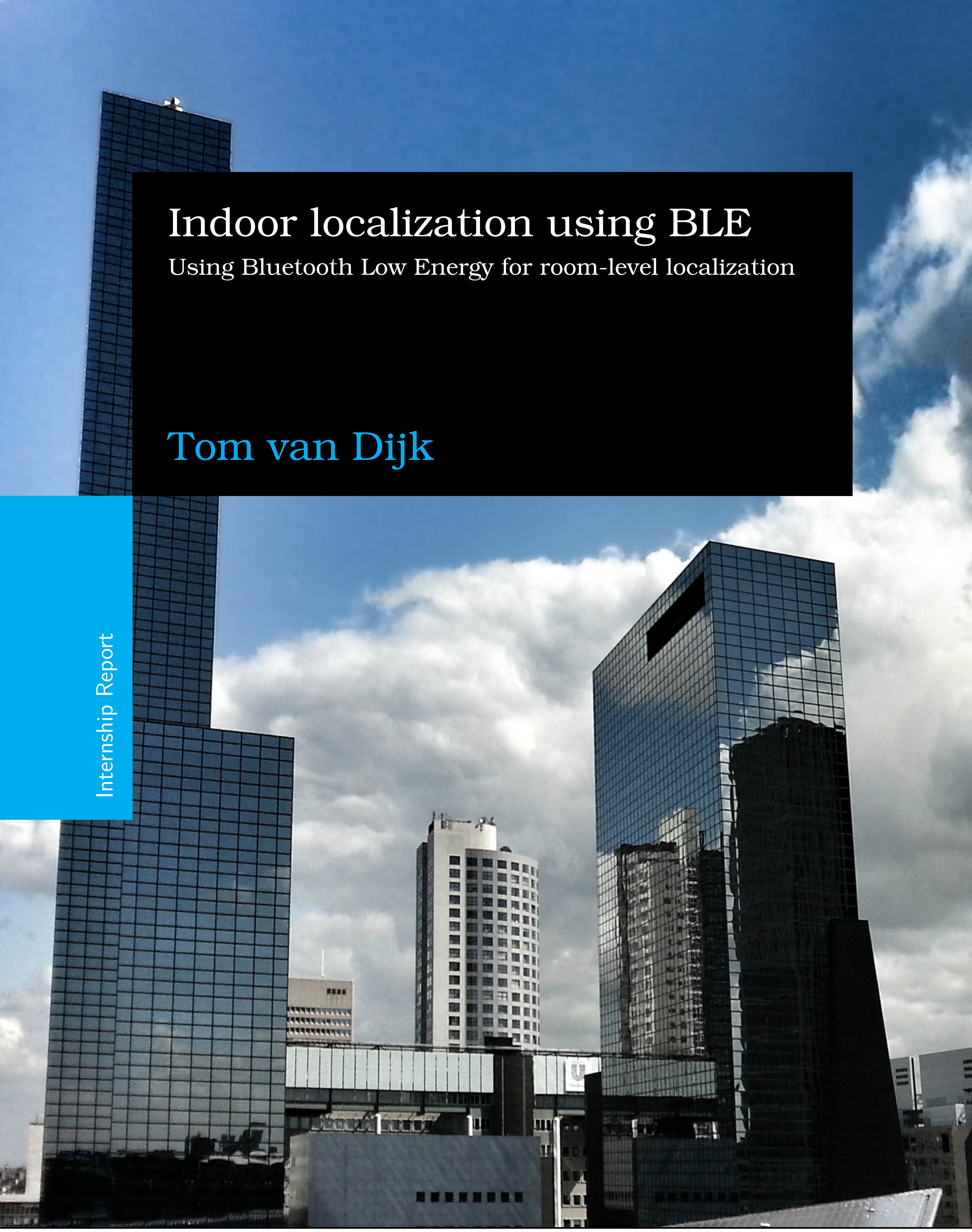


Indoor localization using BLE

Using Bluetooth Low Energy for room-level localization

Tom van Dijk

Internship Report

TU Delft Delft University of Technology

Indoor localization using BLE

Using Bluetooth Low Energy for room-level localization

Internship Report

Tom van Dijk

April 29, 2016

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

The work in this report was performed at DoBots (Distributed Organisms B.V.). Their cooperation is hereby gratefully acknowledged.

Abstract

One goal of home- and office automation is that devices will respond to your presence. For instance, lights can be turned on when you enter a room. To perform this function, it is necessary to know where people are inside the building.

The first part of this report focuses on Device-free Localization. Device-free Localization is a promising technique that can be used to locate people without requiring them to carry any mobile device. The good tracking results reported in literature were repeated in simulation. However, performance decreased when number of beacons was lowered to a realistic amount. The worse performance combined with unpredicted reflections from far-away objects meant that device-free localization did not produce usable results in practice.

In the second part of this report, fingerprint-based techniques are used to perform room-level localization. Fingerprints are a collection of observations that are unique for a specific location. Using supervised learning techniques, the room in which a fingerprint was observed could be determined with more than 99% accuracy.

Table of Contents

List of Figures

List of Tables

“In the future, airplanes will be flown by a dog and a pilot. And the dog’s job will be to make sure that if the pilot tries to touch any of the buttons, the dog bites him.”

— *Scott Adams*

Chapter 1

Introduction

1-1 Home automation

Over the last ten years, the number of intelligent devices at home has increased tremendously. The mobile phone evolved into today's smartphone, a multimedia device with many sensors and connectivity options. But not just mobile phones have increased in intelligence, the past years have also seen the introduction of, for instance, the smart thermostat. It seems that nowadays intelligence can be added to almost any object: smart watches, smart lightbulbs, even smart refrigerators and toasters.

The widespread intelligence in devices can be used for new applications, such as the field of home automation. Home automation entails the ability to control home appliances remotely or even automatically. For example, devices can be turned on or off based on your presence. Throughout this report, one example will be used: turning on the lights when someone enters a room.

1-2 Problem statement

Turning on the lights when the user enters the room may seem like a very simple task, but before this function can be performed, it is necessary to know where to user is. Specifically, the *room* in which the user is located needs to be known in order to turn on the correct lights. The goal of this research is to find a practical method to perform *room-level* localization.

There are many ways to estimate the position of the user (and the position of devices if necessary), but knowing the coordinates of the user does not directly answer the question of which devices should be turned on or off. The coordinates in itself do not provide any information about the room in which they are located.

To support room-based behavior, a method of transforming a real-world position or set of coordinates into a room number is required: a *floorplan*. While estimating a room number when the floorplan is given is trivial, in this case the floorplan is considered unknown (the user may not have this information readily available, or it may be too labor-intensive to enter this information). Therefore, the floorplan has to be estimated.

The localization and floorplan estimation will be used in an office or residential setting. This poses two important constraints on the possible solutions.

First of all, the available devices are limited. Only smartphones and DoBeacons can be used. This limits the types of sensors that are available. Section 1-4 gives an overview of the available sensors and how they might be used for localization and floorplan estimation.

The second constraint limits the amount of user interaction. The goal of home- or office-automation is to make everyday tasks easier, therefore the interaction with the user should be kept minimal. For instance, asking the user to take out their phone so the camera can be used for indoor localization and mapping is not an acceptable solution. In an ideal case, the system should not require any interaction, but a small calibration step is acceptable if it only has to be performed once and is not too time- or labor-intensive.

1-3 Overview

Initially, a more 'traditional' understanding of floorplan is used: a metric map that contains information about the location of walls, doors, rooms, users and devices in a 2- or 3-dimensional space. This map can then be used to determine whether a location is part of a room by checking if these coordinates fall inside the region that defines this room.

Given the limited types of sensors that are available (Section 1-4) and the wish to minimize user interaction, it is not feasible to find walls directly (Chapter 2). Instead, the floorplan can be estimated using the walking trajectories of people inside the building. To collect these trajectories, a localization system is required. Available options are discussed in Chapter 3. Out of these localization methods, device-free localization is examined more closely in Chapter 4.

Later in this report, a more abstract understanding of floorplan is used. Instead of mapping an estimated position to room numbers, the room number is derived directly from observations. It turns out that the position estimation can be skipped entirely whilst maintaining a high room-level localization accuracy (Chapter 5).

1-4 Available devices

As mentioned in Section 1-2, the available devices are limited. Only a smartphone and DoBeacons are used in this research. This section gives a short overview of the sensors that these devices contain.

1-4-1 Smartphone

The following sensors can often be found on a common smartphone:

- Accelerometer

- Gyroscope
- Magnetometer
- Camera
- WiFi/Bluetooth Received Signal Strength (RSS)

The usefulness of these sensors for indoor localization and/or mapping, however, is limited. The accelerometer and gyroscope do not respond to the environment at all (although they can be used for odometry). The magnetometer can detect local changes in the magnetic field, which can be used to distinguish different locations [1]. However, it does not have a predictable response to the presence of objects like walls or doors. The camera *can* recognize walls, doors and other objects, but as mentioned before it requires too much user interaction to be useful.

That leaves only the WiFi and Bluetooth signal strength measurements. These measurements do provide information about the environment. The RSS may be used to estimate distances to known points in the environment, and can thereby be used for localization. The signal strength is also influenced by obstacles such as walls and doors, which may provide additional information. The interaction between obstacles and Bluetooth RSS is explored further in Chapter 2. Because the DoBeacons only contain a Bluetooth module, WiFi is not considered in this report, but overall similar results are expected.

1-4-2 DoBeacons

DoBeacons are simple devices that contain only one sensor that can be used for indoor localization and mapping: a Bluetooth Low Energy (BLE) module which can be used to broadcast advertisements and measure the RSS of nearby devices. These beacons can be part of other devices (for instance, a power socket that can be turned on and off remotely), and have the ability to communicate with each other via Bluetooth.

Users are not expected to buy hundreds of beacons for the purpose of indoor localization. In a realistic setting the number of beacons is limited to roughly 1 or 2 per room.

Chapter 2

A closer look at Bluetooth signals

In the previous chapter it was suggested that Bluetooth (and WiFi) Received Signal Strength (RSS) measurements are the only sensors on a smartphone that can detect walls in the environment. If walls can be located, these can be used to construct a floorplan. This chapter aims to answer the following question: assuming that the positions of the transmitter and receiver are known, is it possible to detect the presence of walls?

To answer this question, the influence of walls on the RSS is measured and compared to the influence of other disturbances (antenna orientation, the user's body, time-of-day). Several simple tests are performed to provide a better insight into the behavior of Bluetooth signals. These tests are not intended to provide accurate, quantitative models of disturbances, but serve to demonstrate the order of magnitude of different effects.

2-1 Distance

In order to recognize the attenuation caused by walls, the observed RSS has to be compared to a baseline value. The log-distance path loss model presented in [2] provides an estimate of the average path loss $\bar{PL}_{[\mathrm{db}]}(d)$ in dB at distance d, given a known attenuation $PL_{[\mathrm{dB}]}(d_0)$ at known distance d_0:

$$\bar{PL}_{[\mathrm{dB}]}(d) = PL_{[\mathrm{dB}]}(d_0) + 10n\log_{10}\left(\frac{d}{d_0}\right) \tag{2-1}$$

For free space, the path loss exponent $n = 2$. In practice, obstructions or other disturbances can lead to a larger value of n. The inverse of this function can be used to predict the distance to a device.

A measurement was performed where the RSS was measured at a distance of 50, 100 and 200 cm in an empty room. Table 2-1 reports the average RSS and its standard deviation. With $n = 2$, a decrease of 6 dB is expected when the distance is doubled (9 dB if $n = 3$). The RSS does indeed decrease with distance, the observed change lies in the same order of magnitude as predicted by the log-distance path loss model, with an n between 2 and 3.

Distance [cm]	RSS [dBm]	ΔRSS [dB]
50	-47 ± 2.16	0
100	-52 ± 2.30	-5
200	-66 ± 4.07	-19

Table 2-1: Mean and standard deviation of RSS vs. distance when no obstacles are present.

2-2 Influence of walls and doors

If the presence of walls can be detected, these measurements can be used to construct a floorplan of the environment. The presence of walls or doors in the line-of-sight between a transmitter and receiver is expected to attenuate the signal. Values reported in literature suggest that the power loss factor of walls lies between 2 and 6 dB depending on the material [3].

To test the influence of walls and doors, measurements with and without these obstacles were performed. The results are shown in Table 2-2 and 2-3. The influence of walls and doors is small, 1-2 dB. When the measurement with a wall between beacons was performed, the RSS unexpectedly increased, most likely because of nearby reflections or other disturbances. This is already an indication of the relatively small effect of walls on RSS.

	RSS [dBm]	ΔRSS [dBm]
Door opened	-68 ± 1.75	0
Door closed	-70 ± 3.49	-2

Table 2-2: Change in RSS with a door between the transmitter and receiver.

	RSS [dBm]	ΔRSS [dBm]
No wall	-74 ± 4.33	0
Wall	-73 ± 3.32	$+1$

Table 2-3: Change in RSS with and without a wall at a distance of 170 cm.

2-3 Influence of the user's body

When someone carries a smartphone in his/her pocket, their body is expected to block a significant part of the Bluetooth signal. Measurements where a person is located between a Bluetooth transmitter and receiver are reported in Appendix A. The effect of the user's body is in the order of 5-20 dB depending on the distance to the devices.

2-4 Antenna anisotropy

The antennas used in the DoBeacons and development boards are generally not isotropic. To test the influence of antenna anisotropy, RSS measurements were performed while the

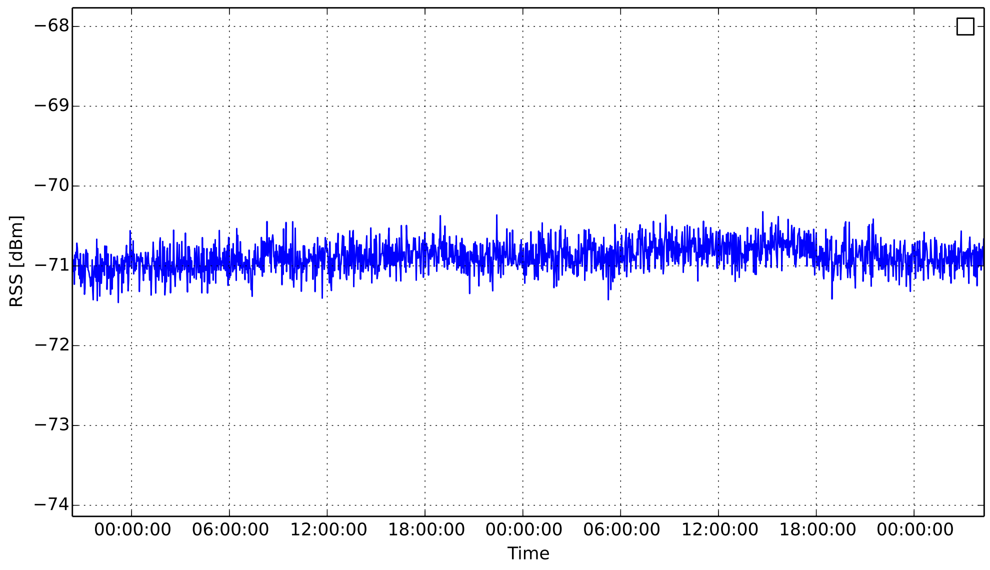


Figure 2-1: Moving-average (60 s window) of RSS over a 48-hour period.

transmitter was rotated. Rotation of the transmitter caused a change in the order of 10-15 dB. Similar values are reported in literature, for instance [4], but the exact value strongly depends on the antennae of both devices.

2-5 Change in signal strength throughout the day

To see whether the observed RSS changes throughout the day because of changes in temperature, humidity or other environmental factors, the signal strength between two beacons was recorded over two days. The measurement was performed during the weekend, so no people were present in the building during the recording.

The resulting RSS is shown in Figure 2-1. The moving average RSS stays within 1 dB of its initial value.

2-6 Conclusion

The simple tests performed in this chapter demonstrate the effects of distance, walls, doors, the user's body, antenna anisotropy and the change in signal strength throughout the day. The influence of walls and doors is significantly smaller than the influence of antenna orientation or shadowing effects from the user's body, both of which are unknown. It is therefore not feasible to detect walls by comparing the observed RSS to the value predicted using the log-distance path loss model.

