Submitted in fulfillment of the requirements for the degree of
Master of Science in Computer Science:
Computer Networks and Distributed Systems

Experimental Validation of Location-Based Vertical Handover Algorithms in IEEE 802.11ah Networks

Tobia De Koninck

Adviser: Prof. Dr. Jeroen Famaey

Supervisor: Dr. Filip Lemic and Serena Santi

IDLab
INTERNET & DATA LAB
University of Antwerp

Faculty of Science
2020

Contents

Acknowledgements

I would like to thank my adviser Prof. Dr. Jeroen Famaey for giving me the opportunity to perform this research. Likewise, I wish to express my appreciation to Serena Santi and Dr. Filip Lemic for conducting previous research on the topic of this thesis. In addition I want to thank them for providing extensive feedback on this manuscript. The assistance by the members of IDLab for obtaining and debugging the hardware setup was greatly appreciated. Finally, I want to thank my parents, brothers and friends for their ongoing support and distraction during the making of this thesis.

Samenvatting

Het Internet of Things (IoT) is een wetenschappelijk, beloftevol paradigma waarnaar meer en meer onderzoek wordt uitgevoerd. Het IoT bestaat uit kleine en energie-beperkte apparaten die verbonden zijn met het Internet. Deze worden gebruikt in verschillende toepassingen, gaande van de gezondheidszorg en industrie tot milieubescherming en -controle. Bijna al deze toepassingen vereisen dat de toestellen op een kleine en lichte batterij werken. Bovendien moeten deze idealiter werken in een groot gebied en ondersteuning bieden voor een relatief hoge netwerksnelheid, tegelijkertijd moeten deze zo energie-efficiënt als mogelijk zijn. Low Power Wide Area Network (LPWAN) is een veelbelovende technologie om IoT apparaten te verbinden met het Internet, omdat deze een heel hoge netwerksnelheid bieden gecombineerd met een laag energieverbruik. Hoewel er verschillende LPWANs technologieën bestaan voor het verbinden van dergelijke toestellen met het Internet, moeten deze allemaal een afweging maken tussen netwerksnelheid en bereik. Daarom is er in de literatuur recent voorgesteld om verschillende LPWAN technologieën te combineren in één enkel toestel. Op deze manier is het mogelijk om de verschillende voordelen van elke technologie te combineren. Een dergelijk toestel kan, bijvoorbeeld, profiteren van het grote bereik van de ene technologie en de respectievelijke hogere snelheid van de andere beschikbare technologie. Deze tweede technologie is dan slechts af en toe beschikbaar. Door het introduceren van verschillende LPWAN in één enkel toestel komt echter de energie-efficiëntie in het gedrang. Om dit probleem op te lossen, moet een toestel enkel proberen te verbinden met een bepaalde LPWAN als

er een relatief hoge kans is dat een dergelijke connectie succesvol zou zijn. Deze thesis focust op het systeem dat nodig is om dergelijke intelligente *handover* beslissingen te nemen met behulp van de locatie-informatie van het multi-Radio Access Technology (RAT) IoT toestel. We maken gebruik van de locatie-informatie van het toestel omdat er talrijke IoT toepassingen zijn die deze locatie-informatie vereisen. Daarom is er geen nadelige invloed op de kostprijs om deze informatie te gebruiken voor het optimaliseren van de handovers. In deze thesis focussen we op IEEE 802.11ah, een nieuwe technologie waarvan de haalbaarheid van locatie gebaseerde handover momenteel nog helemaal niet duidelijk is. Specifiek voeren we een zeer uitgebreide performantie-evaluatie uit van een locatie gebaseerde handover mechanisme voor IEEE 802.11ah. Daarnaast evalueren we de performantie van een veelvoorkomende IoT applicatie die bovenop het handover mechanisme is geïnstalleerd. Onze resultaten tonen de haalbaarheid van locatie gebaseerde handover in IEEE 802.11ah aan, als ook de superieure performantie in vergelijking met de traditionele handover systemen gebaseerd op het luisteren naar beacons of op basis van een Radio Environmental Map (REM). Ten slotte, demonstreren we de haalbaarheid van een dergelijk systeem op een fysieke hardware prototype.

Summary

The Internet of Things (IoT) is a highly promising scientific paradigm that attracts an ever increasing amount of research efforts. The IoT consists of often small and energy-constrained devices connected to the Internet. These are used in various types of use-cases, ranging from healthcare and industry, to environmental protection and monitoring. Nearly all these use cases require that the device is powered by a small and light battery. Furthermore, they should ideally work in a large area and support relatively high data rates, while at the same time being as energy efficient as possible. Low Power Wide Area Network (LPWAN) is a promising candidate for connecting IoT devices to the Internet, as it offers very high range, while simultaneously featuring low energy consumption. Although there exist various LPWANs technologies for connecting such a device to the Internet, all of them have to make a trade-off between data rate and range. Therefore, it has recently been proposed to combine multiple LPWAN technologies into a single device. In this way, the device can benefit from the advantages of each technology. For instance, such a device can benefit from the long range of one technology, as well as from the respectively higher data rate of the other, intermittently available technology. However, by introducing multiple LPWAN technologies into a single device, the energy efficiency is compromised. To mitigate this issue, the device should only attempt to connect to a certain LPWAN when there is a relatively high change of such a connection being successful. This thesis focuses on the system required for making such intelligent handover decisions based on the location information of the multi-Radio Access Tech-

nology (RAT) IoT device. We utilize the location information of the device, as there are numerous IoT use-cases that require this information, so its usage for optimizing the handover comes with effectively no additional costs. Moreover, we focus on IEEE 802.11ah, a novel technology for which the feasibility of location-based handover is currently all but clear. Specifically, we carry out an extensive experimental performance evaluation of a location-based handover mechanism in IEEE 802.11ah. In addition, we evaluate the performance of a typical IoT application deployed on top of this handover mechanism. Our results demonstrate the feasibility of location-based handover in IEEE 802.11ah, as well as its superior performance compared to the traditional handovers based on beacon listening and Radio Environmental Map (REM). Finally, we demonstrate the feasibility of using such a system on a physical hardware prototype.

List of Abbreviations

AP Access Point

CoAP Constrained Application Protocol

CRC Cyclic Redundancy Check

DTLS Datagram Transport Layer Security

ERP Effective Radiated Power

GI Guard Interval

GNSS Global Navigation Satellite System

GPS Global Positioning System

GPX GPS Exchange Format

HTTP Hypertext Transfer Protocol

IoT Internet of Things

IPv4 Internet Protocol version 4

IPv6 Internet Protocol version 6

LPWAN Low Power Wide Area Network

LTE Long Term Evolution

M2M Machine to Machine

MAC Medium Access Control

MAE Mean Absolute Error

MCS Modulation and Coding Scheme

NB-IoT Narrowband Internet of Things

NSS Number of Spatial Streams

RAT Radio Access Technology

RAW Restricted Access Window

REM Radio Environmental Map

RPI Raspberry Pi

RSSI Received Signal Strength Indication

SNR Signal to Noise Ratio

UDP User Datagram Protocol

List of Tables

List of Figures

Chapter 1

Introduction

Lately, more and more devices are being connected to the Internet to form, what is called, the Internet of Things (IoT) [1]. These devices are used in various ways, ranging from simple home gadgets to devices used for tackling complex challenges our society is facing. The IoT application domains include, healthcare, logistics, retail, environment protection and monitoring, etc [1]. Typically, these types of use cases are using sensor networks that consist of small, battery powered devices that regularly send their sensor data to a central server. A major challenge is to make these devices as power efficient as possible [1]. Ideally, a sensor device should have connectivity in a large area, have a high data rate and still use little energy. Hence, using a Low Power Wide Area Network (LPWAN) for providing the connectivity is advised [2]. Recently, a novel LPWAN technology, known as IEEE 802.11ah [3], has been standardised. This new standard builds upon the omnipresent WiFi technology, making it possible to re-use parts of this already established technology [4] and is, therefore, easier to implement compared to other LPWANs. In contrast to the more traditional WiFi standards, IEEE 802.11ah operates in a lower frequency (i.e. 868 MHz), therefore it achieves a high range of 1 km [5]. Furthermore, a single Access Point (AP) supports up to

8000 stations, which makes the technology ideal for dense sensor networks [6] and Machine to Machine (M2M) communication. The technology supports high data rates, up to 346 Mbps, which is an unique property for LPWAN [6]. Just like regular WiFi, IEEE 802.11ah operates in a license free band. Therefore, it is very easy for organisations to setup an IEEE 802.11ah network, without depending on external operators. On top of these many advantages, the technology is highly energy efficient [7]. However, due to the novelty of IEEE 802.11ah it is unclear how this technology works in practice. For example, until very recently it was unclear what is the achievable range of IEEE 802.11ah in practice, which was answered during my internship project [8]. *As an additional example, it is all but clear how IEEE 802.11ah operates in a device supporting multiple Radio Access Technologys (RATs).* In this thesis, we make one important step in answering that question.

When comparing LPWAN technologies, it becomes clear that there is no LPWAN technology which combines all desired properties [9]. In other words, there is no single technology supporting very long range, high data rates and features high energy efficiency. Every technology has some trade-off between these properties. Sigfox, is a LPWAN technology that can achieve a range of 40 km but only support sending a fixed amount of message per day [10]. On the other hand, IEEE 802.11ah makes a different trade-off, preferring a higher data rate over the long range. Therefore, the idea rises to create devices which support multiple RATs [9]. These devices benefit from the advantages of all the RATs they support. Furthermore, they can choose which data they send over which network, which is very valuable when one of the networks is somehow constrained. The drawback of these devices is that due to their multiple radios they would consume more energy if they would always enable both radios. Therefore, a smarter system has to be devised, which only enables a certain radio if necessary. This system would switch between the supported RATs when another technology is more suited at that time [9]. Such a switch is called a *vertical handover*. This requires the device to discover the availability of a network. The currently default way for a device to discover an available network is to continuously listen for beacons. Beacons are short messages sent by the network to inform its availability. However, listening for such beacons implies that the radio is turned on and

hence is consuming power, while the original goal was to turn off the radio as much as possible. Thus, some other discovery mechanism has to be found, that does not require the radio to be turned on. Such a system could work with the help of the current location of the device. Fortunately, many IoT use cases require the usage of a device's location, implying that they include some technology to determine their location [11]. A straightforward technique for this, is to create an Radio Environmental Map (REM) which consists of a mapping of physical locations to the availability of a network. Of course, creating such a map for each deployment would increase the costs of such a network. Furthermore, a lot of effort is required to create an REM. In addition, an REM gets stale, yielding the need for the REMs to periodically be recreated. Obviously, this is practically unfeasible, especially for networks covering large areas [12]. Precisely for theses reasons, the system should be able to somehow estimate whether a network is available solely based on the current location of the device. Existing research shows promising results for this approach [11, 13]. However, *it is currently all but clear whether location-based discovery and handover would work for IEEE 802.11ah, primarily due to its novelty.* Moreover, the performance of both the algorithm, network and applications has not been studied for devices using IEEE 802.11ah. This gap is filled by this thesis. Specifically, we validate the feasibility and performance of location-based handover between the IEEE 802.11ah and IEEE 802.11 b/g/n technologies. We have chosen IEEE 802.11b/g/n because it is an ubiquitous technology and because it is a less constrained network compared to IEEE 802.11ah, allowing the device to offload data-intensive tasks over the IEEE 802.11b/g/n network. Furthermore, because both technologies share parts of the network stack, they intuitively integrate very well.

We show that, location-based handover for IEEE 802.11ah is indeed feasible, first by emulating its performance, as well as experimentally confirming this with a prototype. Moreover, we show that location-based handover significantly outperforms more traditional vertical handover systems based on listening for beacons and using an REM.

1.1 Structure

The thesis is structured as follows. Starting with Chapter 2 we introduce the background concepts, possible use cases and related work. Thereafter, we introduce the considered vertical handover algorithms in Chapter 3. Chapter 4 explains the methodology, such as the hardware and software setup used in the evaluation of the considered algorithms. In Chapter 5 we present and discuss the results of the performed experiments. The thesis is concluded in Chapter 6.

Chapter 2

Background

This chapter starts with explaining the motivation for our research. It also provides more background information on the underlying concepts and technologies this thesis makes use of. The chapter is concluded by an overview of the related work.

2.1 Internet of Things

The IoT can be defined as embedding computing devices, which are connected to the Internet, into everyday devices. Wireless sensor networks consist of geographically scattered sensor devices employed to monitor or record physical parameters of the environment.

A whole range of IoT use cases is based on the principle of Track&Trace systems [14, 15, 1]. In such systems a sensor device is attached to a valuable asset. The sensor devices contain some technology to determine its location (e.g., using a Global Navigation Satellite System (GNSS)). This location is regularly sent by the device to a central server. Such devices are often employed to prevent theft of the asset (e.g., bicycles and vehicles). They are also used, for instance, by taxi and transportation companies and police forces to know the actual location of their vehicles to optimally dispatch their resources.

Many use cases of sensor networks have one thing in common: they require a *wireless* connection to the Internet (or at least some network). Furthermore, in a majority of such IoT use cases, the sensor devices are battery powered, since they have to operate at locations where there is no access to the electricity grid. Besides, such sensor networks typically consists of a vast amount of devices, therefore requiring a fixed connection to the electricity grid is unfeasible. Due to the usage of batteries, the energy usage and efficiency of the sensors device is very constrained. Moreover, these devices are often required to be small and light, resulting in an even lower battery capacity. Recharging these batteries is not an option, as in most use cases the devices are difficult to reach and should operate fully autonomous. Due to these constraints on power usage, both hardware and software should be as energy efficient as possible, including the network connection. Therefore, it is obvious that the technology providing the wireless connectivity should also be power efficient. For example, whether the wireless radio is turned on or not greatly influences the power consumption of the device [16]. Although some Track&Trace systems are powered by the large battery of the tracked vehicle, there are circumstances where the tracking device must powered by a small battery. A modern example are devices to track pets (typically cats and dogs). The whole device (including the battery) should be as light and small as possible because it is worn by an animal. However, the goal of the device is to retrieve the animal when it is lost, therefore the device should ideally be working for an extensive period of time. Again, improving the efficiency of the device results in longer battery life or smaller batteries.

Another example of Track&Trace based systems can be found in the industrial sector. This sector often uses expensive machines which require regular maintenance. However, the frequency of this maintenance depends on the amount and intensity of the usage of the machines. Furthermore, it may also depend on the location where the machine is used. A machine used outdoor may require more frequent maintenance than a machine which is used mostly indoors. To achieve this goal, the equipment contains a small, battery powered tracking device. This devices regularly transmits its location and the wear of the machine. Just like in the other example use cases, such a device requires power efficiency in order to only have a small battery.

A final example application of IoT can be found in the (professional) sports sector. In order to follow a strict training schedule it is important for the trainers to know some parameters of each training. For instance, a cyclist wants to know their power output, heart rate and speed. These measurements are collected by a device and sent to the trainer so that the trainer can adjust the training. As with the other examples, power usage is important in this use case. A cyclist wants to carry as few weight as possible, requiring the device to have a small and light battery.

All these examples outline the challenges of battery powered IoT devices and their required power efficiency. Achieving this power efficiency is an important goal of this thesis.

2.2 Low Power Wide Area Network

A LPWAN is a wireless network in which (sensor) devices can communicate over long ranges at a low bit rate in combination with long battery life. Common examples of LPWAN wireless technologies are IEEE 802.11ah, IEEE 802.15.4g, Narrowband Internet of Things (NB-IoT), LoRa, Sigfox, etc [17, 2]. These technologies widely differ on the bandwidth, range, data rate, payload size and transmission power, although their goal is to have a long range and to be energy efficient.

Sigfox Sigfox is deployed by an operator using proprietary (i.e. using patented technology) base stations. In Europe it uses the license free 868 Mhz band. Sigfox achieves among the best range (10-40 km [10]) and power efficiency at the expense of low throughput (100 bps in UL). When operating in bidirectional mode, a device can only receive a message (DL) after transmitting a message (UP). To comply with duty cycles the system allows for 140 messages in the uplink channel and four messages in the downlink channel. [10, 9, 18].

LoRa Similar to Sigfox, LoRa is designed to achieve a long range (5-20 km) with the trade-off of low throughput [9]. It operates at the same frequency as Sigfox (868 Mhz). Due to the use of a proprietary chip spread

spectrum modulation technique it has high interference resilience. Using a bandwidth of either 250 kHz or 125 kHz it achieves a maximal data rate of 50 kbps [10]. Various classes of devices are distinguished, supporting various levels of bidirectional communication.

NB-IoT Different from Sigfox and Lora, NB-IoT operates in the same, licensed frequency bands as Long Term Evolution (LTE). While achieving a higher throughput (200 kbs) its range is shorter (1-10 km). Furthermore, its immunity to interference is lower compared to Sigfox and LoRa. The same hold for its power efficiency.

IEEE 802.15.4g This technology is the base for many other technologies such as ZigBee. Using a bandwidth between 7.8 kHz and 500 kHz it achieves a data rate between 40 kbps and 800 kbps [9]. It can operate in the 868 Mhz band similar to Sigfox and Lora. The maximum achievable range is 5 km.

IEEE 802.11ah Also known as WiFi-Halow, IEEE 802.11ah, also operates in the 868 MHz. Compared to the other technologies, it is possible to communicate at high data rates (more than 300 Mbps). The trade-off is that it supports a relatively short range of 1 km. In practice this range is even shorter [8, 19]

The aforementioned description of the available technologies explains the major differences between them. Where one technology excels at its supported range (e.g., Sigfox), another technology stands out because of its data rate (e.g., IEEE 802.11ah). It should be clear that creating one technology that excels both at range, throughput and power efficiency would be very difficult. In practice, improving one property, automatically degrades another property. However, combining multiple technologies into a single device results in a device that can profit from the advantages and characteristics of each supported technology.

Besides range and data rate, some technologies are limited by governmental regulations. For example, some bands require that any device using that band obeys to a duty cycle. In Europe, devices using the 868 Mhz frequency band of (e.g, Sigfox, Lora, IEEE 802.11ah) must comply with a duty cycle of

2.8 % (given they obey to some extra rules). Therefore, devices taking advantage of using multiple RATs can choose to offload data to an unconstrained technology.

From time to time, many IoT devices require an update of their firmware in order to improve security or to fix bugs. Therefore these devices should regularly check for new updates and download them once they are available. However, the exact moment at which such an update-check should happen is not very important. Therefore both the update-check and download of the firmware could be performed when the device is connected to an unconstrained network.

In addition to sending the current location of a device, many Track&Trace use cases also gather other data. For example, a delivery service may want to keep track of how long each delivery takes. Or maybe how long the driver has to drive between two addresses and the amount of fuel that takes. While this information is very valuable to the delivery company, it does not need to know that information in realtime. Therefore, this data can be collected by the device and once the driver is back at the warehouse, the device can transfer the full report at once. Such a transfer is ideal to offload to an unconstrained technology.

Similarly, in the example of the professional cyclist, there are some measurements which the trainer wants to know in realtime. These measurements have to be transmitted in realtime, using a technology with a long range. However, the extensive report of the full training has to be known only at the end of the training and it can be transferred using a short-range and unconstrained network.

2.2.1 IEEE 802.11ah

The remainder of this thesis uses IEEE 802.11ah as LPWAN technology, therefore it is discussed in more detail in this section. The novel IEEE 802.11ah, for which work started as early 2012 [4], also known as WiFi Halow, has the unique advantage of that it is built upon the omnipresent WiFi technology. Therefore, the know-how, technology, research, software, etc.

for WiFi can be re-used in WiFi HaLow devices. Where IEEE 802.11b/g/n uses a frequency of 2.4 GHz or 5 GHz, IEEE 802.11ah uses the sub-GHz license-exempt bands. The maximal bandwidth used by IEEE 802.11ah is 16 MHz, this is less than the bands for IEEE 802.11n (and IEEE 802.11ac), which are $\geq$ 20 Mhz.

Besides the frequency, some changes had to be made in comparison with the standard WiFi such that IEEE 802.11ah can be used as a LPWAN technology. It is also more suited for M2M communication [20]. The Medium Access Control (MAC) layer of the IEEE 802.11ah protocol has been adapted to support more stations, to reduce the power usage and finally, to reduce the overhead of many short packets. A IEEE 802.11b/g/n AP has a theoretical limitation of 2007 associated stations [6]. Since in IEEE 802.11ah the goal is to have huge amounts of stations, this limit is changed to allow more than 8000 stations. In order to reduce the overhead of many short packets (which are typical for a sensor network) the standard has incorporated changes to [5, 6]:

Frame headers: IEEE 802.11ah uses a backward incompatible shortened header format for some specific frame types.

Beacons: instead of one type of beacon, IEEE 802.11ah defines both a short and a full beacon. The short beacon contains less information and is sent more frequently than the full beacon.

Grouping stations: stations are grouped into pages, blocks and subblocks. Stations with similar characteristics (e.g., device type, location . . .) are placed in the same group. This allows for various novel mechanisms for reducing the overhead. The grouping of stations (or more precise group sectorization) also helps solving the hidden node problem, which is a major issue in IEEE 802.11ah networks.

