is expected to be highly correlated. For one, the large scale distance dependent path loss is largely the same for both signals, and it is from this path loss the distance to the AP is determined. Nevertheless, independence is a desirable feature for the signal noise, which includes the *small scale fading*.

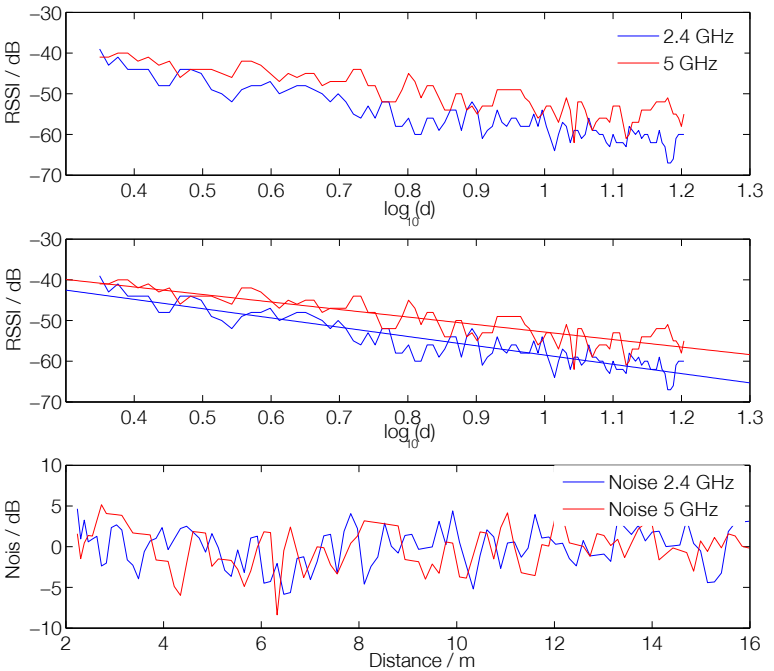


Figure 5.1 Signal power, signal power together with linear trend and signal power minus linear trend for the 2.4 and 5 GHz channels. The measurements were taken while traversing a distance of 15 meters, moving away from the AP. The power is of course attenuated in a similar way due to the increasing distance to the AP, but when this effect is subtracted, the noise of the 2.4 and 5 GHz signal is not entirely correlated.

To investigate the correlation between 2.4 and 5 GHz signals, measurements were taken, with an increasing distance, to an AP transmitting on both frequencies. After the path loss dependent behavior was removed from the signals, the correlation of the noise could be computed. This process is illustrated for one series of data in Figure 5.1

The *correlation coefficient* $\hat{r}_{xy}$ between the two can be estimated using

$$\hat{r}_{xy} = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{(N-1)\sigma_x\sigma_y}, \tag{5.1}$$

where N is the number of measurements, x_i and y_i are the 2.4 and 5 GHz measurements compensated for the path loss, $\bar{x}$ and $\bar{y}$ their means and σ_x and σ_y their standard deviations. These standard deviations can be estimated according to

$$\hat{\sigma}_x = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}. \tag{5.2}$$

To construct a confidence interval for the covariance some statistic processing of the data is required. This presentation will be brief, and interested readers are referred to [Fisher, 1915; Fisher, 1921] for a more in-depth description.

Since the distribution of $\hat{r}_{xy}$ is not Gaussian, *Fisher's z-tranform*

$$z' = \frac{1}{2}\ln\left(\frac{1+\hat{r}_{xy}}{1-\hat{r}_{xy}}\right) \tag{5.3}$$

is used. Under the current conditions, z' is normally distributed with standard deviation

$$\sigma_{z'} = \frac{1}{\sqrt{N-3}}, \tag{5.4}$$

if more than 3 measurements are taken.

The two-sided confidence interval for z' at level α is then

$$I_{\hat{z}'} = \hat{z}' \pm Z_{\alpha/2} \cdot \frac{1}{\sqrt{N-3}}, \tag{5.5}$$

and may be transformed back to $\hat{r}_{xy}$ using the *inverse Fisher's z-transform*,

$$\hat{r}_{xy} = \frac{e^{2z'}-1}{e^{2z'}+1}. \tag{5.6}$$

In Table 5.1 six estimates of the correlation between the noise of the 2.4 and 5 GHz RSSI measurements are presented. Four of these are estimated from data where the distance varies by 15 meters to the AP. Two of these are taken in line-of-sight (LOS) and two in non-line-of-sight (NLOS), and for each case one measurement is taken near the AP and one far from the AP. The other two sets of data are taken at stationary positions in LOS and NLOS respectively.

The table shows that the correlations are significantly different from zero in some cases. At the same time, they are small with high probabilities. This means that additional information is added by using both frequencies, compared to using only one of them. It also means that the impact of the noise from one frequency can be reduced by using the other frequency as well.

Signal path	Estimated covariance, r_{xy}	95 % confidence interval
LOS near AP	-0.10	$[-0.29; 0.09]$
NLOS near AP	0.26	$[0.05; 0.44]$
LOS far from AP	0.26	$[-0.09; 0.55]$
NLOS far from AP	0.24	$[0.04; 0.42]$
LOS stationary	-0.06	$[-0.13; 0.01]$
NLOS stationary	-0.12	$[-0.18; -0.06]$

Table 5.1 Estimated covariances and 95 % confidence intervals for 2.4 and 5 GHz in different environments. There seems to be a small correlation but the signals are not entirely correlated, allowing for additional information to be obtained if both are used.

However, the model errors, i.e. the difference between models and measurements, are still highly correlated between the frequencies. For example, if the user moves from LOS with an AP to NLOS, e.g. to the other side of a wall, both signal frequencies will be reduced in similar ways. Events like these are not modeled accurately, since one can not know when they occur. The result is a large correlation between the model errors for each frequency.

Difference in Modeling

An extensive discussion about how path loss depends on the frequency is done in [Rappaport, 1996], the gist of which is that higher frequencies lose power faster with distance. The difference between the behavior of 2.4 and 5 GHz is not large (a factor ~ 2), however some extra attention is needed to model the two. To extend the models in chapter 3 to the difference in frequency, usually different n or α, depending on the choice of model, will be needed. Furthermore, an AP may transmit 5 GHz with a greater power than 2.4 GHz to mitigate some of the increased path loss, requiring information of the transmitted power for both frequencies.

6

Indoor Positioning - RSSI Approach

The signal propagation models described in Chapter 3 in combination with the particle filter (PF) from Chapter 4 provide a framework for indoor positioning. This framework can be used to perform position estimation using RSSI measurements from multiple WiFi APs.

In this chapter, the considerations about, and parameter choices for, the PF in the context of indoor positioning are presented. The performance is then tested in a few different environments and compared to position estimation using a least squares (LS) algorithm. These tests then serve as a basis for an extensive error analysis, discussing the error behavior depending on different conditions. The resulting knowledge is then used in the next chapter, to ascertain if and how adaptive or crowdsourced signal strength models may be used.

Importance of Geometry

How the APs are positioned in relation to the user (i.e. the position of interest to estimate), highly impacts the quality of the estimation. In Figure 6.1, three different AP geometries are shown along with the distribution of probable positions. It represents an idealized case where the noise in the measurements of the distance to each AP is zero-mean Gaussian. However, it provides an illustrating example of how the geometry impacts the distribution of the estimated position.

In the leftmost case in Figure 6.1 the APs are placed on a straight line, and in this case trilateration of the *true* position, $[0, 20]$, is impossible if no prior information about past positions is known. As the figure shows, APs positioned on a straight line gives two equally probable position estimates (more *red* implies higher probability), i.e. it is not possible to distinguish between a position and its mirror image in the line formed by the APs.

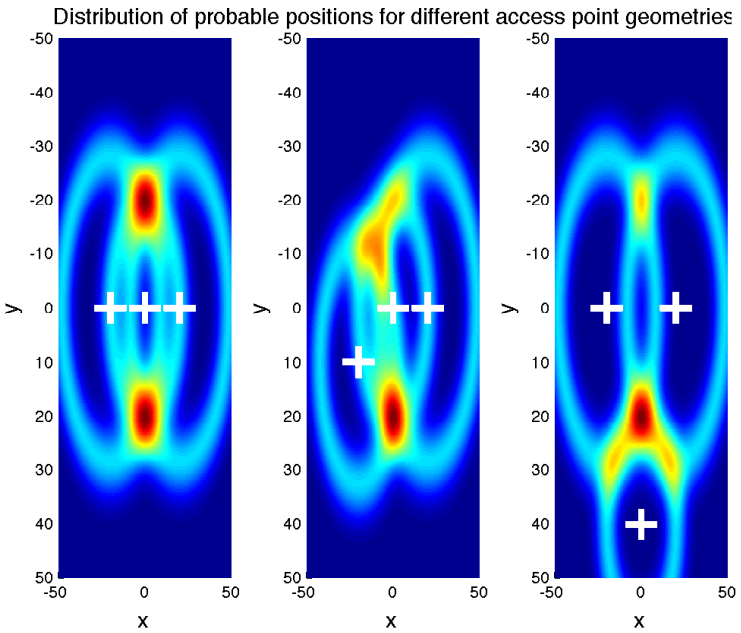


Figure 6.1 Probable positions for different AP configurations (*white* +). *Red* implies high probability and *blue* low probability. The *true* position is $[0, 20]$. It is easy to see that a geometry like the one in the rightmost figure produces the best position estimates while it is hard to determine the position without prior position history if the geometry is similar to the left figure.

The second case displays a scenario where the APs are placed in the corners of an obtuse triangle (largest angle greater than $90°$). In this case it is possible to discern the *true* position as the most probable one. However, higher noise levels and/or biased noise in the measurements will make this increasingly hard, as there are two different positions with relatively high probabilities.

The third case represents a best-case scenario where the position to be estimated is centered in an equilateral triangle. Here, it is easy to distinguish the most probable position, and likewise the *true* position.

The PF uses information about the positions at previous time steps making it more likely to determine the true position even if the current geometry is disadvantageous. The LS estimation, on the other hand, uses only instantaneous measurements and will suffer much more severely from an unfavorable environment.

The positioning will further benefit from a larger number of APs as both the number of measurements and the probability of having an advantageous geometry increase. Still, a few conclusions could be drawn from further simulations, the main one being that measurements taken inside a polygon with corners consisting of the APs tend to provide the best information about the true position regardless of the used estimation algorithm.

6.1 On Parameter Choices

There are four parts of the PF that can be chosen by the user to some extent. The system model, the process noise v_k, the measurement noise e_k and the number of particles N.

System Model

The system model is the most extensive choice, where both the number of states and their respective dynamics need to be determined. Some parts of this are dependent on available measurements and how complex the model is allowed to be. Using only measurements of the RSSI from multiple APs, a simple model using two states x and y, describing the position in a plane, are used. Moreover, a random walk process is used to model the dynamics, resulting in the state update

$$x_{k+1} = x_k + v_{k+1}^x \tag{6.1}$$
$$y_{k+1} = y_k + v_{k+1}^y \tag{6.2}$$

where v_k^x and v_k^y are the process noise on each state.

Process Noise

The process noise is, in this case, rather straight forward to choose, as it represents the spread of particles in between measurements. A normal distribution around a circle with radius r is an appropriate choice,

$$v_k^x \sim r\sin\alpha_k + N(0,\sigma^2) \tag{6.3}$$
$$v_k^y \sim r\cos\alpha_k + N(0,\sigma^2) \tag{6.4}$$

where

$$\alpha_k \sim \mathrm{U}(0,2\pi), \tag{6.5}$$

$\mathrm{U}(0,2\pi)$ is a uniform distribution between 0 and 2π and σ is the standard deviation of the normal distribution. In this case r is an approximation of how far the phone travels between two measurements and σ measures the uncertainty of this distance.

A fair estimation is that a walking person travels between 1 and 2 meters per second, and given an interval Δt between measurements it holds that $r \approx 1.5 \cdot \Delta t$ and $\sigma \approx 0.5 \cdot \Delta t$. Thus,

$$v_k^x \sim 1.5\Delta t \sin \alpha_k + N\left(0, \left(\frac{\Delta t}{2}\right)^2\right) \tag{6.6}$$

$$v_k^y \sim 1.5\Delta t \cos \alpha_k + N\left(0, \left(\frac{\Delta t}{2}\right)^2\right) \tag{6.7}$$

$$\tag{6.8}$$

places the new particles somewhere on a circle of radius $1.5\Delta t$ from the old one and spreads them by a standard deviation of $0.5\Delta t$.

This estimation could be further improved by the use of a technique for counting steps, and this approach will be investigated in later chapters.

Measurement Noise

The measurement noise can be estimated in the following way. Take a set of RSSI measurements from a certain AP, and vary the distance. Then model a received RSSI at each distance and compute the noise as the difference between model and measurement. The standard deviation of measurement noise can then be estimated as the average of the absolute value of the noise. To estimate the noise accurately a large set of data in different environments is required.

This has been done for a quite large number of APs, and in environments ranging from LOS to a highly obstructed signal path. Some of these results are displayed in Figure 6.2.

The resulting variance in the measurements ranged from 9 to 60, with a mean of 25. The reason for the large span is that some of the APs were in LOS for most of the measurements even though the distance to them was fairly large. Other APs were, on the other hand, highly obstructed while at a quite short distance, and yet others followed the models well.

During the position estimations in the rest of this chapter the mean of the variance, 25, will be used. Further, the measurement noise will be assumed to be Gaussian with zero mean. This assumption is valid for data sets covering large distances. However, over a short traveled distance the noise is expected to be biased in some cases as it is highly dependent on the environment. This is somewhat mitigated by having measurements from numerous APs, allowing positive and negative biases to cancel.

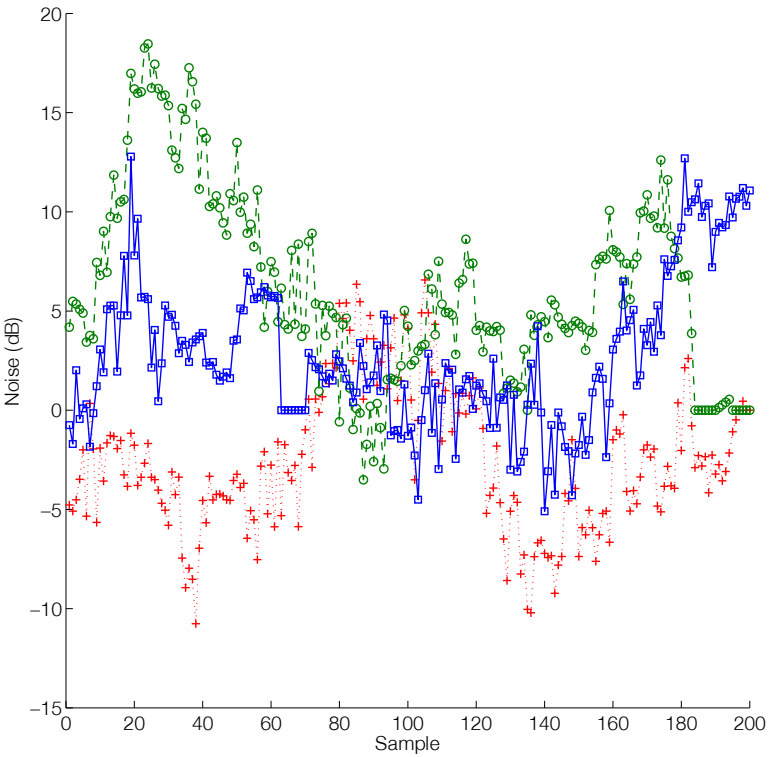


Figure 6.2 The difference between measurements and model for a few different APs for different environments and distances. Here it is clearly visible that the indoor signal environment is hard to model. Depending on the number of objects between transmitter and receiver the received signal strength at a given separation varies substantially. Because of this large measurement noise, the particle filter needs to take into account that the received signal strength at a given distance has a large variance.

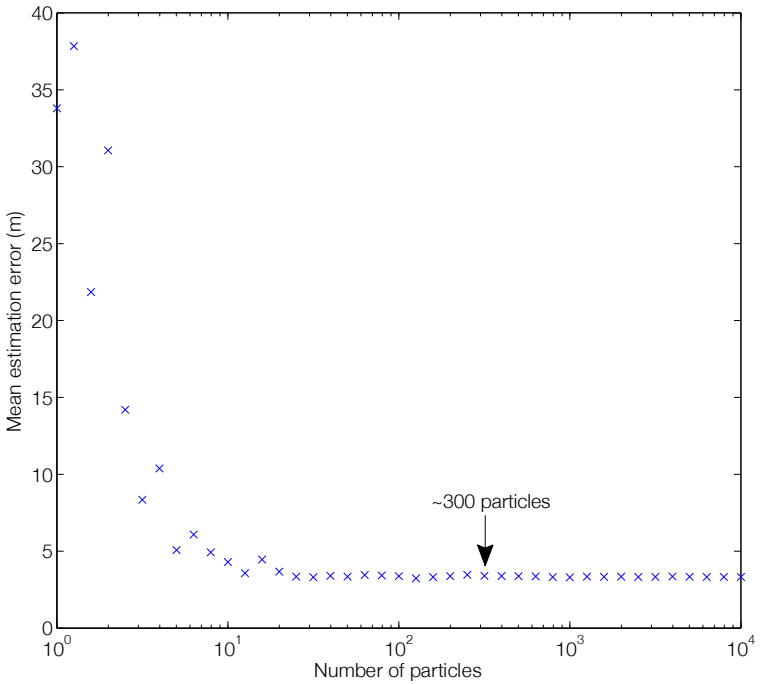


Figure 6.3 Mean error in estimated position versus number of particles, N, for a two dimensional positioning. For a small number of particles, the mean estimation error can be greatly improved by increasing the number of particles. However, when reaching around 100 particles, this improvement stops, and only minor differences are noticed.

Number of Particles

The number of particles required to maintain a high enough distribution over the relevant part of the state space, is dependent on the size of the state space and the distance traveled between filter updates. In this case, the size of the state space is two, as the positioning is done in two dimensions. The distance between filter updates, i.e. distance between consecutive RSSI measurements, changes for different test cases. The range starts at 0.5 meters with a close to zero variance, when measurements are taken every 0.5 meters around a predefined path. The other limit of the range is around 4 meters with considerably larger variance when measurements are taken around every third second while walking a predetermined path.

In Figure 6.3, the mean positioning error for the same data set is shown, for different numbers of particles, N, ranging from 1 to 10 000. The data was generated by measurements taken with 3 meters of relative spacing along a predetermined path. For small N, the error improves dramatically when N is increased. However, for N larger than ~ 40 no further improvements are noticeable and the mean estimation error settles at ~ 3.5 meters.

For some of the test cases used later in this chapter, the distance between measurements is considerably larger than 3 meters. To keep the particle density high even in those cases, $N = 300$, is chosen and used for all tests in the following sections.

6.2 On Model Choice

From Chapter 3, a model on the form

$$R_x(d) = C - 2 \cdot 10 \log_{10} d - \alpha d \tag{6.9}$$

is appropriate for the RSSI measurements. In this model, $R_x(d)$ is the RSSI measurement at a distance d from the transmitter, C is related to the power of the transmitter and α models path loss related to signal path obstruction.

The models are discussed in Chapter 3 but appropriate values of the constants C and α are yet to be determined. This may be done in a couple of fashions, two of which will be presented here. The value of α is dependent on the environment. The value of C is related to the power of which the WiFi APs transmit, so different types of APs will most likely have different values.