Chapter 3

Indoor localization for floorplan estimation

3-1 The need for indoor localization

In Chapter 2 it was shown that it is not feasible to directly extract information about walls from Bluetooth Received Signal Strength (RSS) measurements. Disturbances resulting from the orientation of the antenna are significantly larger than the power loss caused by walls. Even if the orientation is known, other disturbances like the body of the user or parameter estimation errors can cause large differences in RSS.

Because it is not possible to directly observe the presence of walls, information to construct a floorplan has to be derived from other signals in the environment. These signals should be measurable and should contain information about the shape of the environment. Only two signal sources were found that agree with these requirements:

- Doors. The state of a door (open, closed) influences nearby RSS measurements through shadowing. Since doors are always placed in walls, the (estimated) position of doors can provide information about the location of walls. The actual usability of this signal is low, since only walls with doors can be observed. Furthermore, doors can only be detected reliably when they are located exactly between beacons.
- People. People walk around in the environment but are unable to pass through walls. By tracking their movements, it is possible to see which areas are traversable, and this information can be used to construct a floorplan [5, 6]. People can be tracked by different means, for instance using odometry, trilateration or device-free localization.

3-2 Indoor localization techniques

Using the motion traces of people walking around in the environment, it is possible to construct a floorplan. Different methods have been proposed: in [6] an occupancy grid map is used, while [5, 7] use clustering techniques.

The floorplan construction methods described in [6, 5] have in common that they do not depend on a specific source of the motion traces. A diverse group of tracking methods are available, each with its own advantages and disadvantages. This section gives an overview of the available methods, and ends with a selection of the most suitable approach.

3-2-1 Proximity

One of the simplest forms of indoor localization is based on the proximity to beacons in known positions. The receiver is assumed to be at the position of the beacon with the highest RSS. This method does not depend on a signal propagation model, and is therefore easy to implement and does not require tuning.

However, with a limited number of beacons the resolution is very coarse. While localization at room level may be possible depending on the placement of the beacons, the motion traces are not accurate enough for floorplan construction using the methods described above. A further complication is that DoBeacons are generally placed inside walls. An estimated position at the beacon location is therefore guaranteed to be incorrect, and it may be difficult to distinguish between neighboring rooms.

3-2-2 Multilateration

Instead of selecting the location of the beacon with the highest RSS, it is also possible to estimate positions using the RSS values of multiple beacons. By estimating the range to beacons based on the RSS, it is possible to find a position that minimizes the error between the expected and measured distances. Unlike proximity-based solutions, this method can find positions between beacons, thereby greatly improving the resolution.

This comes at a cost: a signal propagation model is required to transform the RSS measurements into approximate distances. In itself this is not a prohibitive disadvantage, but the propagation model is sensitive to disturbances, especially at larger distances.

"If we were to assume a modest measurement noise such as 3 dBm, this would result in a ranging uncertainty of the same order of magnitude as the distance to the source; within a metre of the transmitter a positioning uncertainty of only a few centimetres would be possible, however, at 10 m the ranging error would be around 5 m" [8].

With the limited number of beacons per room and the large disturbances found in Chapter 2, the average tracking error will be in the order of several meters, making this method unsuitable for floorplan construction.

3-2-3 Fingerprinting

Fingerprinting-based methods use the concept of 'fingerprints', a unique collection of measurements taken at a specific location. Fingerprints can be compared to each other to judge their similarity, where similar fingerprints are assumed to come from locations that are in close proximity. Localization can be performed by comparing the current measurements to the recorded fingerprints and selecting the location with the most similar result (or interpolating between multiple results). An overview of localization methods is given in [9]. Two often-cited examples of fingerprint-based tracking systems are *RADAR* [10] and *Horus* [11]. Fingerprint-based localization can be performed without assumptions about signal propagation or knowledge of beacon locations and can use reflections and shadowing effects in a *static* environment to its advantage.

The fingerprints do not have to be based on previously deployed beacons. It is also possible to use signals from external sources such as cell phone signals [12]. It is even possible to use local disturbances of the ambient magnetic field as fingerprints [1].

The downside of using fingerprints is that a fingerprint map (a collection of position-labeled fingerprints) has to be created. Compared to the other tracking methods the calibration phase is more time-consuming, as the required amount of positioned fingerprints is generally larger than the amount of beacons that may otherwise have to be located. Furthermore, changes in the environment can cause changes in local measurements. Not only is it necessary to create a map, but it also needs to be corrected after changes in the environment have occurred.

It is possible to avoid the time-consuming calibration step by using the similarities between samples to estimate their position, or by relaxing the required resolution to room-level localization. This is investigated further in Chapter 5.

3-2-4 Odometry

Using the sensors on a common smartphone, it is possible to estimate motion traces by odometry. The accelerometer is used as a pedometer and detects forward motion, the magnetometer is used as a compass and determines the current heading.

Since the steps are integrated over time, odometry is susceptible to drift. It is essential to correct this drift, otherwise the motion traces become too inaccurate for floorplan construction. To correct this drift, an absolute estimate of the position is required. Different methods have been proposed: using fingerprints and using anchor points.

The fingerprinting methods described above provide an absolute estimate of the position and can therefore be used to correct drift. However, this means that the disadvantages of fingerprinting also apply to this solution, as a map of fingerprints needs to be maintained.

It is also possible to correct drift using just the accelerometer to recognize *anchor points* [5, 13]. Anchor points are locations that can be recognized reliably and have a known location. The *CrowdInside* and *UnLoc* floorplan construction systems presented in [5, 13] use the accelerometer to recognize anchor points such as stairs, escalators and elevators. These anchor points can then be used to correct and align the motion traces. Other sensors such as GPS near building entrances may also be used to detect anchor points [5, 13].

Heading estimation for odometry using a mobile phone is a non-trivial problem. The orientation of the mobile phone relative to the user is not known, and not guaranteed to be constant. Solutions exist to estimate this orientation or correct it afterwards (for instance, using anchor points), but situations where this orientation is not constant (e.g. taking the phone out of your pocket) are difficult to correct.

Unlike the other methods proposed here, odometry can only estimate a location relative to the last known position. Especially when using anchor points, this means that the current location is unknown until at least one anchor point has been observed.

3-2-5 Device-free Localization

The final method under consideration is device-free localization [14, 15, 16]. Unlike the previously presented solutions, device-free localization does not require people to carry a mobile phone. Instead, the presence of people is detected by shadowing effects on signals between beacons. The RSS between beacons is compared to a baseline recorded when no people are present.

To estimate the position of obstacles, the location of all beacons should be known (or estimated [17]). However, no conversion is being made between RSS and distance, so no signal propagation model is required. Only shadowing effects are considered in this approach.

Motion traces are created by passing the changes in RSS through a tracking filter. The tracking filter uses an observation model to estimate the presence and location of people based on the change in RSS, and can use a prediction model to estimate the motion of targets between observations.

Since people can only be observed between beacons, a sufficient number of beacons is required to provide adequate coverage. The required number of beacons is generally higher than for the other methods proposed here.

The fact that users do not need to interact with the system is a big advantage. Depending on the use case, this might make it easier to collect a large amount of traces and as a result the floorplan can be constructed in a smaller amount of time. The detection of people without smartphones also has other interesting applications for home automation outside of floorplan construction, since it does not require users to carry their phone at all times, and can also recognize guests that do not carry a previously registered device.

3-3 Trade-off and selection

The fingerprinting-based solutions are in general quite simple to implement and tune. They can cope with unknown beacon positions and do not make assumptions about signal propagation. The required number of beacons is low, especially when using signals coming from an external source. The weak point of fingerprinting methods, however, is the time-consuming calibration phase and difficulty to cope with changes in the environment.

A combination of odometry with (RSS-) based fingerprinting does not offer many advantages compared to fingerprinting itself. The estimated position is expected to be more accurate since more information is available, but this comes at the cost of having to implement a more complex system. Odometry does provide advantages when the number of beacons or radio coverage is very low.

Odometry using anchor points has the advantage that it does not depend on beacons at all. Instead, anchor points are used to correct drift. This method makes more assumptions about the environment, as it assumes that enough anchor points are present to correct the

	Fingerprinting BLE/WiFi RSS	Fingerprinting other signals	Odometry with fingerprinting	Odometry with anchor points (*CrowdInside*)	Device-free Localization
Simplicity	+	+	-	--	--
Unknown beacon locations	++	++	++	++	--
Number of beacons	+	++	+	++	-
Calibration	--	--	--	+	+
Change in environment	--	--	--	++	-
User interaction	+	+	+	+	++
Assumptions	++	+	++	--	+
Localization	++	++	++	--	+

Table 3-1: Comparison of strengths and weaknesses of tracking methods. '+'s indicate an advantage relative to other methods, '-'s a disadvantage. None of the proposed methods is better than the rest on all criteria, so there is no clearly optimal solution.

drift before it becomes too large. It is also more difficult to implement than odometry with fingerprinting, as a reliable classifier should be developed to recognize anchor points. It is not able to estimate a location when no anchor points have been observed, which means that the location estimator needs to be running continuously.

Device-free localization is the only method that requires the position of beacons to be known. In exchange, the calibration is simple compared to the fingerprinting methods. Only a few model parameters need to be estimated, and "tracking performance is relatively robust to the parameter choice" according to [16], so it may be possible to use predetermined values for these parameters. This method may have difficulties distinguishing changes in the environment from obstacles, but it can slowly adjust its baseline RSS values to handle these cases. The main advantage of device-free localization is that it does not require users to carry a phone or other measurement device.

An overview of these considerations is shown in Table 3-1. None of the methods has a clear advantage over its alternatives, each has its own strengths and weaknesses. Because of its additional benefits outside of tracking, device-free localization is chosen as a tracking filter and is described in more detail in Chapter 4. Fingerprinting provides a nice alternative if the calibration phase can be shortened or skipped altogether, this is explored further in Chapter 5. Because of time constraints, no further investigation has been performed on odometry-based solutions.

Chapter 4

Concept 1: Device-free Localization

When people pass between two Bluetooth beacons, their shadowing effect causes a momentary drop in Received Signal Strength (RSS). An example of this shadowing effect is given in Figure 4-1. Device-free Localization tracks people by detecting their presence using this drop in RSS. The algorithm to perform this tracking task consists of two major parts: baseline estimation and a tracking filter.

To detect the shadowing effect caused by people, RSS measurements are compared against a baseline value. The idea behind baseline estimation is very simple: collect samples when no people are present, and average these to estimate a baseline. There are, however, some practical problems that need to be solved first. The baseline estimation is described in Section 4-1.

The tracking filter uses the difference in RSS to estimate the locations of people. In this implementation, this function is performed by a particle filter, which uses a prediction model to predict the current state and an observation model to correct the prediction when new measurements arrive. This filter is described in more detail in section Section 4-2.

4-1 Baseline estimation

Obstacles are detected by their shadowing effect, i.e. the drop in RSS compared to an obstacle-free baseline. The shadowing effect is usually determined by comparing the instantaneous RSS between beacons to a value that has been determined during calibration, but other methods have been proposed as well:

- Comparing a moving average of RSS over a short timespan to a moving average over a long timespan [14]. This reduces the influence of noise and can adapt to changes in baseline RSS, but using a short moving average instead of the instantaneous RSS leads to an increased response time and may have difficulty responding to short peaks.

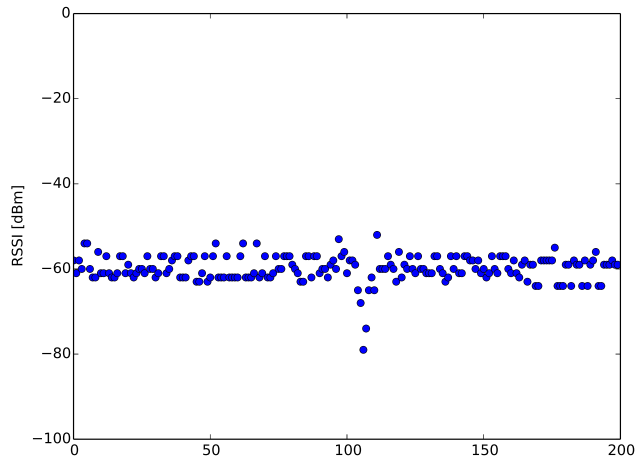


Figure 4-1: An example of the shadowing effect that occurs when a person walks between two beacons.

- Comparing the variance determined over a sliding window to a baseline variance [14]. This method gives a more robust indication of obstacles as it can handle positive changes in RSS caused by reflections. However, using this method it is not possible to detect stationary obstacles as these do not significantly change the variance of the RSS.
- Detecting large changes in RSS by looking at its derivative [18]. This method is also more suitable for moving objects, but because it is taking a difference between samples it is very sensitive to noise.

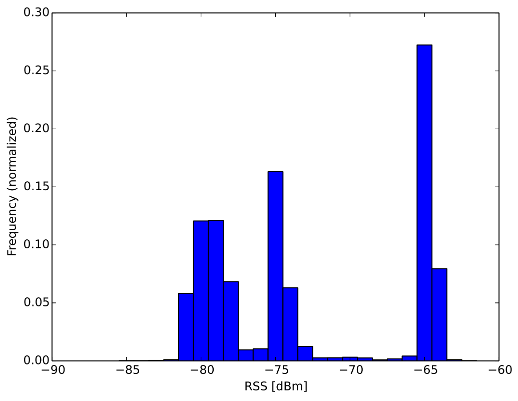
The observable region between beacons is small (approximately 0.2 m [16], Appendix A). In order to reliably detect moving objects when they cross this region, a high bandwidth is required because the shadowing effect only lasts for a few samples. Using a moving average to reduce noise is therefore not an acceptable solution. Instead, the noise will be accounted for in the tracking filter using the estimated variance σ_z^2 of the RSS measurements.

Since the measurements in Section 2-5 showed that the RSS does not change throughout the day, a constant baseline will be assumed. This baseline value is estimated during a calibration phase when no people are present.

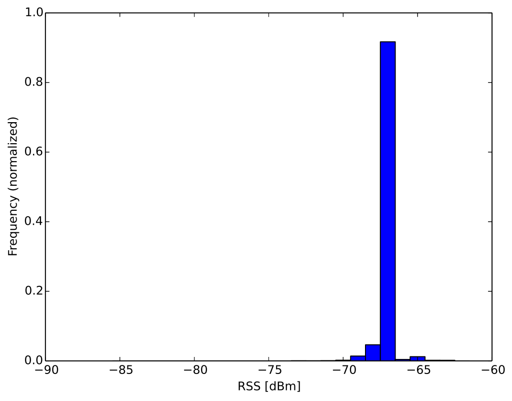
4-1-1 Influence of Bluetooth advertising channels

The models presented in [19, 16] assume that additive Gaussian noise is present in the measurements. This is, however, not necessarily the case. Consider the histogram of RSS values shown in Figure 4-2a. Instead of a single normal distribution, three peaks can clearly be distinguished.

By default, Bluetooth transmits advertisements on three separate channels. This redundancy allows BLE advertisements to keep working even when one or two of these channels are not available due to interference with other devices. As shown in [8], the narrow bandwidth of the



(a) Advertising on 3 channels.



(b) Advertising on 1 channel.

Figure 4-2: Influence of the BLE advertising channels on the RSS distribution. Histograms indicate the relative frequency of observed RSS values. When advertising on three channels, three distinct peaks are observed in the RSS distribution. When advertising on one channel, only one peak remains.