Restricted Access Window (RAW): the collision probability is decreased by limiting the subset of stations accessing the channel and at the same time spreading their access attempts over time. This is especially important in networks with thousands of stations. RAW also improves

the power efficiency: there are less transmissions, moreover, the stations can sleep when they are not allowed to transmit. [7]

Association and authentication: when an IEEE 802.11ah network starts operating (e.g., after a power failure of the AP), all the stations will try to connect. Obviously, in networks with thousand of stations this becomes problematic, since the AP is unable to handle thousands of association requests. IEEE 802.11ah solves this with a novel authentication mechanism that limits contention.

In order to achieve a high power efficiency, the IEEE 802.11ah contains extensive power management features. These mechanisms are typically based on the idea of stations alternating between an awake state and a doze state. In the doze state, the device turns of its radio and stops communicating, in order to safe energy.

Two final important properties of IEEE 802.11ah are the available Modulation and Coding Scheme (MCS) and the corresponding achievable link speed. There are 11 MCSs defined in IEEE 802.11ah of which MCS 10 is a special case aiming to achieve a longer range. Every MCS can be used with a channel width of 1, 2, 4, 8 or 16 MHz. Furthermore up to 3 spatial streams can be used. Table 2.1 lists a subset of the possible combinations and their achievable data rate. The maximal data rate (at the physical layer) achievable is 346.66 Mbps when the Number of Spatial Streams (NSS) is four, the Guard Interval (GI) is 4 μs, the data channel is 16 MHz and MCS 9 is used [3].

2.3 Vertical Handover

The previous sections uncovers the problem where lots of IoT use cases require at the one hand high coverage and high data rates and at the other hand a low energy consumption. However, at present these requirements cannot be fulfilled by a single technology. Therefore the need for devices supporting multiple RAT rises.

Because not every technology is available at every physical location, it is necessary to switch between these technologies. Such a switch between the

MCS Index	Modulation	Coding rate	Data rate (kbps)	
			1 Mhz	2 Mhz
0	BPSK	1/2	300	650
1	QPSK	1/2	600	1300
2	QPSK	3/4	900	1950
3	16-QAM	1/2	1200	2600
4	16-QAM	3/4	1800	3900
5	64-QAM	2/3	2400	5200
6	64-QAM	3/4	2700	5850
7	64-QAM	5/6	3000	6500
8	256-QAM	3/4	3600	7800
9	256-QAM	5/6	4000	N/A
10	256-QAM	1/2[1]	150	N/A

Table 2.1: *IEEE 802.11ah MCSs for 1, 2 MHz, NSS = 1, GI = 8 us (source: [21])*

available technologies is called a *vertical handover*, in contrast to handovers in, for instance, cellular networks where a handover is typically performed between two neighbouring cells. Before a station initiates a handover it has to find out whether the desired technology is available. There are two common methods for detecting whether a technology is available. In the first case, a station listens for beacons sent by the AP. A station using the second method actively broadcasts probes. The station detects the availability of technology if it receives a probe response. Both methods have a great drawback: they require that the radio is turned on the whole time and it must actively transmit and receive probes or beacons [22]. The active probe mechanism also has the disadvantage that it is actively using the frequency bands and therefore also reduces the capacity of the band, which could be restricted by a duty cycle [11].

Many IoT use cases require that the position of the station is known. In practice different methods are used for determining this position [13]. First of all a GNSS, (such as Global Positioning System (GPS) or Galileo) can be used. This provides a theoretical accuracy ranging from 5 m (GPS) to 1 m

(Galileo). An issue with these GNSS is that they consume additional power. Currently, research is performed on localization using only the LPWAN technologies [13]. Since the location of the device is available in many use cases, it can be used in the handover algorithm [11].

Determining whether to perform an handover to a certain technology using the device's current location does not require any connectivity for that technology. Instead the radio can be turned off until the algorithm indicates the connectivity would be good enough. Using this mechanism results in less power usage, making the device more power efficient. We used IEEE 802.11ah as the long-range technology combined with IEEE 802.11b/g/n for short-range technology. As explained earlier using IEEE 802.11ah involves obeying the duty cycle, fortunately this is not the case for IEEE 802.11b/g/n. Since the range of IEEE 802.11ah is much larger than the one of IEEE 802.11 b/g/n it is likely that the radio of the IEEE 802.11b/g/n technology is turned of for longer periods (or even for most of the time). Moreover, when the device is moved outside the range of both IEEE 802.11ah and IEEE 802.11 b/g/n, both radios can be turned off. The proposed approach instructs to turn the radio on once the device is again in the range of one of the technologies.

Because IEEE 802.11ah is a novel technology, it is still unclear whether the location-based discovery and handover is feasible for this technology, which is the gap we are aiming to fill in this work.

2.4 Related Work

In the context of 5G, the authors in [23] explain the importance of using location-aware communications. For example, they argue that the location information can aid in the reduction of the energy consumption of a device. Although they argue about the importance, they do not provide concrete methods to achieve this. In contrast, we implemented such a location-based discovery and handover mechanism and evaluate its performance. Furthermore, we focus on the novel IEEE 802.11ah combined with a second technology.

In [24] the authors propose a different kind of handover algorithm: it does not require the location of the device, instead it relies on a combination of a state machine and counters in order to more efficiently poll for networks. However, as earlier argued, a major part of the IoT use cases require the usage of the location of the device. Therefore, our approach can be used in the major part of IoT setups. Furthermore, they focus on Sigfox, Lora and DASH7, whereas our research focuses on the novel IEEE 802.11ah technology. Although they propose an architecture for running an application on a multi-RAT device, they do not evaluate the performance of it. Our evaluation includes the evaluation of the performance of a realistic application.

The authors in [11] provide an approach for location-based discovery of LP-WAN technologies in outdoor environments. In this approach the decision is made based on the estimated or expected Signal to Noise Ratio (SNR). The authors extensively validate the performance of the mechanism, for example, the number of correct, false positive and false negative decisions made using simulations. In contrast to our work focusing on IEEE 802.11ah, their work focuses on LoRa and Sigfox. In addition, they evaluate only system-level parameters such as the number of correct decisions and the number of false decisions, while we go one step further and also evaluate the performance of a typical application.

Related research studies the feasibility of this mechanism when using IEEE 802.11ah [13]. They perform simulations using the ns-3 event based simulation framework [21]. On the other hand, our research examines the feasibility of the system on a real hardware prototype. Moreover, we test the feasibility of running an application on top of the system. Furthermore, using emulation which is based on realistic survey-based data, we perform an extensive amount of experiments. Similarly to [11], [13] focuses on the performance of the mechanism, such as the accuracy of the algorithm, association time and energy consumption. Our research evaluates, in addition to those metrics, also the performance of a very common application. These papers also compare their results with the default discover procedure based on continuously listening for beacons, while we also compare it with a second reference algorithm.

The authors in [25] research the use of the physical location of a device in order to decide whether a Device-to-Device link can be setup. While they use WiFi for Device-to-Device communication, our research assumes another kind of setup, which consists of an AP and a station supporting both IEEE 802.11ah and IEEE 802.11b/g/n. Furthermore, because they use a single technology, they focus on discovery, whereas our research studies the vertical handover between multiple technologies. Similarly to [11] and [13] they use an estimation of the SNR. Again they take the erroneous of the location information into account. Because our research is based on real surveyed data, our location data contains some realistic error caused by the GPS device.

While the efficient discovery of a network is important, other research also indicates the requirements for an efficient heterogeneous network [26]. For example, they present application requirements, such as, throughput, delay and jitter which may influence the decision of what is the best network at a given time. Other preferences include the cost involved when using a specific network or even the current battery level of the device. Similarly, our evaluation focuses on the performance of an application instead of solely examining the performance of the system itself. However, where this research ([26]) provides insight in the requirements for realising an efficient heterogeneous network, they do not propose how to achieve these goals. Our research takes an important step in order achieve these goals by providing a mechanism for efficient discovery and handover, which is an integral part of their proposed architecture.

Our research further builds upon [11, 13]. Instead of relying on purely simulation based validation, we provide evaluation of the system using realistic, experimentally derived measurements. Furthermore, we perform the validation with a realistic application running on top of the whole system. Therefore, we can also evaluate the application's performance instead of only the performance of the algorithm. Finally, we test the feasibility of the handover algorithms and the application on a physical prototype.

Chapter 3

Vertical Handover Algorithms

This chapter starts by explaining the mechanisms which are used to decide between multiple technologies. Next, the vertical handover algorithms considered in this thesis are introduced. In addition to the basic algorithms, some possible optimisations are introduced.

3.1 Deciding Between Multiple Technologies

When a multi-RAT device is used it is likely that at some locations multiple technologies are supported. For example, in our use case, the range of IEEE 802.11ah is larger than the range of IEEE 802.11b/g/n. Since both APs are placed at the same location, the area covered by IEEE 802.11b/g/n is also completely covered by IEEE 802.11ah. Therefore, the need for some procedure deciding which connection to actually use arises.

Each vertical handover algorithm presented in this chapter makes decisions for a single connection type. A device supporting both IEEE 802.11ah and IEEE 802.11b/g/n therefore makes independent decisions on whether to use IEEE 802.11ah or IEEE 802.11b/g/n. These decisions are provided as an *advice* to the decision procedure. This procedure then chooses which connection it actually uses. In other words, the decision procedure is an abstraction over the vertical handover algorithms, in order to break a tie. Specifically, if both

the algorithm for IEEE 802.11ah and IEEE 802.11b/g/n advices to connect, the decision procedure decides to use the IEEE 802.11b/g/n link. The main advantage is that the performance of the vertical handover algorithm can be analysed without the influence of the other link types. For the same reason our decision making mechanism is based on simple priorities. IEEE 802.11 b/g/n has priority 10, while IEEE 802.11ah has priority 5. Since the radio coverage of IEEE 802.11ah is larger, it is logical to give IEEE 802.11b/g/n a higher priority since it otherwise would not be used at all. Besides simple priorities, other conditions and requirements could be taken into account [26]. In order to analyse the performance of the vertical handover algorithm, the priority system has the advantage that this is fully deterministic, i.e. our experiments are not influenced by it.

Algorithm 1 lists the used mechanism in pseudo-code. The algorithm is straightforward: when a link specific algorithm advices to perform an handover, it only effectively performs the handover when the priority of that link is higher than the priority of the current active link. The handover is always performed when there is currently no active link. In case a link specific algorithm advices to disconnect, the decision process performs this directly. The `KeepLink` and `NoHandOver` advices are only used for studying the performance of the algorithm, therefore no action is required by the decision making process.

Algorithm 1: MakeHandoverDecision(Link-Type, Advice)
Makes the final decision to handover between the possible networks based on their respective priority and the advice of the link specific algorithm.

```
Global: Priority-Of-Link = {WiFi = 10, Ah = 5}
Global: Active-Link
1  Priority-Of-Advice = Priority-Of-Link[Link-Type];
2  Priority-Of-Current = Priority-Of-Link[Active-Link];
3  switch Advice do
4      case PerformHandOver do
5          if Priority-Of-Advice ≤ Priority-Of-Advice then
6              Hand over to Link-Type;
7          end
8          break;
9      case Disconnect do
10         Disconnect from Link-Type;
11         break;
12     case KeepLink do
           // No action required
13         break;
14     case NoHandOver do
           // No action required
15         break;
16 end
```

3.2 Handover Using Beacons

This section illustrates one of the reference algorithms which is typically considered the default handover algorithm [13, 3]. The algorithm is listed in Algorithm 2. Beacons are sent with a fixed amount of time between them, for each such interval the algorithm checks whether a beacon was received. If a beacon is received (and the device was not yet associated) the device can start the association procedure. In the other case, when no beacon is received, the device should disconnect.

Note that associating and disconnecting is a relative slow process [27]. In the case of IEEE 802.11b/g/n the device has to first associate itself, followed by the authentication procedure. For IEEE 802.11ah there is an more optimal association and authentication procedure, although it can still take multiple seconds to fully connect [6]. Furthermore, there is a chance that a single beacon is missed while the connection is still working fine. Remember that the frequencies used by IEEE 802.11ah and IEEE 802.11b/g/n are license exempt and that many technologies operate in these bands. Therefore, experiencing an increased amount of packet loss for a short period is expected, especially at the borders of the network's range. This is confirmed for IEEE 802.11ah by the (hardware based) benchmark described in Section 5.2.2. However, for IEEE 802.11b/g/n this was not observed. To prevent this large overhead (i.e. of re-associating just after a disconnect), just because of missing one beacon, the algorithm could be extended to allow up to β missed beacons: Algorithm 3. The effect of this change is studied in Chapter 5.

A major drawback of both the simple and extended algorithm is that the radio is turned on for the whole time. An adjusted algorithm which only periodically checks for beacons could reduce the power efficiency. However, because the availability of a network is less checked, its discovery is delayed compared to the default algorithm.

Algorithm 2: Beacon-based Handover algorithm for single technology.

Input: LinkType

```
1  for Beacon interval elapsed do
2      Wake up to receive beacon for one beacon interval;
3      if Beacon Received then
4          if Already connected to Link then
5              Advice = KeepLink;
6          else
7              Advice = PerformHandOver;
8          end
9      else
10         Missed-Beacons++;
11         if Already connected to Link then
12             Advice = Disconnect;
13         else
14             Advice = NoHandOver;
15         end
16     end
17     MakeHandoverDecision(LinkType, Advice);
18 end
```

Algorithm 3: Extended Beacon-based Handover algorithm for single technology.

```
Input: LinkType
Input: Allowed-Missed-Beacons
Global: Missed-Beacons = 0
1  for Beacon interval elapsed do
2      Wake up to receive beacon for one beacon interval;
3      if Beacon Received then
4          Missed-Beacons = 0;
5          if Already connected to Link then
6              Advice = KeepLink;
7          else
8              Advice = PerformHandOver;
9          end
10     else
11         Missed-Beacons++;
12         if Already connected to Link then
13             if Missed-Beacons ≥ Allowed-Missed-Beacons then
14                 Advice = Disconnect;
15             else
16                 Advice = KeepLink;
17             end
18         else
19             Advice = NoHandOver;
20         end
21     end
22     MakeHandoverDecision(LinkType, Advice);
23 end
```

3.3 Handover Using a REM

The algorithm discussed in this section assumes the existence of an REM which maps a physical location to the number of packet loss observed at that location. Such mapping is typically created using either dedicated measurement campaigns or using crowd-sourcing [8]. Because of the great accuracy of this process, intuitively one would except good results. However, in some cases, such as the following ones, it may not be feasible to perform such measurements [28]:

Very wide-ranging deployments where it is simply too cumbersome to perform measurements in the whole area;

Deployments in private or restricted areas for example industrial plants where only authorized personnel or equipment are allowed;

Future, unknown deployments before deploying a network, a telecom operator typically plans the deployment very carefully. It is, however, infeasible to perform individual measurements for each possible deployment.

The algorithm (Algorithm 4) periodically (e.g., each beacon interval) looks up the observed packet loss at the current location in the REM. If the value is below a certain value, the algorithm advices to associate (if not already connected). Else the algorithms advices to disconnect from the current link. Note that the lookup of the packet loss inside the REM can efficiently be implemented using a k-d tree [29]. The requirement of both the storage and related lookup of values can be a problem for constrained devices. This procedure runs periodically in order to have a good trade-off between fast discovery and load on the devices. If the check is run more often, there will be more load but faster discovery and vice versa.

Algorithm 4: REM-based Handover algorithm for single technology.

Input: Allowed-Packet-Loss
Input: LinkType

```
1  for Beacon interval elapsed do
2      Surveyed-Packet-Loss = GetSurveyedPacektLossForCurrLoc();
3      if Surveyed-Packet-Loss ≤ Allowed-Packet-Loss then
4          if Already connected to Link then
5              Advice = KeepLink;
6          else
7              Advice = PerformHandOver;
8          end
9      else
10         if Already connected to Link then
11             Advice = Disconnect;
12         else
13             Advice = NoHandOver;
14         end
15     end
16     MakeHandoverDecision(LinkType, Advice);
17 end
```

3.4 Handover Using Location

This section explains the final and most important handover algorithm. The algorithm was first presented in [11, 13]. Using a device's location, the algorithm estimates the current SNR, even when the radio is turned off. The SNR, which is the ratio between signal and noise indicates the quality of the signal. Therefore the SNR and distance to the AP of a device are directly correlated. When a device is further away from the AP the path loss increases (i.e. the remaining power of the signal decreases) and thus the SNR decreases. However, when the noise increases the SNR also decreases. Typically, noise increases when there is some interference. The most basic algorithm Algorithm 5 periodically estimates the SNR and compares it to the required SNR, denoted by σ. When the estimated SNR is above the re-

quired SNR (and optionally an extra tunable threshold) the algorithm turns the radio on. Next, it listens for a beacon during one beacon period, only when a beacon is received, the association procedure is started.

Some adjustments and optimisations can be made for the algorithm to work in our multi-RAT setup. First of all, the algorithm should also advice when to disconnect from a network. After the device has turned on its radio, it starts listening for a single beacon. When the algorithm do receive a beacon (either when it is connected or not) it should look at the *actual* SNR related to that beacon. The idea, is to only connect to the network when the actual SNR is higher than the required SNR (i.e. σ), and thus not only look at the estimated SNR. Because the actual SNR also has to be higher than a certain value, the algorithm can be configured to only start the association procedure when the connection is actually feasible. This is especially the case when a beacon is actually received but the packet loss is too high for the application to work, which can happen at the border of the radio coverage-area. Furthermore, looking at the actual SNR makes the algorithm more resilient to faulty estimations. To explain this, assume that the device is located at a distance which is close enough to the AP so that in normal circumstances the network should work fine. If there is an external source of interference at such location, the actual SNR will be lower than the estimated SNR. This is something the estimation cannot really take into account. Even the REM-based algorithm can only provide such information when the interference source was active at time the data was collected. In addition, the algorithm can be adapted in a similar way as in Algorithm 3, by allowing to miss up to β beacons before disconnecting from the network. Finally, the last possible optimisation has a similar goal as the previous one, namely reducing the overhead of re-associating after a (false-positive) disconnect. The provided algorithm does not only disconnect when β beacons are missed, but also when a beacon is received of which the *actual* SNR is below σ. The algorithm can be changed such that it only disconnects when the *actual* SNR is below $\sigma - \Omega$. The Ω parameter behaves like an offset, allowing the algorithm to stay connected even when the SNR is lower than σ. Therefore there is a difference between the minimal required SNR at which the algorithm may perform a handover and the SNR at which

it must disconnect. This allows the SNR to have a short dip, preventing the number of times the algorithm will re-associate directly after disconnecting. The complete algorithm is provided in Algorithm 6. The following section explains how the SNR can be estimated.

Algorithm 5: Estimation-based Handover [13]

```
for Beacon interval elapsed do
    Calculate estimated SNR;
    if Estimated SNR ≥ required SNR + threshold then
        Wake up to receive beacon for one beacon interval;
        if Beacon received then
            Start Association Procedure;
        else
            Sleep;
        end
    else
        Sleep;
    end
end
```

3.4.1 Estimating the Signal to Noise Ratio

In order to estimate the SNR solely based on a device's location (without using a REM), a model of the SNR based on the characteristics of the radio technology and environment is required. Such a model is widely known as a propagation model [30]. In the literature there exists various standardised propagation models. Such as, COST-231 Hata [31], COST-231 Walfisch-Ikegami [31] and ITU-R Below Rooftop [32] models. Furthermore, sometimes new propagation models are created which are more tailored to a specific technology. Examples for IEEE 802.11ah include the Ah Macro, Ah Micro and Ah Indoor models [33]. However, due to the novelty of IEEE 802.11ah, there is currently not much research about suitablity of these propagation models. In [8], we preformed validation of an exhaustive set of propagation

models for IEEE 802.11ah in three different scenarios [1]. Therefore we can employ these suggested models for this thesis. Note that propagation models can also be used in situations where the deployment area is only partially covered by measurements [28]. In such cases the propagation models can be used to interpolate results for the uncovered areas. In Section 4.2.4 the exact used models are discussed.

While a propagation model is used to predict or estimate the path loss in function of the distance between the station and the AP, the algorithm requires an estimation of the SNR. Before converting the path loss (L_b) to the SNR, it has to be converted to the Received Signal Strength Indication (RSSI), this requires the knowledge of the transmission power of the sender (P_{tx}) and the sensitivity of the receiver (S_{rx}). These conversions are provided in Equation (3.1) and Equation (3.2) respectively.

$$RSSI\ [dBm] = P_{tx}\ [dBm] - L_b\ [dB] \tag{3.1}$$

$$SNR\ [dB] = RSSI\ [dBm] - S_{rx}\ [dBm] \tag{3.2}$$

[1] This was performed as a Research Internship, unrelated to this thesis.