Model Driven Approach

Given measurements at known locations, preferably with both obstructed and LOS paths to the APs, it is possible to compute the model parameters.

The value of α from (6.9) can be calculated as

$$\alpha = \frac{C - 2 \cdot 10 \log_{10}(d) - R_x(d)}{d}. \tag{6.10}$$

The estimation of α will improve if the paths between transmitters and receiver are representative for the current environment.

To find the value of C, measurements of the RSSI should be performed in LOS of the AP, at different locations and with different distances to the AP. The value can then be calculated from

$$C = R_x(d) - 2 \cdot 10 \log_{10}(d), \tag{6.11}$$

i.e. from (6.9) with $\alpha = 0$ (LOS).

Frequency, GHz	α	$\sigma(\alpha)$	C	$\sigma(C)$
2.4	0.332	0.319	-40.7	2.97
5.0	0.551	0.375	-32.3	4.72

Table 6.1 Estimated model parameters , α, C , and their standard deviations.

This has been done at 20 different locations with 20 measurements of the RSSI from available APs taken at each location. The estimated values of α and C are presented in Table 6.1.

Error Estimation Approach

The other method is more heuristic, but might produce equal or better performance. Instead of finding the values of α and C by measurements it is possible to find the parameters resulting in the smallest average positioning error. This is done by taking a set of RSSI measurements at a few different known locations and use them in the position estimation algorithm with different α and C, to find the parameters producing the smallest mean error.

To illustrate this, the error for $\alpha \in [0;1]$ and $C \in [-55;-25]$ is shown in Figure 6.4 for 2.4 and 5 GHz. The parameters yielding the smallest average positioning errors are $C = -38$ and $\alpha = 0.3$ for 2.4 GHz, and $C = -39$ and $\alpha = 0.2$ for 5 GHz.

Final Parameter Choice

A few interesting observations can be made from the two methods above. The first of these is that the range of possible values is large, indicated by the standard deviations obtained from the model driven method. This is somewhat to be expected as the obstruction between transmitter and receiver is vastly different from location to location. Thus, the obtained parameters will be an average of the different environments, giving rise to their high standard deviation.

Further, it is noteworthy that the parameters obtained from the two approaches differ, especially for 5 GHz. One explanation for this is that the PF is able to find an accurate position given, for example, a positive model error for 2.4 GHz and a negative error for 5 GHz from the same AP. However, the parameters found trying to minimize the model error still produced small positioning errors when used together with the PF.

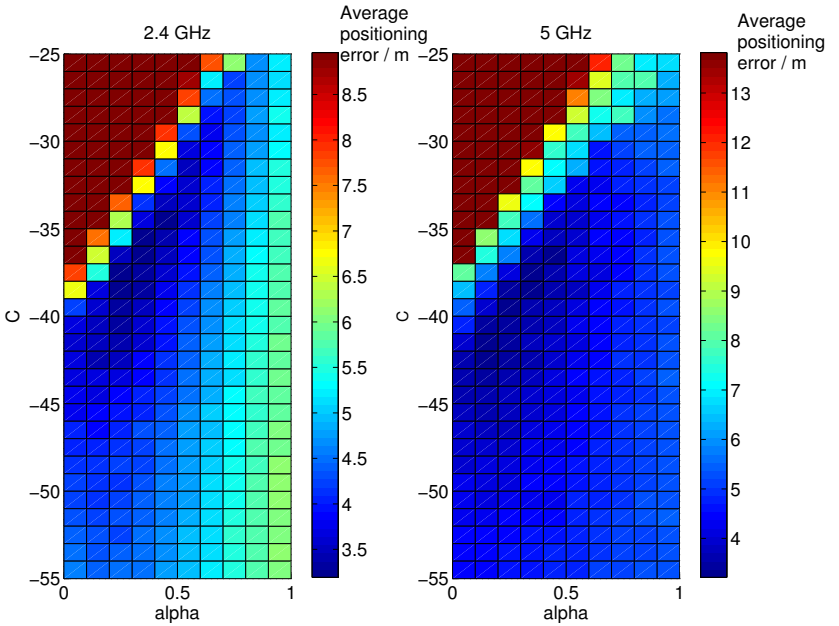


Figure 6.4 Average estimation error as a function of α and C for 2.4 and 5 GHz RSSI measurements. Errors larger than 13 m for 2.4 GHz and 9 m for 5 GHz are capped to put emphasis on the interesting behavior. The interesting take from this figure is that there is a large range of parameters that provides estimates with similar properties, at least if comparing the mean error. The values producing the best results are the ones predicting the correct received signal power most of the time. If these values, on the other hand, are changed by a small amount in the wrong direction, very bad estimates could be obtained. This happens when the received power at a certain location is estimated as higher than it actually is. The overall conclusion is that, when using a particle filter, it is considerably better to underestimate the signal power, than to overestimate it.

Frequency, GHz	C	α
2.4	-38	0.4
5	-36	0.5

Table 6.2 Model parameters for equation (6.9), to be used to model the RSSI for the following sections.

Looking at Figure 6.4 the PF exhibits an interesting behavior. The positioning error is relatively small for a large range of parameters, but a change to substantial errors happens suddenly for certain parameters. Parameter choices along the approximate line from $C = -39$ and $\alpha = 0$ to $C = -25$ and $\alpha = 0.7$ need to be changed only by a small amount to produce either good or truly poor results. A deeper investigation shows that this line roughly translates to the models switching from estimating a too low signal power to estimating a too high one, for a certain location. The smallest positioning errors are obtained when the modeled and received powers are about the same, placing them slightly below the line described earlier. However, these parameters are sensitive if the environment changes, resulting in estimating too high signal powers for the given location. A more robust choice is to overestimate the signal attenuation resulting in a higher α. This choice might produce slightly larger positioning errors, but is less sensitive to changing environments.

The conclusion of the discussion above is that it is far better to underestimate the signal strength at a given location than to overestimate it. Taking this into account, the values stated in Table 6.2 will be used to model the received RSSI value at a distance, d, from the AP. These values are more restrictive concerning the signal strength than the parameters found previously. However, these choices provide a more robust estimation if the environmental properties change. The effect of a small change in the true value of C, due to either variations in the power transmitted by the AP or different properties of the receiving device, will also be somewhat mitigated by a more restrictive choice of parameters.

6.3 Test Setup

During the remaining sections of this chapter two different test cases will be used.

Case 1:
Measurements of the RSSI are taken at equally spaced points with 0.5 meters spacing over a predetermined path. Only measurements from APs with known positions are used. The measurements are taken while the measuring phone is at rest at each point.

Case 2:
Measurements of the RSSI are taken continuously while the user is walking a predetermined path at a relatively constant speed. The time between each measurement is approximately 4 seconds. Only measurements from APs with known positions are used.

For *Case 1*, the distance between measurements can be changed in increments of 0.5 meters by using only a subset of the available data.

Case 2 provides a more natural use-case, where the user is moving while trying to find the position.

The tests were performed in a building called Glasgow, one of Sony's office buildings, on the 6:th floor. For different tests, different paths and different setups of APs are used. Further, all tests are performed in the same type of open office environment, but the differences between the exact environments for each test are fairly large. During the tests, the user held the smartphone in one hand, with the screen pointing upwards. The smartphones used were one Xperia Z Ultra and one Xperia Z1, both manufactured by Sony. These performed similarly except for a constant bias on the RSSI.

For both test cases, the estimated path is generated by connection of subsequent estimated positions.

6.4 Results

The results of the different tests performed are presented in the sections below. The different test paths are shown on a floor plan in Figure A.1 in Appendix A.

Environment One

The methodology from *Case 1* was used during this test and measurements from seven APs along a path of approximately one hundred meters. The locations of the APs together with the path taken may be found in Figure 6.5.

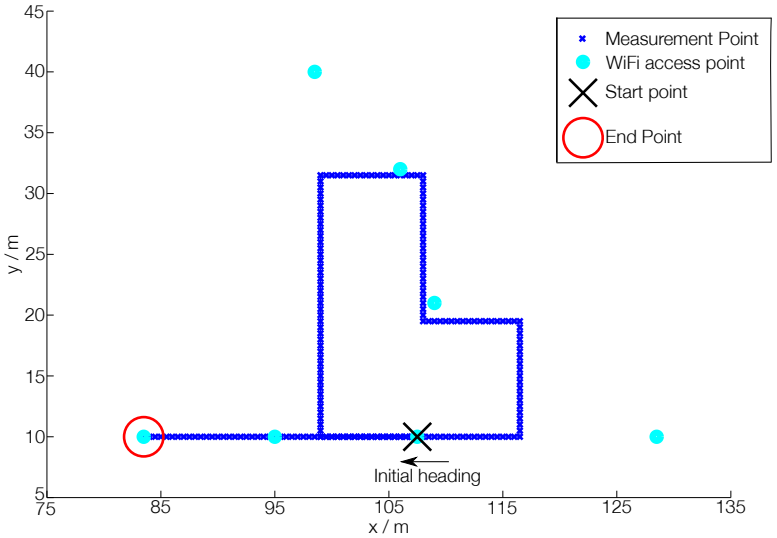


Figure 6.5 The path (*blue* line) used in *Environment one*, along with measurement points (*blue* `x`), start point (*black* `x`) and end point (*red* `o`).

	Error / meter		
	2.4 and 5 GHz	2.4 GHz	5 GHz
Mean	1.7	2.6	2.9
Maximum	4.5	5.3	5.8

Table 6.3 Attributes of the absolute positioning error for *Environment one*. It is clear that using 2.4 and 5 GHz WiFi in combination produces better results, both in terms of mean as well as maximum, than using either one alone.

The estimated walked path using RSSI measurements is presented in Figure 6.6 and the absolute position errors in Figure 6.7. Some additional data concerning the error is presented in Table 6.3 together with the positioning errors when using only 2.4 and 5 GHz for reference. In this environment, the placement of the AP in relation to the path walked is beneficial. During the walk, measurements from at least four APs are available and almost always on both 2.4 and 5 GHz. Further, the geometry of the APs is such that we always stay inside or at the border of a polygon where the APs compose the corners. These properties contribute to the fairly small positioning errors in Table 6.3.

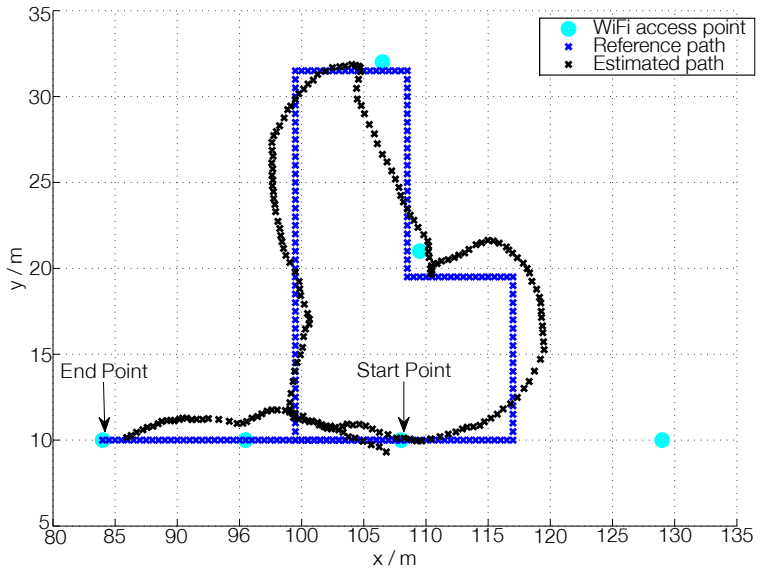


Figure 6.6 Comparison between the true (*blue*) and estimated (*black*) path for *Environment one*. The mean positioning error is 1.7 meters and the estimated path follows the true one relatively well during the entire walk.

Attribute	Meter
Mean	4.0
Maximum	11.5
Median	3.2

Table 6.4 Attributes for the absolute positioning errors for *Environment two*.

Environment Two

The test in *Environment two* uses the methodology of *Case 1*, and the measurements are taken over a distance of 176.5 meters.

A plot of the estimated path along with the true one is shown in Figure 6.8, and some attributes of the absolute positioning error can be found in Table 6.4. The maximum positioning error here is considerably worse than the maximum from *Environment one*. This is largely due to an unfavorable geometry of the APs during the last part of the path where only measurements from the three rightmost APs were available.

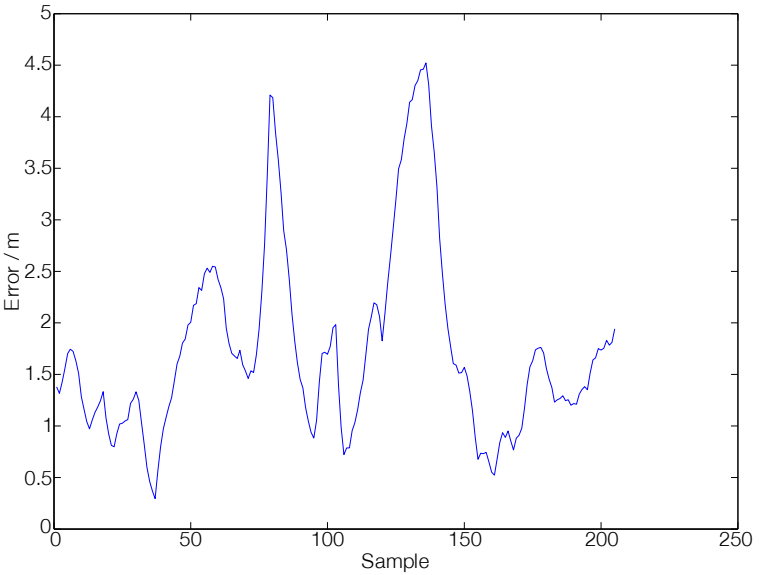


Figure 6.7 The absolute errors in estimated position for *Environment one*.

These APs compose a quite obtuse triangle further impairing the positioning.

Environment Three

For *Environment three*, which was a substantially longer path of 273 meters, the *Case 2* methodology was used. Measurements were received around once every 3rd second, resulting in the measurements being taken over a distance of around 4.5 meters, and a total of 61 different measurement points were collected.

In Figure 6.9, the true path along with estimated positions and path may be viewed. For this test case, it is hard to give an absolute measurement of the error. However, the measurements were taken while moving at constant speed and the time between each measurement was more or less constant. Thus, the estimated positions should be approximately equally spaced over the path. Estimates of the error could thus be obtained dividing it into a number of equally spaced points according to the number of measurements taken. Then these points can be compared with the estimated position in each scan, and the absolute difference in distance used as a measure of the positioning error.

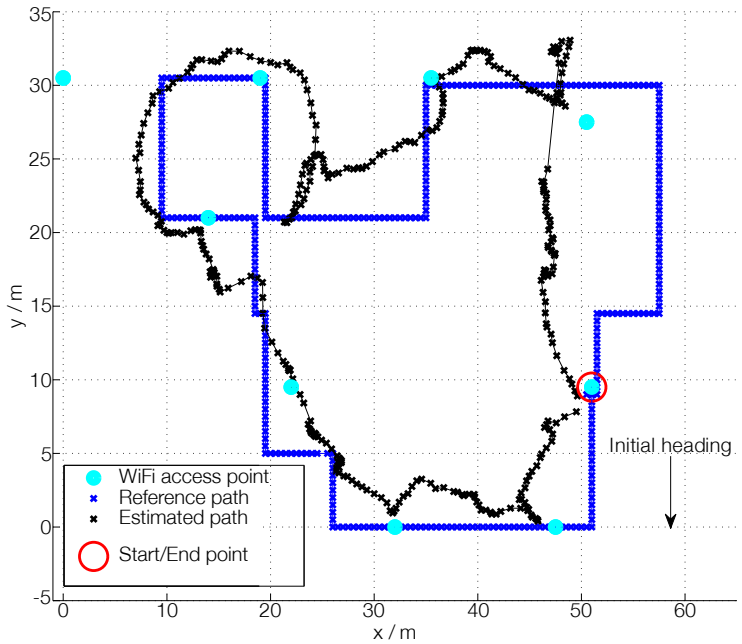


Figure 6.8 The true path (*blue*), along with estimated path (*black*), start point (*red* ∘), initial heading (*black* arrow) and AP (*cyan* ∘). The estimates follow the true path well for most parts of the walk, but a part near the end stands out where the errors are large (with a maximum of 11.5 meters). This is due to the unfavorable geometry, where only signals from the three rightmost APs are obtained. The mean error for this estimation is 4.0 m and the maximum error is 11.5 m.

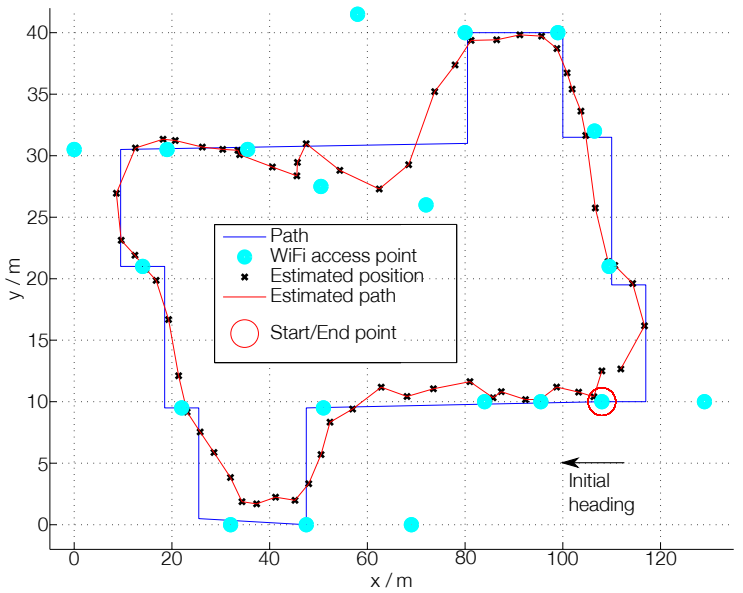


Figure 6.9 The true path (*blue*), along with estimated path (*red*), estimated positions (*black* `x`), start point (*red* `o`), initial heading (*black* arrow) and AP (*cyan* `o`). The estimated path is fairly close to the true one, but some of the estimated locations are around 10 meters from the true one. The mean error is 5.5 m.

The estimated path differs from the true one at times, but the difference is, in absolute terms, not large. Also, most of the estimated positions are equally spaced. However, some of them stand out and the minimum distance between two is 0.49 m and the maximum is 8.2 m. Another interesting observation is that the mean distance between two consecutive estimated positions is 4.1 m, giving a total distance of 250 m, which is 23 m less than the true path. This is also visible in Figure 6.9 in that the estimated path has a tendency to cut corners.

For this environment similar results were produced both when the phone was held in hand, placed in a front jeans pocket and carried in a bag. However, holding the phone in a certain manner to cover the antenna, resulted in a significant worsening of the performance. This will be investigated further in Chapter 7.