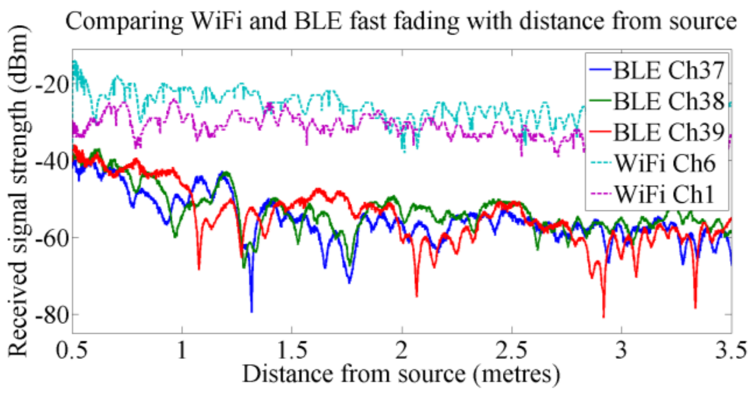


Figure 4-3: "The RSS variation over both time and distance for an iPhone moving on a slow conveyor belt away from a BLE beacon and two WiFi Access Points. The motion was so slow (around a millimetre per second) that velocity Doppler effects can be ignored." *Image/caption source: [8].*

Bluetooth channels makes them more susceptible to interference by reflections. The results presented in [8] (Figure 4-3) clearly show the influence of this interference, which can be recognized as sharp notches in the measured RSS values. It is important to notice that the observed RSS is not the same for the three channels, even though their measurements are performed at the same distance. This causes the separate peaks in the histogram shown in Figure 4-2a and also explains why no histograms with more than three peaks were found.

This multi-modal distribution is a problem for baseline estimation. Instead of one baseline as assumed in literature, each channel has its own baseline RSS value. Unfortunately, the Bluetooth stack[1] that is used on the DoBeacons does not report the channel on which an advertisement is received. It is therefore not possible to know which baseline to compare a sample to.

To solve this problem, the DoBeacons are reconfigured to only transmit advertisements on a single channel. This produces the histogram shown in Figure 4-2b. Instead of a multi-modal distribution, there is now a single, sharp peak, which greatly improves the quality of the measurements.

4-2 Tracking filter

The tracking filter uses the drop in RSS to estimate the position of people. The tracking problem can be formulated as a Sequential Bayesian Filtering problem, but in general these filters cannot be implemented directly so approximations have to be made. Since the problem is highly nonlinear (especially the observation model), Kalman filters are not expected to perform well, so a different approach is required.

In [20, 19, 16] a particle filter is used to perform tracking. A particle filter uses a particle-based discrete approximation of the posterior probability $p(x_k|z_{1:k})$. In [21], a Bayesian Grid Array (BGA) is used instead of a particle filter. The BGA uses a grid-based approximation of $p(x_k|z_{1:k})$. Compared to particle filters, a grid-based discretization ensures that the entire state space is covered at all times, while a particle filter concentrates particles at the expected target positions, providing a better coverage in the most likely areas but poor coverage in other parts of the state space.

In this research, the particle filter approach is taken because its feasibility for tracking a variable number of targets has already been shown in literature [22], while BGA has only been shown to work with a single target. Additionally, the particle filter approach makes it easier to use a more complicated prediction model if this need arises (one that keeps track of the targets velocity), which would be difficult to implement in the BGA solution because of the exponential growth of the grid when states are added.

4-2-1 Particle filter

Several approaches towards particle filter based tracking have been proposed. In [16] these are compared in terms of accuracy and computational demands.

[1]Nordic S110 Softdevice 8.0

The Sequential Importance Resampling (SIR) or Bootstrap particle filter is the simplest option of all, but it is difficult the use for multi-target tracking as the number of required particles increases exponentially with the number of targets.

A Multiple Particle Filter (MPF) [20] solves this problem using a fixed number of particles per target, of which the average is taken as this target's location during the weight update. Instead of the exponential complexity of the SIR filter, the MPF grows linearly with the number of targets. This filter can be used to track multiple targets, but requires the number of targets to be given.

The Additive Likelihood Moment (ALM)/Probability Hypothesis Density (PHD) filter [23, 19, 22] is also able to track multiple targets, but approaches this problem in a different way. Where the SIR filter and MPF model the multi-target state as a vector $\mathbf{X}_k = \begin{bmatrix} x_{k,1} & x_{k,2} & \cdots & x_{k,n} \end{bmatrix}$ (with $x_{k,i} \in \mathcal{X}$ the states of the n individual targets), the PHD filter represents this state as a random finite set $X_k = \left\{ x_{k,1} \quad x_{k,2} \quad \cdots \quad x_{k,n} \right\} \subseteq \mathcal{X}$. This set representation is better able to handle a changing number of targets than a vector of fixed length. Instead of estimating the multi-target state probability density function $p(\mathbf{X}_k | z_{1:k})$, the filter estimates the *Probability Hypothesis Density (PHD)* [23] of random finite set X_k. The PHD is a function defined on the single-target space $\mathcal{X}$, which in a multi-target tracking setting has as property that the integral taken over a region in the state space $\mathcal{X}$ is equal to the expected number of targets in that region. The advantage of estimating the PHD instead of the probability density function is that it is defined on the single target space $\mathcal{X}$ instead of the multi-target space $\mathcal{X}^n$. As a result, the required number of particles is drastically lowered in multi-target applications.

In [16], the performance of the PHD filter is compared to the Markov Chain Monte Carlo (MCMC) filter [24]. Both the MCMC and PHD filters have a good accuracy, but the MCMC filter needs more particles and CPU time to get the same accuracy as the PHD filter.

In order to track a variable number of targets, an estimate of the number of targets is required. The PHD filter estimates the number of targets by finding clusters in the particle set. In [22] an extension to this filter is presented: the Cardinalized Probability Hypothesis Density (CPHD) filter. This filter explicitly estimates the cardinality of the probability hypothesis density, but requires a particle birth/survival model to do so.

4-2-2 Prediction model

Two models are combined in the particle filter: a prediction model and an observation model. The prediction model predicts the current state of the environment after a certain amount of time has elapsed, based on the previously estimated state. For each particle, the prediction model provides a probability distribution $p(x_k | x_{k-1})$ of the particle's next state.

Prediction models of varying levels of detail can be used. Simple random walkers with a two-dimensional state consisting of the x- and y-position, or more complex models that also keep track of the target's velocity. In this case, a random walking model is chosen because of its simplicity. The state is predicted according to $x_k = x_{k-1} + v_{k-1}T$, where T is the sampling period and v_{k-1} is a random two-dimensional velocity sampled from the normal distribution $\mathcal{N}(0, \sigma_v^2 I_{2\times 2})$ with $\sigma_v = 1.5\,\mathrm{m\,s^{-1}}$.

4-2-3 Observation model

The observation model predicts the expected measurements based on the estimated state of the target(s). The shadowing effect $g_j(x_k)$ of a single target at x_k on link j in an open space can be predicted using the *exponential model* [17]:

$$g_j(x_k) = \phi \exp\left(-\frac{\lambda_j(x_k)}{\sigma_\lambda}\right) \tag{4-1}$$

The distance $\lambda_j(x_k)$ is defined as:

$$\lambda_j(x_k) = d_1(x_k) + d_2(x_k) - d_{12} \tag{4-2}$$

where $d_1(x_k)$ and $d_2(x_k)$ are the distances between the target and the beacons, and d_{12} is the distance between the beacons. ϕ and σ_λ are tuning parameters that need to be determined experimentally.

As shown in [19, 24], the shadowing effects are additive:

$$g_j(X_k) = \sum_{n=1}^{N_k} g_j(x_{k,n}) \tag{4-3}$$

The *exponential model* assumes that the measurements are performed in an open space. In other words, it only considers the shadowing effect and does not take reflections into account. For indoor environments, the *magnitude model* [16] is more suitable, because it also takes reflections into account. Instead of looking at just the difference in RSS, the magnitude model assumes that reflections are also caused by the presence of people, and therefore looks at the *absolute* difference in RSS. The magnitude of this absolute difference is still modeled using the same formula as the multi-target exponential model, but the parameters can be slightly different. In this report the values from [16] are used: $\phi = 4\,\text{dB}$, $\sigma_\lambda = 0.2\,\text{m}$. A short test was performed to demonstrate the validity of the observation model (Appendix A). This test produced slightly different parameters ($\phi = 5.26\,\text{dB}$, $\sigma_\lambda = 0.07\,\text{m}$), these did not improve the tracking accuracy.

4-2-4 Target number estimation

The PHD tracking filter implementation presented in [22] uses k-means to estimate the target locations from the collection of particles. k-means requires the number of clusters to be given, but this number is not known if the number of targets is variable. To estimate the number of targets, [22] proposes to run k-means for all values of k (up to a predetermined limit), and select the k that maximizes the average silhouette coefficient [25].

There are two practical problems with this method. First of all, the silhouette coefficient can not be determined for less than two clusters. It is therefore not possible to recognize situations in which no targets or only one target is present. Secondly, k-means is not able to reject outliers. While the influence of outliers on the estimated cluster means is small, they are considered in the target number estimation and may cause an overestimation of the number of targets.

Instead of a combination of k-means with silhouette, in this research the use of Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [26] as a clustering algorithm is proposed. DBSCAN is a density-based clustering algorithm that can form clusters when a given number of samples $minPts$ lies within a given radius ϵ. Unlike k-means, it is not necessary to know the number of clusters beforehand. Another advantage of DBSCAN is that it is able to recognize and reject outliers.

The computational complexity of the k-means/silhouette solution is dominated by the quadratic complexity of the silhouette calculation. DBSCAN has a worst-case complexity of $O(N^2)$, but in practice this can be reduced to $O(N \log N)$ when an appropriate indexing structure is used to perform region queries [26].

The k-means/silhouette combination has no tuning parameters. For DBSCAN, values for $minPts$ and ϵ are required. The values for these parameters have been determined by trial-and-error and are set to 20% of the number of particles per target and 0.6 m respectively. Once the feasibility of using DBSCAN is shown, these can be tuned further to optimize the performance of the tracking filter.

In the experiments performed in this chapter, k-means is used when the number of targets is constant since it is impossible to estimate a fixed number of clusters using DBSCAN. When the number of targets is variable, both k-means/silhouette and DBSCAN will be used and their results will be compared.

4-2-5 Implementation

The implementation of the tracking filter follows the auxiliary particle filter based PHD filter algorithm as described in [22]. The tracking filter is written in JavaScript (ES5). This allows the code to be reused in both Android and iOS apps using cross-platform frameworks such as Cordova, Phonegap or React Native. The code is available at https://github.com/tomvand/dfljs.

4-3 Performance evaluation: simulation

4-3-1 Method

A simulated environment is provided which contains beacons with known positions and actors that cause a shadowing effect between beacons. The simulated measurements follow the additive exponential model with additive Gaussian noise and have a fixed probability of actually being received.

The simulation and tracking filter are set up as follows: the tracking filter uses 500 particles per target and 500 auxiliary particles. The simulation runs for 10 seconds with a sampling period of 0.25 s. The actors use a random walking model with x and y accelerations sampled from $\mathcal{N}(0, \sigma_v^2)$ where $\sigma_v^2 = 0.2\,\mathrm{m/s^2}$. Measurements follow the additive exponential model and include an additive Gaussian noise with variance $\sigma_z^2 = 1.0$. Measurements are received with a probability of 0.40 (otherwise they are ignored). The performance is evaluated over 10 runs with random initial configurations.

In cases where the number of targets is constant, the estimation accuracy is evaluated at each timestep using the Optimal Mass Transfer (OMAT) performance metric [27]:

$$d_p(X,Y) = \left(\frac{1}{n} \min_{\pi \in \Pi} \sum_{i=1}^{n} d(x_i, y_{\pi(i)})^p \right)^{1/p} \tag{4-4}$$

where Π is the set of all possible permutations of the clusters, x_i is the position of target i, y_i is the position of a cluster and $d(x,y)$ is the distance between x and y. In other words, each target is assigned a cluster such that the average distance between targets and clusters is minimized. In this report, a value of $p = 2$ is used.

When the number of targets can vary, the OMAT metric cannot be used because the number of clusters may not be equal to the number of actual targets. The Optimal Subpattern Assignment (OSPA) metric is an extension to the OMAT metric that puts an upper bound c on the tracking error, and also assigns this error c to clusters that do not correspond to any target or vice versa [28]:

$$d_p^{(c)}(X,Y) = \left(\frac{1}{n} \min_{\pi \in \Pi} \sum_{i=1}^{m} d^{(c)}(x_i, y_{\pi(i)})^p + c^p(n-m) \right)^{1/p} \tag{4-5}$$

where $d^{(c)}(x,y) = \min\{d(x,y), c\}$, $X = \{x_1, \cdots, x_m\}$ and $Y = \{y_1, \cdots, y_n\}$ with $n > m$ (when $n < m$, $d_p^{(c)}(Y,X)$ is calculated). (Note that when the estimated number of targets is correct and $d(x,y)$ does not exceed c, the OMAT and OSPA errors are equal.)

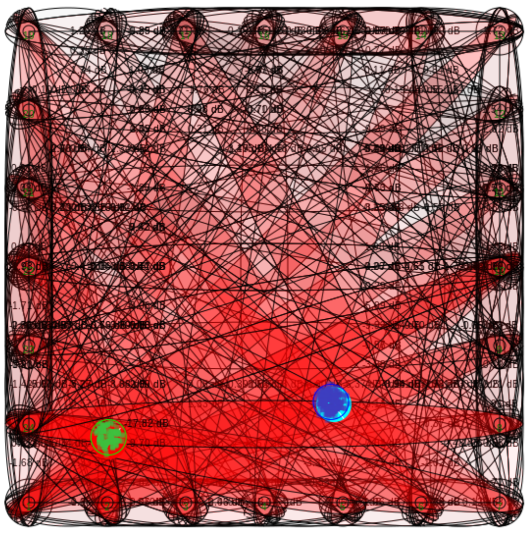
The tracking filter is first tested in an environment similar to the test setup in [16] (Figure 4-4a). The environment consists of 24 beacons surrounding an $8\,\mathrm{m} \times 8\,\mathrm{m}$ area. Two targets perform random walking starting at a random initial position within the bounds of the environment. Because the number of targets is known and fixed, clustering is performed using k-means.

The same test is performed again in a more realistic environment. This environment is a model of the physical test setup, and consists of six beacons set up in an $8\,\mathrm{m} \times 6\,\mathrm{m}$ room (Figure 4-4b). The same filter settings and performance measure are used.

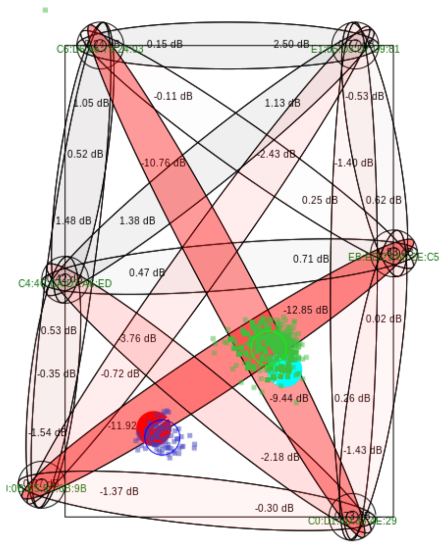
Finally, the test is performed again, this time with a variable number of targets. The target number changes when a random walker leaves the bounds of the environment. An additional random walker was added, because the silhouette-based target number estimation cannot recognize less than two targets. Two clustering and target number estimation methods are used: a combination of k-means and silhouette as described in [22], and DBSCAN as proposed in this report. The performance is evaluated using the OSPA metric.

4-3-2 Results

The performance of the tracking filter was first evaluated in a simulated setup with 24 beacons surrounding an $8\,\mathrm{m} \times 8\,\mathrm{m}$ room. The 24 beacons provided good coverage of the area (Figure 4-4a). Over 10 trials, the average OMAT error was $0.08\,\mathrm{m}$. The best and worst tracking results are shown in Figure 4-6.



(a) $8\,\mathrm{m} \times 8\,\mathrm{m}$ room surrounded by 24 beacons.



(b) Physical test setup with 6 beacons.

Figure 4-4: Illustration of the tracking filter in action in the two simulated environments. The ellipses indicate the measurement area between beacons. They are drawn with a semi-minor axis of σ_λ and give an idea of the coverage in the two environments. The color of the ellipse indicates the observed disturbance caused by the targets, which are displayed as solid disks. Finally, the particles display the state of the particle filter after resampling. The particle color indicates which cluster they belong to, and the circles indicate the estimated target positions.

	OSPA error [m]		
	c=1.0 m	c=2.5 m	c=5.0 m
k-means/silhouette	0.88	2.03	3.93
DBSCAN	0.68	1.39	2.58

Table 4-1: Average OSPA tracking error in the simulation environment based on the physical test setup with a variable number of targets (0-3) for two target number estimation and clustering methods: k-means combined with silhouette, and DBSCAN.