Algorithm 6: Extended Estimation-based Handover algorithm for single technology.

```
Input: Required-SNR
Input: Allowed-Missed-Beacons
Input: Offset
Input: LinkType
Global: Missed-Beacons = 0
for Beacon interval elapsed do
    if Not Already connected to Link then
        Estimated-SNR = Calculate estimated SNR;
        if Estimated-SNR ≥ Required-SNR then
            Wake up to receive beacon for one beacon interval;
            if Beacon received then
                Missed-Beacons = 0;
                if Actual-SNR < Required-SNR then
                    Advice = NoHandOver;
                else
                    Advice = PerformHandOver;
                end
            else
                Advice = NoHandOver;
            end
        else
            Advice = NoHandOver;
        end
    else
        Receive beacon during one beacon interval;
        if No Beacon received then
            Missed-Beacons++;
            if Missed-Beacons ≥ Allowed-Missed-Beacons then
                Advice = Disconnect;
            else
                Advice = KeepLink;
            end
        else
            Missed-Beacons = 0;
            if Actual-SNR < (Required-SNR - Offset) then
                Advice = Disconnect;
            else
                Advice = KeepLink;
            end
        end
    end
    MakeHandoverDecision(LinkType, Advice);
end
```

Chapter 4

Measurement Methodology

This chapter starts by discussing the methodology used for the measurement collection. The hardware and software setup is discussed, followed by the used protocols. Thereafter the collected data is presented, which is used for the emulation-based evaluation. The second part of this chapter explains how the emulation setup works. This includes the working of the emulation, protocols and software setup. Furthermore, an overview of the used propagation models is described. Finally, an overview the metrics used in the evaluation is given.

4.1 Measurement Collection

4.1.1 Hardware Setup

The hardware setup used for the measurement collection is shown in Figure 4.1. What follows is a discussion of every component and its relevant specifications.

4 × Open-Mote B This is a IEEE 802.15.4 development board of which we use the sub-GHz radio for providing IEEE 802.11ah access, in the same way as [8]. Due to the ongoing unavailability of IEEE 802.11ah

hardware, we use these boards since the sub-GHz IEEE 802.15.4 radio included in the modules features a partial support for a subset of the modulation and coding schemes of IEEE 802.11ah [34]. The IEEE 802.15.4 standard defines, in addition to the MCS, the potential use of an "option" parameter, with the possible values being 1, 2, 3, and 4. This option gives, in addition to the MCS, more possibilities to alter the working of the physical layer (such as, the number of active tones). For this thesis we always used option 1, which utilizes a bandwidth of 1 MHz (Table 4.1), i.e., the minimal bandwidth in IEEE 802.11ah (Table 2.1). Note that in Table 4.1 the MCSs marked with "*" are not compliant with the IEEE 802.15.4 standard, although they are supported by the *Open-Mote B*. The first three MCSs (i.e., 0, 1, 2) of IEEE 802.15.4 use frequency repetitions and are therefore incompatible with IEEE 802.11ah. Therefore we only consider MCS 3 (referred to as MCS 1 in IEEE 802.11ah) in this work. From [8] we conclude that using MCS 5 in IEEE 802.15.4 (MCS 3 in IEEE 802.11ah) results in a shorter range, therefore it was not taken into account for these experiments. The device is configured to transmit at its maximal transmission power of 14.5 dBm. The sub-GHz radio used in the *Open-Mote B* has a receiver sensitivity of -123 dBm [35]. However, in [8] we observed that no signal was received after the RSSI gets below -115 dBm. In total four *Open-Mote Bs* are used, two of them are programmed for transmitting packets while the two other are programmed for receiving packets. This reduces the complexity of the firmware, because devices should either send or receive packets. Therefore, there is less need to handle special cases. For example, the device may be busy receiving a packet, during which the firmware gets interrupted to transmit a packet.This causes less problems with the boards and as a result provides more realistic measurements.

Raspberry Pi The Raspberry Pi (RPI) is connected to the IEEE 802.11 b/g/n network using the built-in WiFi radio. One *Open-Mote B* is used for transmitting packets on the IEEE 802.11ah network while a second one is used for receiving packets. Both devices are connected to the RPI using USB and therefore the RPI can both receive and transmit IEEE

802.11ah packets. The WiFi chip has a receiver sensitivity varying between -71.5 dBm and 98.7 dBm depending on the used MCS [36]. The *Adafruit GPS Hat* provides the RPI with its current location. Since this part of the setup is being moved within the measurement area, it is powered by a power bank.

WiFi Access Point In order to provide a IEEE 802.11b/g/n network a Routerboard RB951G-2HnD is used, which only supports the 2.4 GHz band. It is configured to transmit at its maximal transmission power of 14 dBm. However, when taking into account the antenna gain, a total of 17 dBm transmission power is used. This is below the allowed Effective Radiated Power (ERP) in Belgium for the 2.4 GHz band of 20 dBm. The device has a receiver sensitivity ranging from -79 dBm to -96 dBm depending on the used MCS [37].

Laptop A laptop is used to run the software acting as the base station. It is connected to the two *Open-Mote Bs* in order to transmit and receive packets over the IEEE 802.11ah network. By connecting the laptop to the *WiFi Access Point* using an Ethernet connection it is also able to send packets over the IEEE 802.11b/g/n network. This part of the network (i.e. the *laptop*, the *WiFi Access Point* and the second *Open-Mote B*) forms the server-side or base station part of the setup. Therefore it has a fixed location and hence it can be powered by the electricity grid.

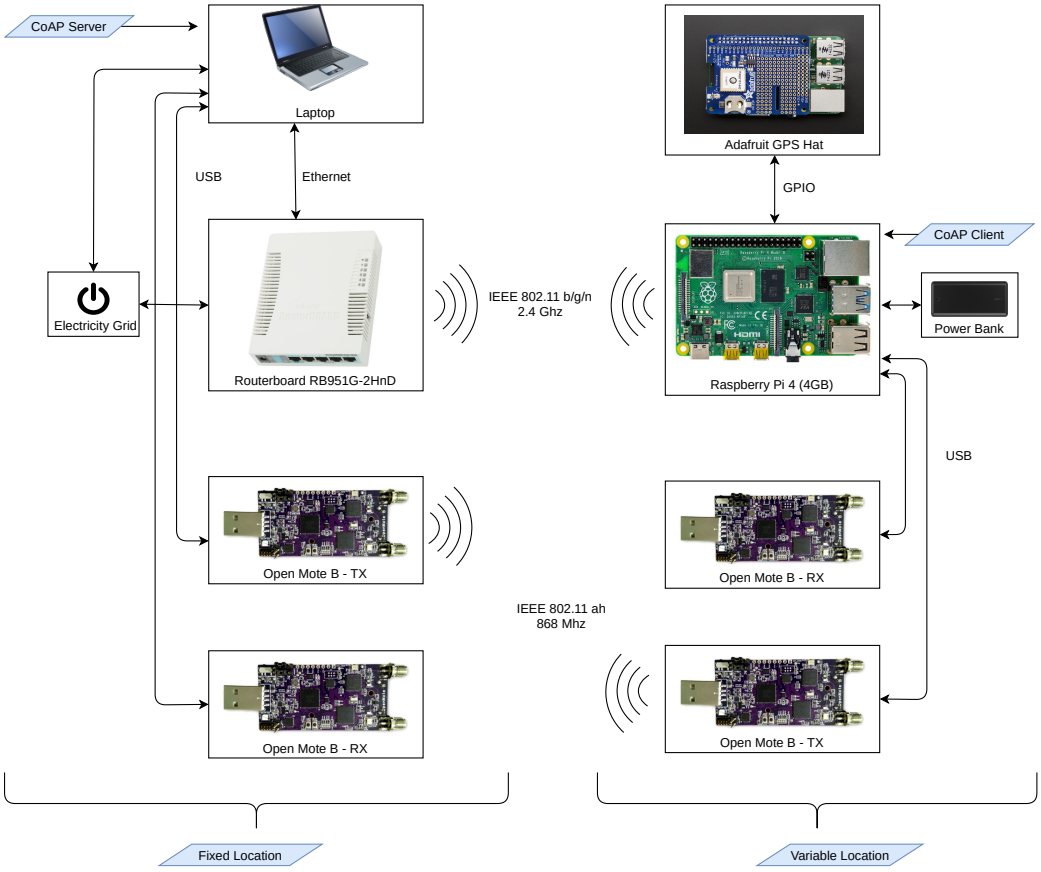


Figure 4.1: *Hardware setup and configuration.*

MCS Index	Modulation	Coding rate	Frequency Repetition	Data rate (kbps)	802.11ah MCS
0	BPSK	1/2	4x	100	N/A
1	BPSK	1/2	2x	200	N/A
2	QPSK	1/2	2x	400	N/A
3	QPSK	1/2	No	800	1
4	QPSK	3/4	No	1200*	2
5	16-QAM	1/2	No	1600*	3
6	16-QAM	3/4	No	2400*	4

Table 4.1: *IEEE 802.15.4g MCSs for option 1 (source: [38])*

4.1.2 Protocols

This section gives an overview of the protocol used during the measurement campaigns. We assume that an Internet Protocol version 6 (IPv6) only (i.e. no Internet Protocol version 4 (IPv4)) network is used and that User Datagram Protocol (UDP) is used as Transport Layer Protocol instead of the more complex Datagram Transport Layer Security (DTLS). Figure 4.2 gives a complete overview of the protocols used for the IEEE 802.11b/g/n link. The same overview for IEEE 802.11ah is shown in Figure 4.3. Because the devices used for transmitting the IEEE 802.11ah packets do not support the IEEE 802.11ah MAC layer, the choice was made to omit the MAC, IPv6 and UDP protocols from this setup. The idea is that it is better to have no implementation of MAC than to have a partial, possible incorrect implementation. In other words, the application-layer packets are directly sent over the physical layer. Furthermore, the MAC layer is not required for achieving the goals of this thesis.

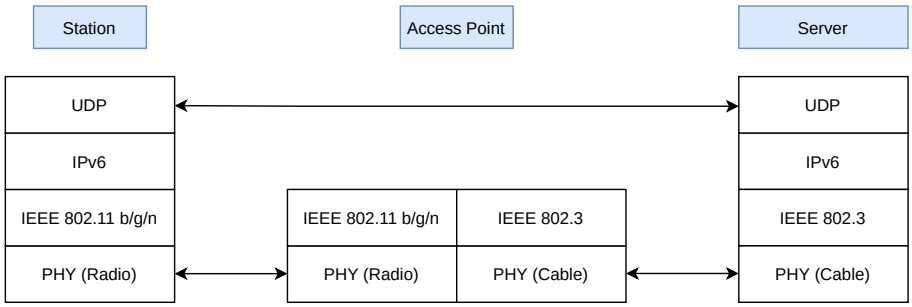


Figure 4.2: *Protocol stack for the IEEE 802.11b/g/n connection. The Server and Access Points are separate devices connected through Ethernet (IEEE 802.3).*

4.1.3 Data Collection

For IEEE 802.11ah

The aforementioned hardware setup is used for performing the measurements. In this case, only two of the four Open-Mote B's are used. One Open-Mote B is configured to transmit packets while the other is configured to receive packets. The transmitting module is connected to the RPI. The whole setup

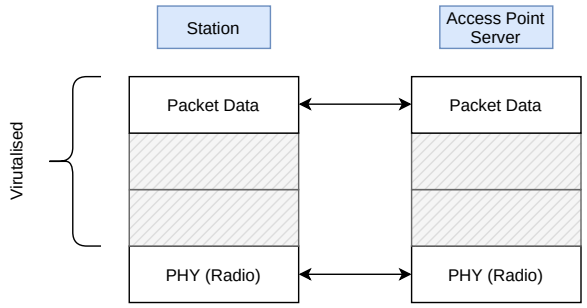


Figure 4.3: *Protocol stack for the IEEE 802.11ah connection. Packets are immediately transmitted over the physical channel, without taking other layers into account.*

is held by a person walking around the measurement environment. Simultaneously, the receiving module is placed on a pole at 1.5 m above the ground, logging the collected measurements. Both modules run a program adapted from the OpenWSN project [39]. The transmitter module continuously transmits packets of 509 bytes with 4 additional bytes for the Cyclic Redundancy Check (CRC). In an interval of 5 seconds the transmitter sends 158 packets, with the resulting throughput of 129 kbps. This is lower than the maximum physical-layer data rate supported by MCS 1 and 3 in, as shown in Table 2.1, in order not to overload the devices. Each time the transmitting Open-Mote B sends a packet, the RPI writes the packet number and current location to a log file. When the other module receives a packet, this message is logged to a computer over a serial interface. This second log contains the length of the packet, packet number, timestamp, SNR, and the indication of the CRC correctness. The timestamps on the transmit and receive sides are then correlated for mapping of the received packets and the transmitting locations. The measurement campaign took roughly 2.5 h, which resulted in a total of 341280 sent packets and 121658 received packets. The transmission power was configured to the maximal supported value by the radio chip, i.e. 14.5 dBm [35]. Both Open-Mote B modules utilized Atmel AT86RF215 transceivers that report RSSI as a signed integer between -127 and 4. The -127 value indicates an invalid RSSI value, which never occurred during the tests. In addition, the transceivers report the measured noise-floor, which was, in combination with the observed RSSI values, used for determining the SNR. Figure 4.4 gives an indication of the locations covered during the

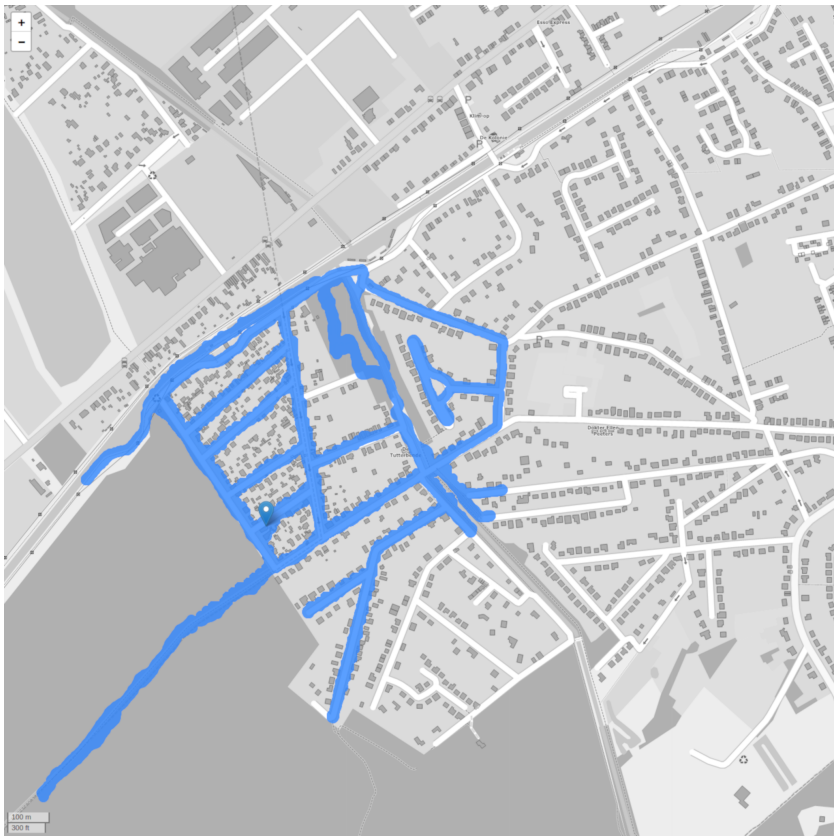


Figure 4.4: *Indication of measurements during the IEEE 802.11ah measurement campaign.*

measurement campaign. The observed packet loss is depicted in Figure 4.8, while the RSSI is shown in Figure 4.7. Locations where 100 % packet loss is observed were hidden. Sometimes packets of which the CRC did not pass were received, obviously they were counted as a lost packet. However, these lost packets can still be used to observe the RSSI, therefore the figure showing the RSSI contains more data points.

Figure 4.5: *Explanation for the colour codes used in the figures depicting the RSSI.*

Figure 4.6: *Explanation for the colour codes used in the figures depicting the packet loss.*

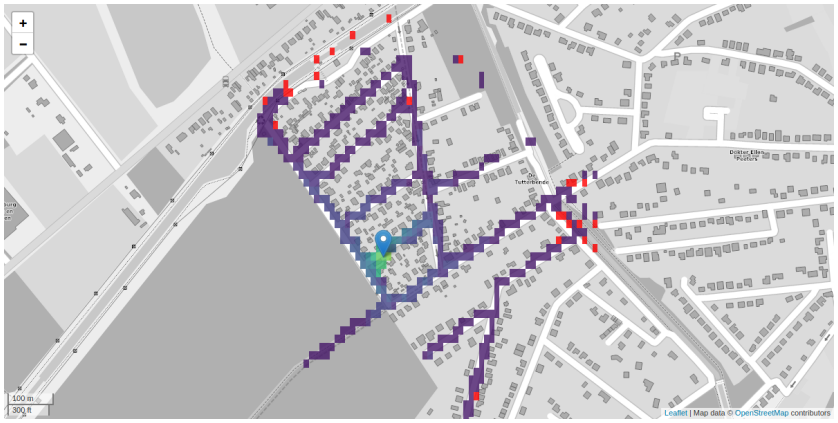


Figure 4.7: *Visualisation of the observed RSSI when using IEEE 802.11ah. The meaning of the colours is visible in Figure 4.5. The AP is located at the blue marker.*

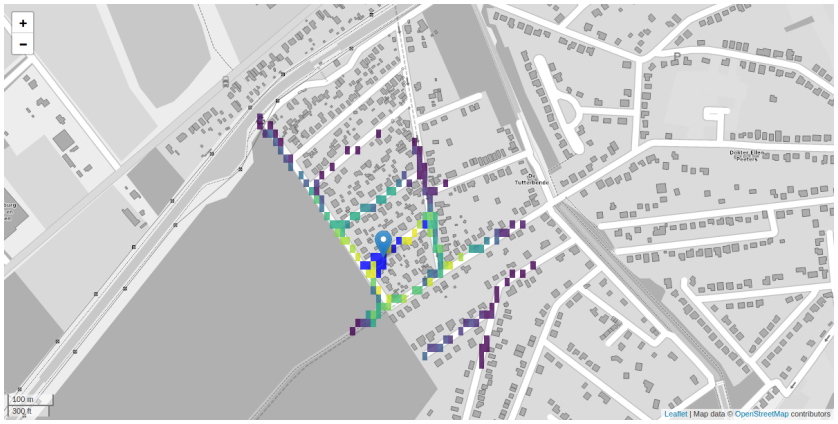


Figure 4.8: *Visualisation of the observed packet loss when using IEEE 802.11ah. Blue indicates zero percent packet loss, the meaning of the other colours is visible in Figure 4.6. The AP is located at the blue marker.*

For IEEE 802.11b/g/n

The method used throughout the measurement campaign for IEEE 802.11 b/g/n is very similar to that of IEEE 802.11ah. In this case, only the RPI is used together with an existing WiFi AP. A laptop is connected to the AP, acting as the receiver. The RPI sends an UDP packet of about 500 bytes every 15 ms. Therefore about 330 packets are sent every 5 seconds achieving a throughput of 266 kbps (without taking the overhead of the network layers into account). The RPI logs the current timestamp, current location and the unique id of each packet it sends. Furthermore, it periodically (every 250 ms) logs the current location along with the current RSSI. The RSSI is obtained via the `/proc/net/wireless` pseudo file of the Linux system running on the RPI. Because the laptop logs each packet it receives, the packet loss (in intervals of 5 seconds) can easily be calculated. The measurements locations are visualised in Figure 4.9, while the observed RSSI and packet loss are shown in respectively Figure 4.10 and Figure 4.11. Again, locations with 100 % packet loss are not shown. The measurement campaign took about one hour in which 240000 packets were sent of which 204203 packets were received.

Figure 4.9: *Indication of measurements during the IEEE 802.11b/g/n measurement campaign.*

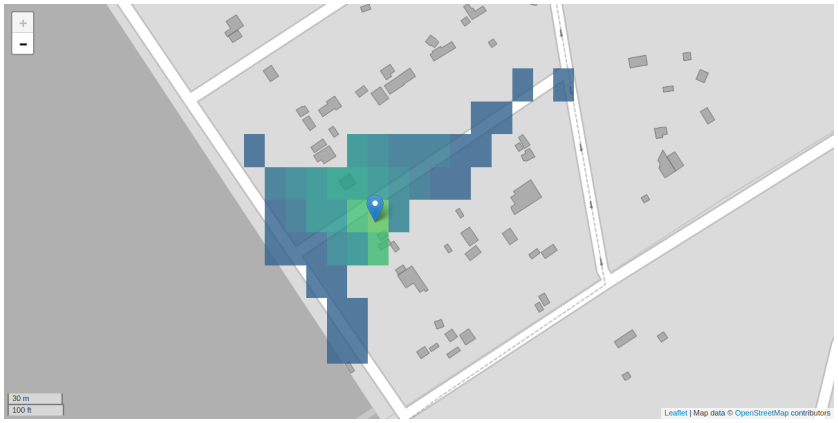


Figure 4.10: *Visualisation of the observed RSSI when using IEEE 802.11b/g/n. The meaning of the colours is visible in Figure 4.5. The AP is located at the blue marker.*

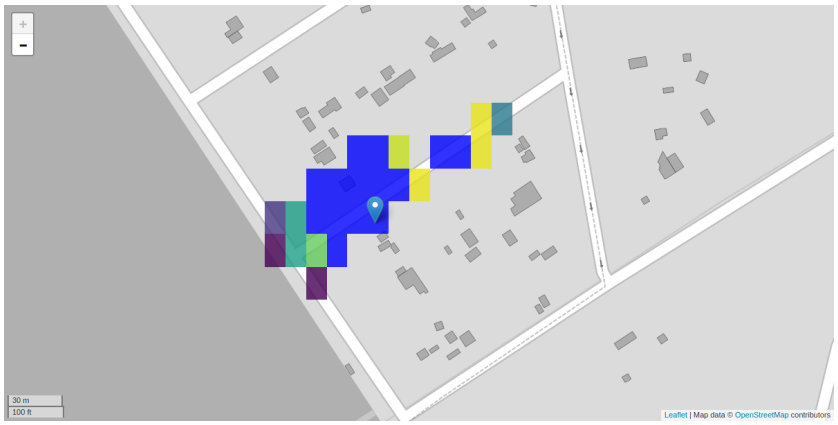


Figure 4.11: *Visualisation of the observed packet loss when using IEEE 802.11 b/g/n. Blue indicates zero percent packet loss, the meaning of the other colours is visible in Figure 4.6. The AP is located at the blue marker.*

4.2 Emulation Setup

4.2.1 Emulation of the Physical Layer and Device Location

This section explains how the emulation of the physical layer is implemented. Each packet sent by the emulation software is passed through this layer. Therefore, it is responsible for emulating the packet loss. Furthermore, it should emulate sending and receiving beacons. To achieve these goals, the layer has to know the RSSI and packet loss at the location of the experiments. The previous two sections described how these data were collected. The data have to be stored in a searchable format such that it can be used by an emulation algorithm, A k-d tree is an ideal data structure for this [29], allowing to look up some data in a geographical area. The emulation consists of two main parts.