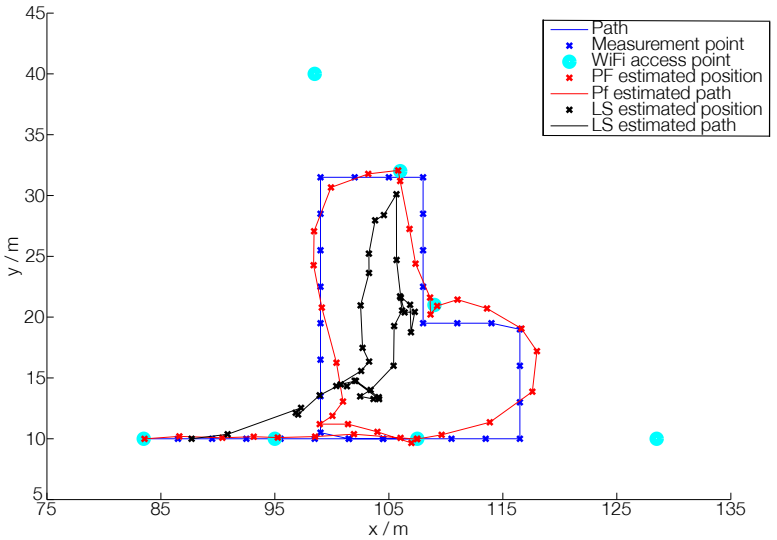


Figure 6.10 Least squares estimation versus PF estimation for *Environment one*. True path (*blue*), LS estimated path and position (*black*) and PF estimated path and position (*red*). It is clearly visible that the PF outperforms the LS algorithm in every respect in this estimation. The mean error is 1.7 meters for the PF and 5.7 meters for the LS estimation.

Least Squares - a Comparison

A common way of estimating a position from an RSSI measurement is to use an LS estimation, for additional information see Section 4.5. Here the difference between the estimations from a PF and from an LS algorithm will be investigated.

The comparison will be made for *Environment one* and *two* from the previous sections, but with new datasets. Further, the methodology from *Case 1* will be used, but with 3 m instead of 0.5 m sample spacing. In addition, both estimators will use the same path loss models.

The estimated paths from both the PF and LS estimators are displayed in Figure 6.10, along with the true path for *Environment one*. It is clearly visible that the PF performs considerably better compared to the LS estimator. The positioning errors from Figure 6.11 make this even more clear and the PF has a smaller positioning error for every sample compared to the LS estimator. The mean position error is 1.7 m for the PF, and 5.7 m for the LS estimation.

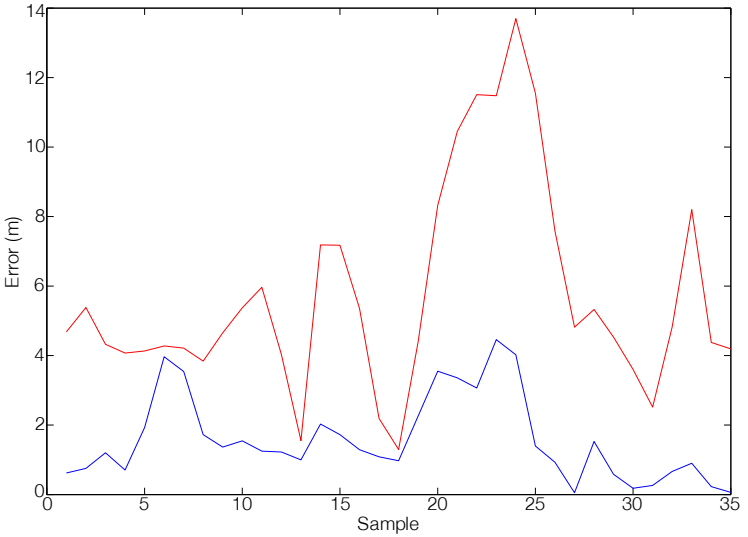


Figure 6.11 Absolute position error for the LS (*red*) and PF (*blue*) estimators for *Environment one*. The PF has a smaller estimation error for each of the estimated locations, and a considerably smaller maximum error.

For *Environment two*, the difference between the LS estimation and the PF are smaller compared to *Environment one*, as shown by the errors in Figure 6.12. The PF, however, does still provide the best position estimates, as the error is lower for the PF in most of the samples. The difference in maximum error is not large, but the mean error for the PF is 3.4 m, and for the LS estimation it is 5.1 m. The PF produces considerably better results compared to the LS method when the environment is favorable, i.e. when a large number of APs are available and/or their geometry is good. When the conditions are disadvantageous both methods perform equally, giving positions around 10 meters off.

The conclusion to be drawn from these investigations is that a PF is the better choice for estimating positions from noisy RSSI measurements, especially if the only sought after attribute is the smallest mean and maximum errors. The LS estimation may be desirable to use if computational power is limited, or if only measurements with long time separation are available.

An interesting aspect is the performance when no previous measurements are available. Then, the PF loses its advantage of using these, whereas nothing changes for the LS. A new test was performed in *Environment two*, and at each measure point,

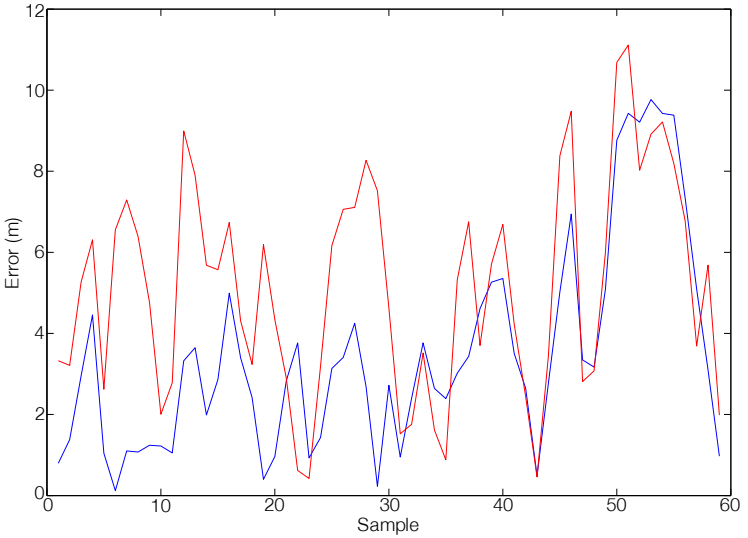


Figure 6.12 Absolute position error for the LS (*red*) and PF (*blue*) estimators for *Environment two*. The particle filter has a smaller mean error, 3.4 m, compared to 5.1. However, the PF is not better for every estimated position and the maximum error of both methods are comparable.

the previous position was treated as completely unknown. This means that the particles had to be spread out over the whole floor before each estimation. Then, one iteration was performed in the PF. The mean error was 5.09 meters for the PF and 5.14 meters for the LS. It is difficult to decide which method that works best in this case.

6.5 Concluding Remarks

Using measurements of RSSI to position devices in an indoor environment can be a satisfactory technique if the need for precision is not too large. In a fairly homogenous building, with a reasonable number of APs placed in an acceptable geometry, the positioning may be quite good. A reasonable number of APs is in this context at least three.

The position estimation may, with satisfactory result, be computed using a PF. The PF outperforms an LS estimation in most cases, and never produces inferior results for a sustained timespan. The worst case errors of the two estimators are comparable

when the environment is unfavorable, but the PF produces better estimates in a beneficial environment and smaller mean errors in the long run.

A few restrictions for the technique must however be mentioned. First and foremost, more or less all buildings need a specific path loss model, and the power which each AP transmits and their locations need to be known. Also, to have a chance to obtain a reasonable position estimate, measurements from at least three different APs must be available at all times. If fewer are available, larger positioning errors are to be expected. If only two APs are available, at least two positions will be equally probable, and if only measurements from one are available, all points at a certain distance may be the true position. The position history of the PF can improve the estimation when the geometry is unfavorable if the initial position is fairly accurate and the time spent in the environment is rather short. The LS estimation lacks this property and will produce erroneous and/or inconsistent estimates independent of previous positions. Further, antennas have different abilities to receive signals, thus the behavior of the specific device must be known. How the phone is carried seems to have small to no effect on the performance, except when the user covers the antenna. When that happens, the performance deteriorates severely, and large errors are introduced.

On the other hand, most newly constructed or renovated buildings already have numerous APs, and there, this technique may provide a simple and cost effective way of providing indoor positioning. If methods for finding APs and estimating the path loss models in a simple way can be attained, the setup may even be quite simple. If some of the user induced signal blockages can be registered and compensated for as well, the technique might be useful for tracking people even when they do not carry their device in hand. This only leaves the problem of characterizing the behavior of the antenna for each device, but as this only needs to be done once for each type of device, it is no overwhelming task.

7

Adapting the Model Parameters

In the previous chapter, the parameters α and C in the path-loss model

$$\log_{10}(P_r(d)) = C - 10n\log_{10}(d) - \alpha \cdot \min(d, d_{\max}) + X_\sigma. \tag{7.1}$$

were assumed to be known. However, these constraints are not necessary if the parameters can be estimated during the localization. This is useful since α depends on the current environment, and C depends on the radiated power from each AP. The first part of this chapter describes an algorithm for determining when the estimation of the location is accurate, and using this to estimate the model parameters.

Another issue is that the signals reaching the phone might be attenuated considerably by the user. The weaker signals would indicate that the phone is further away from each AP than it actually is. In the second part of this chapter, an algorithm to compensate for this is proposed and evaluated.

7.1 Estimation of Position Error

The method described in Chapter 6 has proven to perform better at a closer distance to the nearest AP. At a certain time step, a large maximum RSSI-value indicates that the nearest AP is close. This, in turn, indicates that the position error is relatively small. Figure 7.1 shows this relation. The data is from the test setup *Case 1* for *Environment two*, defined in Section 6.3 and 6.4, respectively.

Whenever one RSSI is above a certain value, the error can be considered small. This value depends on the model of the phone, and on C for the closest AP. In this example, -36 dB is a suitable lower limit. If one RSSI is larger than that at a certain time, the average error is 1.05 meters, and it is unlikely to get an error that is larger than 4 meters. Figure 7.2 shows the error together with a one-sided, upper bounded,

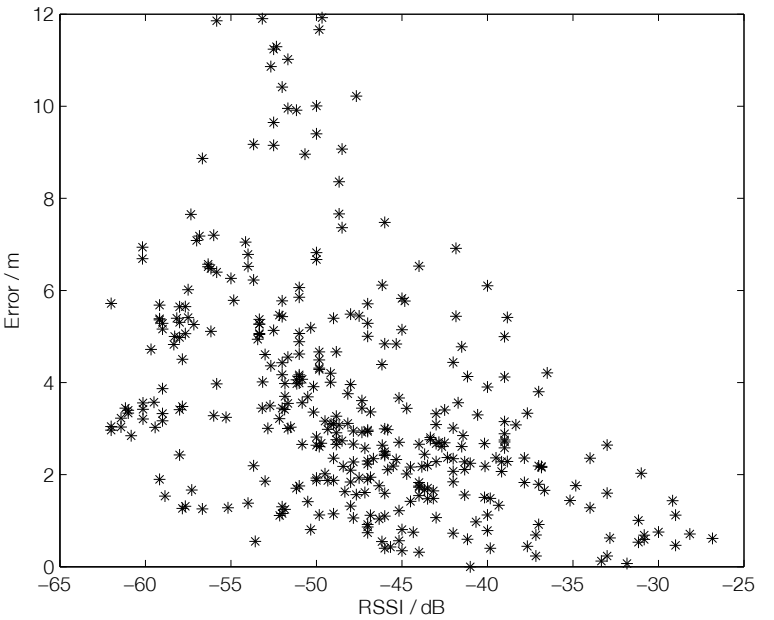


Figure 7.1 Obtained position errors versus the maximum received RSSI. There is a clear correlation between small positioning errors and a large value of the maximum received RSSI. The reason for this is that the RSSI values decrease faster with distance if close to the AP, which gives a better position resolution at short distances, and in extension larger RSSI values.

99 % prediction interval [Bickel and Doksum, 2001]. For other phone models and different APs , the limit has to be modified by adding a constant value, in order to get similar results. If C is not known initially, it has to be estimated in order to determine the limit, before the adaptation can begin. This can be solved when the user walks straight below an AP for the first time. By registering the peak RSSI, a very first approximation of C can be computed, and a reasonable limit determined.

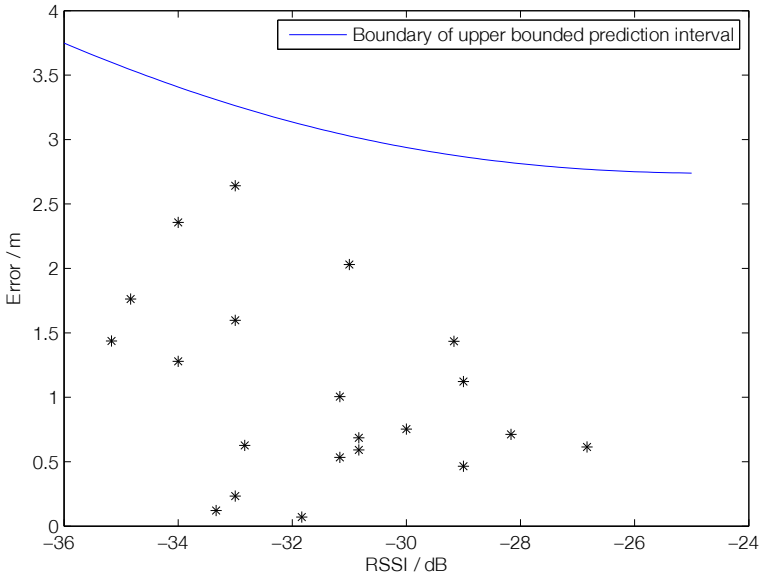


Figure 7.2 Obtained position errors versus the maximum received RSSI and a 99 % confidence interval for the error as a function of RSSI. Here, it is clear that the error can be deemed small (< 3.5 meters) if the maximum received RSSI is larger than −36.

7.2 Updating the Parameters

In order to successfully determine α and C, an accurate estimation of the position is required. The previous section describes how a maximum RSSI larger than a certain limit indicates a low position error. Therefore, the algorithm performs a calibration when this limit is exceeded. Expressions for each parameter can be determined by (7.1), which yields

$$\alpha = \frac{C - 10n\log_{10}(d) - \log_{10}(P_r)}{\min(d, d_{\max})} \tag{7.2}$$

$$C = \log_{10}(P_r) + 10n\log_{10}(d) + \alpha \cdot \min(d, d_{\max}) \tag{7.3}$$

The noise X_σ is unknown and omitted in the parameter update. Since α and C depend on each other, they should not be updated at the same time. Instead, C is updated for the nearest AP only, whereas α is updated using the RSSI from the others. When C is determined, the phone is assumed to be straight below the nearest

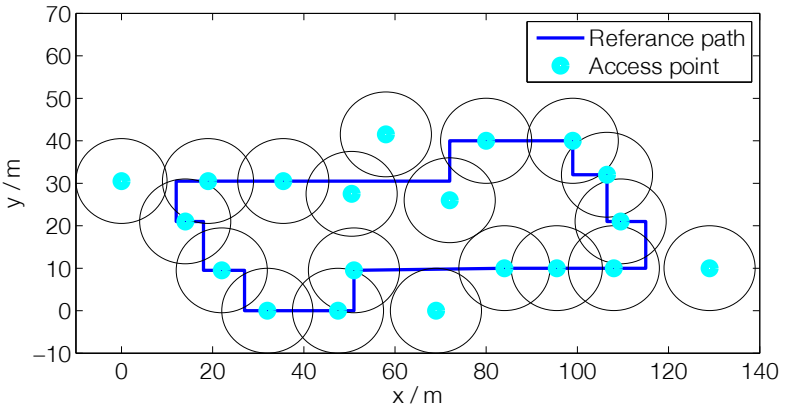


Figure 7.3 *Environment three* divided into sections. Each circle corresponds to a section around a certain AP.

AP, so that the distance d consists only of the vertical distance d_{ver}. In this case, d_{ver} = 2 m. Moreover, this AP is assumed to be within LOS, i.e. α=0. Hence, (7.3) can be rewritten as

$$C = \log_{10}(P_r) + 10n\log_{10}(d_{\text{ver}}) \tag{7.4}$$

Furthermore, initial estimations of the constants are required. These have to be accurate enough for the algorithm to perform a first successful parameter update. Hence, the position estimation has to be accurate enough when an update is performed. The value of C is specific for each frequency on each AP. Therefore, the algorithm should store one value of C for each frequency on each AP. The parameter α can be treated in different ways. Here follows two alternatives for this.

Alternative 1

The parameter, α, is the same for the whole building. After each determination of α_i, α_{ave} is defined as the average of all determinations, including the initial estimate.

Alternative 2

In this algorithm, each value of α_{ij} is stored in a matrix $\vec{\alpha}$. Here, α_{ij} is the parameter to be used in section i, for AP j. Each floor is divided into sections as follows:

Each AP corresponds to one circular section, centered at the AP and with radius r_s. These sections may overlap, and there might be areas that do not belong to any section (see Figure 7.3). Hence, the following three cases may occur for each particle:

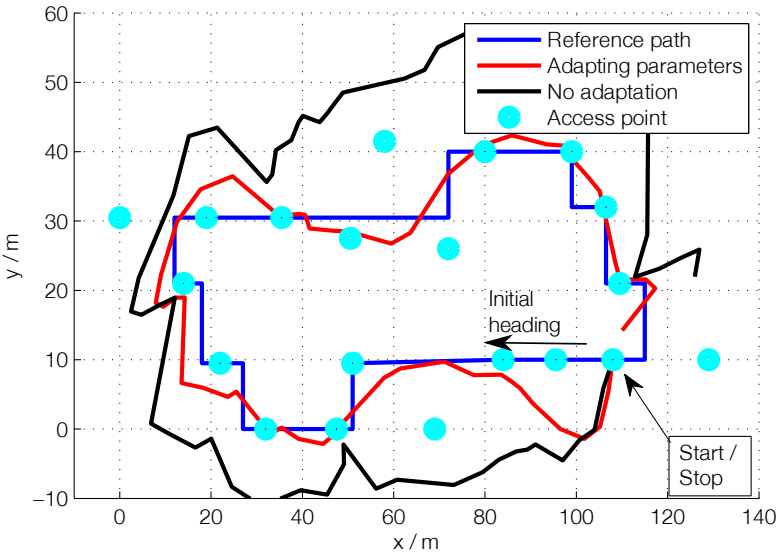


Figure 7.4 Estimated path with and without parameter estimation, using the erroneous parameters, $C = -25$ and $\alpha = 0.3$ as initial values. The mean error was 7.7 m and 18.3 m, respectively. The algorithm performs considerably better when the adaptation is performed, compared to when erroneous model parameter are used. However, parameters obtained after a extensive survey of the signal environment still produces the best results.

1. **Exactly one section** If a certain particle is in section k only, the parameter set from row k in alpha will be used for this particle.

2. **Several sections** If it is in several sections at the same time, an average of the parameter sets for these sections will be used.

3. **No section** If it is outside all sections, a default value of α will be used.

Note that each particle gets its own set of parameters, and that the estimated position of the phone is not involved in choosing this set. Both alternatives have been tested offline using MATLAB. For the environments defined in Section 6.4, the results are similar. The reason why there is no gain in dividing the building into areas is that it is homogeneous enough for alternative 1 to work. In an environment where the obstacle density changes more, alternative 2 would be more suitable.

Initial values		Values after adaptation			
2.4 and 5 GHz		2.4 GHz		5 GHz	
C	α	C	α	C	α
-50	1.0	-37.7	0.28	-35.8	0.35
-45	0.9	-37.7	0.25	-35.8	0.34
-35	0.5	-35.6	0.35	-33.7	0.42
-25	0.3	-31.5	0.47	-29.6	0.54

Table 7.1 Parameter values before and after adaptation. If the initial parameters are chosen far from the true ones, the adaptation manages to find parameters fairly close to the measured ones.