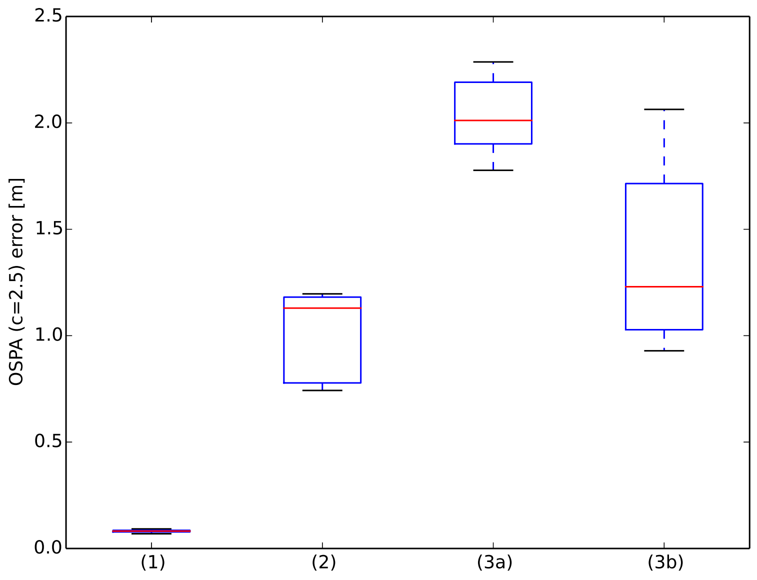


Figure 4-5: Boxplot of the OSPA error for the four simulated test cases. The error was measured over 10 random trials. *(1) 24 beacons, 2 targets, k-means clustering. (2) 6 beacons, 2 targets, k-means clustering. (3a) 6 beacons, 0-3 targets, k-means/silhouette clustering. (3b) 6 beacons, 0-3 targets, DBSCAN clustering.*

In the environment modeled after the physical test setup, the coverage is significantly less dense than in the first environment (Figure 4-4b). The average OMAT error was 1.17 m. Figure 4-7 shows the best and worst tracking results.

Two further tests were performed where the number of targets was no longer kept constant. Since the number of estimated targets does not necessarily match the actual number of targets, the OSPA metric is used to penalize cardinality errors. Table 4-1 lists the OSPA errors for three different values of c: 1.0 m, 2.5 m and 5.0 m.

Two clustering techniques were used. k-means clustering combined with silhouette-based target number estimation and DBSCAN. For all examined values of c, the DBSCAN clustering outperformed the k-means/silhouette clustering (Table 4-1). Finally, the cardinality estimate of both clustering methods has been recorded, these are shown in Figure 4-10.

A summary of the tracking errors in these four test cases is shown in Figure 4-5.

4-3-3 Discussion

When reducing the number of beacons from 24 to 6, the tracking error increases significantly. Most of the error seems to be caused by incorrect clustering. When the targets are close together, their particles form one big cluster instead of two separate clusters. k-means is still looking for a second cluster, which is then formed by outliers.

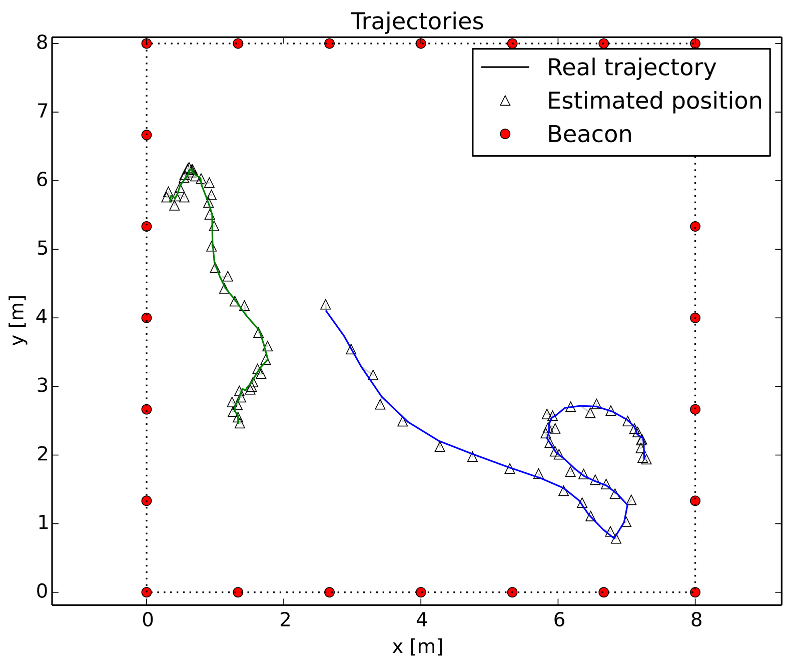
This effect can be observed in Figure 4-7. In the worst case, the targets stay close together during the entire trial. The second cluster is formed by outliers, and often lies far away from the actual position of the second target. In the best case, the targets trajectories are further apart and the effect is less prominent.

In the test setup with 24 beacons, this effect does not occur. The 24-beacon setup provides a good coverage of the entire tracking area, but the 6-beacon setup has several areas with less-than-sufficient coverage (Figure 4-4). Particles will sometimes accumulate in these areas, leading to a cluster of outliers that is not present in the 24-beacons setup.

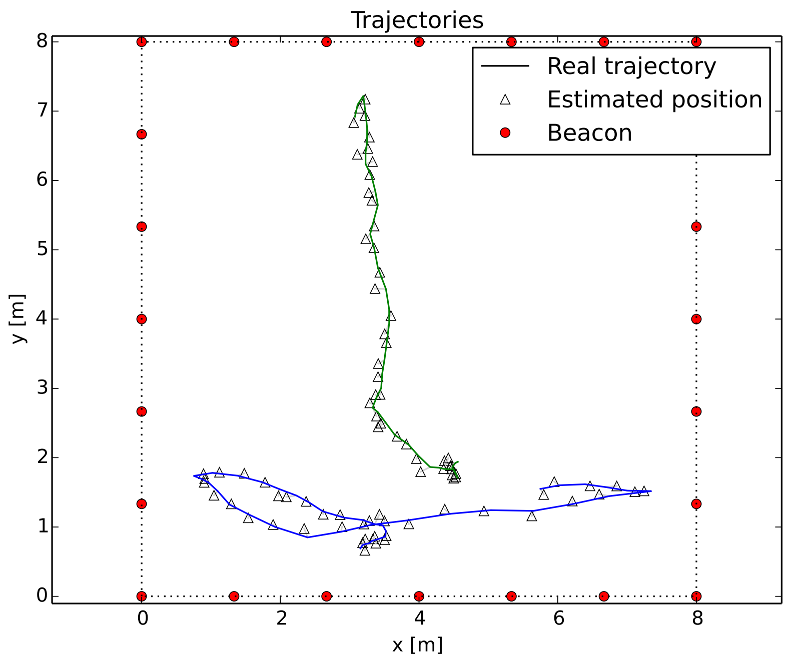
Two clustering- and target number estimation methods are compared in this experiment: k-means/silhouette and DBSCAN. The silhouette can not be determined for less than two clusters, which is a serious disadvantage for practical tracking problems. Furthermore, the outliers caused by insufficient coverage of the tracking area cause the k-means/silhouette combination to overestimate the number of targets (Figure 4-8, 4-10).

DBSCAN provides a better estimate of the number of targets. It does, however, perform badly during the first samples of the simulation (Figure 4-10). For each particle, DBSCAN requires a minimum number of surrounding particles to form a cluster. In this particle filter implementation, the number of particles in the filter depends on the estimated number of targets. It is therefore not possible to immediately recognize all three targets, because the filter does not contain enough particles. After the first target is recognized, the number of particles is increased, which allows the next cluster to be formed, and so on. It is this interplay between the particle filter and DBSCAN that causes the bad performance at the start of the simulation. Once the correct amount of targets has been estimated, this no longer poses a problem.

As shown in Table 4-1, DBSCAN outperforms k-means/silhouette for all values of c, and is therefore the better choice under these circumstances.

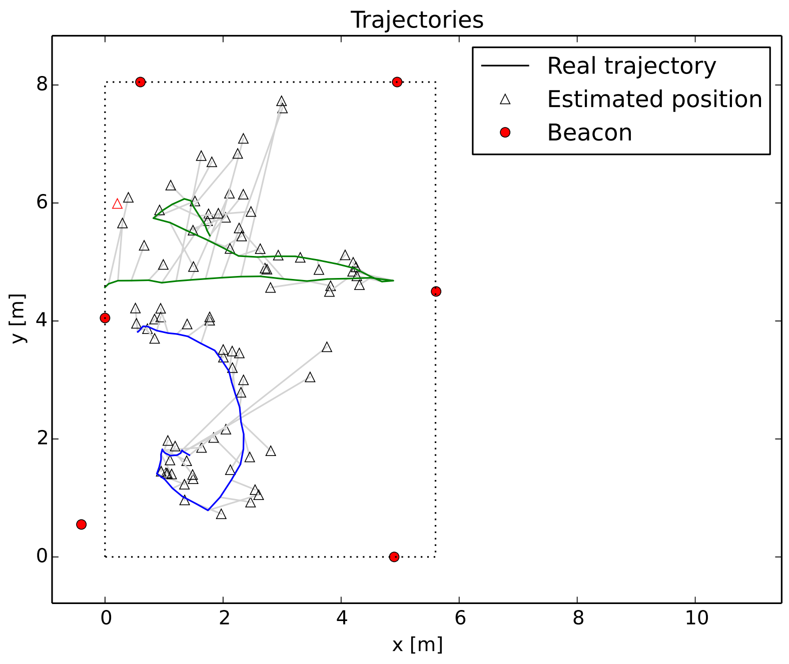


(a) Best result (OMAT error $0.06\,\mathrm{m}$).

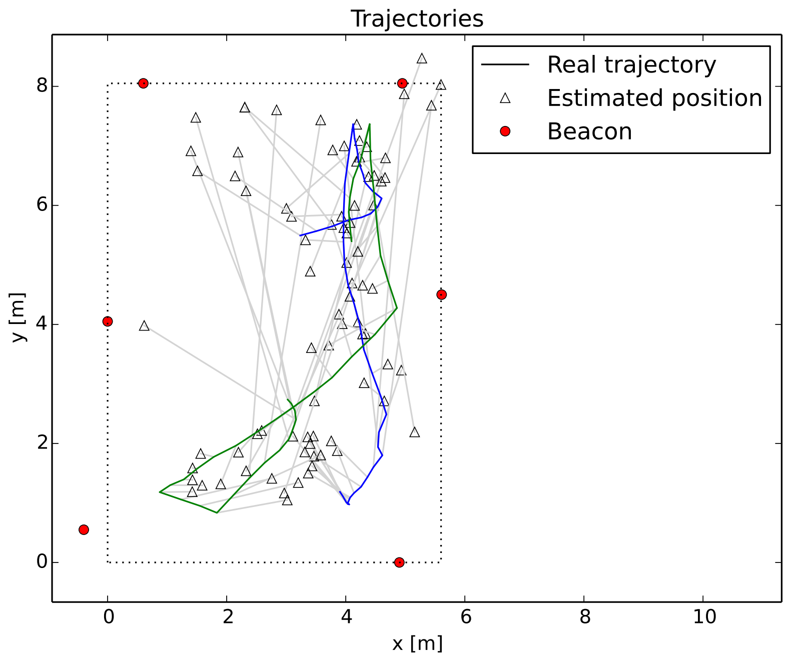


(b) Worst result (OMAT error $0.09\,\mathrm{m}$).

Figure 4-6: Tracking results in the $8\,\mathrm{m} \times 8\,\mathrm{m}$ room with a fixed number of targets.

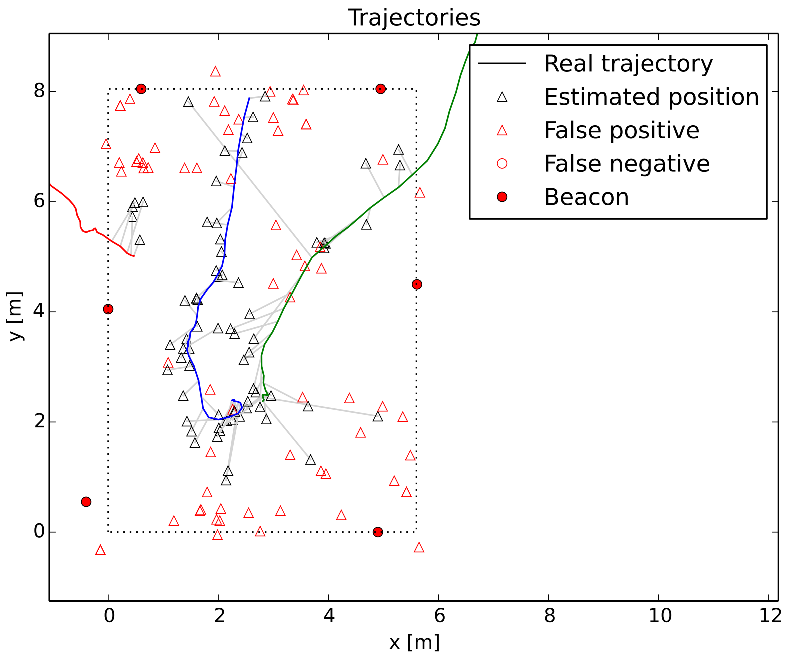


(a) Best result (OMAT error $0.65\,\mathrm{m}$).

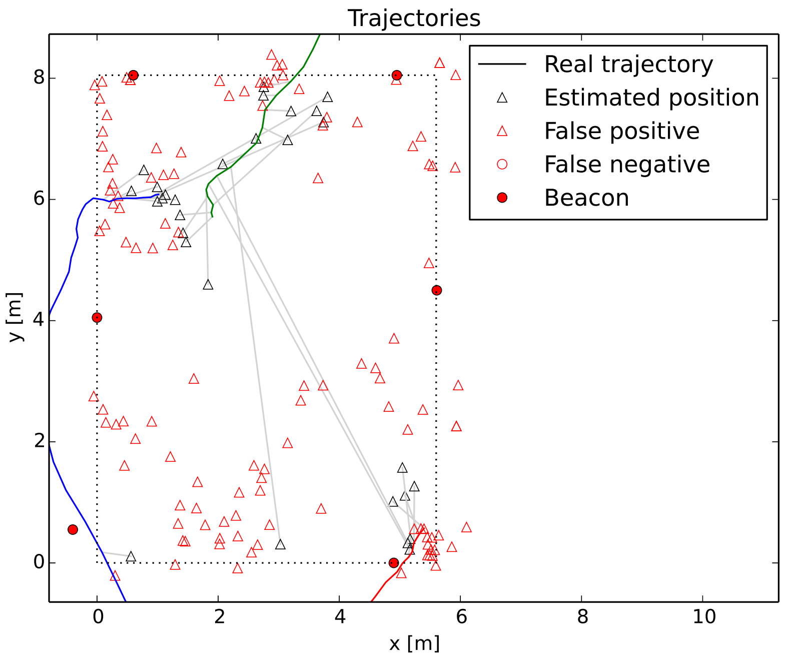


(b) Worst result (OMAT error $1.60\,\mathrm{m}$).

Figure 4-7: Tracking results in the simulation environment based on the physical test setup with a fixed number of targets.