First off all, packet loss should be emulated for packets passing through the emulated physical layer. This is handled by the `PacketFilter` function (Algorithm 7) which determines whether a packet should be dropped or not. For example, when the device is at a location with 45 % packet loss, the function should drop 45 % of the packets. In other words, 45 % of the packets

sent by the server to the client are be dropped by this function. Therefore, this function starts with looking up the observed packet loss at the current location. When no such information is found, it re-executes the lookup but this time using a less granular lookup. The first lookup searches in an area of $310\,\mathrm{m}^2$ whereas the fallback lookup searches in an area of $1242\,\mathrm{m}^2$. If no data is found after the second lookup, the algorithm assumes that there is 100 % packet loss at that location and thus all packets are dropped. In the other case, when one of the lookups yield a result, the average packet loss inside the lookup area is returned. Next, a random integer is generated in the range $[0, 100]$. Only when that random number is higher than the observed packet loss, the packet is sent to the receiving party, otherwise it is dropped. To intuitively explain the working of this mechanism, assume there is a packet loss of 75 %. Now pick 100 perfectly random numbers. Since these are perfectly random numbers, 75 of them will be lower or equal to 75. The idea is thus that if we send 100 packets, 75 of them will generate a random number lower or equal to 75 and therefore will be dropped resulting in 75 % packet loss. Of course, this mechanism results in some variation of the exact packet loss. However, this variation occurs in a real life experiment too. This principle is often used for simulating Bernoulli processes [40].

The second part of the emulator is responsible for generating beacons. This is a very simple process presented in Algorithm 8. First of all, the average RSSI in an area of $310\,\mathrm{m}^2$ around the device's current position is looked up. If the RSSI is found in the data set, this implies that packets were received during the measurement campaign. Therefore, a beacon must be sent. The `PacketFilter` function is used to integrate the packet loss.

The location of the device is emulated by first reading a GPS Exchange Format (GPX) file. Such a file contains a track of locations, each location is accompanied by a timestamp. Therefore, the time between two points is known. The emulator changes the location of the device according to this file while respecting the time between two points. Thus, the emulation happens in real time, at the same pace as recorded by the GPX. In order to have realistic error of the location, the GPX file is created by the GPS board of the RPI. In other words, we move the RPI around the measurement area and record the locations into a GPX file. The emulator supports a slowdown

Algorithm 7: PacketFilter()

Determines whether a packet should be dropped or not, by taking the packet loss into account.

```
1  PL = GetSurveyedPacektLossForCurrLoc(env = 0.0001);
2  if PL == null then
3  |  PL = GetSurveyedPacektLossForCurrLoc(env = 0.0002);
4  end
5  if PL == null then
   |  // No data found indicates not in measurement area and
   |     therefore 100% packet loss.
6  |  return false;
7  end
8  R = Generate Random Int in Range [0, 100];
9  if R > PL then
   |  // Pass packet
10 |  return true;
11 else
   |  // Drop packet
12 |  return false;
13 end
```

Algorithm 8: PeriodicBeaconSender()

Procedure to periodically send a beacon, together with the current RSSI. Beacon is only sent when the PacketFilter allows the packet.

```
1  for Beacon interval elapsed do
2  |  RSSI = GetSurveyedRssiForCurrLoc(env = 0.0001);
3  |  if RSSI != null then
4  |  |  if PacketFilter() then
5  |  |  |  SendBeacon(RSSI) // Effectively send beacon
6  |  |  end
7  |  end
8  end
```

function. Therefore the GPX file can be generated at any pace, while the emulation happens at a faster or slower pace. For this thesis we decided to use the speed of a pedestrian (i.e. with an average of 4 kmph and a maximum of 6 kmph).

4.2.2 Protocols

This section describes the protocols used in the emulation software, in contrast to the protocols used during the measurement campaign. Figure 4.13 and Figure 4.12 give a complete overview of the protocols used for respectively the IEEE 802.11ah and IEEE 802.11b/g/n link. The UDP, IPv6 and MAC layers were left out since there is no need for emulating these. This has no drawbacks when studying the performance of vertical handover algorithms. Adding these layers to the emulation software would unnecessary complicate our setup.

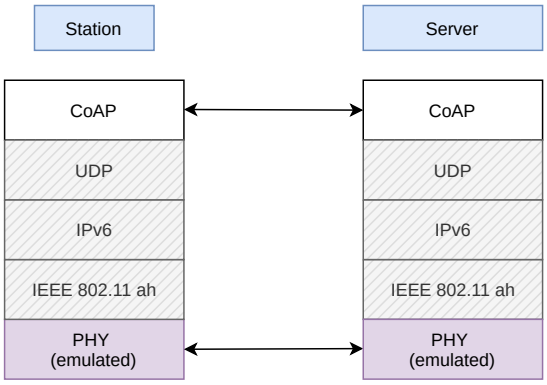


Figure 4.12: *Protocol stack for the emulated IEEE 802.11b/g/n connection. The UDP, IPv6 and MAC layers are left out. The physical layer is emulated.*

The Constrained Application Protocol (CoAP) protocol is used as the application-layer protocol. CoAP is a protocol for the communication between devices which are somehow constrained, such as a constraint on power usage [41, 42]. A goal of CoAP is that it should be easily translated into Hypertext Transfer Protocol (HTTP), such that it can be integrated with the web. The protocol typically runs over either UDP or DTLS. Because UDP does not guarantee reliability, this is optionally guaranteed by CoAP. Each message can be marked as Confirmable. As long as the recipient of the message has

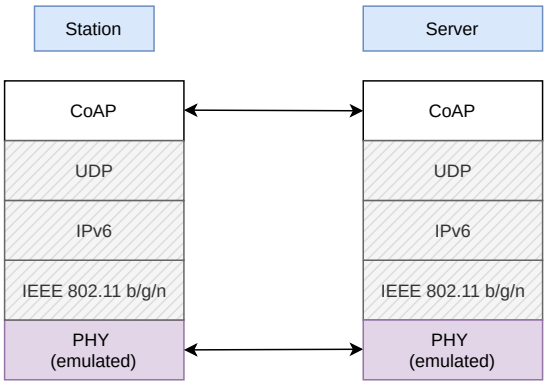


Figure 4.13: *Protocol stack for the emulated IEEE 802.11ah connection. The UDP, IPv6 and MAC layers are left out. The physical layer is emulated.*

not acknowledged the message, Confirmable messages are retransmitted after a default timeout. Exponential back-off is used between the retransmissions. For keeping track of messages and their retransmission, each message contains a Message ID. Because this thesis focuses on (constrained) IoT use cases, the CoAP protocol is a very good choice.

4.2.3 Software Setup

The software stack consists of a CoAP Server and a CoAP Client, this section explains both parts.

Application Server

The Application Servers runs on the laptop and exists mostly of a CoAP server. The Eclipse Californium [43] framework is used for implementing the CoAP server. The server software is written in a combination of Java and Kotlin. A custom `connector` ensures that the server can send an receive packets over both the IEEE 802.11ah and IEEE 802.11b/g/n networks (either emulated or by using the hardware). On startup, the server sends packets destined for the client over both networks. After receiving a packet from the client, the server takes account of the used technology. This technology is then used for transmitting further packets. Therefore, when the client handovers to a different network, the server will switch too. In a more

traditional setup, the server (or a intermediate router) knows to which network the client is associated. However, since there is no full implementation of the IEEE 802.11ah MAC protocol, there is also no association procedure. This also implies that there is no implementation of IEEE 802.11ah beacons available. Therefore, the Application Server is also responsible for sending a special packet (as an imitation of the MAC beacon) every 2048 ms [13]. This packet has a length of 184 bytes in order to mimic a typical beacon length.

The server exposes one CoAP endpoint, namely `/gps`. A client can send a `POST` request to this endpoint, containing the coordinate and (relative) time of the client. The server aggregates these requests into a log.

Application Client

Similar to the server, the CoAP Client uses the Eclipse Californium library for implementing the CoAP. The handover algorithms are implemented in the same program. The purpose of the client is to regularly report its current location and (relative) time to the server. Therefore, it sends a `POST` request every 500 ms to the `/gps` endpoint of the server. Every request by the client should be answered by a `Created` response by the server. A request sent to the server has a timeout of two seconds [41]. In other words, if the client does not receive a response of the server after two seconds, it will not try to retransmit packets.

In a real world scenario, a tracking device would sends its location much less frequent than every 500 ms. It could, for instance, send it every few seconds, every few minutes or even a few times a day, depending on the use case. For the experiments conducted in this thesis, sending the location more frequent, causes the network to be more loaded and gives more insightful results.

4.2.4 Propagation Models

While Section 3.4.1 gave an introduction to estimating the SNR using propagation models, it did not provide a definition of the used propagation models.

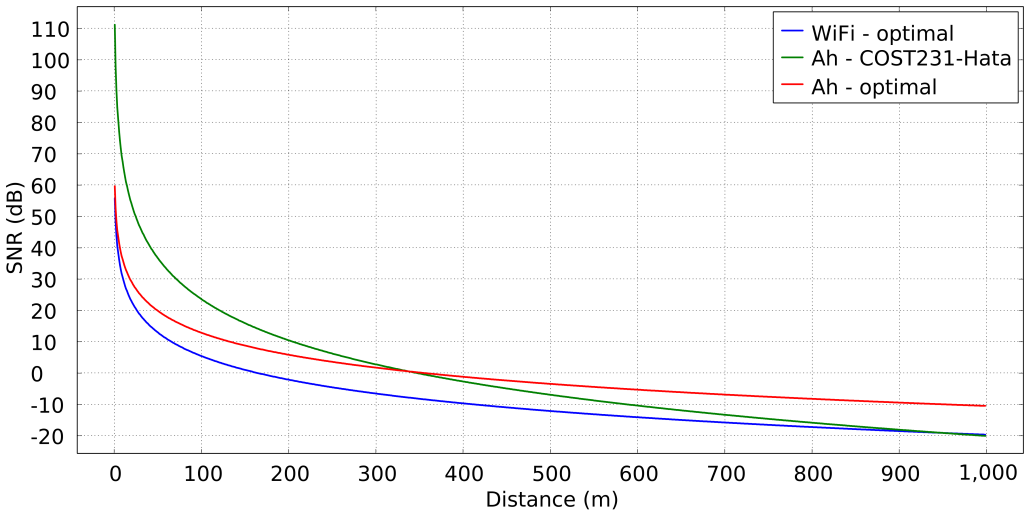


Figure 4.14: *Comparison of all propagation models used.*

IEEE 802.11ah

In our previous research, we have shown that in the case of IEEE 802.11ah, the COST-231 Hata is the best fitting propagation for our measurement location. This section gives a brief definition of the COST-231 Hata model.

The modelled path loss is given in Equation (4.1) [31], with parameters being frequency f [MHz], distance d [km], heights of the AP h_{Base} [m] and of the mobile station h_{Mobile} [m]. The mobile station antenna height correction factor $a(h_{Mobile})$ is defined in Equation (4.2) and the constant offset C_m is defined in Equation (4.3)

$$\begin{aligned} L_b = {} & 46.3 + 33.9 \log_{10}(f) \\ & - 13.82 \log_{10}(h_{Base}) - a(h_{Mobile}) \\ & + (44.9 - 6.55 \log_{10}(h_{Base})) \log_{10}(d) + C_m. \end{aligned} \tag{4.1}$$

$$\begin{aligned} a(h_{Mobile}) = {} & (1.1 \ \log_{10}(f) - 0.7) \ h_{Mobile} \\ & - (1.56 \ \log_{10}(f) - 0.8) \end{aligned} \tag{4.2}$$

Parameter	Value
Frequency	868 MHz
Base Station[1] Height h_{Base}	1.5 m
Mobile Station Heighth_{Moile}	1.5 m
Constant Offset C_m	0 dB
Transmission Power P_{tx}	14.5 dBm
Receiver Sensitivity S_{rx}	-109.0 dBm

Table 4.2: *Parameters used for configuration the COST-231 Hata propagation model.*

$$C_m = \begin{cases} 0\,\text{dB} & \text{for medium sized cities and} \\ & \text{suburban centres with medium tree density} \\ 3\,\text{dB} & \text{for metropolitan centres.} \end{cases} \tag{4.3}$$

The used parameters for the propagation model, for the specific environment of this thesis, are listed in Table 4.2.

Figure 4.14 depicts the estimated SNR as a function of the distance for IEEE 802.11ah. Observe that the SNR decreases more faster at lower distance compared to higher distances.

Since the measurement campaign provides an extensive amount of data, such as the observed RSSI at our experiment location, this can be used to create an optimal propagation model. During the experiments we study what the influence of using this optimal model is, compared to the classic COST-231 Hata model. For this purpose an empirical propagation model based on the log-distance path loss model can be used [44]. The definition of this model contains a stochastic component representing the attenuation caused by fading. However, for this thesis, the predicted SNR is used as parameter for the handover algorithm. It is undesirable to have a stochastic variable as base for such an algorithm, since the output of the algorithm would then depend on the variation of that variable. Hence, the component corresponding with fading was not taken into account. This results in the base formula Equation (4.4), similar to the “Macro Deployment” and “Micro Deployment“

models proposed in [33].

$$L_b = \alpha + \beta \log_{10}(d) \tag{4.4}$$

From the measured RSSI, the SNR is calculated. Next, the Random Search algorithm [45] is used to find the coefficients of the model such that the Mean Absolute Error (MAE) is minimal. Coefficients are searched in the $[-100, 100]$ range. This results in Equation (4.5) of which a plot is shown in Figure 4.14.

$$L_b = 63.69 + 23.40 \log_{10}(d) \tag{4.5}$$

IEEE 802.11b/g/n

For IEEE 802.11b/g/n we did not find any propagation model suited for an outdoor setup and the used frequency. Therefore we opted for an empirical propagation model based on the log-distance path loss model. The data used to fit the propagation model is the same data as used for emulation the physical layer (Section 4.1.3). From this data the SNR is calculated. Finally, the Random Search algorithm was again applied to find the coefficients of the model such that the MAE is minimal. Coefficients are searched in the $[-100, 100]$ range. The result is visible in Figure 4.14, the final model is defined by Equation (4.6) For this model the MAE is 1.351

$$L_b = -76.50 - 4.92 \log_{10}(d) \tag{4.6}$$

4.3 Performance Metrics

This section discusses the metrics used in the evaluation. We distinguish between *Algorithm Metrics* which characterize the system-level performance of the different handover algorithms and *Application Metrics* which give a more detailed view of the effects of a handover algorithms from the application's viewpoint.

4.3.1 Algorithm Metrics

Radio On (*On*) This metric is evaluated for each network technology (i.e. IEEE 802.11ah and IEEE 802.11b/g/n). It is the ratio of the total time the radio is turned on and the duration of the experiment.

$$\text{On in } \% = \frac{\text{Radio On}}{\text{Total Time}}$$

This gives an indication of the power usage of a device: the longer the radio is on, the more energy is consumed.

Connection Effiency (*Eff*) It is the ratio of the time the device is connected to this technology and the time the radio of this technology is powered on.

$$\text{Eff in } \% = \frac{\text{Connected to Technology}}{\text{Radio On}}$$

This measure gives an idea about how efficient the radio is used. When the ratio is low, the radio was turned on for too long, i.e. it was impossible to make a connection but the algorithm turned the radio on. On the other hand, when the ratio is high, the radio was only turned on when a connection was actually possible. Therefore, this gives a good indication about the correctness of the algorithm. However, it should be noted that efficiency alone can be misleading. For example, if the radio is turned on 1 % of the time and the device is connected the whole time, the efficiency is 100 %. However, this does not represent a good result. Therefore, when looking at efficiency, the other metrics should be taken into account too. Finally, note that this metric is calculated for both network technologies separately, i.e. they are calculated as if the device contained only one network technology. This reduces the influence of the performance of one technology on the other technology, making it easier to compare different experiments.

4.3.2 Application Metrics

Packet Loss (*PL*) The Packet Loss metrics expresses the observed packet loss during the whole experiment. It is calculated by taking both pack-

ets sent by the client and server using both technologies.

$$\text{PL in } \% = \frac{\text{Number of received packets}}{\text{Number of sent packets}}$$

As usual it gives an indication of the quality of the connection. In addition, the packet loss also influences the power efficiency of the device. Packet loss causes the network to retransmit packets resulting in more power consumption.

Ratio of successful updates (*Updates*) This metric gives the ratio of successful location updates and total updates. In other words, the radio of location updates received by the server and location updates sent by the client.

$$\text{Updates in } \% = \frac{\text{Updates received by the server}}{\text{Updates sent by the client}}$$

While the *Connection Efficiency* metric gives an idea about how much time the device is connected, this metric indicates how useful the connection is. For example, when the (absolute) connection time is high but the number of successful updates is low this means that the connection was not good enough to send location updates. Conversely, the connection quality is high when both the connection time is high and the number of successful updates is high.

95th percentile of distance between server and client (*Distance*) This final metric illustrates the maximal distance at which the client sent a localisation update. Instead of using the absolute maximal distance, the 95th percentile is used in order to prevent the effect of outliers. Because the emulation contains a stochastic component to emulate the packet loss, the exact location at which localisation updates can be sent varies between the different runs of an experiment. In order to reduce this variation, the 95th percentile of all localisation updates of a (single) experiment is used.

Chapter 5

Experiments

This chapter discusses all performed experiments. Firstly, all experiments using the emulation software are presented. That section starts with some experiments using the reference algorithms, followed by experiments using the location-based discovery and handover algorithm. Thereafter, some parameter tuning is performed to test the influence of these parameters. Finally, some experiments are performed using the hardware prototype in order to demonstrate the feasibility of our example application on a real-world IoT device.

5.1 Emulation-based Experiments

These experiments were performed using the emulation software described in Section 4.2. Figure 5.1 visualises the used trajectory at the measurement area. The track is about 5 km long, taking about 75 minutes to complete at an average speed of 4.24 kmph. The distance to the AP, shown in Figure 5.2, clearly illustrates that the device moves multiple times away from the AP, followed by moving closer. This is ideal to test the performance at the border of the coverage range.

In order to review the experiment results in a good way, it is interesting to know more about the circumstances of the measurement location. While in

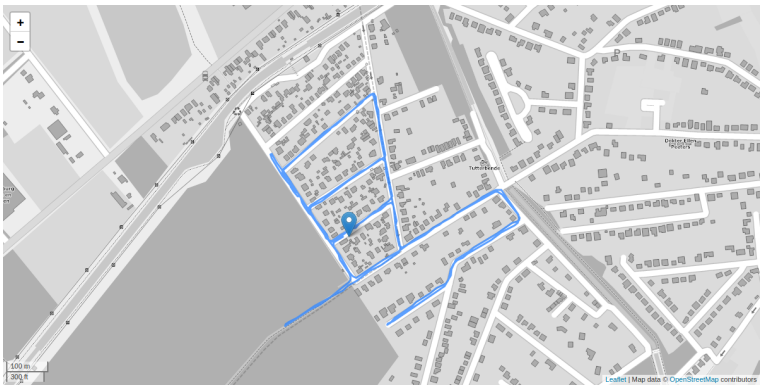


Figure 5.1: *Trajectory used during the emulations.*

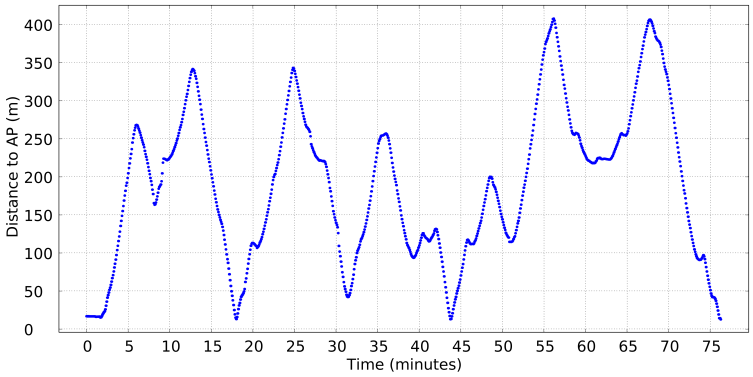


Figure 5.2: *Distance between the device and AP during the experiments.*

Section 4.1.3 a map was provided with the observed RSSI and packet loss, it is now possible to provide the same information related to the trajectory. Figure 5.3 and Figure 5.4 present the observed packet loss as a function of the time. As expected, the packet loss for IEEE 802.11b/g/n indicates more locations for which there is no data, since the range is much shorter.