In order to evaluate the parameter updates, the algorithm was given erroneous initial parameter values. Then, the phone was carried one round in *Environment three*, and after that the updated parameters were noted. During the round, 61 RSSI measurements were taken by the phone, giving an approximate spatial separation of 4.5 m. The resulting parameters are shown in Table 7.1

Implausible parameter values (e.g. $C = -50$ or $C = -25$) are more reasonable after the adaptation. Moreover, the algorithm moves the parameter values towards the region with relatively low position error in Figure 6.4. The bottom row in Table 7.1 was investigated in more detail, as an example of how the adaptation affects the positioning. Figure 7.4 shows the positioning with and without the parameter adaptation, respectively, with $C = -25$ and $\alpha = 0.3$ as initial values.

If more data points were collected, it is probable that the estimated parameters would converge closer to the measured parameters from 6.2.However, the WiFi signals are severely noisy and the estimated positions where the measurements were obtained are not exact. Therefore, a complete convergence is not expected.

7.3 Compensating for Signal Blocking

The RSSI is affected by how the phone is carried by the user. If, for example, the user blocks the signals with the hand, the RSSI will get lower, which indicates that the phone is further away from each AP than it actually is. In this section, this effect, and the possibility of compensating for it, will be evaluated.

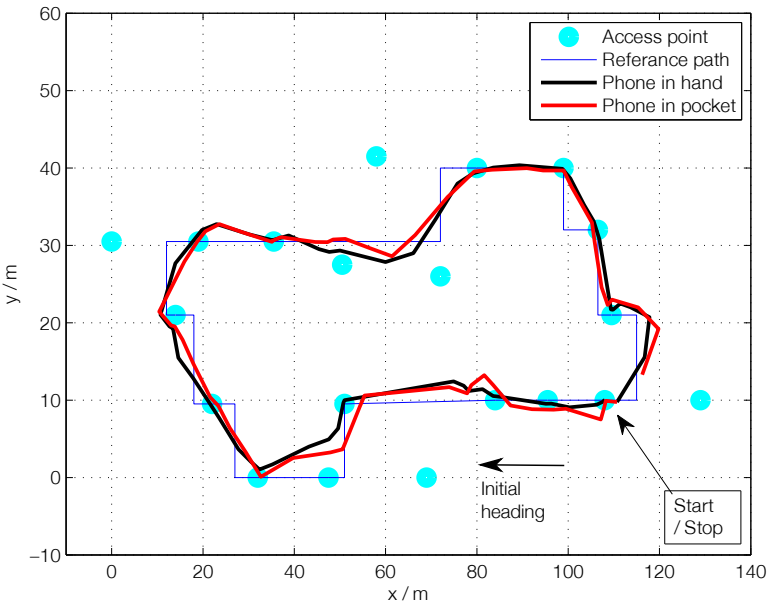


Figure 7.5 Phone in pocket. Wearing the phone in the pocket did not affect the positioning significantly. The mean error was 5.3 m when carried in the hand, and 5.6 m when carried in the pocket. There seems to be little difference between carrying the phone in the pocket compared to the hand, which both the mean estimation error and the estimated path point to. A third estimation when the phone was carried in a bag was also performed, but the behavior was so close to the other two, it was decided to be omitted.

1. **Evaluation of different use cases** In the tests this far, the phone has been held so that the signal is blocked as little as possible. In this section, three different use cases are tested, all in *Environment three*.

2. **Phone in pocket** In this test, the phone was first carried one round in the hand. After that, it was moved to the jeans side pocket, and then another round was walked.

3. **Blocking with hands, once** Again, the phone was carried one round in the hand. Then, the user tried to block as much of the signals with the hands as possible during the second round.

4. **Blocking with hands, repeatedly** Like the previous test, but during the second round, the signals were blocked and unblocked repeatedly.

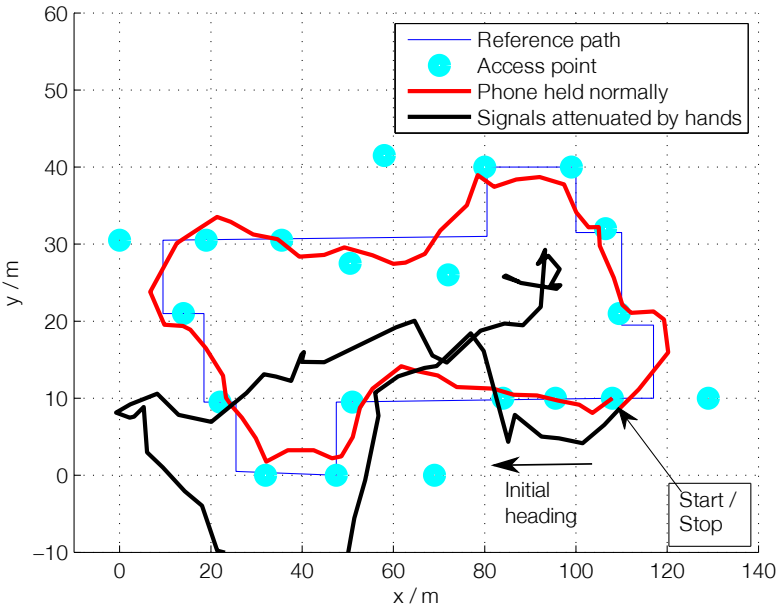


Figure 7.6 Blocking with hands, once. Attenuation by the hands had a severe impact on the positioning. The mean error was 6.3 m while carrying the phone normally, and 38.5 m while blocking. Here, a substantial block of the incoming signals was performed. A large positioning error is present soon after the blocking was initialized.

The results are shown in Figure 7.5 , 7.6 and 7.7.

Carrying the phone in the pocket did not affect the RSSI significantly, and hence the positioning was unaffected. However, blocking the signals with the hands lowered the RSSI considerably, causing the positioning to be inaccurate. Hence, this use case has to be detected and compensated for.

Compensation

The main strategy for detecting the attenuation is to note when the RSSI decreases fast. However, moving away from the APs, or walking behind a wall, may lower the RSSI in a similar way. In order to distinguish the signal attenuation by the user only, the algorithm focuses on the six largest RSSI-values. If fewer than six signals are available, every received signal is used. If the average of these decreases more than 6 dB in one time step, this indicates a hand attenuation. Moving away from the APs would not make the RSSI decrease that fast, and if LOS is lost to some

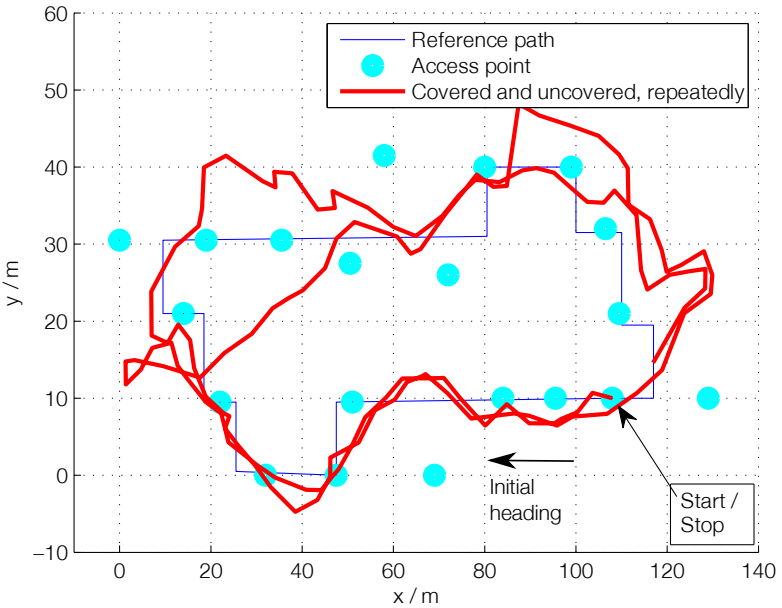


Figure 7.7 Blocking with hands, repeatedly. Attenuation by the hands had a severe impact on the positioning. The mean error was 9.3 meters. The error here is not as large as for the case where the signals were attenuated for a long stretch of time. However, a large mean error was obtained.

APs there is a chance that LOS is established to another. If, on the other hand, LOS is lost to several APs without gaining LOS to other APs, e.g. by entering a room without an AP, the compensation may be triggered. However, this is more likely to improve than to degrade the positioning as the standard RSSI models do not incorporate quick drops in signal level caused by walls and similar objects. If a user enters a room without an AP, the RSSI measurements would drop significantly for all APs, but the compensation would ideally remove this effect, allowing the walls to be taken into account by the positioning algorithm.

When an attenuation is detected, a constant term H is added to the model, which corresponds to how much the signals have been reduced. Hence, the new model is given by Equation (7.5). When the phone is unblocked, this is detected in an analogous way, and then H is set to 0.

$$\log_{10}(P_r(d)) = C - 10n\log_{10}(d) - \alpha \cdot \min(d, d_{\max}) + H + X_\sigma. \tag{7.5}$$

Test 1) and 2) were performed again, but this time with the compensating algorithm. The results are shown in Figure 7.8 and 7.9. Compared to the results shown in Figure 7.6 and 7.7, the compensation algorithm results in a significant improvement to the positioning. This is especially true when the signals are attenuated for a relatively long time, clearly visible when comparing Figure 7.6 to 7.8. However, the positioning is aided by the compensation algorithm even when the attenuation is present for a short time.

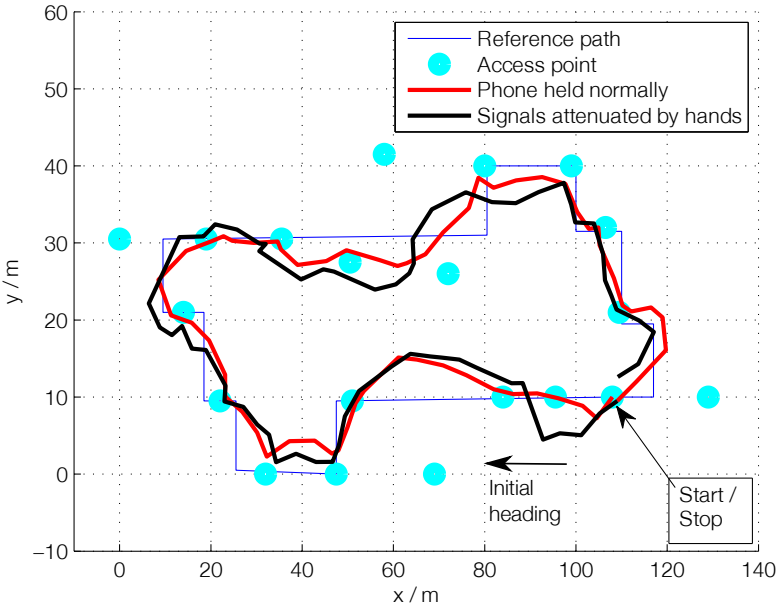


Figure 7.8 Blocking with hands, once. The algorithm compensates for the attenuation. The mean error was 7.3 m while blocking, and 7.2 m while carrying the device normally. Compared to Figure 7.6, this algorithm has a much better performance, and it is hard to discern between the compensated and the non-blocked part of the estimation.

7.4 Conclusion

In this chapter, an algorithm that updates the model parallel to the positioning has been suggested. The purpose is to increase the generality of the application, and to make it improve while in a new environment. If the initial parameter values are inaccurate, this part of the algorithm improves the positioning significantly. However,

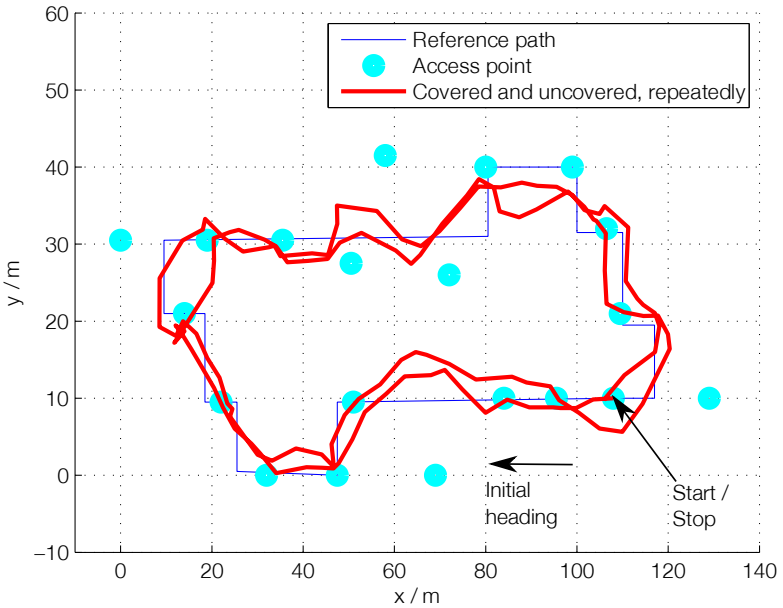


Figure 7.9 Blocking with hands, repeatedly. The algorithm compensates for the attenuation, which supports the positioning. See Figure 7.7 for a comparison with the same use case without compensation.

if the initial parameter values are too far from the true ones, the positioning never gets accurate enough for the parameter update to work. Hence, the algorithm has to start with feasible parameter values in order to make them more accurate. In order to achieve this, C could be estimated from the peak RSSI values.

Compensating for the hand attenuation clearly improved the positioning. However, H can not be determined exactly, due to noise and obstacles in the environment. In addition, the hand attenuation is only detected if the phone is covered fast enough. If it goes from uncovered to covered successively, during several time steps, this can not be distinguished from other causes of lowered RSSI. Naturally, going from covered to uncovered works analogously.

8

Modeling the Kinematics

Modeling the kinematics may greatly improve the performance of a positioning system, especially if the estimated position is noisy or biased. The kinematics may also be used to estimate the position in between the updates from the main positioning algorithm. The range of models stretches from simple random walk processes to a more elaborate modeling of a walking person, including step detection and estimations of the step length and heading by using an accelerometer and a gyroscope.

After a short description of random walk processes, quaternions and their use in three-dimensional pose estimation are introduced. A section about positioning using only sensors, named *dead reckoning* (DR), is presented, after which the chapter closes by a discussion of kinematics models used in indoor positioning and their effects on the performance.

8.1 Random Walk

The simplest way of modeling the kinematics, aside from the trivial case of choosing not to model it at all, is to use a *random walk* model for each state. In a random walk the states, x, undergo time updates according to

$$x_{k+1} = x_k + v_k \tag{8.1}$$

where v_k is a random process. Usually v_k is considered independent for each state, and depending on the distribution and standard deviation σ, different random walk processes are obtained. How v_k should be distributed is determined by the intrinsic behavior of the states. A common choice is a normal distribution with an appropriate σ. Such a random walk is called a *Gaussian random walk*, and if σ is very small it resembles a *Brownian motion*. Four series of Gaussian random walks in two dimensions, each starting in $(0\ \ 0)$, are displayed in Figure 8.1.

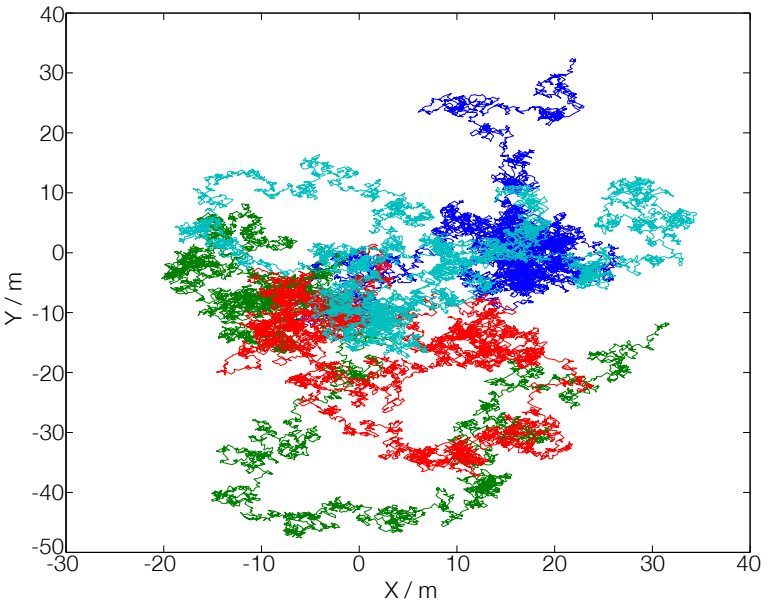


Figure 8.1 Four Gaussian random walks using a standard deviation $\sigma = 1$ and an expectation value of 0. Each walk starts in the origin, and then 10^4 steps are taken. This is a simple way of modeling the kinematics, i.e. of updating the states of the particles. It is clearly visible that a single random walk may end up almost anywhere and that the potential distance from zero grows with the number of steps taken. However, the mean position of many walks are $(0,0)$

8.2 Quaternions

The *quaternions* are an extension of the complex numbers, first described in 1843 by the Irish mathematician William Rowan Hamilton. They were first used in three-dimensional mechanics and today they are, among other applications, used to estimate the pose (position and direction in relation to a fixed frame) in three dimensions [Kuipers, 1998; Mannesson, 2013].

Quaternion Theory

A quaternion, q, is a set of numbers, extending the complex numbers, defined as

$$q = a + bi + cj + dk, \quad [a,b,c,d] \in \mathbb{R} \tag{8.2}$$

where *i, j* and *k* are new basis elements.

$\times$	1	i	j	k
1	1	i	j	k
i	i	-1	k	$-j$
j	j	$-k$	-1	i
k	k	j	$-i$	-1

Table 8.1 All possible basis multiplications for quaternions.

Further, the basis elements obey

$$i^2 = j^2 = k^2 = ijk = -1 \tag{8.3}$$

Thus, from (8.3) all possible basis multiplications can be formed according to Table 8.1. A quaternion is usually seen as a four-dimensional vector consisting of a scalar q_0 and a vector $\vec{q} = (q_1 \;\; q_2 \;\; q_3)^T$, leading to the following notation

$$q = \begin{pmatrix} q_0 \\ \vec{q} \end{pmatrix} = \begin{pmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{pmatrix}. \tag{8.4}$$

Quaternion multiplication is denoted by $\otimes$ and defined as

$$q \otimes r = \begin{pmatrix} q_0 \\ \vec{q} \end{pmatrix} \otimes \begin{pmatrix} r_0 \\ \vec{r} \end{pmatrix} = \begin{pmatrix} q_0 r_0 - \vec{q} \cdot \vec{r} \\ q_0 \vec{r} + r_0 \vec{q} + \vec{q} \times \vec{r} \end{pmatrix} \tag{8.5}$$

for two quaternions q and r, where $\cdot$ denotes the scalar product and $\times$ denotes the vector product. The quaternion multiplication is non-commutative, meaning

$$q \otimes r \neq r \otimes q \tag{8.6}$$

in general. It is, however, associative

$$(q \otimes r) \otimes s = q \otimes (r \otimes s). \tag{8.7}$$

Further, the *unit* quaternion is defined as

$$q_{\mathbf{I}} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{8.8}$$

and the quaternion inverse q^{-1} is

$$q^{-1} \otimes q = q \otimes q^{-1} = q_{\mathbf{I}} \quad \Leftrightarrow \quad q^{-1} = \frac{1}{|q|^2} \begin{pmatrix} q_0 \\ -\vec{q} \end{pmatrix} \tag{8.9}$$

where the quaternion norm $|q|$ is defined as

$$|q| = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2}. \tag{8.10}$$

Quaternions Interpreted as Rotations

Quaternions with the property $|q| = 1$ may be interpreted as rotations, although the reason for this is beyond the scope of this thesis. Instead, here follows a recapitulation of how they can be used. Readers interested in the mechanisms involved are referred to [Kuipers, 1998] for an in-depth discussion of the background and theory of quaternions.