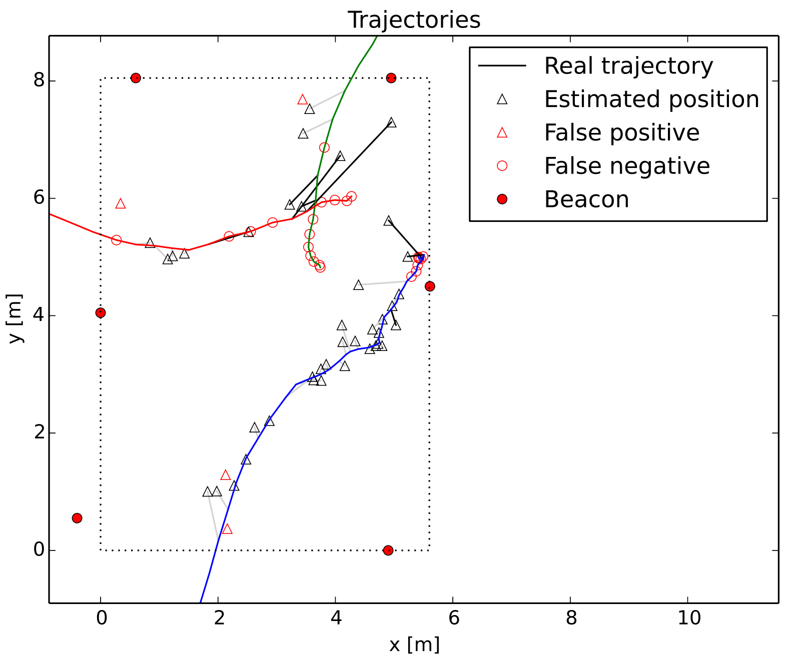


(a) Best result (OSPA error (c=$2.5\,\mathrm{m}$) $1.78\,\mathrm{m}$).

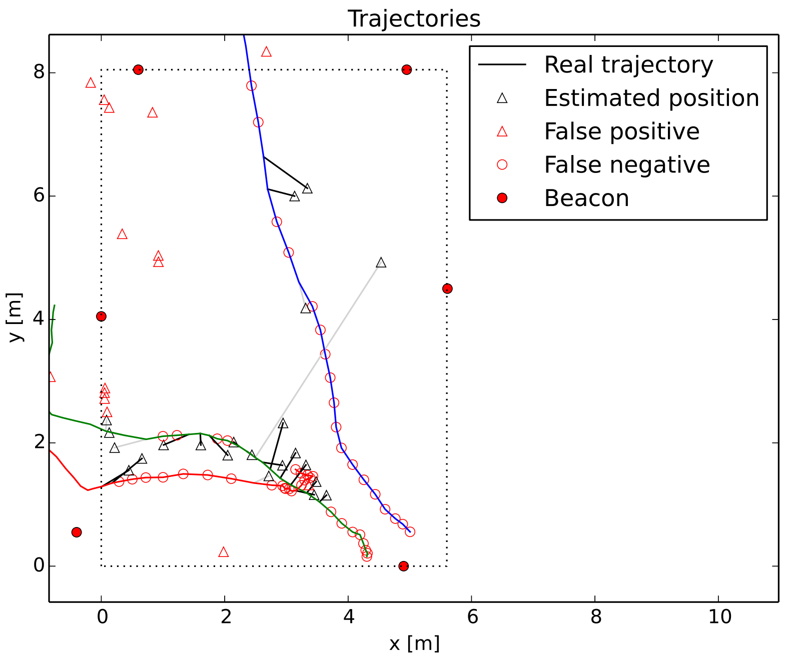


(b) Worst result (OSPA error (c=$2.5\,\mathrm{m}$) $2.29\,\mathrm{m}$).

Figure 4-8: Tracking results in the simulation environment based on the physical test setup with a variable number of targets (0-3) using k-means/silhouette clustering.

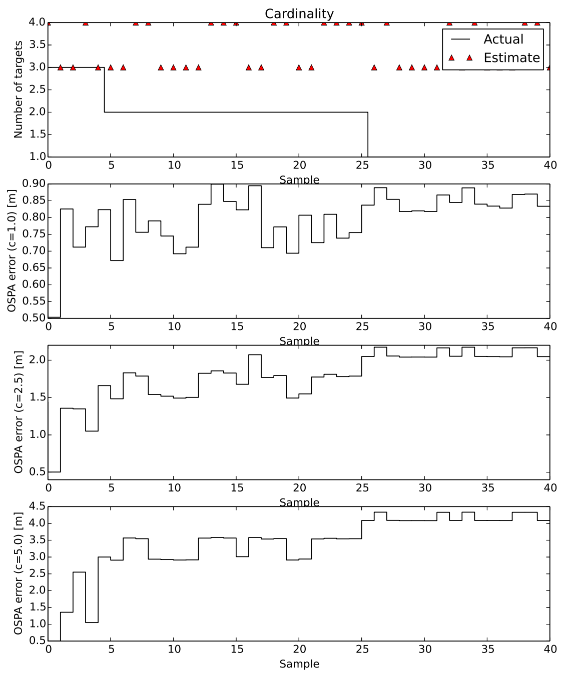


(a) Best result (OSPA error (c=$2.5\,\text{m}$) $0.93\,\text{m}$).

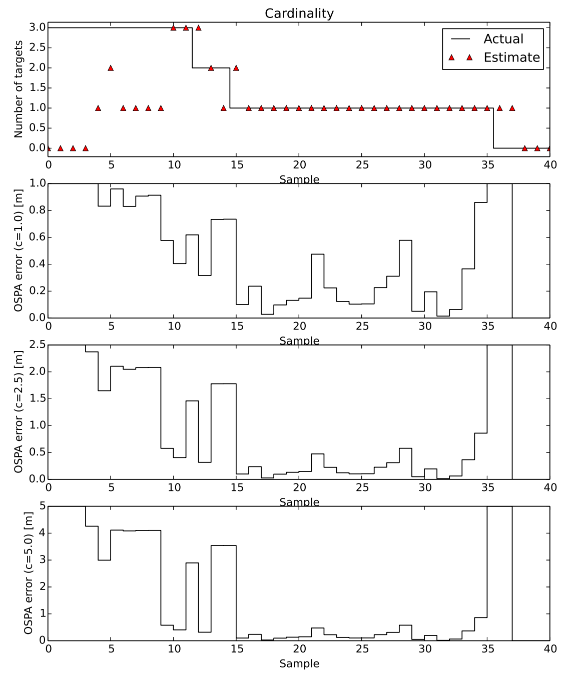


(b) Worst result (OSPA error (c=$2.5\,\text{m}$) $2.06\,\text{m}$).

Figure 4-9: Tracking results in the simulation environment based on the physical test setup with a variable number of targets (0-3) using DBSCAN clustering.



(a) k-means/silhouette



(b) DBSCAN

Figure 4-10: Cardinality estimates and OSPA error metric for the best variable target simulations.

4-4 Performance evaluation: test setup

4-4-1 Method

To test the performance of the tracking filter in a real environment, a physical test setup was built. Six beacons are placed in the corners and on the long sides of an $8\,\mathrm{m} \times 6\,\mathrm{m}$ room. A table surrounded by several chairs is located in the center of the room (Figure 4-11).

Nordic PCA10000 and PCA10001 development boards with Bluenet[2] firmware are used as beacons. The beacons advertise on a single channel, and approximately 4 advertisements per second were observed for each beacon. The RSS information measured by these beacons is logged with a timestamp and is later replayed in the JavaScript test application using the same tracking filter as in the simulation. The measurement area is filmed using a webcam to provide an estimate of the ground truth.

The test starts by measuring the RSS when the room is empty for a period of at least 10 minutes. The recorded RSS is then used to determine the baseline RSS and the measurement variance σ_z^2.

Once the baseline has been established, a person enters the room. First, the localization of stationary targets will be evaluated, so the target remains stationary for several minutes at different locations in the room.

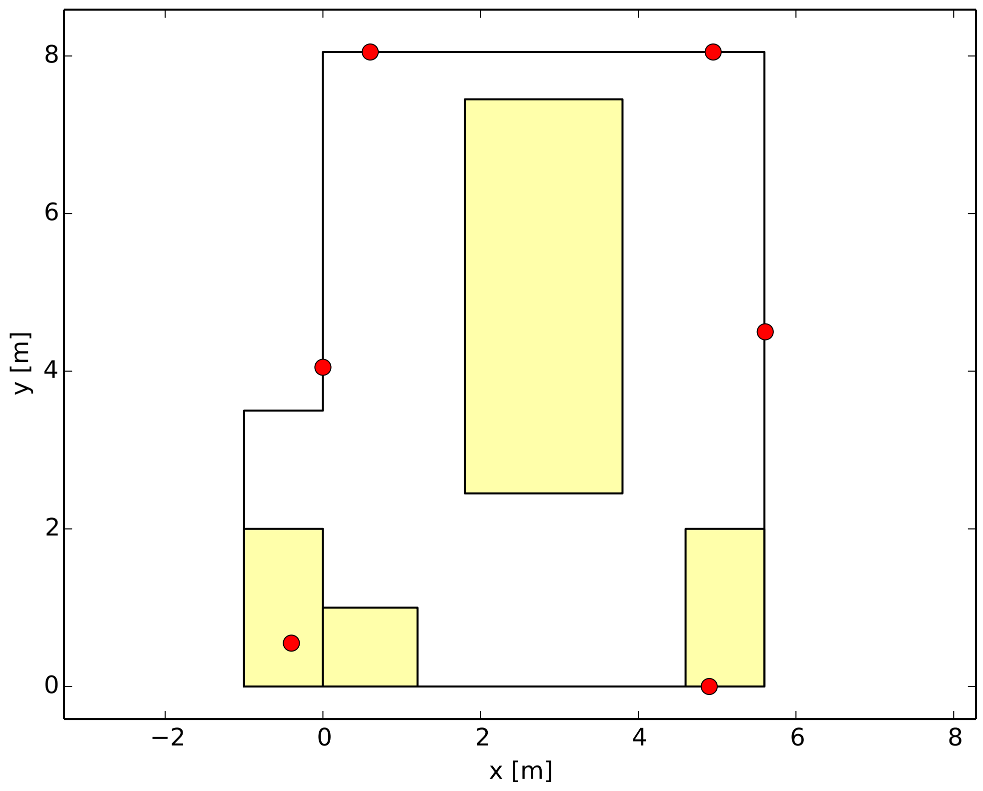
The tracking of moving targets is evaluated next. The target walks around in the room for several minutes. The target's approximate location can be estimated using the webcam. This is not accurate enough to quantitatively evaluate the performance of the tracking filter, but it does provide a rough estimate of the ground truth.

4-4-2 Results

Unfortunately, the results do not come close to the accuracy seen in the simulation. Visual inspection of the results show that the filter is not able to locate the stationary targets, nor the moving targets. The tracking results for one of the stationary target positions are shown in Figure 4-12. The estimated cardinality and OSPA errors are shown in Figure 4-13.

Large changes in RSS were observed even though the target was not near the beacons or the measurement region inbetween. These unmodeled deviations combined with the low coverage of the tracking area are suspected to be the cause of the poor tracking performance.

[2]http://dobots.github.io/bluenet/



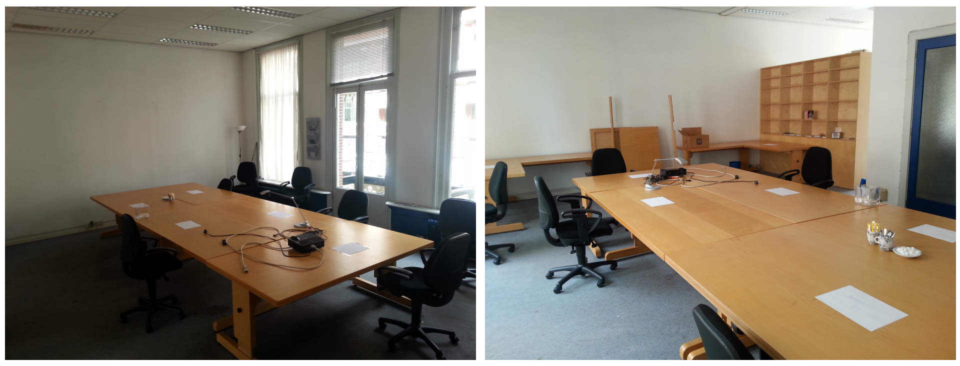


Figure 4-11: Floorplan and overview pictures of the test environment for device-free localization. The floorplan shows beacon locations (red) and tables (yellow).

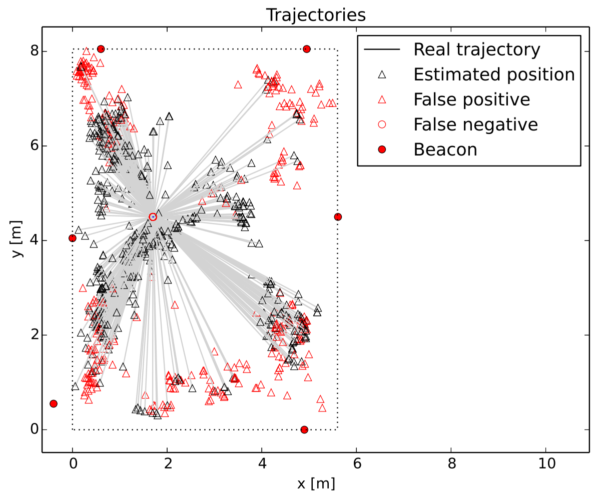


Figure 4-12: Tracking results in the physical test setup with one stationary target.

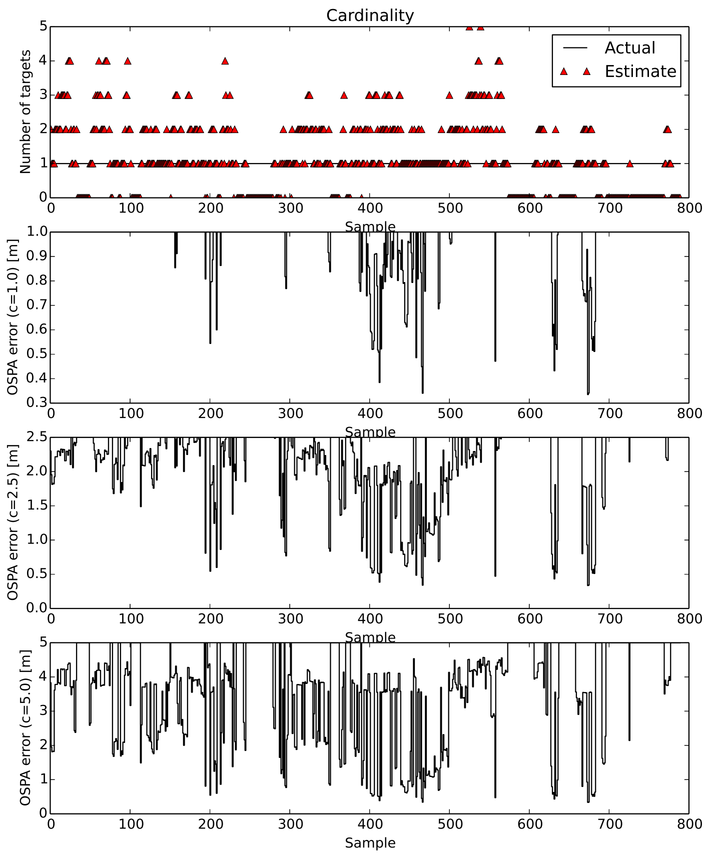


Figure 4-13: Cardinality estimates and OSPA error metric in the physical test setup.

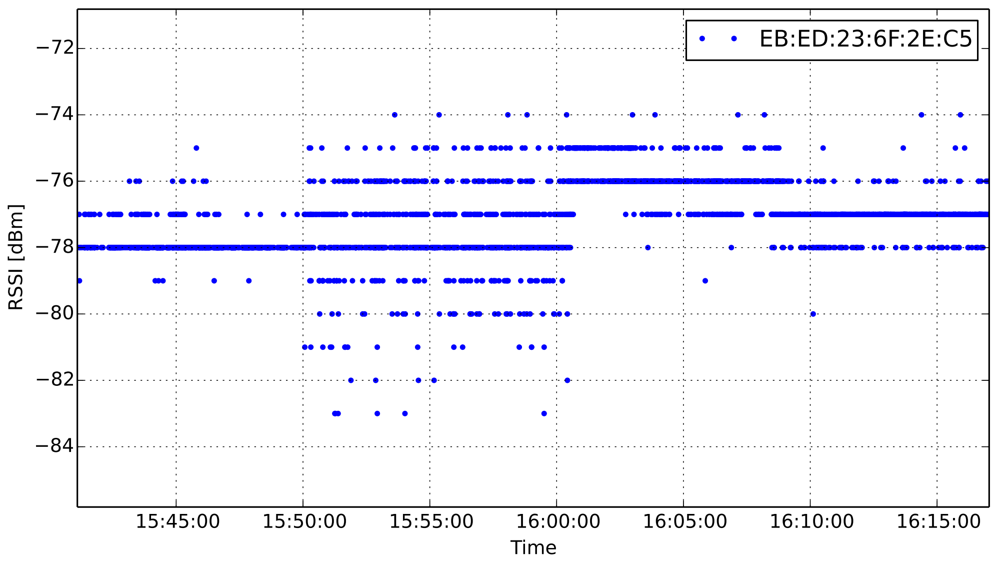


Figure 4-14: Influence of people outside the measurement region. A person enters the room at 15:50 and walks around until 16:00. From 16:00, the person sits still until the room is vacated at 16:10.