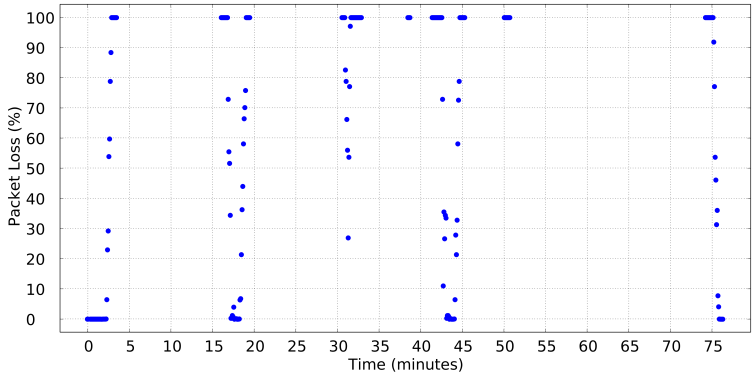


Figure 5.3: *Packet loss observed for IEEE 802.11b/g/n at the trajectory during the measurement campaign*

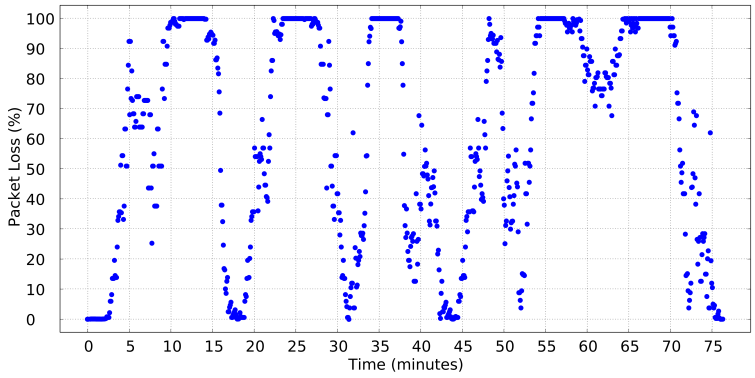


Figure 5.4: *Packet loss observed for IEEE 802.11ah at the trajectory during the measurement campaign*

5.1.1 Reference Experiments

Using Beacons

The experiments described in this section were performed using Algorithm 2. Table 5.1 lists the results for these experiments. For Experiment 500, the allowed number of missed beacons was set to one, i.e. the classic, default handover algorithm. The other experiments vary this parameter for both IEEE 802.11ah and IEEE 802.11b/g/n from one up to three. Experiment 500 yields the worst results: it has the lowest efficiency, lowest number of updates and shortest distance. Increasing the β parameter improves the efficiency, number of updates and the distance, although the observed packet loss also increases. Nevertheless, Experiment 508 could be denoted as the best

experiment: it has the highest efficiency, second most updates and longest distance. While the packet loss is worse and therefore the power efficiency decreases this is not important because the radio is powered on the whole time when using Algorithm 2. Figure 5.5 and Figure 5.6 depict the location updates received by the server. Clearly, Experiment 508 (with higher β) has better coverage at the borders, where there is more packet loss. Because in that experiment the client only disconnects after having missed three beacons, the connection stays active for a longer period. Therefore the client has more opportunities to send location updates, albeit with some packet loss. Finally, Figure 5.7 and Figure 5.8 visualises the observed packet loss during the experiment. There are no significant differences noticeable.

	Configuration						Results in %						
	Ah			WiFi			Ah		Wifi			Application	
ID	σ_{Ah}	β_{Ah}	Ω_{Ah}	σ_{bgn}	β_{bgn}	Ω_{bgn}	**Eff**	**On**	**Eff**	**On**	**PL**	**Updates**	**Distance**
500	N/A	1	N/A	N/A	1	N/A	37	100	9	100	20	42	172
501	N/A	1	N/A	N/A	2	N/A	38	100	10	100	20	44	176
502	N/A	1	N/A	N/A	3	N/A	38	100	11	100	21	44	172
503	N/A	2	N/A	N/A	1	N/A	47	100	9	100	24	47	187
504	N/A	2	N/A	N/A	2	N/A	47	100	10	100	25	46	176
505	N/A	2	N/A	N/A	3	N/A	49	100	11	100	26	48	189
506	N/A	3	N/A	N/A	1	N/A	53	100	9	100	28	47	183
507	N/A	3	N/A	N/A	2	N/A	53	100	10	100	29	47	187
508	N/A	3	N/A	N/A	3	N/A	54	100	10	100	28	47	190

Table 5.1: *Results for experiments using Algorithm 2.*

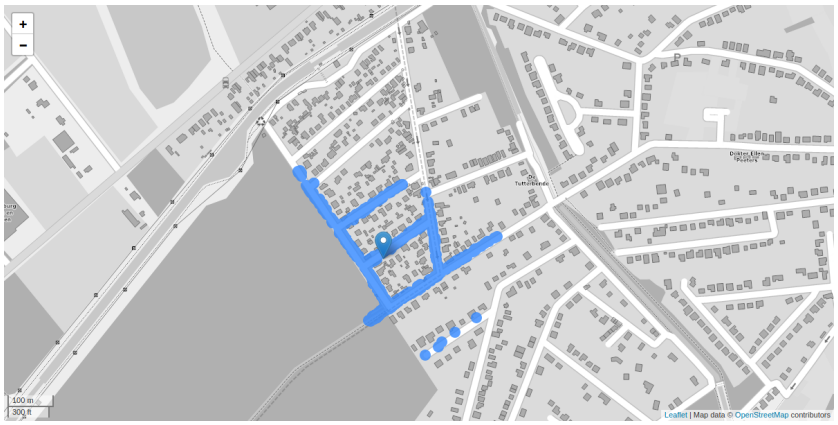


Figure 5.5: *Localisation updates received by the server during Experiment 500.*

Figure 5.6: *Localisation updates received by the server during Experiment 508.*

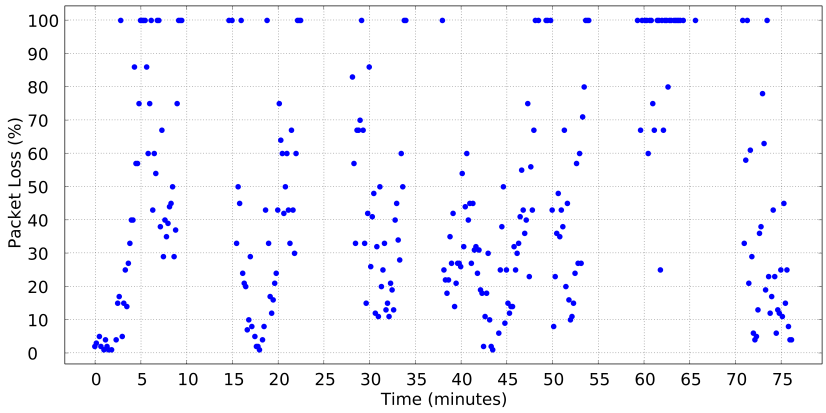


Figure 5.7: *Observed packet loss during Experiment 500.*

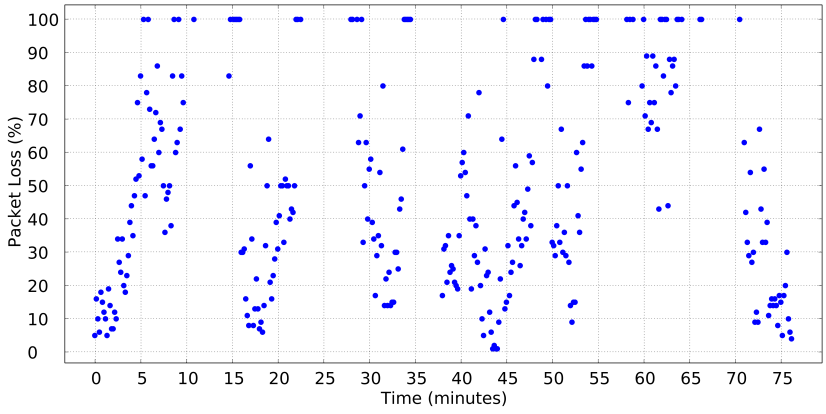


Figure 5.8: *Observed packet loss during Experiment 508.*

Using a Radio Environment Map

The experiments covered in this section are using Algorithm 4. To recapitulate, it uses the surveyed packet loss to know whether it is feasible to handover at the device's current location. Because the algorithm does not take beacons into account, the efficiency of the algorithm is always calculated as 100%. Table 5.2 lists the results of the experiments. The first two lines contain the worst and the best results of the beacon based experiments. Based solely on the application metrics, one would intuitively expect the experiment with the lowest amount of allowed packet loss to be the worst. Experiment 320 confirms this expectation. Analogously, the best result is when 95% packet loss is allowed (Experiment 337), however, there the Ah radio is turned on for 67% of the time. This result achieves to successfully send 50% of the status updates reaching a distance of 218 m. Choosing the *overall* best result depends on the trade-off between power efficiency and application performance. For instance, an use case could desire that the radio is only on for about 55 % of the time. In that case, Experiment 334 is the best result looking at the combination of achievable distance and number of updates. Note that choosing the best result always involves this trade-off, which becomes clear in the following sections. From the figures indicating the location updates, Figure 5.9, Figure 5.10, Figure 5.11, for respectively Experiments 320, 334 and 337, it is clear that the covered area increases when a higher amount of packet loss is allowed. Figure 5.12 depicts the observed packet loss experiment during the experiment where 10 % packet loss is allowed. The vast amount of observations is below 10 %, although their are some outliers. This can be explained by the variation in packet loss at each location, whereas the algorithm only takes the average packet loss into account. Finally, Figure 5.13 and Figure 5.14 display the observed packet loss for Experiment 334 and 337 respectively.

Configuration							Results in %						
	Ah			WiFi			Ah		Wifi		Application		
ID	σ_{Ah}	β_{Ah}	Ω_{Ah}	σ_{bgn}	β_{bgn}	Ω_{bgn}	Eff	On	Eff	On	PL	Updates	Distance
500	N/A	1	N/A	N/A	1	N/A	37	100	9	100	20	42	172
508	N/A	3	N/A	N/A	3	N/A	54	100	10	100	28	47	190
320	Packet loss ≤ 10						100	14	100	7	3	35	113
321	Packet loss ≤ 15						100	19	100	7	5	39	143
322	Packet loss ≤ 20						100	21	100	7	6	42	143
323	Packet loss ≤ 25						100	22	100	7	8	43	147
324	Packet loss ≤ 30						100	25	100	8	9	44	148
325	Packet loss ≤ 35						100	28	100	8	15	44	147
326	Packet loss ≤ 40						100	32	100	9	13	46	153
327	Packet loss ≤ 45						100	36	100	9	16	47	159
328	Packet loss ≤ 50						100	38	100	9	17	49	159
329	Packet loss ≤ 55						100	43	100	9	19	50	164
330	Packet loss ≤ 60						100	44	100	10	24	49	176
331	Packet loss ≤ 65						100	46	100	10	21	50	176
332	Packet loss ≤ 70						100	49	100	10	25	50	179
333	Packet loss ≤ 75						100	52	100	11	25	49	181
334	Packet loss ≤ 80						100	54	100	11	27	49	190
335	Packet loss ≤ 85						100	58	100	11	31	51	195
336	Packet loss ≤ 90						100	61	100	11	27	50	213
337	Packet loss ≤ 95						100	67	100	12	29	50	218

Table 5.2: *Results for experiments using Algorithm 4.*

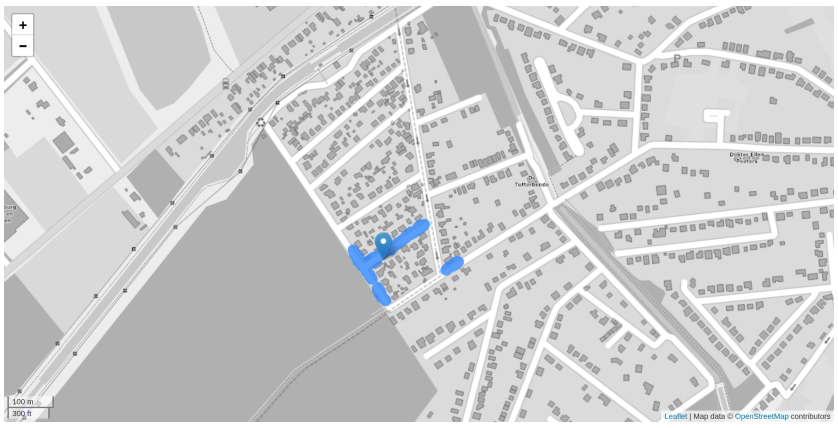


Figure 5.9: *Localisation updates received by the server during Experiment 320.*

Figure 5.10: *Localisation updates received by the server during Experiment 334.*

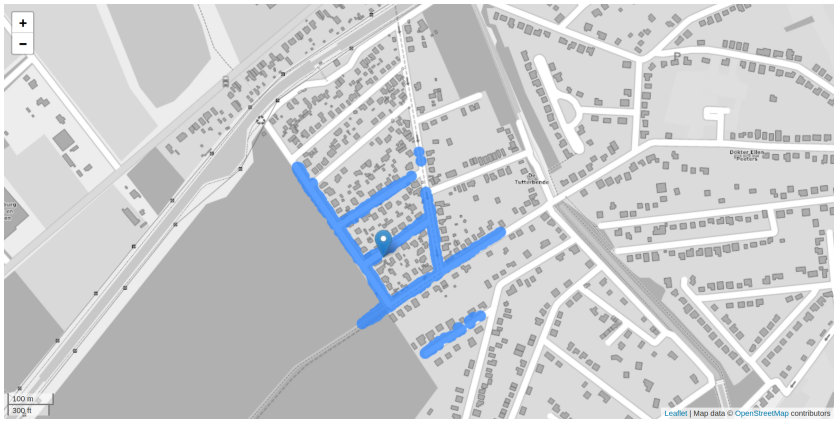


Figure 5.11: *Localisation updates received by the server during Experiment 337.*

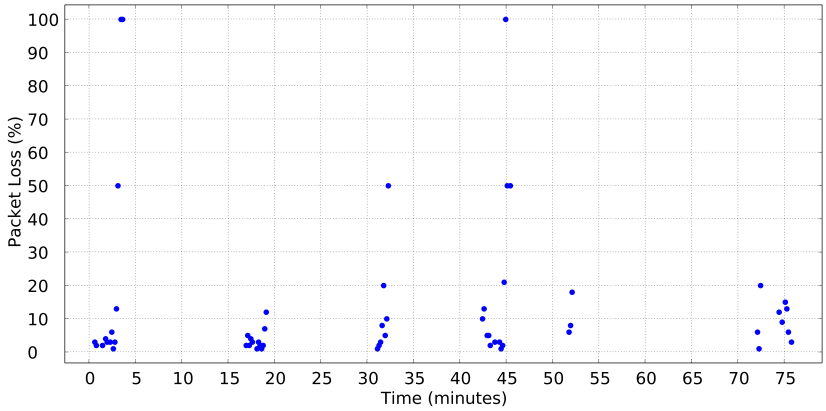


Figure 5.12: *Observed packet loss during Experiment 320.*

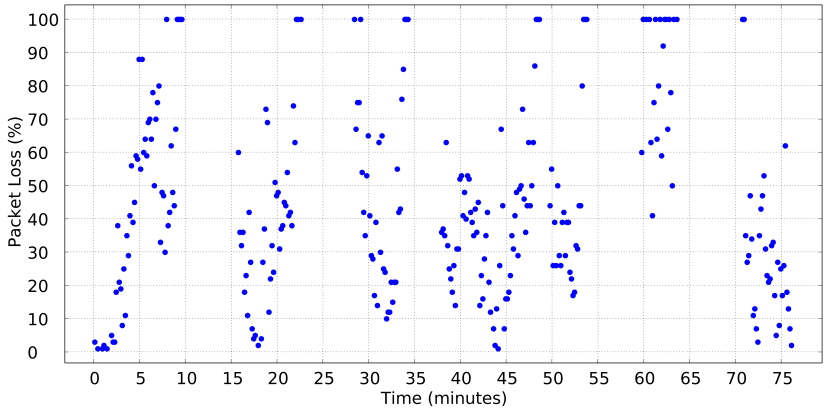


Figure 5.13: *Observed packet loss during Experiment 334.*

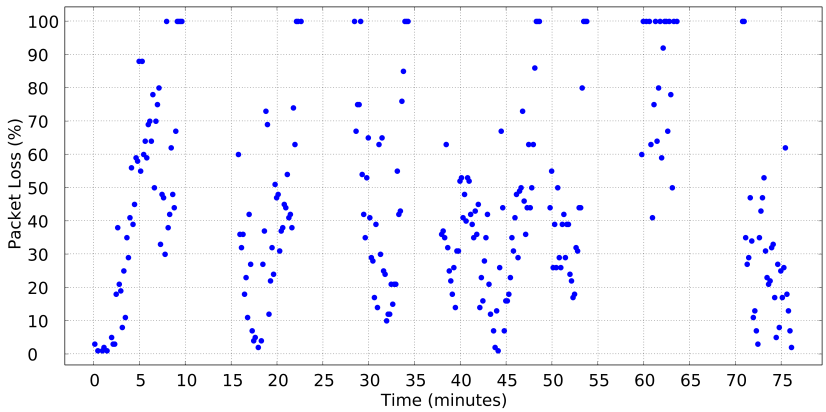


Figure 5.14: *Observed packet loss during Experiment 337.*

5.1.2 Estimation-based Experiments

All of the following experiments use Algorithm 6, i.e. using the location to estimate the SNR. Table 5.3 lists the results of experiments with the aim to explore some sensible parameters. The β (allowed missed beacons) and Ω (offset) parameters were all fixed to respectively 2 and 0. Both for IEEE 802.11ah and IEEE 802.11b/g/n the σ (required SNR) increases from 0 to 50 in steps of 10. The highest distance is achieved in the experiment where $\sigma = 0$. However, in that case the radios of IEEE 802.11ah and IEEE 802.11 b/g/n are turned on respectively 93 % and 54 % of the time. There is no single best result, because the trade-off between efficiency and application performance has to be made depending on the use case. However, when we again look for an experiment where the radio is turned on for about 55 %, Experiment 8 seems like a good fit. It has an efficiency of respectively 61 % and 94 % for IEEE 802.11ah and IEEE 802.11b/g/n while still having good coverage. When choosing a configuration which consumes less power (e.g., Experiment 12), the maximal distance and number of updates significantly decrease. Recall, Experiment 500 and 508 where the reference beacon listening algorithm is used. Compared to Experiment 500, Experiment 0 has slightly larger coverage and a higher amount of updates but a 14 percentpoint more efficient use of IEEE 802.11ah and IEEE 802.11b/g/n radio time. The performance of Experiment 0 is very similar to that of Experiment 508, although Experiment 0 consumes slightly less power and has a slightly less coverage. Experiment 334 (using an REM) consumes drastically less power than Experiment 0 while achieving similar coverage and number of updates. In addition, Experiment 337, which also uses an REM reaches a larger distance and more updates while consuming less energy. Experiment 8 has lower coverage than Experiments 500 and 508, although a similar number of updates. However, it uses its radio for respectively IEEE 802.11ah and IEEE 802.11b/g/n 47 % and 95 %, and thus saving a lot of energy. Experiment 334 has similar power consumption as Experiment 8 but has larger coverage.

Figure 5.18 provides an overview on the performance of the algorithm for Experiment 8. This plot contains two lines, each line representing the estimated SNR for a single technology. The colour of the line indicates the `Advice` of

the handover algorithm. The following colours are used to distinguish the different cases:

Blue The algorithm advices to perform a handover.

Green The device is connected to the link and advices to stay connected.

Yellow The link specific algorithm (Algorithm 6) advices to handover, however the decision algorithm (Algorithm 1) prefers the other link.

Black The algorithm advices to not perform a handover.

Red The algorithm advices to disconnect from the current link.

This plot makes it possible to compare the estimated SNR (i.e. the line) to the actual observed SNR (i.e. the dots). Furthermore, by examining the colour of the line, one can observe at which point the algorithm makes a decision. For example, in Figure 5.18, the algorithm decides to handover from IEEE 802.11b/g/n (the bottom line) to IEEE 802.11ah (the upper line) after about 2.5 minutes. We can also see that this happens when the estimated SNR for IEEE 802.11b/g/n becomes lower than 20 dB. To give another example, at time 16 we see that the algorithm decides to use IEEE 802.11ah for one minute, after which it handovers to IEEE 802.11b/g/n. From minute 60 until 65 we see that some IEEE 802.11ah beacons are received, however, the algorithm does not decide to perform a handover, because the SNR is too low. This plot is very useful for optimising the working of the algorithm.