Rotations are defined by two elements, the amount (angle) of rotation and around what axis the rotation is performed. Suppose that a rotation of θ is performed around the three-dimensional unit vector $\vec{n}$. The quaternion representation of this would be

$$q = \begin{pmatrix} \cos\frac{\theta}{2} \\ \vec{n}\sin\frac{\theta}{2} \end{pmatrix}. \tag{8.11}$$

Finding the vector $\vec{v}$ rotated clockwise by θ around $\vec{n}$, noted $\vec{v}_r$, by use of quaternions is done as

$$\begin{pmatrix} 0 \\ \vec{v}_r \end{pmatrix} = q \otimes \begin{pmatrix} 0 \\ \vec{v} \end{pmatrix} \otimes q^{-1}. \tag{8.12}$$

Many methods of computing rotations suffer from singularities when the rotations are performed around a certain axis. This is not the case for quaternions, e.g. if $\vec{v} = (0\ \ 1\ \ 0)$ is rotated an angle θ around $\vec{n} = (1\ \ 0\ \ 0)$, $v_r = (0\ \ \cos\theta\ \ \sin\theta)$ is obtained as expected. For e.g. *Euler angles*, this may not be the case, depending on the frame of reference [Slabaugh, 1999].

From here on, we use the naming convention

$$\vec{v} \equiv \begin{pmatrix} 0 \\ \vec{v} \end{pmatrix}. \tag{8.13}$$

The rotation of a vector, $\vec{v}$ by a quaternion, q, and subsequently by p, is equivalent to rotating $\vec{v}$ by the corresponding quaternion product $(p \otimes q)$. This is easily proven by multiple applications of (8.7),

$$(p \otimes q) \otimes \vec{v} \otimes (p \otimes q)^{-1} = p \otimes (q \otimes \vec{v} \otimes q^{-1}) \otimes p^{-1}. \tag{8.14}$$

When using sensors, in particular a gyroscope, to compute the rotation, it is common to end up with integrated angular velocities on the form $\vec{r} = (r_x\ \ r_y\ \ r_z)^T$. Each element in $\vec{r}$ characterizes the rotation around an axis in an orthogonal coordinate system, in a relatively short timespan. To describe this rotation using quaternions, the direction around which the rotation is performed can be viewed as the direction of $\vec{r}$, and the vector norm $|\vec{r}|$ may be considered the angle of rotation. The corresponding quaternion, q, is then

$$q = \begin{pmatrix} \cos\frac{|\vec{r}|}{2} \\ \frac{\vec{r}}{|\vec{r}|}\cos\frac{|\vec{r}|}{2} \end{pmatrix} \tag{8.15}$$

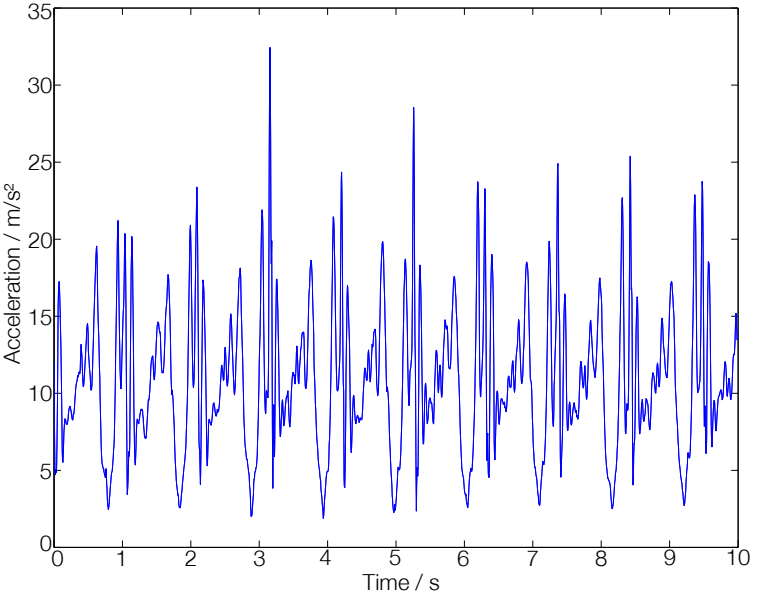


Figure 8.2 Acceleration during 10 seconds of a typical walk, when the phone is carried in the hand. Here, it is hard to distinguish when a step has been taken. In order to make this easier, the signal is first differentiated, to center it around zero (see Figure 8.3), and then filtered (see Figure 8.4).

8.3 Characterization of Steps

Characterizing a step allows a kinematics algorithm to keep track of the approximate distance traveled, as the length of a particular individual's steps is relatively regular. A way to identify whether a step has been taken, could be constructed using an accelerometer [Ying et al., 2007]. The accelerometer gives output on the form $\vec{a} = (a_x \;\; a_y \;\; a_z)^T$, each element representing the acceleration along each of the phone's coordinate axis. However, the orientation of the phone during a walk affects how each element in the acceleration vector behaves. A phone placed in a pocket or a bag gives vastly different outputs. A more robust way is to consider the magnitude of the acceleration, $|\vec{a}|$, defined similar to the vector norm as

$$|\vec{a}| = \sqrt{a_x^2 + a_y^2 + a_z^2}, \tag{8.16}$$

which is considerably more consistent between use cases.

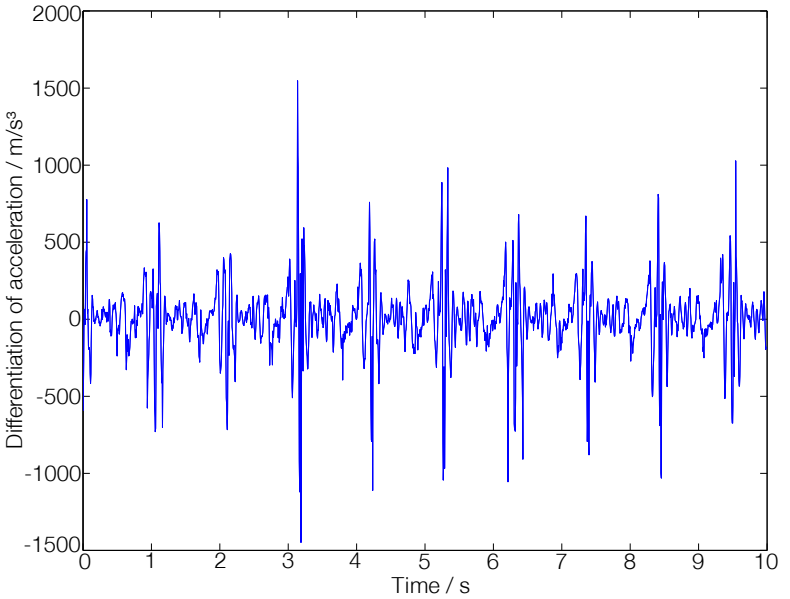


Figure 8.3 Differentiation of acceleration during 10 seconds of a typical walk. This signal is low-pass filtered before the steps are detected (see Figure 8.4).

A 10 second sample of the average acceleration measured by a phone placed in the front pocket during a typical walk is displayed in Figure 8.2. A flaw of this approach is easily noticed. Due to gravity, the average acceleration is centered around 9.81 m/s^2. A way around this is to differentiate the signal according to

$$\Delta f(x) = \frac{f(x+\Delta t) - f(x)}{\Delta t}, \tag{8.17}$$

yielding the signal depicted in Figure 8.3. This signal is centered around zero, but possesses unnecessarily high frequency behavior. The sampling frequency of the accelerometer is $\sim$ 20 Hz, giving a frequency content of the signal between 0 and 10 Hz [Proakis and Manolakis, 2007]. The normal step frequency for a person is no larger than 3 Hz, which promotes filtering the signal. Doing this with a fourth order *Butterworth filter* with a cut-of frequency of 3 Hz results in the signal shown in Figure 8.4.

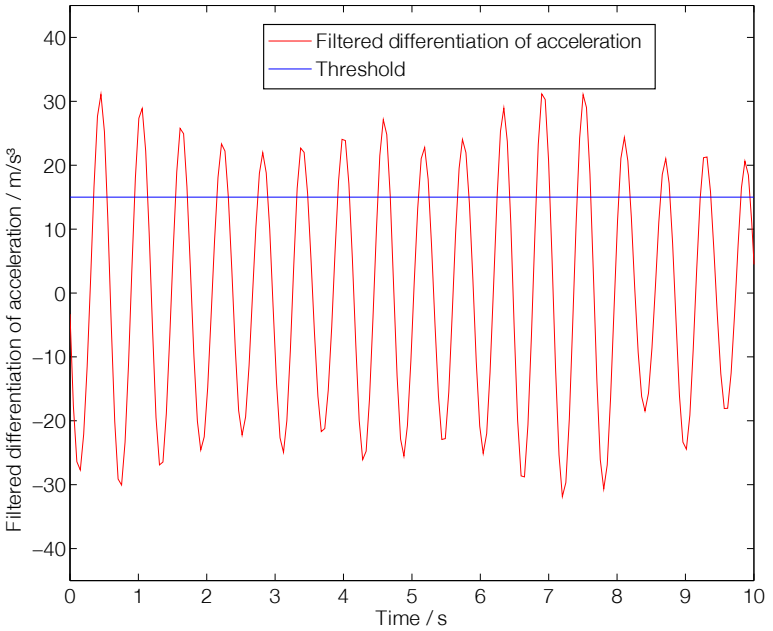


Figure 8.4 Filtered differentiation of acceleration during 10 seconds of a typical walk with the phone carried in hand (red), and a threshold level (blue). Here, each step is considerably easier to distinguish than in Figure 8.2 and 8.3. When the signal goes from below to above the threshold, this is used to indicate that a step has been taken. In this case, 17 steps were taken and registered. This step counter works regardless of if the phone is carried in a bag, in a pocket or in the hand.

Here, the steps are clearly distinguishable and each peak corresponds to a step taken. In Figure 8.4 a threshold for determining whether a step has been taken or not is also present. A simple rule to count the number of steps taken could be to count the number of times the signal passes from below to above the threshold. Using this rule 17 steps were taken during the 10 seconds interval displayed in Figure 8.4.

An algorithm corresponding to the rule above has been implemented in Android using Java. To test the performance of the algorithm two walks of 100 steps respectively were taken, one with the phone in hand and one with the phone in the front right jeans pocket. When the phone was carried in the hand, 101 steps were registered and when carried in pocket, 100 steps were registered. This shows that the simple algorithm produces satisfactory results in these common use cases. The technique could be refined further, e.g. by identifying the step frequency or by using more sensors.

Step Length Considerations

The step length of an individual, moving in a straight line with a constant speed, is more or less constant. Different individuals or the same individual moving at different speeds or performing turns have a quite large span of step lengths. In the parts of this thesis where a fixed step length is used, the step length has been determined by the authors walking a predetermined distance comprised of walking straight interspersed with turning both left and right, at a constant speed. The average step length is then simply computed as the distance divided the by number of steps and a value of 0.77 m was obtained.

For real world usage however, the length of a user's steps must be determined. This can be done in various ways. One is to have the user estimate his/her own step length with the same method as above. Another is to adaptively tune the step length by using approximately known distances traveled and the number of steps taken whilst traveling. A way of doing this is to use some sort of positioning technique (GPS, WiFi etc.) to estimate the distance while the step counter determines the number of steps.

An alternative method is to add a state in the PF corresponding to step length, and spread the particles accordingly. Particles with either too long or too short step lengths would agree badly with measured RSSI-values and thus be given small weights. A downside of this is that increasing the number of states would require more particles as discussed in Section 4.1, and consequently increase the computational power required to implement the filter.

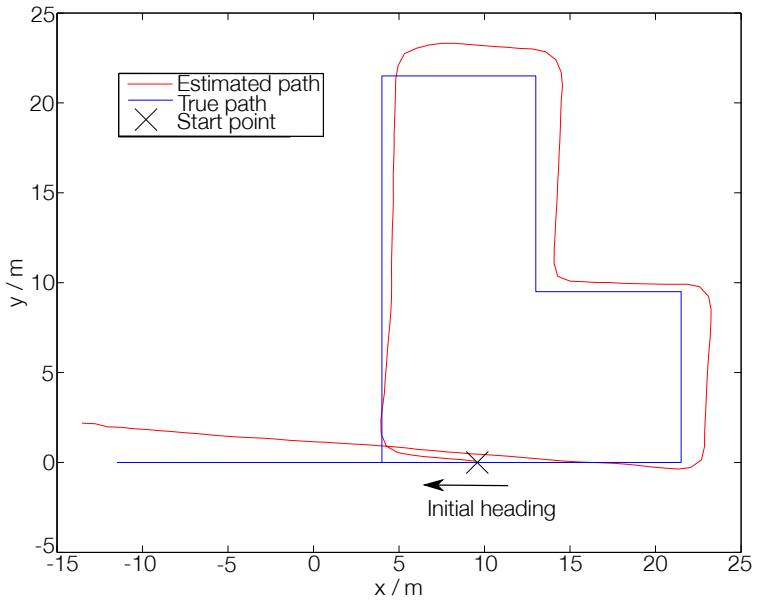


Figure 8.5 True versus, by PDR, estimated path using gyroscope and pedometer. The initial heading is in the negative x-direction. The error gets larger with time, but is only about three meters off after a of 110 meters walk. However, as Figure 8.6 shows, the performance only deteriorates form here.

8.4 Dead Reckoning

Dead reckoning (DR) is the process of calculating the current position based on a previously determined position and advancing that position based on a known model [Beauregard and Haas, 2006]. DR has been used by ships and airplanes for a long time, and in more recent time in networked computer games [Murphy, 2011]. However, the error is cumulative and the performance deteriorates with time. This is shown in Figure 8.6. The distance around the rectangle is approximately 105 meters, and the error in heading and position is clearly visible.

DR estimations are subject to cumulative errors, i.e. errors are added over time and depending on the accuracy of the DR system, the estimated position deviates from the true one. This implies that the position and heading estimates need to be corrected at a certain time interval to provide an accurate position. For phones, a method which extends normal DR, called *pedestrian dead reckoning* (PDR) could be used [Beauregard and Haas, 2006]. Position and direction estimation would most

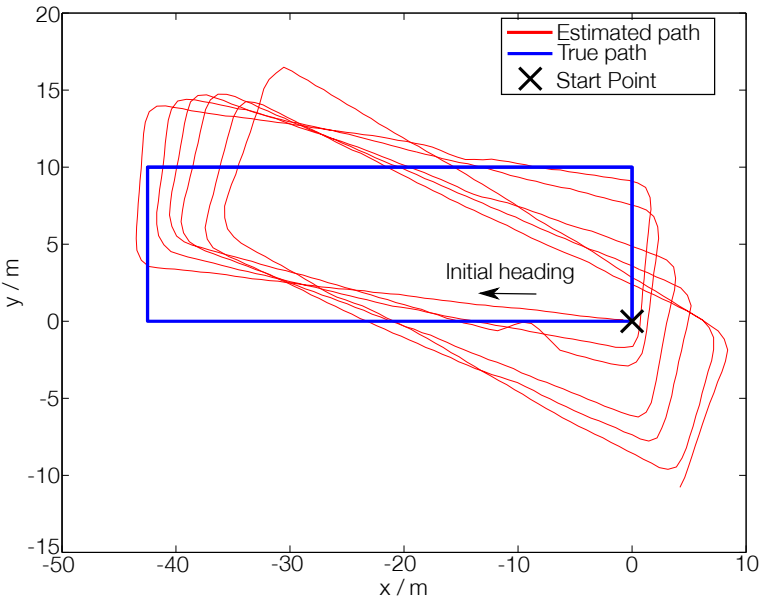


Figure 8.6 True versus, by PDR, estimated path using pedestrian dead reckoning using the same method as in Figure 8.5 but for a longer time. The error gets larger with time and after approximately 630 meters walked, the error is around 12 meters and the heading is almost 30° off.

likely include some form of step counting for distance measurement combined with either gyroscope or a sensor fusion including the magnetometer for heading estimation. As an example, the true and estimated paths using a pedometer and a gyroscope during a walk of approximately 110 meters, are presented in Figure 8.5. The behavior is fairly good but there is a heading error present soon after the start, and this grows as the walk continues. The distance estimated by the pedometer is also larger than the true distance traveled. Further, the estimation error grows with time, as seen in Figure 8.6, making PDR useful only for short term estimation, if no correction of position and direction is performed.

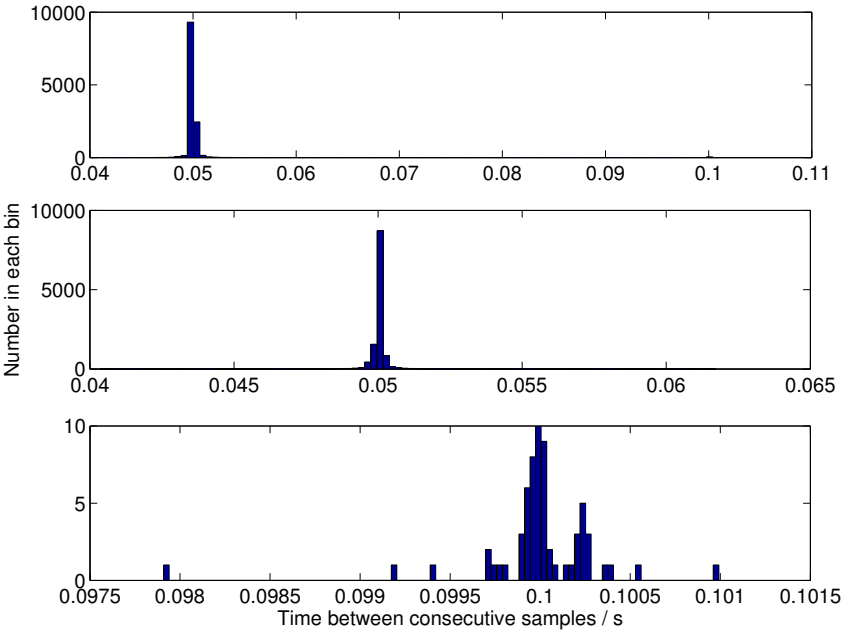


Figure 8.7 Histogram over time difference between consecutive samples given a specified sampling frequency of 20 Hz. Each interval is divided into 100 bins and the width of each bin is $7 \cdot 10^{-4}$, $2.5 \cdot 10^{-4}$ and $4 \cdot 10^{-5}$ seconds respectively. The time differences shown in the bottom figure shows occasions when a sample has been skipped by the subsystem, which occurs about once every 2000 samples. Due to the large number of data points these are almost indistinguishable in the first plot.

8.5 Dealing With a Non-Constant Sampling Rate

The Android platform employs a sampling strategy called *event based sampling*. An application may request sensor data with a specified time separation, e.g. 50 ms for a sampling frequency of 20 Hz. However, the system will forward data to the application when a new value is available in conjunction with the specified rate. The result is sensor data sampled at slightly different time intervals. Further, if the system is busy when the requested data is to be obtained, it may choose to skip that sample, resulting in a de facto doubling in sample time for the next sample. Disregarding sensor samples is more common when the system is under heavy load, which results in more samples being skipped.