4-4-3 Reflections by far-away objects

Inspecting the results from the test setup showed that many outliers were observed. These changes occurred when the target was not near the respective link, and both positive and negative changes in RSS were observed. These disturbances only occur when one or more persons are present. It is suspected that these disturbances are caused by interference between reflections from people, even though they are far away from the beacons.

To verify that this is the case, an additional test was performed. In this test, two beacons were set up in an empty room. The measurement starts with a ten-minute measurement of RSS between the beacons while the room is empty. Then, a person enters the room and walks around for ten minutes. A minimum distance of two meters is kept from the beacons and the link between them. This is followed by ten minutes of sitting still inside the room, in order to see if this effect also occurs with stationary targets. After the person leaves the room, a final ten-minute measurement is performed to see whether the signal returns to its baseline value.

The RSS recorded during this test is shown in Figure 4-14. The result is clear. Walking around in the room significantly increases the variance of the RSS. Sitting still also causes a deviation, in this case constructive interference is observed as the measured RSS is approximately 2 dB higher than the baseline value. When the person leaves the room, the signal does not return to its baseline value. During the test, one chair was moved slightly, this is the most likely cause of the failure to return to the baseline value.

4-5 Conclusion

The reasoning behind device-free localization is sound. In simulation, the method can produce good tracking results even under the presence of noise and incomplete measurements. It has also been shown to work in physical test setups, for instance in [16, 22]. These tests, however, were performed with a larger number of beacons.

Sadly, the results do not seem to hold when a smaller number of beacons is used. The interference by reflections is too large and can not be distinguished from actual obstacles. Including the reflections in the measurement model is not feasible, as these are near-impossible to model if no map of nearby reflective surfaces is available.

Because it was not possible to collect accurate motion traces, no steps have been made towards the interpretation of this data to create a floorplan. Although other tracking methods presented in Chapter 3 could be used to provide motion traces, a review of the functions indoor localization has to perform shows that the exact location may not actually be necessary. The next chapter describes a method to perform room-level localization without an intermediate position estimation step.

Chapter 5

Concept 2: Fingerprinting

In Chapter 3, fingerprinting was proposed as a localization method. This method, however, suffers from a time-intensive calibration phase. The calibration takes a long time because each fingerprint has to be labeled with its exact position.

A review of the functions that depend on localization (e.g. turning on lights) shows that the position of the user does not need to be known exactly. In fact, localization at room-level may be more suitable for practical applications, since it drastically simplifies the detection of users *in a specific room.* When the resolution is reduced to room-level accuracy, the localization can be simplified. For fingerprint-based techniques, this means that samples only have to be labeled with the room they were recorded in, so no exact position has to be determined during calibration, instead they can be collected just by walking around. The expected workload for calibration is therefore drastically reduced. Instead of estimating a position, localization can be reduced to a classification or supervised learning problem. Room-level localization is further described in Section 5-1 to 5-3.

Fingerprints can also be used to construct a 2-/3-dimensional map of the environment. While the fingerprints are elements in a high-dimensional space (with dimension equal to the number of beacons), they originate from a low-dimensional problem and are therefore expected to have a low intrinsic dimensionality. Using dimensionality reduction techniques, these fingerprints can be mapped back to 2-/3-dimensional space, providing a rough estimate of the floorplan (Section 5-4).

In order to respond to the users presence, it is necessary to know the room in which devices are located. While this information can be provided manually, two automated approaches are suggested in Section 5-5.

Knowing where the user will be in a few seconds provides more possibilities for home or office automation. As an example, turning on lights *after* the user has entered a room means that the room was still dark when he/she entered, which is not desirable. Even worse, when the sampling rate of the localization system is low, just using a switch to turn on the lights might be a faster and simpler solution. So instead of turning on lights after the user enters the room, it is better to turn them on *just before* this happens. Several methods of predicting the next room are presented in Section 5-6.

In this chapter, it is assumed that the device performing localization measures the Received Signal Strength (RSS) from beacons in fixed locations. This can also be reversed: using the beacons to measure the RSS of the mobile device. The DoBeacons are able to measure the RSS from other devices and communicate these values.

5-1 Data collection

To evaluate the effectiveness of the localization methods proposed in this chapter, a dataset with room-labeled fingerprints was created. Room-labeled fingerprints were collected in seven rooms spread over three floors. Twenty-two beacons (7-8 per floor) were placed along walls in this environment. Per room, 50-150 labeled samples were collected depending on the size of the room. The fingerprints were collected by walking around inside each room while trying to visit all traversable areas. Fingerprinting the entire area took approximately 15 minutes.

Fingerprints were collected using a laptop with a CSR USB Bluetooth Low Energy (BLE) dongle. The fingerprints were sampled by collecting 100 advertisements and averaging the RSS per beacon. Not all beacons were observed in every sample, beacons that were not observed were assigned an RSS of $-100\,\text{dBm}$.

5-2 Dimensionality reduction

Before any form of classification is applied, the dimensionality of the data is reduced to improve the effectiveness of the classifiers. Three dimensionality reduction techniques will be considered: two unsupervised methods (Principal Component Analysis (PCA) and Isomap [29]) and one supervised method (Linear Discriminant Analysis (LDA)). These methods provide a mapping from signal-space to a lower-dimensional space, whilst trying to maintain the distance between samples. Both methods can transform new samples after the training phase.

PCA tries to find a linear mapping that maximizes the spread between samples by projecting them on an orthogonal base of which the first axis points in the direction of largest variance, where for each following axis the orientation is such that the remaining variance on that axis is maximized.

Isomap finds a nonlinear mapping that tries to maintain the distances between samples along the manifold in which they are located. It approximates the geodesic distance along the manifold by finding the shortest path on a graph generated using k-nearest neighbors.

Since the fingerprints will be used for classification, it may be possible to improve the overall accuracy by taking the labels into account during dimensionality reduction. LDA tries to find a linear projection that maximizes the spread between classes instead of all samples. A larger spread of room clusters may improve classification results. It should be noted that a minimum number of classes (rooms), one more than the number of components, is required for LDA.

In a multilateration problem, at least four distance measurements are required to uniquely find a position in a 3-dimensional space. Since the RSS used for fingerprints largely depends on distance, a similar requirement for the number of measurements is expected, i.e. at least four RSS measurements are required to provide unique fingerprints for 3D locations. The classification accuracy is therefore expected to decrease significantly when the dimensionality is reduced to three or less components. Following the same reasoning, the localization accuracy is expected to become worse when less than four beacons are available for localization.

The accuracy evaluation and optimization of dimensionality reduction parameters are performed in Python using *scikit-learn* [30]. The tuning will be performed in combination with the classifiers presented in the next section.

5-3 Room-level localization

Room-level localization can be treated as a classification/supervised learning problem. Based on the labeled fingerprints recorded during calibration, the label belonging to the currently observed RSS vector should be predicted as accurately as possible.

In this section, classifiers are trained to determine the room in which a fingerprint was recorded. Three classifiers will be evaluated: k-nearest neighbors, radius neighbors and Gaussian Naive Bayes. k-nearest neighbors and radius neighbors are non-generalizing classification methods. These methods can be used on clusters of arbitrary shapes, but may be sensitive to irrelevant features. Gaussian Naive Bayes does generalize the training data and assumes a Gaussian distribution of samples within clusters. Other methods (support vector machines, decision trees, ensemble methods) were considered, but initial testing showed poor performance compared to the methods evaluated here, so these classifiers were left out to reduce the computation time for the final parameter optimization.

Two situations will be considered: one where all rooms are labeled, and one where some fingerprints are taken in previously unseen rooms. In practice, it may not be worthwhile to record fingerprints in every room. In this case, fingerprints taken outside of calibrated areas should be recognized as such to prevent false positives in nearby rooms.

5-3-1 Room classification when all rooms are labeled

Method

To find the best combination of dimensionality reduction techniques and classifiers, the accuracy of all combinations is estimated. For each pair of dimensionality reductors and classifiers, a grid search is performed to optimize the tuning parameters to provide the best possible classification accuracy.

The following dimensionality reduction techniques are taken into account: PCA, LDA, Isomap and raw data. The tuneable parameters are the number of components (0-10) for PCA, LDA and Isomap and the number of neighbors (1-10, 25, 50 or 100) for Isomap.

For the classifiers, the accuracy scores of Gaussian Naive Bayes, k-nearest neighbors and radius-neighbors were examined. The tunable parameters are the number of neighbors (1-10,

	K-Nearest Neighbors	Radius Neighbors	Gaussian Naive Bayes
Raw	97.7	89.8	91.7
LDA	97.2	96.7	95.7
PCA	**99.2**	98.8	97.0
Isomap	98.0	97.0	96.8

Table 5-1: Accuracy of dimensionality reduction techniques and classifiers with optimized parameters. Classification accuracy is shown in percentages.

25, 50 or 100) for k-nearest neighbors, the radius (5-25 dB in steps of 5 dB) for radius-neighbors and the dissimilarity metric (euclidean, manhattan or chebyshev) for k-nearest neighbors and radius-neighbors. The Gaussian Naive Bayes classifier does not have tunable parameters.

The accuracy of all combinations was estimated by evaluating the prediction accuracy using stratified 10-fold cross-validation on the labeled data. Stratified cross-validation was used to guarantee that enough training samples were available for every room.

Results

The classification accuracy results are shown in Table 5-1. For all classifiers, dimensionality reduction using PCA produced better results than LDA or Isomap. Radius neighbors performed worse than k-nearest neighbors with all dimensionality reduction techniques, and Gaussian Naive Bayes performed worse than radius-neighbors except when applied to raw data.

The best classification accuracy (99.2% correctly predicted labels) was found for k-nearest neighbors in combination with PCA. The parameters were 5 components for PCA and 5 neighbors with a euclidean distance metric for k-nearest neighbors.

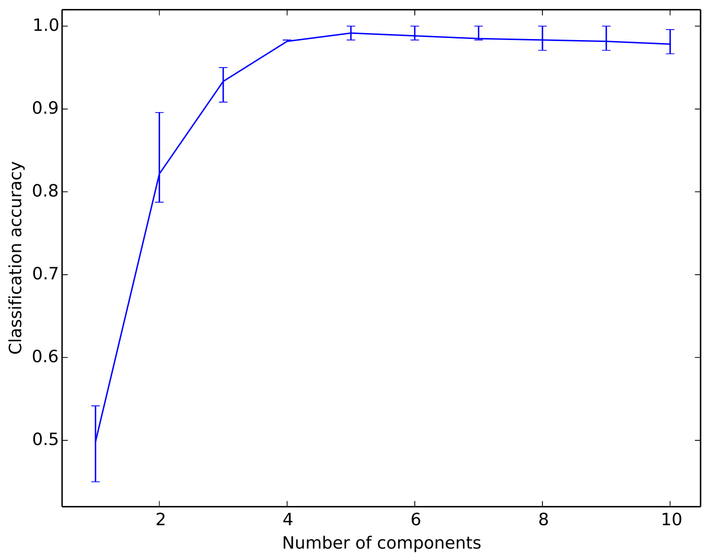
The second-best results were found with PCA and radius neighbors (98.8%). Here, the optimal parameters were 5 components for PCA and a 15 dB radius with a euclidean distance metric for radius neighbors.

In Figure 5-1 the effect of parameter changes relative to the optimal tuning is shown. The accuracy change caused by the number of components for PCA levels out after 4 components. As predicted in Section 5-2, the classification accuracy decreases for smaller values. Larger values do not significantly change the classification accuracy.

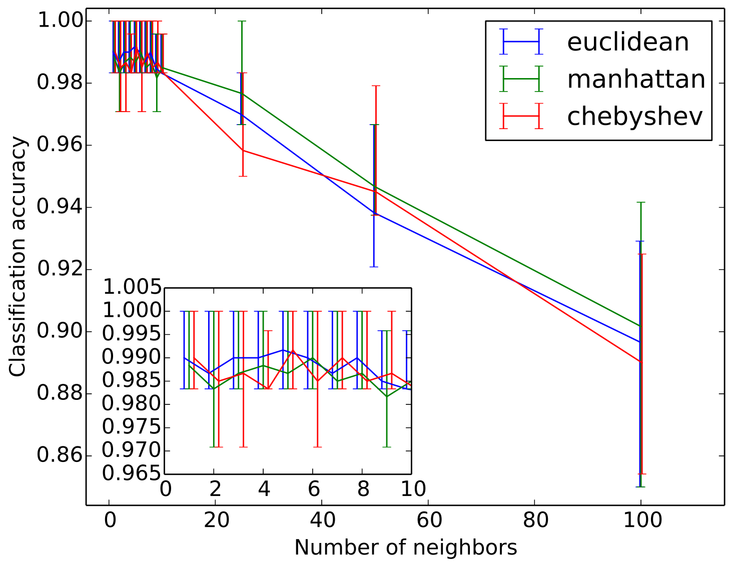
The number of neighbors for k-nearest neighbors has no clear influence on the classification accuracy for values between 1 and 10, but decreases the accuracy if it is increased further. There is no clear difference in performance between the dissimilarity metrics, although the average accuracy for euclidean distances seems slightly higher for 1-10 neighbors.

5-3-2 Room classification when not all rooms are labeled

In real-life applications of indoor localization, it is unlikely that all rooms will be labeled. When visiting an unlabeled room, false positive classification of nearby rooms should be prevented. Therefore, the classifier should be able to recognize that the current fingerprint does not belong to any of the recorded rooms.



(a) Number of components for PCA.



(b) Number of neighbors and distance metric for k-nearest neighbors.

Figure 5-1: Influence of parameter changes relative to the optimum settings on the classification accuracy of PCA followed by k-nearest neighbors. Error bars indicate the 25th and 75th percentile scores.

A large number of classifiers is not able to recognize outliers. It may be possible to use outlier- or novelty-detection techniques such as a one-class Support Vector Machine (SVM) next to these classifiers to estimate whether the current fingerprint is an outlier or not. Radius-neighbors is an exception, as it can be used for classification and outlier detection at the same time. It recognizes outliers when no neighboring points are located in the specified radius.

The tuning of an outlier detector is an optimization problem, since a trade-off has to be made between the number of false positive or false negative results. The tuning depends on the density of samples in each room and therefore it may be difficult find a single optimal value for all applications. When the sampling density is increased, outlier detection will improve because the difference in fingerprint density between labeled and unlabeled areas will be larger.

Method

The dimensionality is first reduced using 5-component PCA. Then, once for each of the seven rooms, the data is randomly separated into training and test datasets, keeping 80% of samples for training. One room at a time is labeled as an outlier, the others are labeled as inliers. The outlier data are then removed from the training datasets and added to the test datasets.

Two outlier detectors are applied to these datasets: one-class SVM and radius neighbors. Again, the parameters are optimized to produce the best classification results. The tunable parameters for one-class SVM are the upper bound of training samples that are considered outliers ν (0.001-0.1), the kernel type (linear, polynomial, Radial Basis Function (RBF)) and the kernel size γ (0.001-1000) for the RBF kernel. For radius neighbors, the radius (5-20 dB) is tuned.

Results

On average, radius neighbors had an outlier classification accuracy of 96.1% with an optimal radius of 10 dB. The accuracy of one-class SVM was 54.5% with an RBF kernel where $\nu = 0.001$, $\gamma = 0.1$.

The influence of the radius on the outlier classification accuracy is shown in Figure 5-2. The best average accuracy is achieved with a radius of 10 dB. If the radius is lowered, the number of samples that are incorrectly labeled as outliers increases. The opposite also holds. If the radius is increased, the number of unrecognized outliers increases.

Discussion

Unlabeled rooms can be reliably detected using radius neighbors. In terms of room classification, k-nearest neighbors had a slightly better accuracy than radius neighbors. These methods can be combined by using k-nearest neighbors for classification and the distance to the nearest neighbors for the detection of outliers.