To conclude, the location-based discovery and handover algorithm is able to achieve a comparable maximal range in which the application has sent location updates as the beacon listening based algorithm, while using half of the energy for IEEE 802.11ah and even a reduction of 95 % percent-point in the case of IEEE 802.11b/g/n (Experiment 8 compared to Experiment 500). This result is achieved without performing many optimisations to the parameters of the algorithm. Comparing to the algorithm using an REM, our algorithm achieves slightly less optimal results. However, using an REM is almost always unfeasible.

Configuration							Results in %						
	Ah			WiFi			Ah		Wifi		Application		
ID	σ_{Ah}	β_{Ah}	Ω_{Ah}	σ_{bgn}	β_{bgn}	Ω_{bgn}	**Eff**	**On**	**Eff**	**On**	**PL**	**Updates**	**Distance**
500	N/A	1	N/A	N/A	1	N/A	37	100	9	100	20	42	172
508	N/A	3	N/A	N/A	3	N/A	54	100	10	100	28	47	190
320			Packet loss $\leq$ 10				100	14	100	7	3	35	113
334			Packet loss $\leq$ 80				100	54	100	11	27	49	190
337			Packet loss $\leq$ 95				100	67	100	12	29	50	218
0	0	2	0	0	2	0	51	93	23	43	23	46	186
1	0	2	0	10	2	0	52	93	75	13	22	47	186
2	0	2	0	20	2	0	51	93	94	5	24	53	176
3	0	2	0	30	2	0	51	93	100	3	20	52	179
4	0	2	0	40	2	0	49	93	0	0	20	51	176
5	0	2	0	50	2	0	51	93	0	0	19	50	170
6	10	2	0	0	2	0	59	53	23	43	19	40	147
7	10	2	0	10	2	0	58	53	74	13	19	42	147
8	10	2	0	20	2	0	61	53	94	5	17	47	146
9	10	2	0	30	2	0	59	53	100	1	16	47	147
10	10	2	0	40	2	0	60	53	0	0	21	46	147
11	10	2	0	50	2	0	59	53	0	0	17	47	143
12	20	2	0	0	2	0	34	30	23	43	11	24	57
13	20	2	0	10	2	0	34	30	74	13	8	25	62
14	20	2	0	20	2	0	34	30	94	5	5	29	57
15	20	2	0	30	2	0	34	30	100	0	11	29	65
16	20	2	0	40	2	0	34	30	0	0	14	29	57
17	20	2	0	50	2	0	34	30	0	0	13	29	62
18	30	2	0	0	2	0	42	13	23	43	9	23	57
19	30	2	0	10	2	0	42	13	74	13	7	24	50
20	30	2	0	20	2	0	41	13	94	5	9	18	32
21	30	2	0	30	2	0	41	13	100	3	11	18	34
22	30	2	0	40	2	0	42	13	0	0	7	18	34
23	30	2	0	50	2	0	42	13	0	0	2	18	33
24	40	2	0	0	2	0	56	8	23	43	15	23	57
25	40	2	0	10	2	0	56	8	73	13	15	22	50
26	40	2	0	20	2	0	58	8	94	5	7	16	24
27	40	2	0	30	2	0	57	8	100	3	5	14	23
28	40	2	0	40	2	0	57	8	0	0	11	14	23
29	40	2	0	50	2	0	57	8	0	0	1	14	23
30	50	2	0	0	2	0	73	5	23	43	7	23	57
31	50	2	0	10	2	0	74	5	73	13	16	22	50
32	50	2	0	20	2	0	73	5	94	5	2	16	24
33	50	2	0	30	2	0	74	5	100	3	5	12	21
34	50	2	0	40	2	0	74	5	0	0	1	11	20
35	50	2	0	50	2	0	74	5	72	0	1	11	20

Table 5.3: *Results for experiments using Algorithm 6 using a broad set of parameters.*

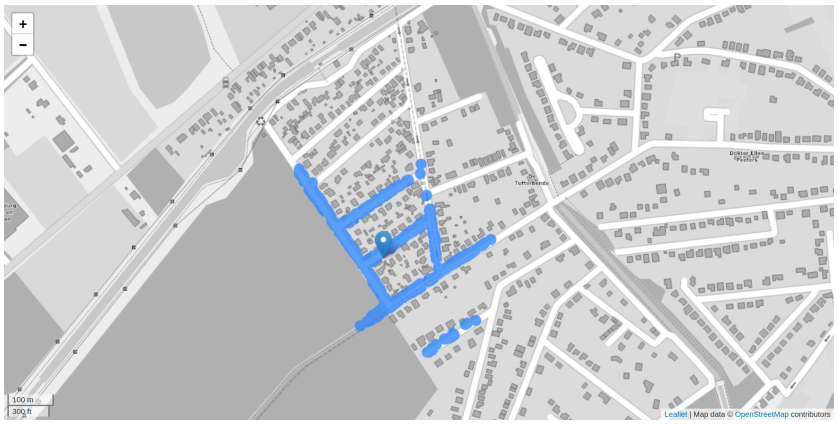


Figure 5.15: *Localisation updates received by the server during Experiment 0.*

Figure 5.16: *Localisation updates received by the server during Experiment 8.*

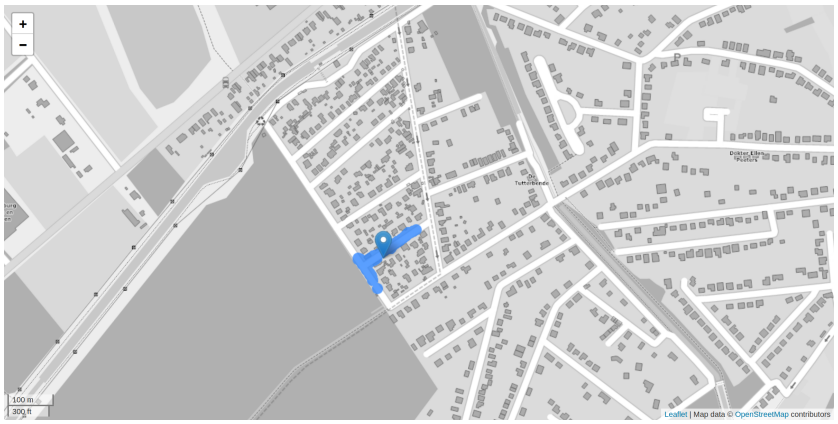


Figure 5.17: *Localisation updates received by the server during Experiment 12.*

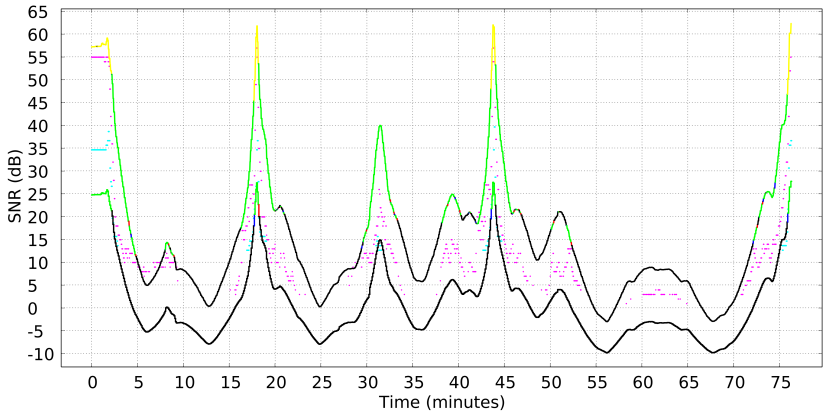


Figure 5.18: *Estimated SNR for IEEE 802.11ah (top line) and IEEE 802.11 b/g/n (bottom line) and received beacons (purple for IEEE 802.11ah and cyan for IEEE 802.11b/g/n) during Experiment 8.*

Using the information obtained in the previous set of experiments, we can try to optimize the different parameters of the algorithm. Experiment 8 is a reasonable candidate to further optimise, since it provides a good trade-off between energy efficiency and application performance. Therefore, the parameter σ_{ah} was varied between 1 and 19 in steps of 2, while the σ_{bgn} parameter was varied from 11 to 19 in steps of 2. A subset of the results are shown in Table 5.4, redundant results were left out. Experiment 82 seems to provide a very good trade-off between performance and efficiency. It provides a larger range than the original Experiment 8, while being comparably efficient. The coverage of Experiment 82 is displayed in Figure 5.19, which compared to that of Experiment 8 (Figure 5.16) indeed shows a slightly better coverage. Furthermore, the experiment achieves a higher amount of updates. Although, this is indeed a better result, it also shows that the algorithm is performing well, even without tuning the parameters.

Figure 5.19: *Localisation updates received by the server during Experiment 82.*

Configuration							Results in %						
	Ah			WiFi			Ah		Wifi			Application	
ID	σ_{Ah}	β_{Ah}	Ω_{Ah}	σ_{bgn}	β_{bgn}	Ω_{bgn}	**Eff**	**On**	**Eff**	**On**	**PL**	**Updates**	**Distance**
76	9	2	0	5	2	0	63	55	48	21	19	42	159
77	9	2	0	7	2	0	64	55	62	16	27	41	154
78	9	2	0	9	2	0	63	56	70	14	22	45	159
79	9	2	0	11	2	0	64	55	78	12	24	45	160
80	9	2	0	13	2	0	64	55	86	10	19	45	154
81	9	2	0	15	2	0	64	56	91	7	23	48	153
82	9	2	0	17	2	0	62	55	93	6	20	49	159
83	9	2	0	19	2	0	65	55	100	5	26	47	148
700	9	2	0	20	2	0	64	55	94	5	14	49	147
701	10	2	0	5	2	0	61	53	47	21	15	42	140
702	10	2	0	7	2	0	59	53	63	16	16	41	148
703	10	2	0	9	2	0	58	53	69	14	15	42	142
704	10	2	0	11	2	0	58	53	79	12	15	42	147
705	10	2	0	13	2	0	59	53	85	10	17	42	143
706	10	2	0	15	2	0	60	53	93	6	16	48	148
707	10	2	0	17	2	0	60	53	100	5	14	47	143
708	10	2	0	19	2	0	54	51	94	5	12	44	147
8	10	2	0	20	2	0	61	53	94	5	17	47	146
84	11	2	0	5	2	0	55	51	48	21	22	38	143
85	11	2	0	7	2	0	55	51	62	16	20	39	143
86	11	2	0	9	2	0	54	51	70	14	25	40	147
87	11	2	0	11	2	0	55	51	78	12	22	40	143
88	11	2	0	13	2	0	54	51	85	10	21	41	145
89	11	2	0	15	2	0	54	51	91	8	20	44	142
90	11	2	0	17	2	0	56	51	93	6	22	45	137
91	11	2	0	19	2	0	55	51	100	5	25	44	142

Table 5.4: *Results for experiments using Algorithm 6, using more fine-tuned parameters based on the results of Table 5.3.*

With Increased Offset Before Disconnect

This section examines the influence of the Ω parameter. This parameter controls the SNR at which the algorithm advices to disconnect. When this parameter (an offset) is increased, the SNR at which the device will disconnect becomes lower. The other parameters (i.e. σ) are those of Experiment 8 and 82. Table 5.5 lists the results for these experiments. Comparing the experiments based on Experiment 8, it is clear that the application performance is very similar (i.e. within the expected variation of different runs). However, the efficiency improved, especially in the case of Experiment 804, where the efficiency of IEEE 802.11ah is increased by 5 percent-point and the efficiency of IEEE 802.11b/g/n is improved by 3 percent-point. There is no clear improvement when comparing the coverage of Experiment 804 (Figure 5.20) with Experiment 82 (Figure 5.19). In the case of the experiments based on Experiment 8, only the efficiency of IEEE 802.11b/g/n is improved while still achieving very similar application performance. But, in this case there is clearly an improvement in the density of the coverage between Experiment 803 (Figure 5.21) and Experiment 8 (Figure 5.16). A very important observation is that in both cases the actual time that the radio is turned on remained constant. Therefore, we can conclude that (slightly) increasing the Ω parameter improves the efficiency of the algorithm without affecting the power usage.

Configuration							**Results in %**						
	Ah			**WiFi**			**Ah**		**Wifi**			**Application**	
ID	σ_{Ah}	β_{Ah}	Ω_{Ah}	σ_{bgn}	β_{bgn}	Ω_{bgn}	**Eff**	**On**	**Eff**	**On**	**PL**	**Updates**	**Distance**
82	9	2	0	17	2	0	62	55	93	6	20	49	159
801	9	2	1	17	2	1	63	55	95	6	14	48	154
804	9	2	2	17	2	2	67	55	96	6	16	49	153
806	9	2	3	17	2	3	64	55	96	6	16	49	147
8	10	2	0	20	2	0	61	53	94	5	17	47	146
800	10	2	1	20	2	1	63	53	100	5	14	47	143
803	10	2	2	20	2	2	61	53	100	5	14	48	147
805	10	2	3	20	2	3	61	53	100	5	13	48	147

Table 5.5: *Results for experiments using Algorithm 6, where the Ω parameter is tuned.*

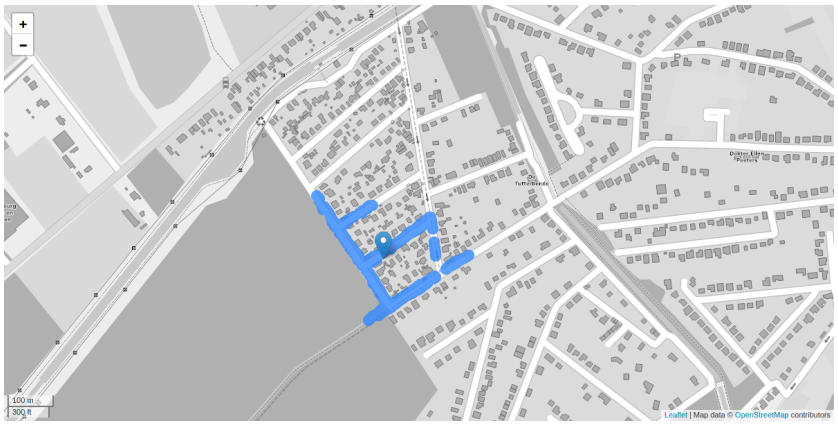


Figure 5.20: *Localisation updates received by the server during Experiment 804.*

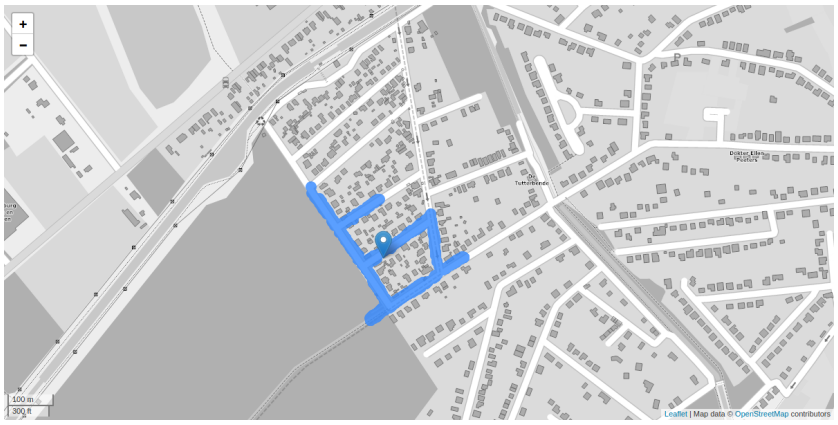


Figure 5.21: *Localisation updates received by the server during Experiment 803.*

With Increased Number of Missed Beacons

The experiments presented in this section (Table 5.6) are again based on Experiment 8 and 82. This time the β parameter was changed in order to study the influence of tuning the amount of missed beacons before the algorithm disconnects. Note that, the previous experiments would disconnect after missing two beacons for the reasons explained in Section 5.2.2. Therefore, it is interesting to have a look at what would happen if the algorithm disconnects after missing one beacon. This is the purpose of Experiment 811 and 810, based on respectively Experiment 82 and 8. Compared to their relevant experiments, both have the worst results on all metrics. The efficiency of IEEE 802.11ah has decreased by at least 8 percent-point while the range and number of updates also decreased. Therefore, we can already conclude

that it is better to allow for more than one missed beacon. Looking at the other experiment, it is clear that the efficiency of IEEE 802.11ah increases when increasing β, while the application performance stays almost constant. The coverage for Experiment 815 (Figure 5.22) and 814 (Figure 5.23) do not show an obvious difference to the experiments on which they are based.

Configuration							Results in %						
	Ah			WiFi			Ah		Wifi			Application	
ID	σ_{Ah}	β_{Ah}	Ω_{Ah}	σ_{bgn}	β_{bgn}	Ω_{bgn}	**Eff**	**On**	**Eff**	**On**	**PL**	**Updates**	**Distance**
811	9	1	0	17	1	0	54	55	91	6	12	46	147
82	9	2	0	17	2	0	62	55	93	6	20	49	159
813	9	3	0	17	3	0	67	55	93	6	15	49	154
815	9	4	0	17	4	0	70	56	93	6	16	49	154
810	10	1	0	20	1	0	52	53	93	5	10	46	142
8	10	2	0	20	2	0	61	53	94	5	17	47	146
812	10	3	0	20	3	0	63	53	94	5	13	47	143
814	10	4	0	20	4	0	64	53	94	5	14	47	148

Table 5.6: *Results for experiments using Algorithm 6, where the β parameter is tuned.*

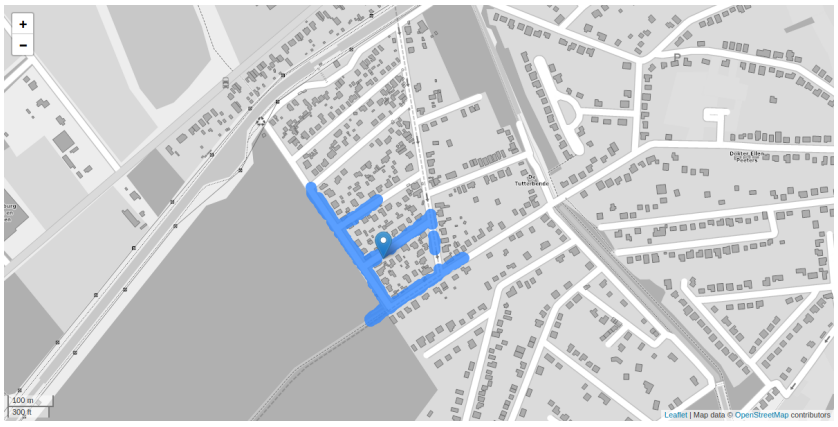


Figure 5.22: *Localisation updates received by the server during Experiment 815.*

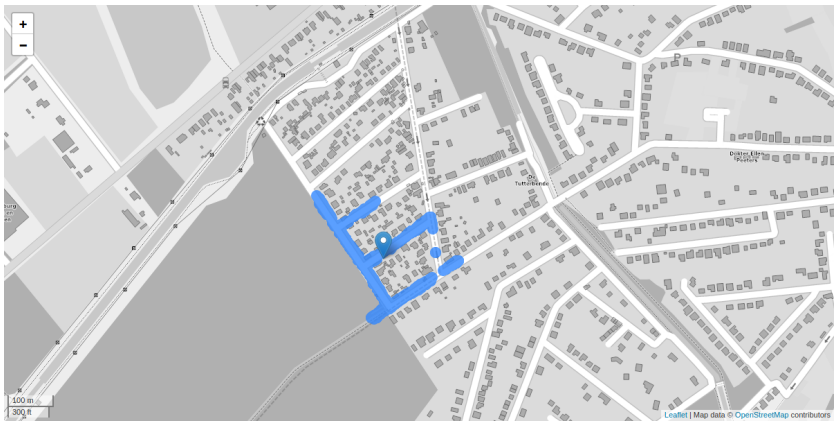


Figure 5.23: *Localisation updates received by the server during Experiment 814.*

With Optimal Fitted Propagation Model

The goal of the last set of experiments is to find out whether using an optimal propagation model instead of a more default, already established propagation model makes any difference. Therefore, the experiments performed in this section use the optimal fitted propagation model defined in Section 4.2.4. Changing the propagation model used by the handover algorithm changes the value of the optimal required SNR (i.e. the σ parameter). Therefore, this section again uses a broad set of parameters instead of building upon the previous best experiments. To be more precisely, both the σ_{ah} and σ_{bgn} parameters vary between 0 and 30. Table 5.7 provides an overview of the conducted experiments. Experiment 630 provides a good trade-off between efficiency and application performance when looking at an experiment for which the radio is not turned on for more than 55 %. Compared to Experiment 8 (of which the IEEE 802.11ah radio is turned on for 55 % and the IEEE 802.11b/g/n radio is turned on for 6 %), Experiment 630 achieves slightly more coverage and a higher amount of server updates. However, the radio is turned off for a shorter amount of time (respectively 52 % and 6 %). Furthermore, the efficiency of IEEE 802.11ah is increased by 15 percent-point. Therefore, the device is connected for a longer time while the radio is turned on for the same amount of time. The coverage of Experiment 630 (Figure 5.24) is clearly larger and more dense than that of Experiment 82 (Figure 5.19).