Figure 8.7 presents a histogram over the time between consecutive samples for a set of 12 375 accelerometer data points with a specified time between samples of 50 ms. The first figure shows that most time differences are around 50 ms, which is to be expected. If the data is studied more closely, a small number of points is visible around a separation of 100 ms. These are generated by the system disregarding a sample, resulting in an approximately double time difference. The two following figures consist of the data divided into two intervals, one with time differences ranging from 40.3 ms to 61.7 ms, and one with a range from 97.9 ms to 101 ms. The first of these intervals consists of 12 311 data points, and the second of 64 points.

Only a small fraction of the samples are skipped by the system. However, these might introduce unwanted behavior when the data is filtered. The *Butterworth* low pass filter used in the step detection algorithm is designed by a cutoff frequency being chosen as a fraction of the *Nyquist frequency* (half the sampling frequency). A small difference from the desired sample time only affects the cut-off frequency and filter behavior slightly. A missed sample, on the other hand, causes the cut-off frequency and the behavior of the filter to change drastically.

This must be addressed for the aforementioned step detection algorithm to function as intended. A straight forward way to do so is to use *linear interpolation*. Given two data points, x_0 and x_1, separated in time by Δt, a new sample in between could be constructed as,

$$x_{1'} = x_0 + \frac{x_1 - x_0}{2}. \tag{8.18}$$

The resulting sequence, x_0, $x_{1'}$ and x_1, now has a time separation of $\Delta t/2$ and $x_{1'}$ is located in the middle of x_0 and x_1 on the straight line connecting the two. This would of course not capture any of the signal's behavior in between the two data points, but the frequency content missed would be of the order of $1/(2 \cdot \Delta t)$ and higher. Thus sampling the signal at a sufficiently high rate would allow for some interpolation without missing the frequency behavior of interest. This simple interpolation scheme can easily be extended to accommodate for more samples being introduced between x_0 and x_1.

There is a possibility for the system to skip several consecutive measurements. For the step detection algorithm previously described, the frequencies of interest were below 3 Hz, resulting, due to the *Nyquist sampling theorem*, in a minimum sampling frequency of 6 Hz. In the algorithm, a sampling time of 50 ms is used and to fully resolve the sought after frequencies, a sampling time of 166 ms or faster is needed. This implies that at least three consecutive samples would need to be skipped to corrupt the desired frequencies. From the data set illustrated in Figure 8.7, the approximate probability of missing a single sample is $64/12375 \approx 0.005$. If the system uses the same priorities when deciding if a sample should be skipped regardless of if the previous sample was skipped, which in itself is unlikely, the probability of two consecutive samples being skipped is $2.5 \cdot 10^{-5}$. Further, to skip

three samples a collection of, on average, 8 million samples would be needed. With a sampling frequency of 20 Hz, this would take 400 000 seconds, or a little over 4.5 days. Even a tenfold increase in the probability of skipping a sample would potentially cause either a missed detection or extra step detected on average once every 7th minute, which still is acceptable.

8.6 Direction of Movement

To have a good estimation of the movement, not only the distance traveled needs to be known. More important is the direction in which the movement takes place. In addition, this task is more complicated than finding the distance by the number of steps taken. Many approaches are available using a wide span of technologies.

Integrated Gyroscope

An enticing opportunity to determine the direction of movement using the gyroscope is, given a known initial heading, to integrate the gyroscope output using e.g. the trapezoidal rule. This would give three angles describing the gyroscope's current orientation in reference to the initial one. This approach, however, has two major flaws. The MEMS gyroscopes possess biases and drifts making the estimated orientation drift over time. This makes the gyroscope suitable only for estimations over short intervals of time. Further, the gyroscope only measures its own angular velocity and in extent orientation, this makes it hard to discern between a true turn and a turn of the gyroscope without strapping the gyroscope to the user with a fixed orientation.

A way to get around some parts of the last problem is to use the accelerometer to approximately determine how the device is oriented in reference to gravity. The gyroscope may then be used to compute only the approximate angle with which the user has turned around the direction of gravity. This is beneficial simply because all ”normal” turns performed by a human are around the direction of gravity. This makes the orientation of the device in reference to the user somewhat obsolete, but false turns may still be induced by turning the device around the direction of gravity.

Sensor Fused Heading Estimation

There are different approaches for combining magnetometer, gyroscope and possibly other sensors to produce a sensor fused estimation of the heading. Most Android phones implement such an algorithm, using gyroscope, magnetometer and accelerometer to produce an estimate of how the phone is oriented in the world coordinate system. The algorithm uses gyroscope and accelerometer to determine when the magnetometer output is corrupted, and uses their measurements to estimate the heading until the magnetometer can be trusted.

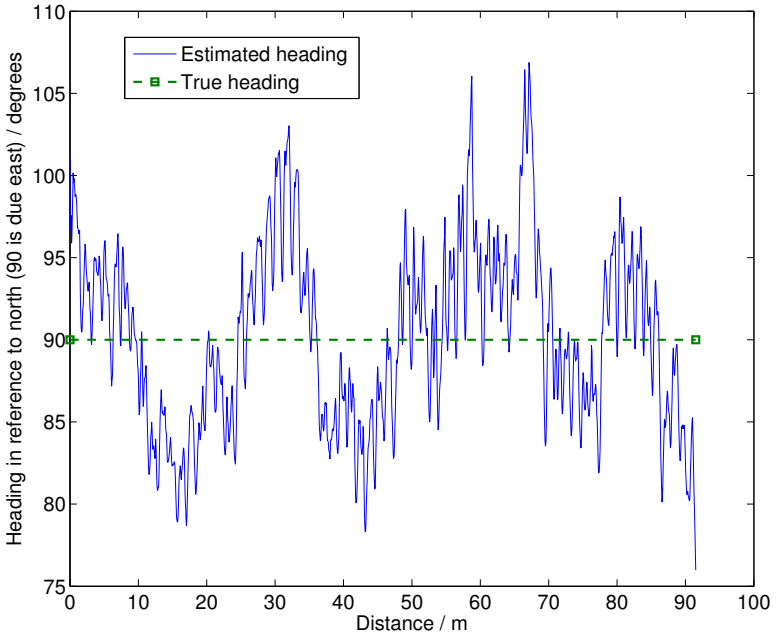


Figure 8.8 True heading versus heading output from Android rotation vector sensor. The true heading was due east, marked by the green dotted line. The mean of the estimated heading is 89.7°, which is satisfactory. However, it deviates from this mean as much as 15° and is not accurate enough for our purpose.

In Figure 8.8 the estimated heading for such an algorithm is shown along with the true heading. The measurement was taken while walking due east for 91.5 meters in a corridor indoors. The difference between the mean estimated heading, 89.7°, and the true heading, 90°, is minuscule, but the estimates vary around $\pm 15°$ from the true heading. This method of estimating the heading does not possess longterm drifts, but, as the estimation relies on the magnetometer for longterm calibration, it may be corrupted by different environmental factors, see Section 2.3.

Heading as a Particle Filter State

Another method for estimating the direction of movement is to extend the PF with a state representing the heading of each particle. A kinematics model could be

$$\begin{aligned} x_{k+1} &= x_k + r \cdot \cos\theta_k \\ y_{k+1} &= y_k + r \cdot \sin\theta_k \\ \theta_{k+1} &= \theta_k + v_{k+1} \end{aligned} \tag{8.19}$$

where x and y represent the particle's two dimensional position, r the approximate distance traversed between time step k and $k+1$, θ the particle heading in the range of $[0, 2\pi]$ radians and v_{k+1} the process noise at time step $k+1$. During the state update particles will be spread according to their heading and the distance traveled since last update. Particles with correct heading and previous position will reach the correct position and be assigned large weights. A problem arises when after a long stretch of walking straight, the user makes a turn, as most particles at that point have headings corresponding to the user before the turn was made. Thus, to be able accommodate for turns, the process noise, v_k, must have a sufficiently large standard deviation.

Gyroscope measurements may be added to the kinematics model by adding them to the θ state update rule as

$$\theta_{k+1} = \theta_k + \Delta\theta_k + v_{k+1}, \tag{8.20}$$

where

$$\Delta\theta_k = \sum_{n=1}^{N} \frac{\vec{\omega_{t_n}} + \vec{\omega_{t_{n+1}}}}{2} \cdot (t_{n+1} - t_n) \tag{8.21}$$

with N as the number of gyroscope measurements between t_k and t_{k+1}, $\vec{\omega}$ as the sensor measurements and t_n as the time of the n:th measurement. $\Delta\theta$ is a measurement of the change in direction between k and $k+1$, allowing the kinematics model to more accurately capture turns. Used in this sense the gyroscope is only used to compute the angular change between two time steps. Thus, if the time between state updates is short enough, biases and drifts of the gyroscope may be neglected. Further, this model allows for a process noise, v_k, with a much smaller standard deviation as most of the dynamics of a turn is captured by the gyroscope.

The problem of "false" turns, which arises when the device turns when the user does not, is still present. However, the effect can be reduced quite substantially when the gyroscope is used collaboratively with the PF. Each particle is an independent entity and if a turn is detected, a portion, $p \in [0, 1]$, of the particles may be selected to ignore the turn. Later, when the particles are given their weights, these particles will, depending on whether an actual turn was made, either be given large or small weights. Thus, if a turn was made, the portion, p, of particles that did not turn will not be re-sampled and vice versa.

Adding the Step Counter The kinematics model formulated in (8.19) and (8.20) may be integrated with a step counter, like the one described previously. The revised kinematics model then becomes,

$$\begin{aligned} x_{k+1} &= x_k + N \cdot (d + r_{k+1}) \cdot \cos\theta_k \\ y_{k+1} &= y_k + N \cdot (d + r_{k+1}) \cdot \sin\theta_k \\ \theta_{k+1} &= \theta_k + \Delta\theta_k + v_{k+1} \end{aligned} \tag{8.22}$$

where N is the number of steps taken between time step k and $k+1$, d is a predetermined step length and r_k is a small correction factor to account for fluctuations in step length. Further, r_k and v_k are Guassian random variables with zero mean and σ_r and σ_v as standard deviations.

Given a well determined step length of the user and an accurate gyroscope, the standard deviations of both r_k and v_k can be kept small. The resulting kinematics model will give a particle spread more related to the user's actual movement. However, inferior behavior is to be expected if the user's step length is far from the one used by the model. A solution to this is, as stated in Section 8.3, to include step length as an additional particle state. The extended model becomes

$$\begin{aligned} x_{k+1} &= x_k + N \cdot d_k \cdot \cos\theta_k \\ y_{k+1} &= y_k + N \cdot d_k \cdot \sin\theta_k \\ \theta_{k+1} &= \theta_k + \Delta\theta_k + v_{k+1} \\ d_{k+1} &= d_k + r_{k+1} \end{aligned} \tag{8.23}$$

with the same notation as previous with the exception of d_k being the particle's step length at time step k. Particles with appropriate step lengths will agree more with the measurements and tend to sustain and multiply. This has the advantage of being independent of the user's step length as the algorithm adapts for each user. On the other hand, we now have introduced two additional filter states compared to the simple case of only two states corresponding to x- and y-positions. Adding states forces an increase in the number of particles which in turn leads to a more computational and power demanding algorithm. Increasing the number of states by a factor two requires the number of particles to increase to the power of two to keep the same density of particles in the state space. This would result in the need to use between 10^5 and 10^6 particles, which would be too computationally demanding for today's smartphones, at least without extensive optimization. But, adding the additional states increases the knowledge of how the user has moved between measurements, giving a much more narrow area where a true position is probable. Further, the possible variance, especially of the step length state, is fairly small. The step length for most persons is in the region of 0.5 to 1 meter, and when an initial estimate has been determined, there are only a few cases where this length changes by more than a few centimeters. Concerning the heading, particles would

initially need to be spread in all directions. However, when an approximate heading is obtained, the spread only needs to be enough to correct for biases and drifts of the gyro and small heading changes caused by the user inadvertently turning the device a small amount. Larger ”false” changes in heading will be caught by the partial spread discussed earlier. In conjunction, this decreases the number of particles needed from almost a million to between 3 000 and 10 000, depending on the time between the RSSI measurements.

9

Indoor Positioning - Sensor Fused Approach

In this chapter, a method of how to include the information from the sensors in the positioning algorithm will be presented. The dead reckoning presented in the previous chapter works well for short distances (see e.g. Figure 8.5), whereas information from WiFi signals do not have the disadvantage of drifts. The main strategy is now to combine these, in order to get the best properties from both parts.

9.1 Extension of Particle State Space

In the previous chapters, the particles consisted of two states; the coordinates of the position. Now, we introduce the direction of movement, and the step length of the user, as two new states.

In Section 8.6 two extensions to the state space, (8.22) and (8.23), to accommodate for heading and for heading plus step length in congregation with a step counter are presented. Both of these models will be used to estimate positions in later parts of this chapter.

Two sensors will be used to aid in the positioning, an accelerometer and a gyroscope. The accelerometer will have two different uses. It will serve as the basis for the step counting algorithm and to estimate the orientation of the phone in relation to the gravity. Using this orientation, the gyroscope will be used to measure the changes in heading.

Initially, the position of each particle is uniformly random distributed in $[0, 2\pi]$, and the step length is normally random distributed around 0.75 m. When the user takes a step, each particle moves its own step length in its own direction. Then, particles with good estimations of the step length and direction will reach accurate estimations of the position, and be more likely to multiply in the next step.

Parameter	Magnitude
C for 2.4 GHz	-38 dB
C for 5 GHz	-36 dB
α for 2.4 GHz	0.4 dB / m
α for 5 GHz	0.5 dB / m
Number of particles N	10^4

Table 9.1 Parameter magnitudes used in the algorithm.

Parameter	Distribution
Initial direction θ	$U(0, 2\pi)$
Process noise v^{θ}	$N(0, 10^{-4})$ (m)
Initial step length r	$N(0.75, 4 \cdot 10^{-2})$ (m)
Process noise v^{r}	$N(0, 10^{-4})$ (m)
Measurement noise e^{RSSI}	$N(0, 25)$ (dB)

Table 9.2 Probability distributions used in the algorithm.

The new states offer two advantages. First, this kinematics model is more accurate than a random walk, which means that the particles are moved to more probable positions at each step. Secondly, the average direction and step length can be used for PDR between the RSSI-measurements, providing a more frequent position estimation output.

In order to keep the particle density in the state space high enough, the number of particles used has to be larger for this algorithm. In order to model the noise, the directions and step lengths are spread randomly with normal distributions around their current values, during each state update (see Table 9.2). Standard deviations of 0.01 meters for the step length, and 0.01 radians for the direction, seemed to be reasonable for the models. The states are updated each time a step is taken by the user.

9.2 Test Setup

In order to test this algorithm, one of the authors walked one round in *Environment three*, at a normal walking pace, holding the smartphone with constant orientation in relation to the body. The test path is shown on a floor plan in Figure A.1 in Appendix A. The initial position and direction were treated as unknown. The parameter values used in this test are listed in Table 9.1, and the probability distributions of the models are listed in Table 9.2.

The test was then repeated, but this time, the initial average step length was set to 0.4 m. The purpose of this was to investigate how the states adapt to the actual step length, which was 0.83 m.

The positioning estimation error is computed by dividing the path into equidistant points according to the number of steps taken during the walk. The position error is then computed as the absolute distance between the estimated position and the corresponding point on the path.

9.3 Results

This section presents the results obtained when fusing sensor data together with WiFi measurements. First, the results obtained while a predetermined step length was used are presented. Afterwards, the position estimates when the step length is treated as a part of the state space is presented along with positioning from an LS algorithm, a PF using only WiFi measurements and a PDR algorithm. This section is followed by a presentation of findings concerning the step length estimation when the algorithm is tested with an erroneous initial value. The last section deals with detection and compensation of ”false” user induced turns.

Fixed Step Length

In Figure 9.1, the estimated path of a walk in *Environment three* is presented. The positioning errors are shown in Figure 9.2 and have a mean of 1.3 meters and a maximum of 6 meters. The time until an approximate initial heading is found is clearly visible in the plot of the error, corresponding to the peak at 7 samples. A known initial heading would of course improve on this error.

Estimated Step Length

Figure 9.3 shows the estimated path of four different positioning algorithms, LS, PDR and PF with and without sensor data. The positioning error is, for each algorithm, displayed in Figure 9.4 and to high-light some of the features of the error, the mean, max and median values for each are available in Table 9.3. In this test, WiFi measurements combined with information from the sensors performed best. Furthermore, Figure 9.4 shows that the error from the PDR seems to grow with time, which is a typical property of this estimation method. The error of the PF with sensor data is larger when the step length is also estimated. This behavior is expected since the estimation relies on relatively noisy RSSI measurements to estimate the step length. However, this walk was conducted at an approximatively constant speed by a single user, and a different user may have a quite different stride length. In such a case, the positioning will improve if the step length is estimated. Additionally, a user may change stride length over the course of a walk, which will be caught by the estimation.

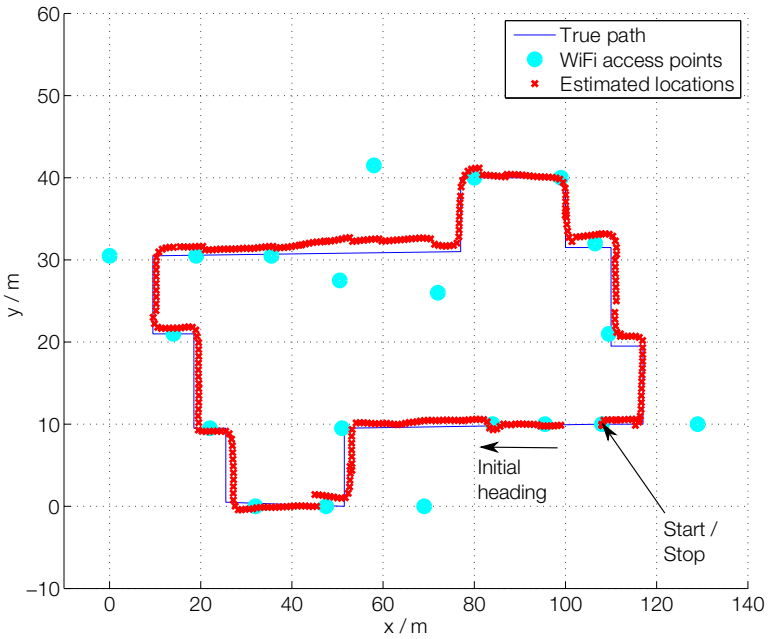


Figure 9.1 Position estimates using both WiFi and pedestrian dead reckoning, along a reference path of 273 meters. This method improves the positioning significantly. The mean error is 1.3 meters (see also Figure 9.2). The start and stop point is (109.5;10), and the initial heading is in the negative x-direction. Here, the step length is considered as known and set to 0.83 m. See Table 9.3 and Figure 9.3 for a comparison with other methods.

Positioning algorithm	Maximum error / m	Average error / m	Median error / m
LS - WiFi Only	11.2	4.8	4.1
PF - WiFi Only	11.6	3.5	3.1
PDR	7.7	4.2	4.8
PF - WiFi & Sensors	5.8	1.8	1.8

Table 9.3 Position error using four different algorithms utilizing sensors and/or WiFi along the true path in Figure 9.3. Combining WiFi and PDR gave the lowest error. Plots of the error are shown in Figure 9.4.