The selection of the radius depends on the fingerprint density. If the fingerprint density is low, a larger radius is required to prevent false negatives, but this reduces the accuracy of detecting outliers. In this dataset, on average 2 fingerprints were recorded per m^2. The tuning

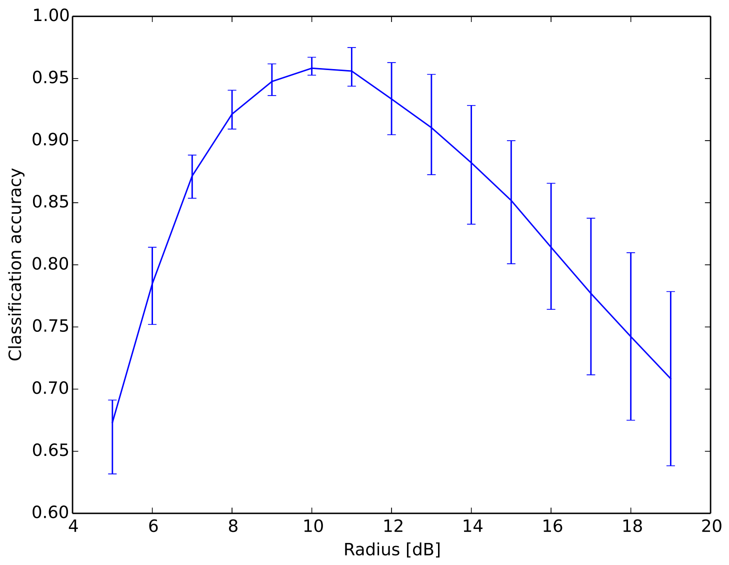


Figure 5-2: Influence of radius on outlier classification accuracy when using radius neighbors after 5-component PCA. Error bars indicate the 25th and 75th percentile scores.

of the radius can be performed automatically using the method described above. For this experiment, evaluating the accuracy for 15 radii took approximately 1 second[1].

5-3-3 Conclusion

The results from this section show that classification of RSS fingerprints can be used successfully for room-level localization. When all rooms are labeled, a localization accuracy of 99.2% is observed. This accuracy is slightly lower for unlabeled rooms (96.1%). Calibration can be performed in several minutes, and only needs the room in which fingerprints are recorded as input.

5-4 Floorplan estimation

The estimation of a building's floorplan can be seen as a special case of dimensionality reduction, since the high-dimensional signal strength vector has to be mapped to a 2- or 3-dimensional space. Unlike the dimensionality reduction for classification, the goal of this reduction is to preserve the distance between samples as well as possible, instead of keeping clusters easily separable.

Two dimensionality reduction methods were found that focus on maintaining the distances or dissimilarities between samples: PCA and Isomap. PCA finds an orthogonal mapping that minimizes the loss in variance of the projected samples. Isomap aims to maintain approximate

[1]Calculations were performed on an HP 8530w with a 2.80 GHz Intel Core2 Duo T9600 CPU.

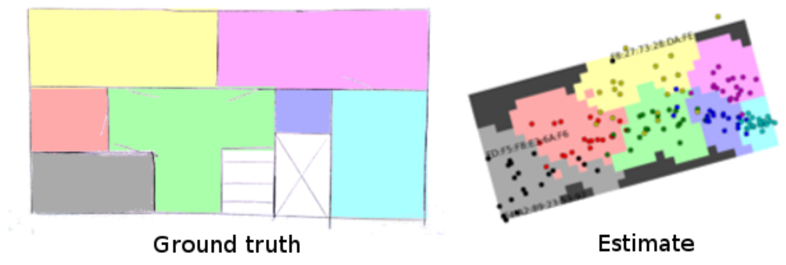


Figure 5-3: Example of floorplan estimation using Isomap. The estimated floorplan shows the correct adjacency of rooms when compared to the ground truth. The colored markers indicate fingerprint locations after transformation using Isomap, colored regions indicate the room area estimated using radius neighbors on the transformed data.

geodesic distances along the estimated manifold, and will therefore estimate larger distances when the number of neighbors k is decreased.

Although the RSS dissimilarities can not be directly converted into a distance, the application of dimensionality reduction techniques on fingerprints can already produce a rough estimate of the shape of the environment, as shown in Figure 5-3. A short test performed on the data collected for this experiment suggests that floorplan estimation using Isomap or PCA also works in 3D.

In [31], a small number of position-labeled fingerprints is used to estimate a mapping between the Isomap results and world coordinates. This mapping warps and scales the transformed fingerprints to match the position labels as well as possible. After this transformation, an average localization error of 2.0 m is reported.

5-5 Localization of devices

For home and office automation, it is not only necessary to know the location of users, but also the locations of the devices that should respond to their presence. It is possible to ask the user to indicate the room in which devices are located, but this task may be automated to some extent. In this section, two methods are proposed to determine the room in which devices are located.

A simple but effective solution is to evaluate all recorded fingerprints and assume that the device is located where its RSS is maximized. This method will work if fingerprints have been collected near all devices, but it is not possible to localize devices that are not located in any of the fingerprinted rooms. Still, this method can be quite effective. In the dataset used here, 20 out of 22 beacons are placed in the correct room using this method, the other two beacons were placed in rooms adjacent to their true location.

A different approach is to use fingerprint-based localization to find these devices similar to the way users are localized. In this chapter localization is performed using fingerprinting on the mobile device, but localization of devices may be performed by measuring the mobile device's RSS at each beacon. The DoBeacons are already able to record and communicate the RSS of nearby devices, but because of time constraints this option was not explored further.

5-6 Next room prediction

As mentioned in the introduction of this chapter, it is not only necessary to know the room users are located in at this moment, but also to predict their location a few seconds ahead. This prediction can then be used, for instance, to turn on lights so the user does not have to enter a dark room.

In Section 5-3, it was shown that the current room can be recognized with more than 99% accuracy. In this section, the next room will be predicted. Different approaches to predicting the next room are examined in Section 5-6-1. In Section 5-6-2, a new classifier is used to predict the next room. The prediction results are shown in Section 5-6-3.

5-6-1 Prediction methods

Proximity

One of the simplest ways to predict the next room is to look at other rooms in proximity of the user. All of the classification techniques considered here perform classification based on some form of score (distance for k-nearest neighbors and radius neighbors, probability for Gaussian Naive Bayes), which makes it easy to find the second-best estimate. In case of distance-based classification, the underlying indexing structures can also be used to find the distance to nearby rooms.

Proximity, however, does not indicate that the user is actually moving towards that room. It may be possible that the user is stationary inside the current room. The number of false positives can be lowered by only predicting room transitions when the user is moving, for instance using the accelerometer. Still, proximity is a very crude approximation as it is not guaranteed that there actually is a passage to the other room at this location.

Adjacency

Instead of estimating the proximity to any point in nearby rooms, estimating the proximity to passages leading to this room reduces the number of false positives.

Passages between rooms can be found by observing the current room estimate. When this estimate changes, the last fingerprint is stored as the location of a passage to the next room. Future transitions between these rooms can than be predicted by observing the proximity to the estimated position of this passage.

However, like the proximity-based prediction, proximity to a passage does not necessarily mean that the user is about to move to that room.

Extrapolation

The previous methods all relied on proximity to other rooms or passages. Instead of relying on proximity, it may be possible to predict the next fingerprints by extrapolating from the last few samples.

An advantage of this method is that it does not require any training, the prediction is only based on the last samples. It is assumed that the change in RSS can be approximated linearly for small movements. Under this assumption, the previous RSS values are extrapolated one or more steps ahead. This predicted fingerprint is than passed through the room classifier to predict the next room.

In practice, this method produces poor results. If a small number of fingerprints is extrapolated, this method is very sensitive to noise. This could be solved by filtering or using a larger set of fingerprints to sample from, but taking into account the low sample rate (approximately 1 Hz), the reaction time becomes too long and the movements can no longer be considered small.

Time-shifted classifier

Extrapolating the last few fingerprints produced poor results, but this does not mean that the history of fingerprints does not contain information about the next room. When the user walks to a different room, he/she will most likely encounter the same fingerprints before entering that room since he will walk along the same path (i.e. towards and through the door). Instead of blindly extrapolating the past fingerprints, a classifier can be trained to predict the next room using the past fingerprints.

Under the assumption that the room classification is correct, a dataset of past fingerprints and room labels can be created on-the-fly. Once enough samples have been collected, a classifier can be trained. Instead of training this classifier using the past fingerprints and the current room estimate, it is trained using a *time-shifted* version of the current room estimate. For instance, given a sequence of fingerprints $x_{k-10}, x_{k-9}, \ldots, x_k$ and corresponding room labels $y_{k-10}, y_{k-9}, \ldots, y_k$, the classifier is trained using the following input-output pairs:

$$(x_{k-10}, y_{k-9}), (x_{k-9}, y_{k-8}), \ldots, (x_{k-1}, y_k) \tag{5-1}$$

The next room can then be predicted by classifying x_k. Instead of just one fingerprints, a history of multiple fingerprints can be used by stacking these to produce new input data:

$$\left(\begin{bmatrix} x_{k-9} \\ x_{k-10} \end{bmatrix}, y_{k-8}\right), \left(\begin{bmatrix} x_{k-8} \\ x_{k-9} \end{bmatrix}, y_{k-7}\right), \ldots, \left(\begin{bmatrix} x_{k-1} \\ x_{k-2} \end{bmatrix}, y_k\right) \tag{5-2}$$

5-6-2 Method

A new dataset was recorded for next-room prediction. This dataset consists of unlabeled fingerprints collected while walking around in the same environment in which the room-level localization dataset was recorded. The fingerprints are stored in order, so that the history of past fingerprints can be used to improve the prediction accuracy if required. The current-room classifier is used to provide a ground truth room label. To improve the accuracy of the ground truth, single-sample outliers were removed by replacing these with the surrounding room labels.

To evaluate the performance of the next-room predictor, its output is compared to the ideal output: a time-shifted version of the current room. The current room is determined using the room classifier described earlier in this chapter. Single-sample outliers were removed by replacing these with the preceding room label to further increase the accuracy of the desired output.

The prediction accuracy is evaluated for predictions 1 to 5 timesteps into the future. Classification is performed using k-nearest neighbors following dimensionality reduction using PCA. The number of components (1-10) for PCA and the number of neighbors (1-20) for k-nearest neighbors are optimized. The dimensionality reduction is applied after the past fingerprints have been stacked. Furthermore, the number of past fingerprints used for the prediction (1-5) is optimized. The prediction accuracy for each combination of parameters is evaluated using stratified 10-fold cross-validation on the stacked and time-shifted datasets.

5-6-3 Results

The prediction accuracy 1 to 5 samples ahead in the future is shown in Figure 5-4. An average prediction accuracy of 89% is achieved for predictions 1 sample ahead, this accuracy decreases if the number of predicted samples increases.

Figure 5-5 shows the influence of changes in dimensionality reduction or classifier parameters relative to the optimal settings. Figure 5-5a shows that the accuracy levels out around 10 components. This value is higher than for the estimation of the current room, which can be explained by the fact that more than one past fingerprint is used to make this prediction. This is further supported by the observation that the optimal number of components increases when the next room is predicted further into the future. Figure 5-5b shows that the optimal number of past fingerprints used to make the prediction increases when predictions are made further into the future. Finally, the influence of the number of neighbors is shown in Figure 5-5c. In all cases, a low number of neighbors (1-5) seems beneficial for the prediction accuracy.

An example of the output of this predictor is shown in Figure 5-6. This figure shows the current room (grey) and the desired (red) and actual (blue) output of the predictor. In the training sample, the predictor achieves a prediction accuracy of 100%. A high accuracy is expected here because all transitions have already been seen before.

On the test set, the predictor achieves an accuracy of 79%, which is slightly lower than the expected value found in Figure 5-4, but still between the 25th and 75th percentile accuracies.

Two main causes of errors are observed: errors during room transitions and errors while the user stays within the same room. Errors observed while staying in the same room may be prevented by only allowing room transitions when motion is detected, therefore the number of false predictions can likely be reduced in practice. During transitions, most errors seem to occur because the prediction is 1 or 2 samples off. In nearly all cases shown here, the prediction is still at least 1 sample ahead of the current room, and thereby still performs its function of predicting the next room.

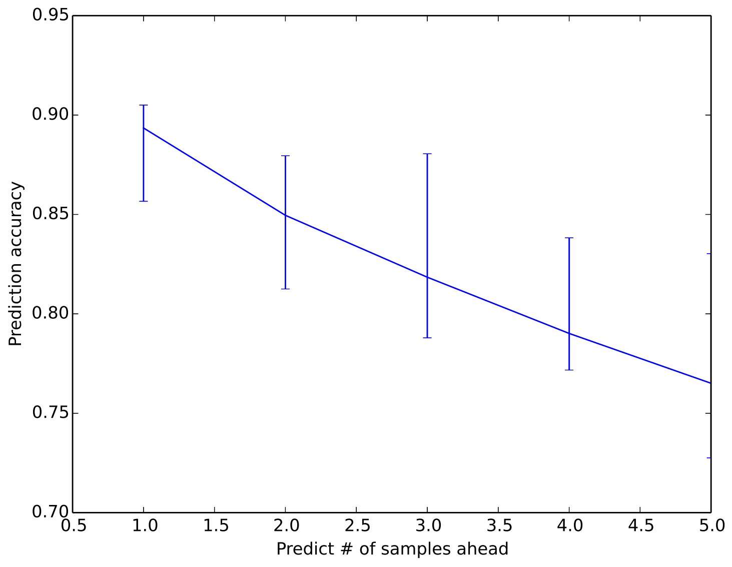
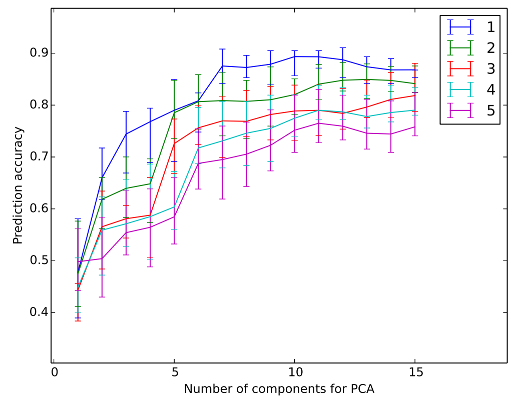
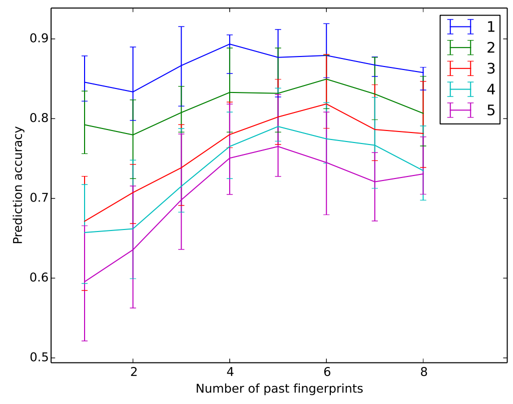


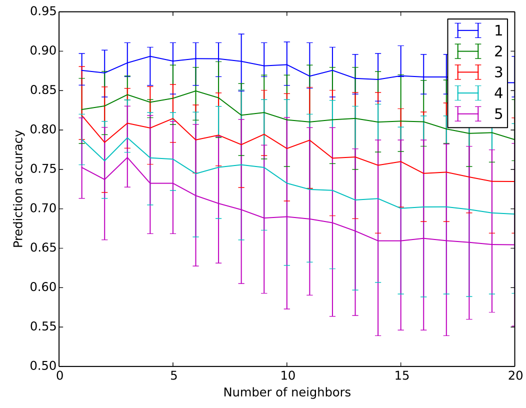
Figure 5-4: Next room prediction accuracy with optimized classifier parameters vs. the number of samples into the future that the next room is predicted.



(a) Number of components for PCA (after time-shifting and stacking of fingerprints).



(b) Number of past fingerprints to stack and use for prediction.



(c) Number of neighbors for classification using k-nearest neighbors.