If there is an use case that requires the radio to be used even more efficiently, the parameters of Experiment 606 could be used. The trade-off is of course that the application performance is reduced. In this experiment, the IEEE 802.11ah radio is turned on for 35 % percent of the time, while the IEEE 802.11b/g/n radio is turned on for 5 % of the time. Experiment 14 has comparable radio on times, respectively 35 % and 5 %, but uses a non-optimised propagation model. There is a big difference in application performance between both experiments, Experiment 606 achieves a range of 118 m whereas Experiment 15 only achieves 57 m. Similarly, Experiment 606 results in 16 percent-point more location updates. This major difference is also visible when comparing the coverage figures for both experiments: Figure 5.25 for Experiment 606 and Figure 5.26 for Experiment 14. This can be explained by the great improvement of 46 % in efficiency for the IEEE 802.11ah technology seen in Experiment 606.

Figure 5.27 provides a visualisation of the working of the algorithm during Experiment 606, including the estimated SNR and the SNR of the received beacons. Compared to the same plot for Experiment 14 (Figure 5.28) it is clear that the estimated SNR is much closer to the actual SNR.

From both comparisons, we can conclude that an optimal propagation model improves the efficiency and application performance. Particularly in situations where the power usage (i.e. radio on time) is very constrained, we see a major improvement of the application performance and efficiency.

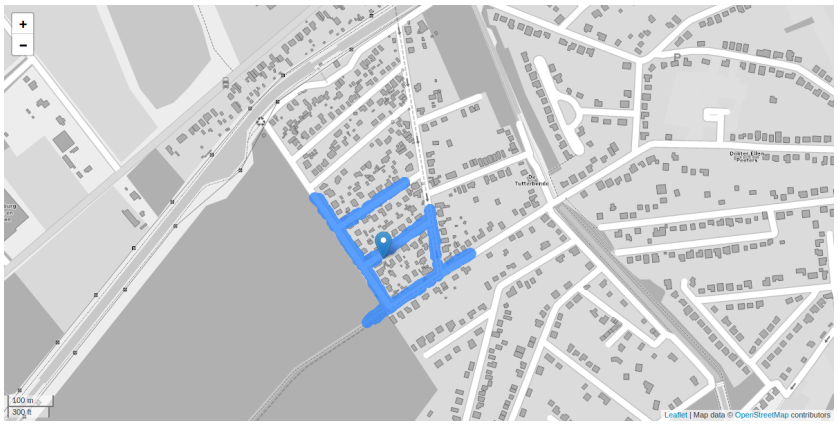


Figure 5.24: *Localisation updates received by the server during Experiment 630.*

Configuration							Results in %						
	Ah			WiFi			Ah		Wifi		Application		
ID	σ_{Ah}	β_{Ah}	Ω_{Ah}	σ_{bgn}	β_{bgn}	Ω_{bgn}	Eff	On	Eff	On	PL	Updates	Distance
82	9	2	0	17	2	0	62	55	93	6	20	49	159
14	20	2	0	20	2	0	34	30	94	5	5	29	57
600	0	2	0	0	2	0	51	94	24	43	22	46	179
601	0	2	0	10	2	0	50	94	74	13	22	47	186
602	0	2	0	20	2	0	50	94	93	5	20	52	179
603	0	2	0	30	2	0	51	94	100	3	19	54	186
624	3	2	0	0	2	0	60	80	24	43	24	46	176
625	3	2	0	10	2	0	59	80	74	13	25	48	186
626	3	2	0	20	2	0	58	79	93	5	20	51	181
627	3	2	0	30	2	0	58	79	100	3	20	50	179
628	6	2	0	0	2	0	75	52	23	43	21	45	164
629	6	2	0	10	2	0	74	52	74	13	20	45	154
630	6	2	0	20	2	0	77	52	93	5	19	51	161
631	6	2	0	30	2	0	77	52	100	1	18	50	162
632	9	2	0	0	2	0	81	39	23	43	23	41	137
633	9	2	0	10	2	0	79	39	73	13	21	41	133
634	9	2	0	20	2	0	81	40	94	5	20	48	134
635	9	2	0	30	2	0	80	39	100	3	20	47	134
604	10	2	0	0	2	0	80	35	23	43	15	38	118
605	10	2	0	10	2	0	79	35	73	13	14	40	119
606	10	2	0	20	2	0	80	35	94	5	13	45	118
607	10	2	0	30	2	0	80	35	100	3	11	45	118
636	12	2	0	0	2	0	87	24	23	43	13	33	104
637	12	2	0	10	2	0	87	24	74	13	22	35	107
638	12	2	0	20	2	0	88	24	93	5	8	40	105
639	12	2	0	30	2	0	88	24	100	3	20	41	103
640	15	2	0	0	2	0	92	15	24	43	22	27	71
641	15	2	0	10	2	0	92	15	74	13	16	28	72
642	15	2	0	20	2	0	92	15	93	5	8	33	70
643	15	2	0	30	2	0	93	15	99	0	15	34	71
608	20	2	0	0	2	0	92	10	24	43	10	24	57
609	20	2	0	10	2	0	93	10	74	13	13	24	54
610	20	2	0	20	2	0	95	10	93	5	4	26	47
611	20	2	0	30	2	0	95	10	100	1	3	27	47
612	30	2	0	0	2	0	100	4	23	43	10	23	58
613	30	2	0	10	2	0	99	4	73	13	10	22	53
614	30	2	0	20	2	0	96	4	94	5	1	16	24
615	30	2	0	30	2	0	97	5	99	0	8	16	31

Table 5.7: *Results for experiments using Algorithm 6, with optimised propagation models.*

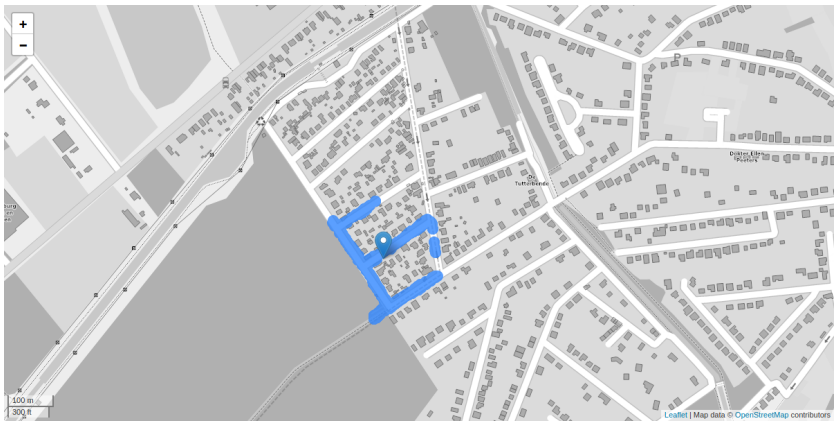


Figure 5.25: *Localisation updates received by the server during Experiment 606.*

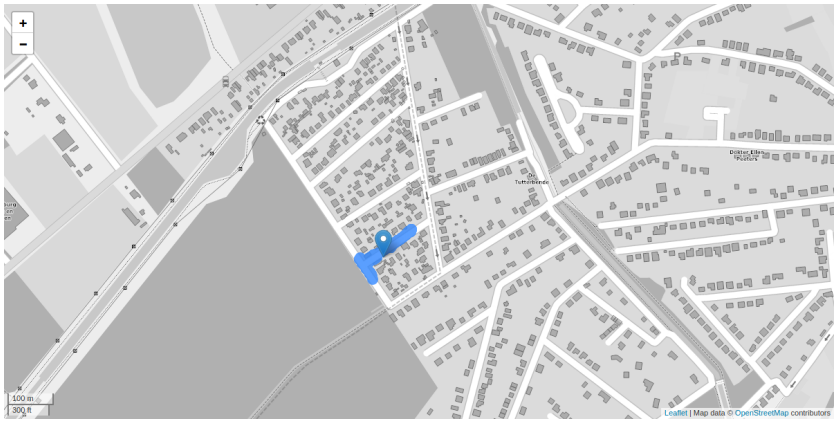


Figure 5.26: *Localisation updates received by the server during Experiment 14.*

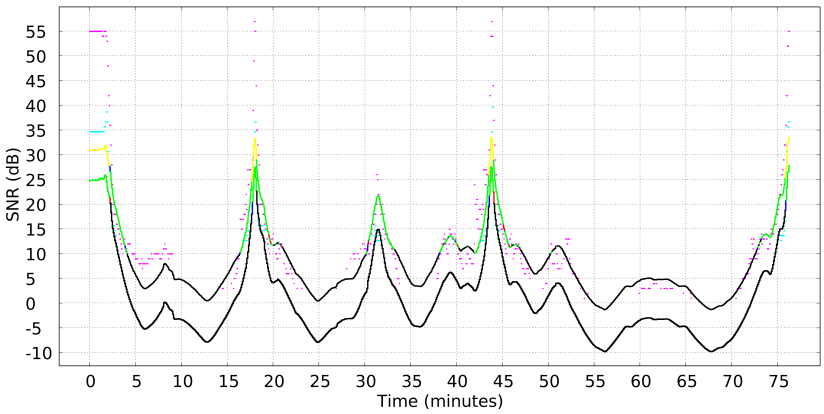


Figure 5.27: *Estimated SNR for IEEE 802.11ah (top line) and IEEE 802.11 b/g/n (bottom line) and received beacons (purple for IEEE 802.11ah and cyan for IEEE 802.11b/g/n) during Experiment 606.*

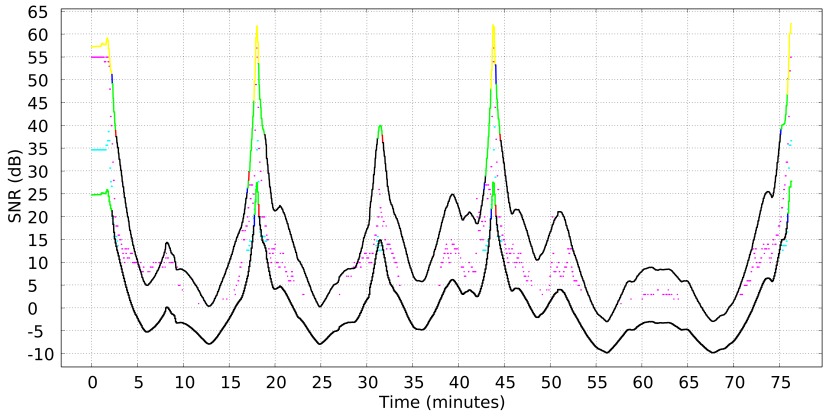


Figure 5.28: *Estimated SNR for IEEE 802.11ah (top line) and IEEE 802.11 b/g/n (bottom line) and received beacons (purple for IEEE 802.11ah and cyan for IEEE 802.11b/g/n) during Experiment 14.*

5.1.3 Overview

This section covers a final comparison between the most interesting experiments. To start Figure 5.29 and Figure 5.30 display the experiments for the four type of algorithms in the case where IEEE 802.11ah radio is turned on for about 55 % of the time. Note that the included Experiment 508 actually uses the radio for 100 % of the time, because it uses the continuously listening for beacons based approach. In Figure 5.29 it is clear that the location-based approach greatly reduces the radio-on time, while still achieving similar performance of the application.

The same visualisations are shown in Figure 5.31 and Figure 5.32 but then for the case where the radio is turned on for about 35 % of the time. Again the location-based approach outperforms the beaconing-based approach.

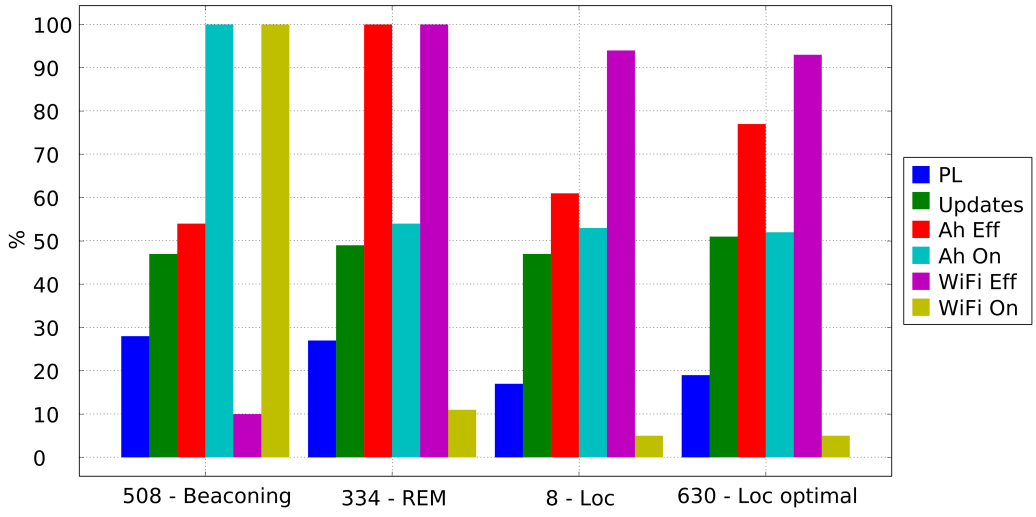
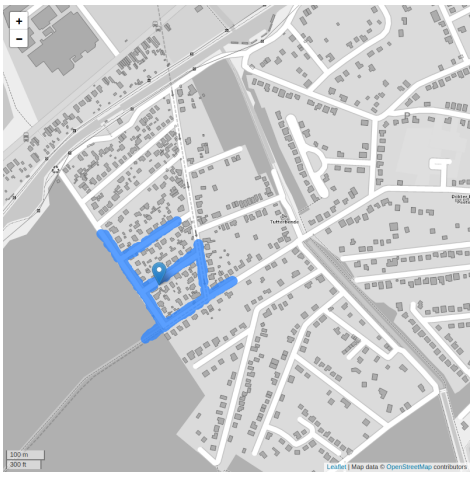


Figure 5.29: *Performance overview of the experiments where the IEEE 802.11ah radio is turned on for about of the time. Experiment 508 uses listening for beacons, Experiment 334 uses the REM based approach, Experiment 8 uses the location-based algorithm without optimal parameters and finally Experiment 630 uses the location-based algorithm with an optimal propagation model. The blue bar indicates the observed packet loss, the green bar indicates the ratio of successful updates, the red and purple bars represent the efficiency of respectively IEEE 802.11ah and IEEE 802.11b/g/n, finally the light blue and yellow bars indicate for how long the respectively radio were turned on.*

(a) *Experiment 508 (beaconing)*

(b) *Experiment 334 (REM)*

(c) *Experiment 8 (location-based, non-optimised)*



(d) *Experiment 630 (location-based, optimised)*

Figure 5.30: *Overview of the coverage of the experiments where the IEEE 802.11ah radio is turned on for about 55 % of the time.*

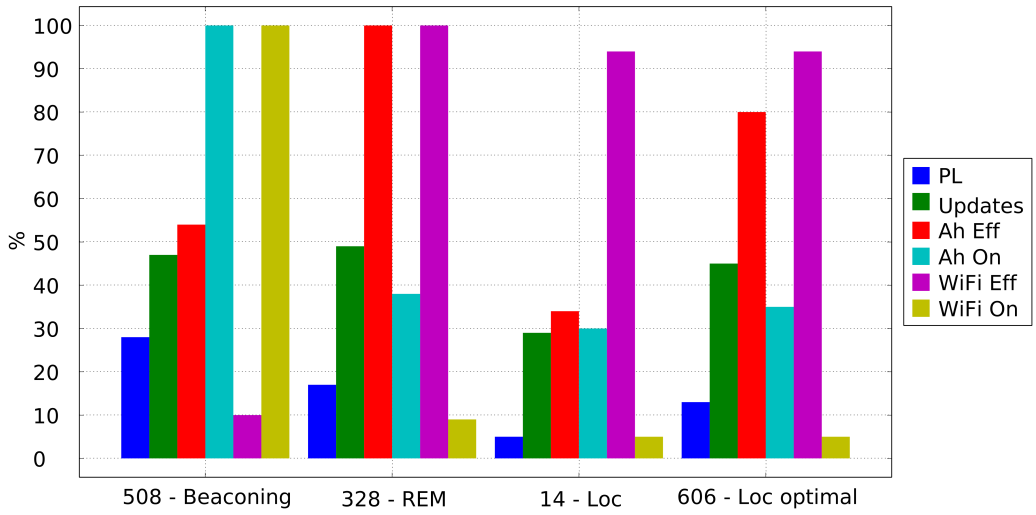
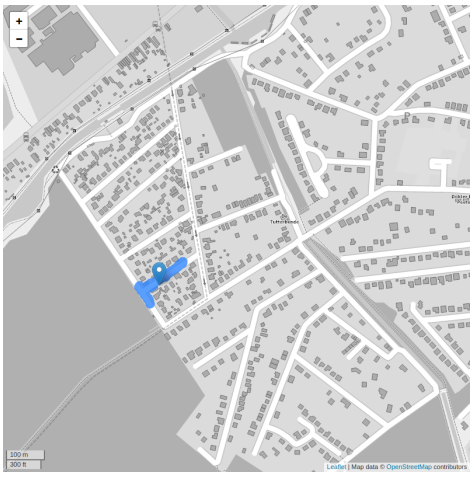


Figure 5.31: *Performance overview of the experiments where the IEEE 802.11ah radio is turned on for about 35 % of the time. Experiment 508 uses listening for beacons, Experiment 328 uses the REM based approach, Experiment 14 uses the location-based algorithm without optimal parameters and finally Experiment 605 uses the location-based algorithm with an optimal propagation model. The blue bar indicates the observed packet loss, the green bar indicates the ratio of successful updates, the red and purple bars represent the efficiency of respectively IEEE 802.11ah and IEEE 802.11b/g/n, finally the light blue and yellow bars indicate for how long the respectively radio were turned on.*

(a) *Experiment 508 (beaconing)*

(b) *Experiment 328 (REM)*



(c) *Experiment 14 (location-based, non-optimised)*

(d) *Experiment 606 (location-based optimised)*

Figure 5.32: *Overview of the coverage of the experiments where then IEEE 802.11ah radio is turned on for about 35 % of the time.*

5.2 Hardware-based Experiments

The aim of these experiments is to illustrate the characteristics of the technology and to test the feasibility of running the example application on the prototype hardware. This hardware prototype is using the exact same hardware as discussed in Section 4.1.1. The laptop runs the *Application Server* discussed in Section 4.2.3, while the RPI runs the *Application Client* as described in Section 4.2.3.

Firstly, the application is tested when running solely on the IEEE 802.11 b/g/n network. Then, the same test is repeated but using only the IEEE 802.11ah network. In the final experiment, both technologies are used, however, the IEEE 802.11b/g/n network is turned off three times. Therefore, the performance of the application can be studied when the network has to perform a vertical handover. In all three experiments, both devices (i.e. station and AP) are placed at a distance of approximately three meters of each other.

5.2.1 Using IEEE 802.11b/g/n

This section describes the results for the hardware-based experiment when only the IEEE 802.11b/g/n technology is used. The experiment lasted about ten minutes. The observed distance to the AP during the experiment is shown in Figure 5.33. Although the position of the hardware is fixed, the figure clearly displays some variation in the distance to the AP, due to the error of the used GPS device. The application latency is shown in Figure 5.34. The minimum latency is 3 ms, the average latency is 5.69 ms and the maximal latency is 81 ms. This gives a good indication of the average latency achievable using IEEE 802.11b/g/n in an ideal situation. The server is able to receive 100 % of the location updates from the client. Therefore, we can conclude that it is feasible to run the example application using the IEEE 802.11b/g/n network.

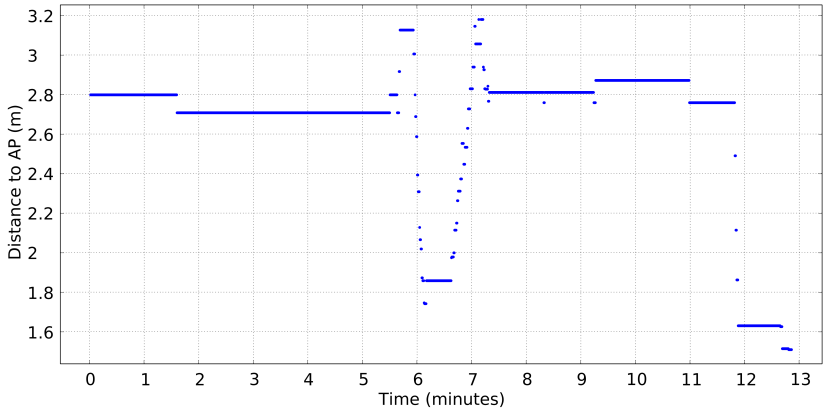


Figure 5.33: *Distance to the AP as observed by the GPS device during the WiFi b/g/n benchmark.*

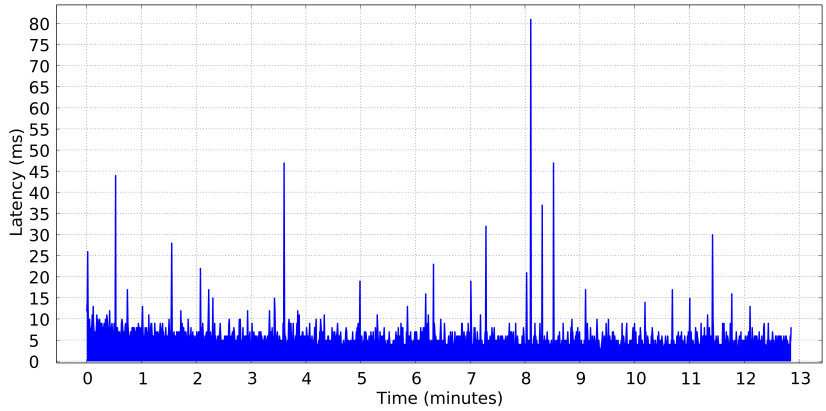


Figure 5.34: *Latency of GPS updates during WiFi b/g/n benchmark.*

5.2.2 Using IEEE 802.11ah

In this section the benchmark results for IEEE 802.11ah are discussed. Again, the experiment lasted for about ten minutes. Figure 5.35 depicts the observed distance to the AP during the experiment. Similar to the previous benchmark experiment, the RPI was held at a fixed position, although the GPS observed some movements. The error (i.e. deviation of the distance) in both benchmark experiments, (Figure 5.33 and Figure 5.35) is similar and stays below the theoretical error of 10 m. For IEEE 802.11ah more packet loss is observed (Section 4.1.3).