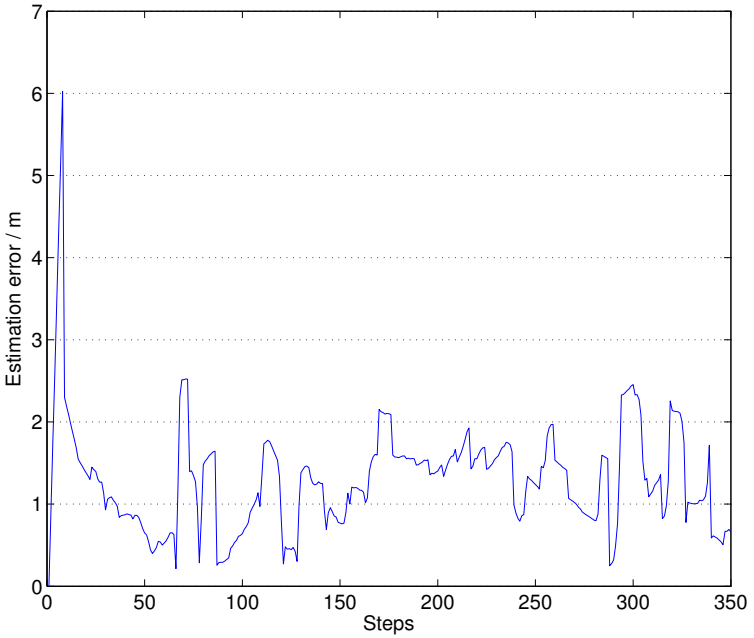


Figure 9.2 Error of the position estimates using both WiFi and PDR. The step length is set to 0.83 m. The error grows towards its maximum value during the first few steps, since the heading is not well estimated in the beginning. The average error is 1.3 meters. See also Figure 9.1.

Evaluation of the Step Length Estimation

Figure 9.5 shows how the average step length of the particles changes with time. The initial estimation was intentionally set too low, in order to investigate how the algorithm would perform. During the very first steps, the estimated value is smaller than the true value. However, after six steps the value has passed 0.75 m, and from this point, the situation resembles one with an initial estimation of 0.75 m.

9.4 Limit the Impact of ”False” Turns

If the orientation of the device changes severely in relation to the user, this will be interpreted as a turn by the user in the algorithm. In worst case, this might result in erroneous position estimates that do not recover. A way to correct this, is to detect when the position estimate is unreasonable, and then change it to a rough estimation.

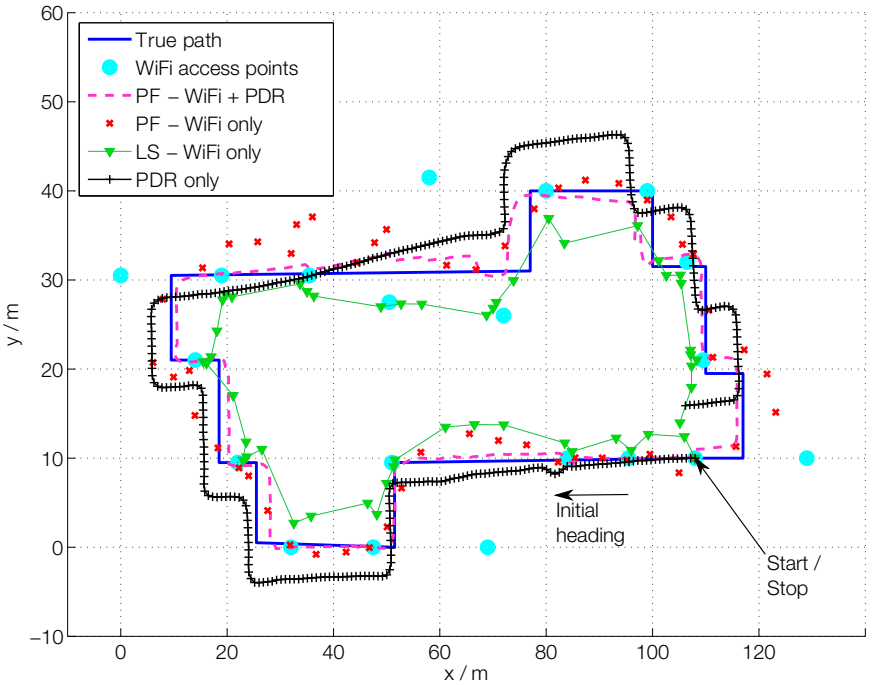


Figure 9.3 Position estimates using four different algorithms utilizing pedestrian dead reckoning (PDR) and/or WiFi, together with a reference path of 273 meters. It is clearly visible that the combination of WiFi signals and PDR gives estimates that are closest to the true path. It also gives more frequent estimates than the methods that only use WiFi. The errors are presented in Table 9.3 and Figure 9.4.

If the RSSI from a certain AP is high (> -42 dB), whereas the distance between the estimate and the AP is more than 7 m, this is used as an indicator of an unacceptably inaccurate estimate. Then, the algorithm is reinitialized around the current AP.

Moreover, when a turn is detected, 15 % of the particles keep their previous directions, in order to include the possibility that the phone merely changed orientation in relation to the user.

This was tested in *Environment three*, where the phone was rotated $\frac{\pi}{2}$ in relation to the user at three points along the path. The rotations were made in the same direction, adding up to a total rotation of $\frac{3\pi}{2}$. This resulted in an average position error of 3.4 m, and a maximum of 24.3 m. The maximum value occurred after the third rotation, and was followed by a reinitialization. The first two rotations resulted in much lower error peaks (6.6 m and 6.8 m).

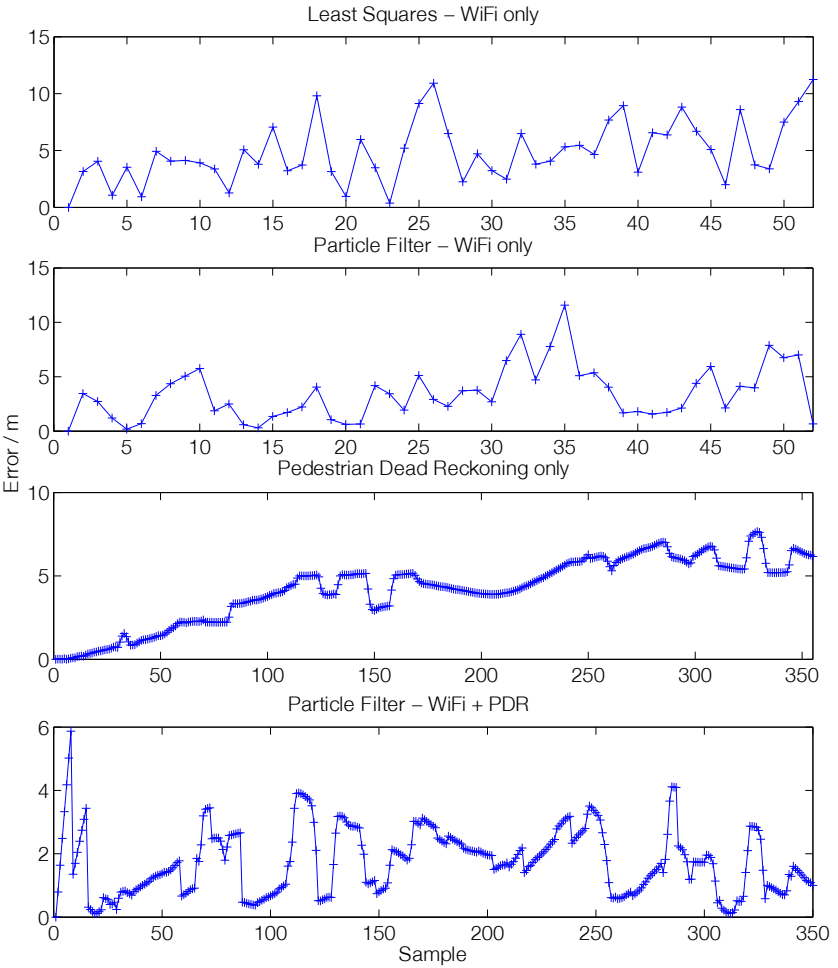


Figure 9.4 Position error of different algorithms. Note that the scales on both the axis are different. Using PDR only gives an error that grows with time. Combining WiFi and PDR gives the lowest error. See also Figure 9.3 and Table 9.3.

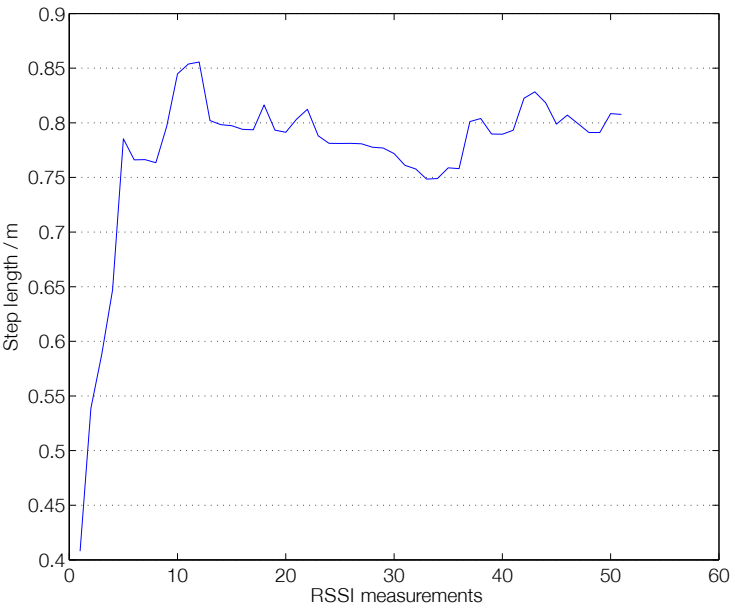


Figure 9.5 Each particle was assigned with a step length as a fourth state. In order to investigate if this could be advantageous when the step length of the user is unknown, a test user carried the device along a path of 273 m. This plot shows the average step length of the particles, with initial value 0.4 m, intentionally set too low. It gets considerably closer to the true step length, which is 0.83 m in this test.

9.5 Concluding Remarks

Using information from the accelerometer and the gyroscope improved the positioning significantly. At the same time, this requires that the orientation of the phone in relation to the user does not change too frequently. While these events may be detected and taken into account, there are situations where it takes some time for the algorithm to exclude a "false" turn, and if these occur too often, the position estimation would always be inaccurate. Even though the sensors contribute with important information about the movement of the user, it would not be an option to use these for PDR only, i.e. without combining them with the WiFi measurements. The main reasons are that the initial states would have to be known, that the expected error grows with time, and that the orientation of the phone would always have to be fix in relation to the user. The conclusion is that WiFi measurements could be used without PDR, but not the other way around.

10

Algorithm Implementation in Android

In the previous chapters, offline computations are used to obtain the position estimate. Here, a real time Java implementation of the developed algorithms will be presented briefly. The PF is quite computationally demanding, and therefore it is important to discern if handheld devices are capable of performing these computations sufficiently fast.

The implementation will be done in a positioning application developed by Sony Mobile, called SonyMap. This application is intended for Android smartphones and used to position the smartphone in Sony Mobile's buildings around the globe.

10.1 Java Implementation

Going from offline simulations to a real time implementation naturally requires certain considerations to achieve desired performance. Further, to scale a positioning algorithm from use in a single building and a single floor to a system that is used in multiple sites around the globe provides additional challenges. In this section, some of the considerations made when the algorithm was implemented are presented.

The focus of the implemented algorithm was usability and accuracy, while keeping the strain on hardware and battery at a minimum. Because of this a few of the techniques described in this thesis were kept outside the implemented algorithm.

Adaptation of Model Parameters
To adapt the model parameters is a good approach either if the environment is unknown or if it is changing within the specific building. However, as stated in Chapter 7, the parameters obtained through the proposed adaptation algorithm are accurate, but not as accurate as the ones obtained by multiple measurements at several known locations. To obtain the best accuracy the choice was made to, instead of adapting the parameters, allow for individual values of C and α to be set for both 2.4 and 5 GHz for each AP.

Compensation for User Induced Signal Attenuation
In Chapter 7 it was shown that most normal use cases (carrying phone in hand, pocket or bag) did not produce sufficient signal attenuation to noticeably impact the positioning. Enough signal attenuation for this was only produced when the user deliberately tried to cover the antenna. Further, the algorithm to detect such an attenuation required a fairly large number of computations and was thus decided to be kept out of the implementation.

Step Length as a Particle Filter State
Estimating the step length proved to be a good addition to the state space, especially when the user's step length was different from the initial filter state or when the step length changed during the positioning. However, to include step length as a filter state required an increase in the number of particles from the current 2 500 to around 10 000, making it cumbersome to run. Further, to accommodate for the possibility of the user changing orientation of the phone, the spread of the particles was increased, removing some of the benefits of having step length as a filter state.

Finding Correct Building & Floor

One of the first obstacles encountered when trying to implement the algorithm was, on a site with several multistory buildings, to find the correct building and floor.

To find the correct building a couple of different techniques could be used. The most simple of these is to use the GPS output when the user enters the building. This, however, is not a very robust way. Either the user may want to start the navigation when inside the building or the GPS output might be inaccurate. A better estimate can be obtained by evaluating the RSSI values of the available known WiFi APs, and position the user in the building from where the strongest RSSI value is obtained. If needed, this could be refined further by taking more than the AP with largest RSSI value into account. However, as the buildings at Sony are separated by fairly large distances, regarding only one AP is sufficient.

A more challenging task is to determine the correct floor within the correct building. The signals are expected to get significantly attenuated when propagating between floors [Rappaport, 1996]. Thus, the strengths of the signals from each floor are used to determine the current one. In this case, it is insufficient to regard only the known AP with largest RSSI value. In areas with openings between different floors, like stairwells, in close proximity to windows or when the distances to APs on other floors are short, it is not unlikely that the largest RSSI value comes from a floor different from the one the user is currently on. To obtain an accurate estimate, information from several APs needs to be used. This can be done in several different ways and during the application implementation two of these were tested.

Below, a floor detection scheme based on the three APs with largest RSSI values are presented [Marques et al., 2012]. The floor the user is currently on is determined by

- If the APs with the second and third largest RSSI values are located on the same floor, that floor is chosen.
- If not, the floor of the AP with largest RSSI value is chosen.

This scheme mitigates the problem of having one strongly transmitting AP on the "wrong" floor. However, if there are at least two APs from the "wrong" floor among the three, this floor will always be chosen. Therefore an alternative method has been suggested by the authors.

Simply determining the mean received power from each floor suffers from the same problems as earlier discussed. If, on the other hand, the aggregate received power from each floor is determined, a more robust method is obtained. As RSSI values are reported from the driver in the range of 0 to -100 dB, where 0 indicates the highest received power, a simple addition will not suffice. Instead, the smallest received RSSI, R_{min}, is used to scale all RSSI values before adding them together. This is done by determining how much to add to the smallest RSSI value to obtain 0 i.e. $-R_{\text{min}}$, and add this to all RSSI values. The set of accumulated power from each floor can then be used to determine a probable floor according to the rules defined in the flowchart depicted in Figure B.1 in Appendix B.

This algorithm uses some heuristic assumptions concerning the power from different floors when estimating the current one. The assumptions are based on consecutive floors being relatively similar, both in layout and AP distribution. Further, it is expected that the antennas of the APs are designed to direct the signal downwards. One of the assumptions is that if the signals from three consecutive floors are comparable, the most probable location is on the middle floor. In the implementation two floors have been comparable if their total powers are within a factor 1.5 from each other.

To test the performance of the algorithms, each was implemented in a version of SonyMap whereafter the applications were installed on identical phones. A $\sim$30 minute walk was then taken throughout the Sony building and the number of faulty floor estimates was noted. The scheme proposed in [Marques et al., 2012] reported the wrong floor 10 times during the walk while the one proposed by the authors was wrong 3 times.

If a consistent floor estimation is of more importance than a speedy change when the user changes floor, both algorithms could be further refined. A check on the number of times a different floor has been reported could be introduced and the floor estimate would not be changed until the algorithm had reported the same floor a specified number of consecutive times.

Coordinate Conversions

In Chapter 2 the longitude/latitude coordinate system is presented. This system is used in most large scale positioning applications as it gives each point on Earth a unique coordinate representation. However, it is impractical to use the longitude/latitude system for positioning on a local scale. This is due tothe same difference in longitude/latitude will equal a different distance in meters on different locations, because of the Earth's spherical surface. As all calculations, in this thesis and in many other applications, are done in meters a conversion between the two is needed.

In this thesis an open source solution called *Jcoord* is used [Stott, 2013], which is published under the *GNU General Public License (GPL)*[1]. This solution splits the Earth into regions within which a distance in longitude/latitude is approximately constant converted into meters. Further, some extrapolation is done within each region to make the conversion even more exact. To convert in the other direction, the region needs to be known, but otherwise the same general computations are made.

Pose Estimation

When implementing the PF, it is of interest to discern between different states or poses that the user is in, like walking or being still. One of the reasons for this is that the state update of the PF can be changed according to the pose. When still, no benefit is drawn from updating the heading, step length or movement and a random walk with a small variance is a good enough kinematics model. Another reason is that if the user is still, there is no need to sample the sensors at a high rate, conserving both computational power and battery. Further, as fewer filter states are required at stationarity, the number of particles can be decreased, which in turn decreases the needed computational power further.

In the implementation, a user has been deemed as still if no steps have been detected over the course of three seconds, and deemed moving if not still. This is a fairly crude method of distinguishing between poses, but in the current use case, the performance is sufficient.

The Particle Filter

In Chapter 4, the PF process is described in three steps, weighting, re-sampling and state update. Performing the operations in this order is not computationally efficient, both concerning memory and computational power. A better way is to combine the state update and weighting, thus when a new set of RSSI values is available for each particle, the state is updated and then the weight computed. During this step a sorted cumulative sum of the particle weights is also computed. Then the re-sampling is performed along with the computation of the estimated position.

[1] GNU General Public License (GPL): `www.gnu.org/copyleft/gpl.html`

If a step counter is added to aid the state update, the states can be updated at each detection of a step, as the time between consecutive RSSI measurements is no longer needed to perform an accurate state update. This further decreases the amount of computing time spent between receiving new RSSI measurements and a position estimate is obtained. In addition, along with every registered step, a new estimation of the position could be computed, allowing for faster updates than the RSSI measurements alone.

As each particle is an independent entity during most of the computations, the process could be greatly improved by parallel computations, either in separate threads on a single core, distributed over several CPU cores or even over GPU cores. In principle the computations for each particle could be run simultaneously utilizing the same number of threads as there are particles. There are, however, one bottle neck in the computations. The state update and weighting must, for each particle, be done before the re-sampling can be initialized. The explanation is simply that the re-sampling is dependent on the weights of all particles. However, the steps of state update and weighting, and afterwards the steps of the re-sampling, can be parallel.

In the authors' implementations a time cap of five seconds was placed on the algorithm, as the RSSI measurements were received with that time interval. Each particle filter process (*State update*, *Weighting* and *Re-sampling*) was parallelized using a number of threads corresponding to the available processor cores. On a device with four cores, the *Xperia Z Ultra*, the algorithm was run with a time consumption of between 200 and 400 milliseconds. This is significantly less than the time between RSSI measurements, allowing other tasks to be run alongside the particle filter algorithm. If either RSSI measurements are received at a faster pace, or more particles are needed, further optimization might be needed.