Figure 5-5: Influence of parameter changes relative to the optimum settings on the prediction accuracy for predictions using PCA and k-nearest neighbors 1-5 samples into the future. Error bars indicate the 25th and 75th percentile scores.

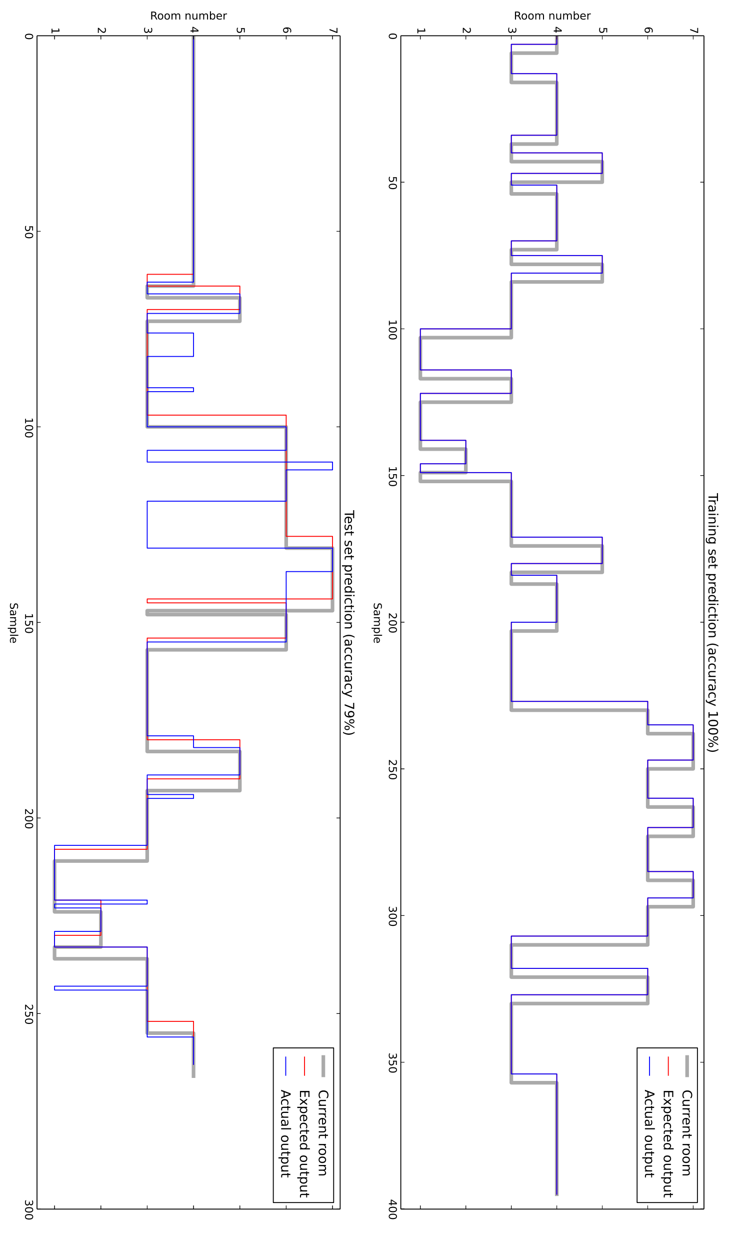


Figure 5-6: Next room predictor output. The next room was predicted three samples ahead using PCA and k-nearest neighbors with optimized parameters. Prediction accuracy is 100% on the training set, prediction accuracy on the test set is 79%.

Chapter 6

Discussion

Two methods of locating users were presented in this report: device-free localization, and fingerprint-based localization. While device-free localization performed adequately in simulation, it did not work in practice. Reflections from far away objects were determined to be the main cause of the poor tracking performance. These reflections can not be practically prevented, nor is it realistic to increase the number of beacons. Therefore, the concept of device-free localization is considered infeasible for this application.

Localization using Received Signal Strength (RSS) fingerprints produced good results. However, the accuracy of this method has only been tested on one dataset. A second dataset was collected during a removal at the test location. This dataset can be used to evaluate the robustness of fingerprint localization when furniture is moved in the environment. Although the data has been recorded and is available for analysis, this has not been performed because of time constraints.

Another direction for further research is to evaluate how well this method can be transferred to other environments. To truly test the effectiveness of this localization method, the experiments should be repeated in other buildings. Variations in the number of beacons and fingerprint densities should also produce more insight into the true effectiveness of using RSS fingerprints for indoor localization.

In order to reduce the need to calibrate each new device, it should be possible to exchange maps between devices. However, it may not be possible to directly transfer the fingerprint map, as the antenna of the other device might produce different RSS values because of a different antenna gain. Further research should be performed to find out whether this is a prohibitive disturbance or not and if fingerprints can be represented in a different way that is not sensitive to these distortions.

Chapter 7

Conclusion

The goal of this research was to find a method to determine the room in which the user is located.

Two concepts were presented. The first concept aimed to estimate a metric map of the environment by tracking the movements of people. Out of several available tracking methods, device-free localization was selected and investigated further. While the performance in simulation was adequate, these results did not carry over to a real-world test setup.

The second concept used Received Signal Strength (RSS) measurements to directly estimate the room label through supervised learning, thereby eliminating the need for position estimation. Using Principal Component Analysis (PCA) and k-nearest neighbors, this method was able to predict the current room label with more than 99% accuracy, making it possible to trigger events based on the presence of people within a specific room. No user interaction is required during localization and the calibration can be performed in a short time, approximately 2 minutes per room. This method complies with all requirements posed in the problem statement in Section 1-2 and can therefore be used in practice to determine whether the user is in a specific room.

Appendix A

Observation model

A short test was performed to verify the accuracy of the magnitude model [16]. Two beacons were placed in an empty room at a distance of 4.5 m. One of the beacons is advertising on a single channel, the other beacon is used to measure the Received Signal Strength (RSS). The RSS is received at integer precision. In most cases, the observed RSS values lie within a single bin (1 dBm wide). When outliers were present the median of RSS values (over 20-30 samples) is listed here.

First, a test subject was positioned between the two beacons at varying distances from the receiver. The resulting RSS is shown in Table A-1. The presence of a person does indeed cause a shadowing effect on the RSS, but the magnitude of this effect is not constant. The attenuation seems to be larger when the person is near the beacons than when he/she is roughly halfway between them. It is speculated that this is a result of the nonzero area of the test subject. The exponential model approximates the person as a point, but at close range a relatively large part of the radiated signal is blocked by the test subject, reducing the accuracy of this approximation.

Next, the test subject was moved perpendicular to the beacon-beacon axis. This produced the results shown in Table A-2. The change in RSS does indeed decrease gradually when λ increases. The parameters ϕ and σ_λ of the magnitude model were fit on these results (Figure A-1), resulting in values of 5.26 dB and 0.07 m respectively.

Finally, the assumption that the shadowing effect is additive is tested. The RSS is measured with a varying number of people between the beacons. The results are shown in Table A-3, which suggests that the shadowing effect is indeed additive.

The estimated parameters are slightly different than those found in [16] ($\phi = 4$ dB, $\sigma_\lambda = 0.2$ m). The results reported here serve to demonstrate the effects predicted by the exponential model, but are expected to be less accurate due to the low resolution of the reported RSS.

Dist. from receiver [m]	RSS [dBm]	|ΔRSS| [dB]
No obstacle	-66	0
0.5	-72	6
1.0	-77	11
2.0	-75	9
3.0	-71	5
4.0	-83	17

Table A-1: RSS with a person between beacons at various distances along the beacon-beacon axis.

		Left		Right	
Off-axis [m]	λ [m]	RSS [dBm]	|ΔRSS| [dB]	RSS [dBm]	|ΔRSS| [dB]
0.00	0.00	-71	5	-71	5
0.10	0.00	-71	5	-72	6
0.20	0.02	-70	4	-70	4
0.40	0.07	-64	2	-64	2
0.60	0.16	-67	1	-66	0
0.80	0.28	-66	0	-66	0

Table A-2: RSS with a person between the beacons, moved perpendicular to the beacon-beacon axis at 2.25m distance.

# people	RSS [dBm]	|ΔRSS| [dB]
0	-66	0
1	-72	6
2	-81	15

Table A-3: RSS with multiple people between beacons.

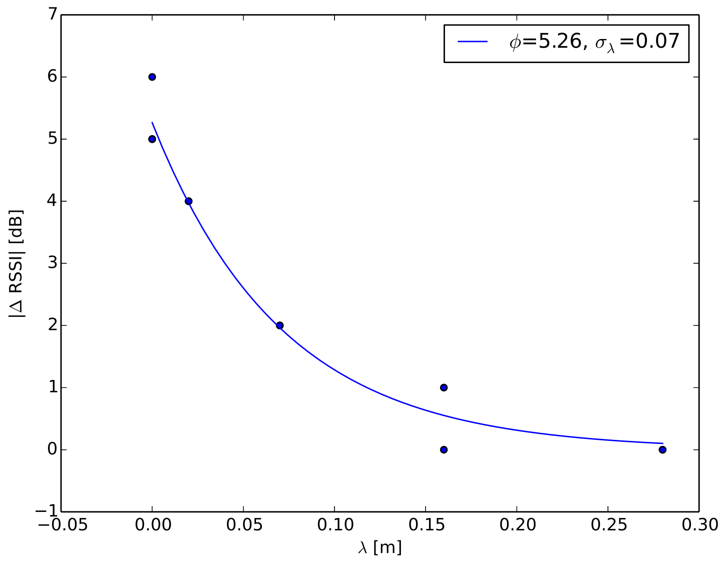


Figure A-1: Magnitude model parameter estimation.

Bibliography

[1] J. Haverinen and A. Kemppainen, "Global indoor self-localization based on the ambient magnetic field," *Robotics and Autonomous Systems*, vol. 57, no. 10, pp. 1028–1035, 2009.

[2] S. Y. Seidel and T. S. Rappaport, "914 mhz path loss prediction models for indoor wireless communications in multifloored buildings," *Antennas and Propagation, IEEE Transactions on*, vol. 40, no. 2, pp. 207–217, 1992.

[3] J. C. Stein, "Indoor radio wlan performance part ii: Range performance in a dense office environment," *Intersil Corporation*, vol. 2401, 1998.

[4] T. Pattnayak and G. Thanikachalam, "Antenna design and rf layout guidelines." `http://www.cypress.com/file/136236/download`. Accessed: 2016-04-27.

[5] M. Alzantot and M. Youssef, "Crowdinside: automatic construction of indoor floorplans," in *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, pp. 99–108, ACM, 2012.

[6] P. Mazumdar, V. J. Ribeiro, and S. Tewari, "Generating indoor maps by crowdsourcing positioning data from smartphones," in *Indoor Positioning and Indoor Navigation (IPIN), 2014 International Conference on*, pp. 322–331, IEEE, 2014.

[7] Y. Jiang, Y. Xiang, X. Pan, K. Li, Q. Lv, R. P. Dick, L. Shang, and M. Hannigan, "Hallway based automatic indoor floorplan construction using room fingerprints," in *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pp. 315–324, ACM, 2013.

[8] R. Faragher and R. Harle, "An analysis of the accuracy of bluetooth low energy for indoor positioning applications," in *Proceedings of the 27th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+'14)*, 2014.

[9] V. Honkavirta, T. Perälä, S. Ali-Löytty, and R. Piché, "A comparative survey of wlan location fingerprinting methods," in *Positioning, Navigation and Communication, 2009. WPNC 2009. 6th Workshop on*, pp. 243–251, IEEE, 2009.

[10] P. Bahl and V. N. Padmanabhan, “Radar: An in-building rf-based user location and tracking system,” in *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 2, pp. 775–784, Ieee, 2000.

[11] M. Youssef and A. Agrawala, “The horus wlan location determination system,” in *Proceedings of the 3rd international conference on Mobile systems, applications, and services*, pp. 205–218, ACM, 2005.

[12] R. Faragher, C. Sarno, and M. Newman, “Opportunistic radio slam for indoor navigation using smartphone sensors,” in *Position Location and Navigation Symposium (PLANS), 2012 IEEE/ION*, pp. 120–128, IEEE, 2012.

[13] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury, “No need to war-drive: unsupervised indoor localization,” in *Proceedings of the 10th international conference on Mobile systems, applications, and services*, pp. 197–210, ACM, 2012.

[14] M. Youssef, M. Mah, and A. Agrawala, “Challenges: device-free passive localization for wireless environments,” in *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pp. 222–229, ACM, 2007.

[15] D. Zhang, J. Ma, Q. Chen, and L. M. Ni, “An rf-based system for tracking transceiver-free objects,” in *Pervasive Computing and Communications, 2007. PerCom’07. Fifth Annual IEEE International Conference on*, pp. 135–144, IEEE, 2007.

[16] S. Nannuru, Y. Li, Y. Zeng, M. Coates, and B. Yang, “Radio-frequency tomography for passive indoor multitarget tracking,” *Mobile Computing, IEEE Transactions on*, vol. 12, no. 12, pp. 2322–2333, 2013.

[17] X. Chen, A. Edelstein, Y. Li, M. Coates, M. Rabbat, and A. Men, “Sequential monte carlo for simultaneous passive device-free tracking and sensor localization using received signal strength measurements,” in *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on*, pp. 342–353, IEEE, 2011.

[18] J. Wang, Q. Gao, Y. Yu, P. Cheng, L. Wu, and H. Wang, “Robust device-free wireless localization based on differential rss measurements,” *Industrial Electronics, IEEE Transactions on*, vol. 60, no. 12, pp. 5943–5952, 2013.

[19] F. Thouin, S. Nannuru, and M. Coates, “Multi-target tracking for measurement models with additive contributions,” in *Information Fusion (FUSION), 2011 Proceedings of the 14th International Conference on*, pp. 1–8, IEEE, 2011.

[20] M. F. Bugallo, T. Lu, and P. M. Djuric, “Target tracking by multiple particle filtering,” in *Aerospace Conference, 2007 IEEE*, pp. 1–7, IEEE, 2007.

[21] J. Wang, Q. Gao, P. Cheng, Y. Yu, K. Xin, and H. Wang, “Lightweight robust device-free localization in wireless networks,” *Industrial Electronics, IEEE Transactions on*, vol. 61, no. 10, pp. 5681–5689, 2014.

[22] S. Nannuru, *Multitarget multisensor tracking.* PhD thesis, McGill University, 2015.

[23] R. Mahler, “Multitarget moments and their application to multitarget tracking,” tech. rep., DTIC Document, 2001.

[24] S. Nannuru, Y. Li, M. Coates, and B. Yang, “Multi-target device-free tracking using radio frequency tomography,” in *Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2011 Seventh International Conference on*, pp. 508–513, IEEE, 2011.

[25] P. J. Rousseeuw, “Silhouettes: a graphical aid to the interpretation and validation of cluster analysis,” *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

[26] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise.,” in *Kdd*, vol. 96, pp. 226–231, 1996.

[27] J. R. Hoffman and R. P. Mahler, “Multitarget miss distance via optimal assignment,” *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 34, no. 3, pp. 327–336, 2004.

[28] D. Schuhmacher, B.-T. Vo, and B.-N. Vo, “A consistent metric for performance evaluation of multi-object filters,” *Signal Processing, IEEE Transactions on*, vol. 56, no. 8, pp. 3447–3457, 2008.

[29] J. B. Tenenbaum, V. De Silva, and J. C. Langford, “A global geometric framework for nonlinear dimensionality reduction,” *science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[31] T. Pulkkinen, T. Roos, and P. Myllymäki, “Semi-supervised learning for wlan positioning,” in *Artificial Neural Networks and Machine Learning–ICANN 2011*, pp. 355–362, Springer, 2011.

Glossary

List of Acronyms

ALM	Additive Likelihood Moment
BGA	Bayesian Grid Array
BLE	Bluetooth Low Energy
CPHD	Cardinalized Probability Hypothesis Density
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
LDA	Linear Discriminant Analysis
MCMC	Markov Chain Monte Carlo
MPF	Multiple Particle Filter
OMAT	Optimal Mass Transfer
OSPA	Optimal Subpattern Assignment
PCA	Principal Component Analysis
PHD	Probability Hypothesis Density
RBF	Radial Basis Function
RSS	Received Signal Strength
SIR	Sequential Importance Resampling
SVM	Support Vector Machine