Performance of the Implemented Algorithm

Allowing for the user to stray from the ideal use case used when the algorithm was developed, i.e. holding the phone with a constant orientation in reference to herself, of course impacts the performance. To test how the final algorithm performed, an application was developed where the user was asked to walk a predefined path, trying to be at the same place along the path at all times as a certain reference position. The difference between this reference position and the estimated one was recorded and the result is displayed in Table 10.1, along with the performance of a least squares algorithm using only 2.4 GHz, and one using both 2.4 and 5 GHz Wifi. The particle filter with sensors shows the best performance, with 37 % and 54 % smaller mean error to each of the least squares methods. The maximum error also decreased a fair amount. Further, it is interesting to note that adding 5 GHz WiFi measurements to the least squares estimation, without giving any special attention to modeling, improved the mean error by 27 %, showing that using dual band WiFi measurements actually improves the performance of indoor positioning algorithms.

	Mean error / m	Max error / m
Least Squares - 2.4 GHz only	7.0	18.2
Least Squares - 2.4 and 5 GHz	5.1	11.3
Partilce Filter plus PDR	3.2	8.0

Table 10.1 Mean and maximum error of the implemented particle filter algorithm along with the same values for a least squares algorithm using 2.4 GHz, and both 2.4 and 5 GHz WiFi. The particle filter shows the best results, with 1.9 and 2.8 meters smaller mean error, compared to the two least squares estimations. Further, it is noteworthy that by adding 5 GHz measurements to the least squares estimator, without no further modeling effort, the mean error decreased by 27 %.

11

Conclusion

In this chapter the contributions made throughout this thesis work are presented. Further, possible extensions and improvements are described in the future work section.

11.1 Contributions

Positioning Using Both 2.4 and 5 GHz WiFi Signals

A PF that uses both 2.4 and 5 GHz WiFi signals for positioning was developed. It was concluded that more information was obtained by using both frequencies rather than one of them, which improved the positioning.

The path-loss models can be improved by adapting the model parameters during the positioning. This is done by using measurements from positions where the position error is expected to be low, and then use this to estimate the model parameters. Furthermore, the algorithm can compensate for signal attenuation induced by the hand of the user.

Sensor Fused Positioning

In addition to the WiFi signals, information from the sensors was used to model the motion of the user carrying the device. From this, an accurate positioning algorithm was implemented, with the constraint that the device must be reasonably fix in relation to the user. This algorithm performed considerably better than more conventional techniques, and gave a mean error of 1.3 meters without the need to know the initial position and heading.

For the case where the phone might be turned in relation to the user, a less accurate but more robust algorithm was developed. This algorithm still produces position estimates with an accuracy of 2-3 meters.

Both algorithms were developed and tested for commonly used smartphones, and hence the sensors in these are accurate enough for the algorithms to work.

Step Counter
A step counter using the accelerometer was developed, as an important part of the algorithm. The magnitude of the acceleration was differentiated and filtered, and when it passed a certain threshold level, this was used to indicate a step. This step counter works regardless of whether the phone is carried in the hand, in a pocket or in a bag.

Heading
The accelerometer was also used to give the direction of gravity as a reference of the rotation, which was measured by the gyroscope, in order to get information about the heading of the user. To handle different hypothesis of the direction, each particle was given a direction as a state in the PF.

Implementation in Android

The robust version of the sensor fused positioning algorithm was implemented as an Android API for use in SonyMap, a real time positioning application developed by Sony Mobile.

Determine Floor
In the online implementation, a new algorithm to determine the current floor by using WiFi signals was included. This proved to be a significant improvement compared to the previous algorithm used in SonyMap.

11.2 Future Work

During the work with this thesis several possible enhancements or extensions to the work have arisen, which have been deemed outside of the thesis scope. Some of these points may, however, be of interest for future work on the subject and are presented with a brief description below.

3D-Positioning

In this thesis only a two dimensional model of the space in which to navigate has been used. Motion along the $z-$axis of the local frame, i.e. changing floor, has been determined using methods other than the PF. Further, possibly beneficial signals originating from other floors or buildings than the one the user is currently estimated to be in are disregarded.

A more general approach is to consider a full three dimensional model of space. For this to be possible some extensions to the work done in this thesis must be done. A start could be to introduce the current floors as a discrete particle state, removing the need to have a separate algorithm to determine it. To make this possible a multi-floor (and multi-building) model of the RSSI values from each AP, taking into account the attenuation from floors and ceilings, needs to be developed. The 3D model could also contain a continuous state corresponding to the altitude, making it possible to determine both x- and y- position along with the height of the phone.

Map Data

Using a Map

The work in this thesis assumes no knowledge, apart from the locations of the APs, concerning the layout of the building. This makes the developed system applicable in many different environments with relative ease. However, if map data is available this could be used in several ways to improve the positioning. One of these is to put constraints on the path of each particle i.e. forcing the particles to enter rooms only through doors and not to travel through furniture and walls. Another is that a user only is able to travel between floors when in proximity of elevators or stairs.

Creating a Map

Another possibility is to use the estimated paths of a large number of users to estimate where walls and furniture are located. This could then be combined to successively build a map of the building that could be used to aid future positioning.

Using More Signals and/or Sensors

To estimate a position, only RSSI measurements from WiFi APs and readings from an accelerometer and a gyroscope were used in this thesis. However, smartphones today contain sensors to measure a much wider array of signals and environmental properties. Some of these have a great potential to aid in the position estimation.

A barometer is a sensor measuring the atmospheric pressure and this varies with the height above ground. Thus, a barometer could aid in the estimation of floor, either to be used to weight particles in a 3D model or to aid the heuristic floor estimation algorithm.

Investigation of the magnetometer values have shown that these, in most cases, vary too much in indoor environments to be useful for heading estimation. If, on the other hand, the occasions when the magnetometer readings are correct could be identified, this could be used to correct the heading.

In addition to WiFi, a smartphone has a large number of other wireless capabilities, like LTE, 3G, GSM, Bluetooth, NFC and RF-ID. These signals could all be used to improve the positioning, either by having a model of signal power as a function of distance, or, in the case of NFC and RF-ID, by sheer proximity.

Measuring the Time Difference

Instead of measuring signal strength from a transmitter an enticing possibility is to measure the difference in time between a sent and received signal. A few different methods of using this are given a brief explanation in Section 5.2. If the measurements of time difference are made at both 2.4 and 5 GHz, as the speed of light puts a limit on the time of flight, the frequency yielding the shortest distance of flight is also the most accurate measure of the distance. This is due to, given perfect time synchronization, the signal taking the shortest path between transmitter and receiver, also yields the shortest time, and the signal can not take a shorter path than the actual distance.

Improved Pose Estimation

When the developed algorithm was implemented only two poses were considered, being still and walking. Furthermore, the distinction between the two was rather crude, a user was deemed as still if no steps had been taken for a certain amount of time, and deemed moving if not still. This is, however, a fairly good model of how people moving on a single floor in an indoor environment behave.

There is, of course, a multitude of poses that would be of interest to discern between, like walking, running, sitting, standing, riding an elevator or escalator etc. These poses could aid the positioning in several ways. Discerning between sitting, walking and running would give a more accurate particle spread and if the user is estimated to travel up or down the floor estimation could be improved.

Improved Usability and Crowdsourcing

A positioning system for a single or a small set of buildings is in its way useful, at least for persons visiting them frequently. A system, with which it is possible to position in a large number of buildings, is more usable. Such a system could be achieved in different ways, one being that APs could along with their MAC-address also report their location and altitude. However, this would require industrial standardization and either new software for the APs or new APs altogether. Another way is to create a system in which users can add AP locations or estimate the AP locations by trilateration. If such a system also is integrated with GPS, a truly ubiquitous positioning system could be achieved.

Another advantage of having a large number of users is a large amount of data generated in the system. This data could be used, as in Chapter 7, to over time improve the propagation model for each AP or to find misplaced APs.

Smart WiFi Channel Choice

To obtain a total picture of the APs in the surroundings of the measuring device a scan of each possible WiFi channel must be performed. Such a scan claims a large amount of time (around 3.5 second). If, on the other hand, the channel/channels on which the APs in the vicinity are transmitting are known, only these channels should be scanned. This would drastically decrease the amount of time spent scanning, as a single channel takes just over 100 ms to scan. If the scans could be performed at a shorter time interval more information from the interesting APs could be obtained.

Bibliography

Adams, R. and C. Essex (2010). *Calculus - A Complete Course*. Pearson Canada Inc.

Beauregard, S. and H. Haas (2006). "Pedestrian dead reckoning: a basis for personal positioning". *Proceedings of the 3rd workshop on positioning, navigation and communication (WPNC '06)*.

Bentley, J. (1988). *Principles of Measurement Systems*. Pearson Education.

Bickel, P. and K. Doksum (2001). *Mathematical statistics - Basic Ideas and Selected Topics*. Prentice Hall, Inc.

Chon, H. D., S. Jun, H. Jung, and A. S. W. (2005). "Using rfid for accurate positioning". *Journal of Global Positioning Systems*.

Choo, J. H., S. N. Cheong, Y. L. Lee, and S. H. The (2012). "I^2Navi: An Indoor Interactive NFC Navigation System for Android Smartphones". *World Academy of Science, Engineering and Technology*.

Creative Commons (2013). *Creative commons attribution 2.5 generic*. URL: `www.creativecommons.org/licenses/by/2.5/`.

Faragher, R. and . Harle R (2013). *Getting Closer to Everywhere*.

Figueiras, J. and S. Frattasi (2010). *Mobile Positioning and Tracking - From Conventional to Cooperative Techniques*. John Wiley & Sons Ltd.

Fisher, R. (1915). "Frequency distribution of values of the correlation coefficient in samples of an indefinitely large population". *Biometrica (Biometrica Trust)* **10**, pp. 507–521.

— (1921). "On the "probable error" of a coefficient of correlation deduced from a small sample". *Metron* **1**, pp. 3–32.

Godha, S. and G. Lachapelle (2008). "Foot mounted inertial system for pedestrian navigation". *Measurement Science and Technology*.

Google (2013). *Android open source project*. URL: `code.google.com/intl/sv/policies.html`.

Grimes, J. G. (2008). *Global Positioning System Standard Positioning Service Performance Standard, 4th Edition*. Tech. rep. Department of Defense, United States of America.

Gustafsson, F. (2012). *Statistical Sensor Fusion*. Studentlitteratur.

Hansson, A. and L. Tufvesson (2011). *Using Sensor Equipped Smartphones to Localize WiFi Access Points*. Masters thesis. Department of Electrical and Information Technology, Faculty of Engineering, LTH, Lund university.

Hofmann-Wellenhof, B., K. Legat, and M. Wieser (2003). *Navigation*. Springer.

IDC (2013). *International data corporation (idc) worldwide mobile phone tracker, november 12, 2013*. URL: `www.idc.com/getdoc.jsp/?containerId=prUS24442013`.

IEEE Computer Society (2013). *IEEE 802.11™- 2012*. URL: `http://www.standards.ieee.org/getieee802/download/802.11-2012.pdf`.

Karlsson, P. (1992). *Measuring and Modelling of the Indoor Radio Channel at 1700 MHz*. Sigmatryck.

Kuipers, J. (1998). *Quaternions and Rotation Sequences: A Primer with Applications to Orbits Aerospace and Virtual Reality*. Princeton University Press.

Li, B., J. Salter, D. A. G., and C. Rizos (2007). *Indoor Positioning Techniques Based on Wireless LAN*. URL: `www.epress.lib.uts.edu.au/research/bitstream/handle/2100/170/113:Li.pdf?sequence=1`.

Mannesson, A. (2013). *Joint Pose and Radio Channel Estimation*.

Marques, N., F. Meneses, and A. Moreira (2012). "Combining similarity functions and majority rules for multi-building, multi-floor, wifi positioning". In: *2012 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. Ed. by C. Rizos, A. Dempster, B. Li, and T. Gallagher. IEEE, pp. 1–9.

Mautz, R. (2012). *Indoor Positioning Technologies*. Habilitation thesis. Institute of Geodesy, Photogrammetry, Department of Civil, Environmental, and Geomatic Engineering, ETH Zurich.

Morris, A. (1996). *The Essence of Measurement*. Prentice Hall, Inc.

Morrison, A., V. Renaudin, J. B. Bancroft, and G. Lachapelle (2012). *Design and Testing of a Multi-Sensor Pedestrian Location and Navigation Platform*.

Murphy, C. (2011). "Believable dead reckoning for networked games". In: *Game Engine Gems 2*. Ed. by E. Lengyel. Peters, A.K.

Parkinson, B. W. and J. J. Spilker (1996). *Global Positioning System: Theory and Applications, Volume 1*. American Institute of Aeronautics and Astronautics.

Proakis, J. and D. Manolakis (2007). *Digital Signal Srocessing. Principles, Algorithms and Applications, 4th international edition*. Prentice Hall.

Rappaport, T. (1996). *Wireless Communications - Principles and Practice*. IEEE Press.

Retscher, G. and A. Kealy (2005). *Ubiquitous Positioning Technologies for Inteligent Navigation Systems*.

Retscher, G. and M. Thienelt (2004). *NAVIO - A Navigation and Guidance Service for Pedestrians*.

Slabaugh, G. (1999). *Computing Euler Angles from A Rotation Matrix*. Tech. rep. Department of Computer Science, City University, London.

Stott, J. (2013). *Jcoord*. URL: `www.jstott.me.uk/jcoord/`.

Taheri, A., A. Singh, and A. Emmanuel (2004). "Location fingerprinting on infrastructure 802.11 wireless local area networks (wlans) using locus". In: *29th Annual IEEE International Conference onLocal Computer Networks, 2004*.

Walrand, J. (2010). *Communication Networks: A Concise Introduction*. Morgan & Claypool Publishers.

Ying, H., C. Silex, A. Schnitzer, S. Leonhardt, and M. Schiek (2007). In: *4th International Workshop on Wearable and Implantable Body Sensor Networks (BSN 2007)*. Ed. by S. Leonhardt, T. Falck, and P. Mähönen. Vol. 13. IFMB proceedings. Springer Berlin Heidelberg, pp. 80–85.

A

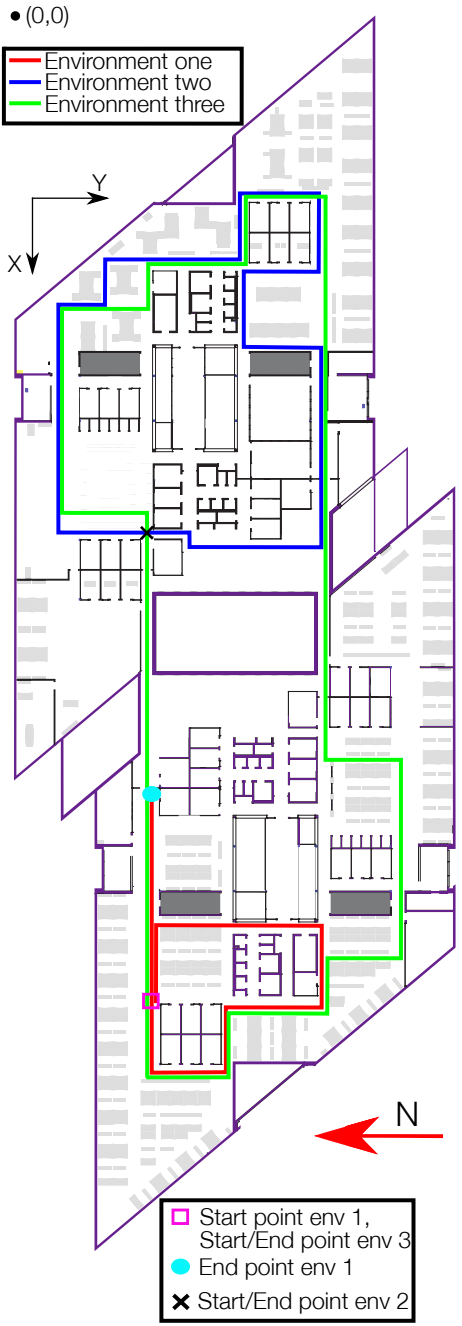


Figure A.1 Paths and environments used during the different tests in this thesis. The floor depicted is the top floor in the Glasgow building at Sony Mobile, Lund.

B

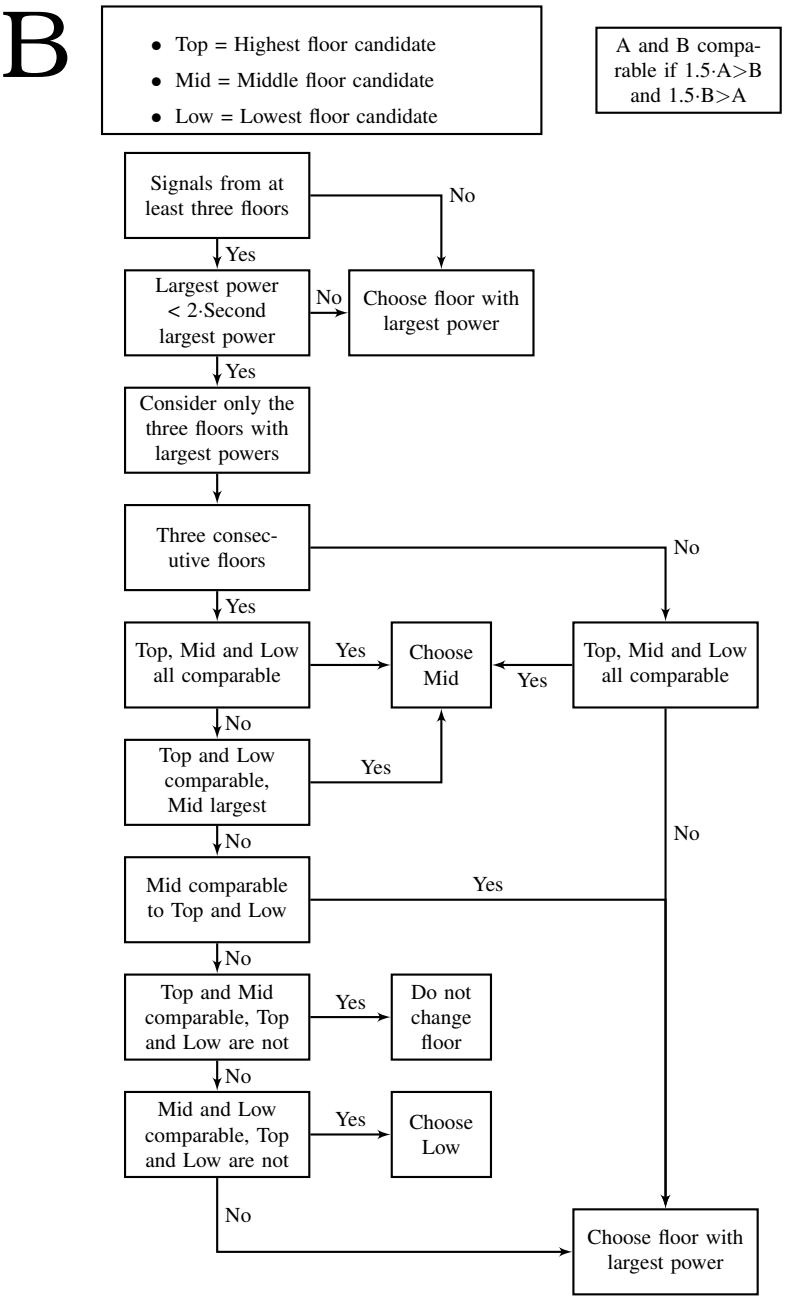


Figure B.1 Flowchart of floor estimation algorithm. It is based on the floors from which the largest total signal intensities are received. A large signal intensity from a certain floor is used to indicate that the user is on/close to that floor.



LUND
UNIVERSITY

http://www.eit.lth.se