The minimal latency during the experiment is 43 ms, the average latency is 52.86 ms, finally the maximum latency is 84 ms. As Figure 5.36 clearly

shows this maximal latency is an outlier. Furthermore this figure indicates that there are gaps between the location updates. Compared to IEEE 802.11 b/g/n, the latency is higher. To give an indication, the *minimal* latency in IEEE 802.11ah (36 ms) is equal to the *maximal* latency in IEEE 802.11b/g/n. Again, 100 % of the location updates were correctly received by the server. Therefore, this experiment makes clear it is feasible to run the example application using IEEE 802.11ah.

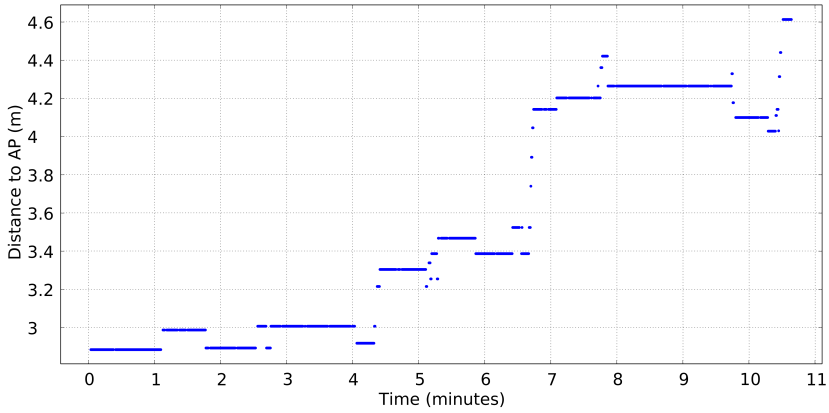


Figure 5.35: *Distance to the AP as observed by the GPS device during the WiFi b/g/n benchmark.*

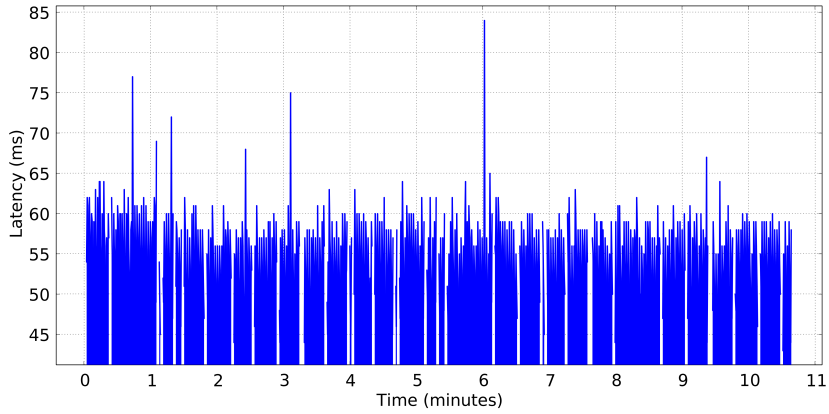


Figure 5.36: *Latency of GPS updates during WiFi HaLow benchmark.*

5.2.3 Using IEEE 802.11ah and IEEE 802.11b/g/n

The experiment discussed in this section, uses both IEEE 802.11ah and IEEE 802.11b/g/n. The prototype devices are placed at a fixed location and a few

meters inbetween the station and the AP. Note that the (non-optimised) location-based handover algorithm was used. It was configured to allow up to two missed beacons (i.e. the β parameter), to prevent the algorithm from disconnecting to early. In order to force some handovers, the IEEE 802.11 b/g/n was turned off three times. The goal of this experiment is to test whether the handover works and whether it is feasible to run our example application on top of it. Figure 5.37 shows the distance to the AP, again there is some variation due to the GPS error. The working of the algorithm is visualised in Figure 5.38. This shows that the algorithm is able to virtually instantaneously switch between the two networks whenever the IEEE 802.11 b/g/n network is turned off. Furthermore, the device switches back to IEEE 802.11b/g/n as soon as that network is back online. Moreover, it is visible that by changing the β parameter to two, the problem of the algorithm disconnecting because of missing a single beacon, is solved. From the latency of the application (Figure 5.38) it is visible that some location updates are lost, especially when the IEEE 802.11b/g/n network is disabled and the device has to handover. However, 96 % of the location updates are correctly received by the server. This is 4 % less than in the situation of using only one technology. As expected, the application is unable to transmit location updates during a handover. From these observations we can conclude that the application is resilient to the device switching between the networks and that it is feasible to run the handover algorithm on the prototype device. Furthermore, the application performance is not significantly reduced when the device has to perform handovers from one technology to another.

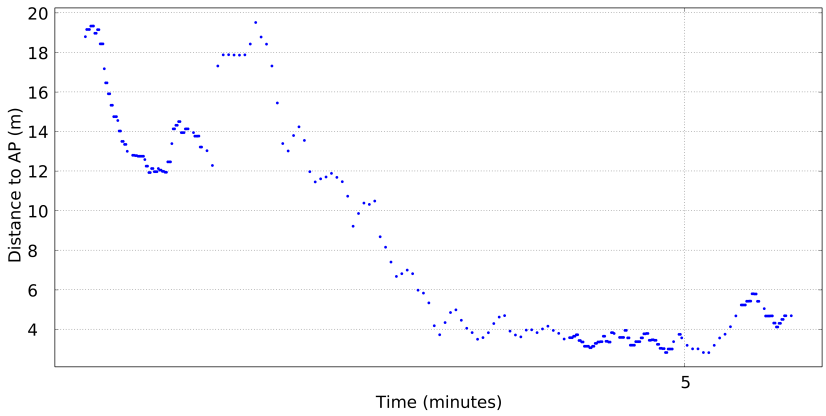


Figure 5.37: *Distance to the AP as observed by the GPS device during the benchmark experiment using both IEEE 802.11ah and IEEE 802.11b/g/n.*

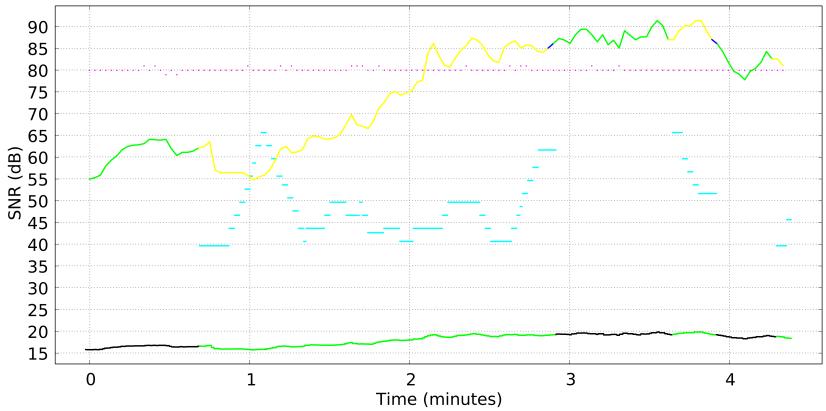


Figure 5.38: *Estimated SNR for IEEE 802.11ah (top line) and IEEE 802.11 b/g/n (bottom line) and received beacons (purple for IEEE 802.11ah and cyan for IEEE 802.11b/g/n) during the benchmark experiment using both IEEE 802.11ah and IEEE 802.11b/g/n.*

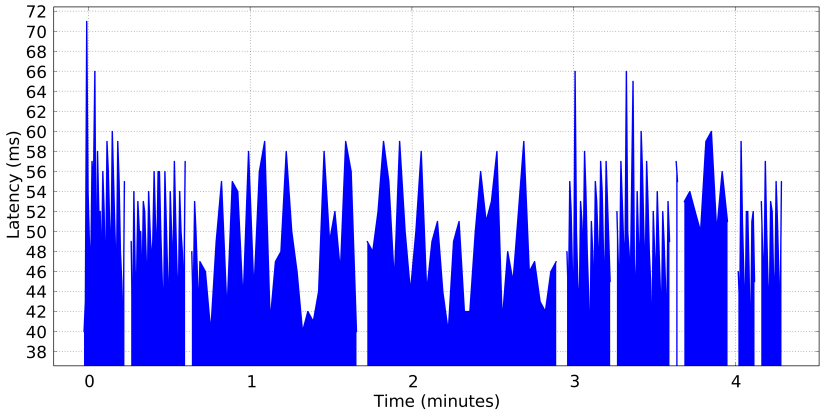


Figure 5.39: *Latency of GPS updates during the benchmark experiment using both IEEE 802.11ah and IEEE 802.11b/g/n.*

Chapter 6

Conclusion

In this thesis we studied the feasibility and performance of using location-based vertical handover between the novel IEEE 802.11ah and the ubiquitous IEEE 802.11b/g/n. Due to the novelty of IEEE 802.11ah, this was not yet covered by existing research. Besides the performance of the algorithm, our research also focused on the performance of a typical IoT application running on top of the algorithm. Using the hardware setup, we measured the RSSI and packet loss inside the measurement area. These comprehensive measurements were used to build an emulator for the physical layer of both IEEE 802.11ah and IEEE 802.11b/g/n. With this emulator we performed a vast number of experiments, more experiments than would be possible with the prototype. In addition to the location-based handover algorithm, we also implemented two reference algorithms. The first one, based on continuously listening for beacon, is the default algorithm in many discovery implementations. This system is unfeasible because of its power usage. The second algorithm uses an REM and therefore requires extensive amounts of work resulting in high costs. Therefore, our proposed algorithm is better suited for IoT cases requiring power efficient devices. Our experiments show that the location-based handover algorithm is able to achieve similar application performance as the continuously polling based algorithm, while using roughly half of the energy for IEEE 802.11ah and even a reduction of 95 % percent-point in the case of IEEE 802.11b/g/n. This was the case where no fully optimised parameters where used in the algorithm. Compared to the

REM based approach, our non-optimised algorithm shows comparable results. However, creating such an REM is almost always unfeasible, because it requires a lot of effort and involves high costs. Furthermore, an REM gets stale resulting in a high returning cost for the operator of the network. Tuning the parameters improves the performance of the algorithm even further, both on its efficiency and application performance. In addition, we saw that using an optimal fitted propagation model improves the efficiency and application performance too. Therefore, we conclude that using location-based discovery and handover is indeed feasible for IEEE 802.11ah, it improves the energy efficiency while still being able to run well performing applications. We also concluded that the performing vertical handovers between IEEE 802.11ah and IEEE 802.11b/g/n on a physical hardware prototype is feasible. Moreover, an application running on the physical device can be adapted in such a way that it is not sensitive to interruptions of the network technology caused by the device switching between network technologies.

As a result of this thesis, the following two papers were submitted for publication:

- **Tobia De Koninck**, Serena Santi, Jeroen Famaey, and Filip Lemic. Experimental Validation of IEEE802.11ah Propagation Models in Heterogeneous Smarty City Environments. In *2020 IEEE Global Communications Conference, Taipei, Taiwan, December 2020*

- Serena Santi, **Tobia De Koninck**, Glenn Daneels, Filip Lemic, and Jeroen Famaey. Performance Evaluation of Location-aware Discovery and Vertical Handover Mechanism in IEEE 802.11ah networks. In *IEEE Access*

Bibliography

[1] Debasis Bandyopadhyay and Jaydip Sen. Internet of Things: Applications and Challenges in Technology and Standardization. *Wireless Personal Communications*, 58:49–69, May 2011. DOI: 10.1007/s11277-011-0288-5.

[2] Usman Raza, Parag Kulkarni, and Mahesh Sooriyabandara. Low Power Wide Area Networks: An Overview. *IEEE Communications Surveys & Tutorials*, 19(2):855–873, 2017.

[3] IEEE Standard for Information technology–Telecommunications and information exchange between systems - Local and metropolitan area networks–Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 2: Sub 1 GHz License Exempt Operation. *IEEE Std 802.11ah-2016 (Amendment to IEEE Std 802.11-2016, as amended by IEEE Std 802.11ai-2016)*:1–594, 2017.

[4] Stefan Aust, Venkatesha Prasad, and Ignas Niemegeers. IEEE 802.11ah: Advantages in standards and further challenges for sub 1 GHz Wi-Fi. In pages 6885–6889, June 2012. ISBN: 978-1-4577-2052-9. DOI: 10.1109/ICC.2012.6364903.

[5] Amina Sljivo, Dwight Kerkhove, Le Tian, Jeroen Famaey, Adrian Munteanu, Ingrid Moerman, Jeroen Hoebeke, and Eli De Poorter. Performance Evaluation of IEEE 802.11ah Networks With High-Throughput Bidirectional Traffic. *Sensors (Basel, Switzerland)*, 18, 2018.

[6] Evgeny Khorov, Andrey Lyakhov, Alexander Krotov, and Andrey Guschin. A survey on IEEE 802.11ah: An enabling networking technology for

smart cities. *Computer Communications*, 58:53–69, 2015. ISSN: 0140-3664. DOI: https://doi.org/10.1016/j.comcom.2014.08.008. URL: http://www.sciencedirect.com/science/article/pii/S0140366414002989. Special Issue on Networking and Communications for Smart Cities.

[7] Le Tian, Jeroen Famaey, and Steven Latré. Evaluation of the IEEE 802.11ah Restricted Access Window Mechanism for dense IoT networks. In June 2016. DOI: 10.1109/WoWMoM.2016.7523502.

[8] Tobia De Koninck, Serena Santi, Jeroen Famaey, and Filip Lemic. Experimental Validation of IEEE802.11ah Propagation Models in Heterogeneous Smarty City Environments. In *2020 IEEE Global Communications Conference*, Taipei, Taiwan, December 2020.

[9] Jeroen Famaey, Rafael Berkvens, Glenn Ergeerts, Eli De Poorter, Floris Van den Abeele, Tomas Bolckmans, Jeroen Hoebeke, and Maarten Weyn. Flexible Multimodal Sub-Gigahertz Communication for Heterogeneous Internet of Things Applications. *IEEE Communications Magazine*, 56(7):146–153, 2018.

[10] Kais Mekki, Eddy Bajic, Frederic Chaxel, and Fernand Meyer. A comparative study of LPWAN technologies for large-scale IoT deployment. *ICT Express*, 5(1):1–7, 2019. ISSN: 2405-9595. DOI: https://doi.org/10.1016/j.icte.2017.12.005. URL: http://www.sciencedirect.com/science/article/pii/S2405959517302953.

[11] Filip Lemic, Arash Behboodi, Jeroen Famaey, and Rudolf Mathar. Location-based Discovery and Vertical Handover in Heterogeneous Low-Power Wide-Area Networks. PP:1–1, August 2019. DOI: 10.1109/JIOT.2019.2935804.

[12] Filip Lemic, Vlado Handziski, Giuseppe Caso, Luca De Nardis, and Adam Wolisz. Enriched Training Database for Improving the WiFi RSSI-based Indoor Fingerprinting Performance. In January 2016. DOI: 10.1109/CCNC.2016.7444904.

[13] Serena Santi, Filip Lemic, and Jeroen Famaey. On the Feasibility of Location-based Discovery and Vertical Handover in IEEE 802.11ah.

In *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6, 2020.

[14] Mohammed Alrifaie, Norharyati Harum, Mohd Fairuz Iskandar Othman, Irda Roslan, and Methaq Shyaa. Vehicle Detection and Tracking System IoT based: A Review. *International Journal of Engineering and Technology*, 5:1237–1241, August 2018.

[15] Mohamed Ben-Daya, Elkafi Hassini, and Zied Bahroun. Internet of things and supply chain management: a literature review. *International Journal of Production Research*, 57(15-16):4719–4742, 2019. DOI: 10.1080/00207543.2017.1402140. eprint: https://doi.org/10.1080/00207543.2017.1402140. URL: https://doi.org/10.1080/00207543.2017.1402140.

[16] Mehdi Amirinasab, Dr Shamshirband Phd, A. Chronopoulos, Amir Mosavi, and Narjes Nabipour. Energy-Efficient Method for Wireless Sensor Networks Low-Power Radio Operation in Internet of Things. *Electronics*, 9, February 2020. DOI: 10.3390/electronics9020320.

[17] Joseph Finnegan and Stephen Brown. A Comparative Survey of LPWA Networking, 2018. arXiv: 1802.04222 [cs.NI].

[18] Brecht Reynders, Wannes Meert, and S. Pollin. Range and coexistence analysis of long range unlicensed communication. In *2016 23rd International Conference on Telecommunications (ICT)*, pages 1–6, 2016.

[19] Ben Bellekens, Le Tian, Pepijn Boer, Maarten Weyn, and Jeroen Famaey. Outdoor IEEE 802.11ah Range Characterization Using Validated Propagation Models. In December 2017. DOI: 10.1109/GLOCOM.2017.8254515.

[20] Toni Adame, Albert Bel, Boris Bellalta, Jaume Barcelo, and Miquel Oliver. Ieee 802.11ah: the wifi approach for m2m communications. *IEEE Wireless Communications*, February 2014. DOI: 10.1109/MWC.2014.7000982.

[21] Le Tian, Amina Seferagic, Serena Santi, Eli De Poorter, Jeroen Hoebeke, and Jeroen Famaey. Extension of the IEEE 802.11ah ns-3 Simulation Module. In *Proceedings of the 10th Workshop on ns-3*, WNS3 '18, pages 53–60. Association for Computing Machinery, 2018.

[22] German Castignani, Andrés Arcia-Moret, and Nicolas Montavont. A study of the discovery process in 802.11 networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 15:25–36, March 2011. DOI: 10.1145/1978622.1978626.

[23] Rocco Taranto, Srikar Muppirisetty, Ronald Raulefs, Dirk Slock, Tommy Svensson, and Henk Wymeersch. Location-Aware Communications for 5G Networks: How location information can improve scalability, latency, and robustness of 5G. *IEEE Signal Processing Magazine*, 31(6):102–112, 2014.

[24] Bart Moons, Abdulkadir Karaağaç, Eli De Poorter, and Jeroen Hoebeke. Efficient Vertical Handover in Heterogeneous Low-Power Wide-Area Networks. *IEEE Internet of Things Journal*, 7(3):1960–1973, 2020.

[25] Filip Lemic, Arash Behboodi, Vlado Handziski, Anatolij Zubow, and Adam Wolisz. Location-Based Decision-Making Mechanism for Device-to-Device Link Establishment. In September 2017. DOI: 10.1109/VTCFall.2017.8288153.

[26] Bendaoud Fayssal, Marwen Abdennebi, and Fedoua Didi. Network Selection in Wireless Heterogeneous Networks: a Survey. *Journal of Telecommunications and Information Technology*, 4:64–74, January 2019. DOI: 10.26636/jtit.2018.126218.

[27] Changhua Pei, Zhi Wang, Youjian Zhao, Zihan Wang, Yuan Meng, Dan Pei, Yuanquan Peng, Wenliang Tang, and Xiaodong Qu. Why it takes so long to connect to a WiFi access point. In pages 1–9, May 2017. DOI: 10.1109/INFOCOM.2017.8057164.

[28] Giuseppe Caso, Luca De Nardis, Filip Lemic, Vlado Handziski, Adam Wolisz, and Maria-Gabriella Di Benedetto. ViFi: Virtual Fingerprinting WiFi-Based Indoor Positioning via Multi-Wall Multi-Floor Propagation Model. *IEEE Transactions on Mobile Computing*, 19(6):1478–1491, 2020.

[29] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, September 1975. ISSN: 0001-0782. DOI: 10.1145/361002.361007. URL: https://doi.org/10.1145/361002.361007.

[30] Tapan Sarkar, Zhong Ji, Kyungjung Kim, Abdellatif Medouri, and Magdalena Palma. A survey of various propagation models for mobile communication. *Antennas and Propagation Magazine, IEEE*, 45:51–82, July 2003. DOI: 10.1109/MAP.2003.1232163.

[31] COST Action 231, European Commission, and and Exploitation of Research DGXIII Telecommunications. *Digital mobile radio towards future generation systems*. European Commission, 1999.

[32] Recommendation ITU-R P.1411-10 Propagation data and prediction methods for the planning of short-range outdoor radiocommunication systems and radio local area networks, 2019.

[33] Ali Hazmi, Jukka Rinne, and Mikko Valkama. Feasibility study of IEEE 802.11ah radio technology for IoT and M2M use cases. In *2012 IEEE Globecom Workshops*, pages 1687–1692, December 2012. ISBN: 978-1-4673-4942-0. DOI: 10.1109/GLOCOMW.2012.6477839.

[34] Xavier Vilajosana, Pere Tuset-Peiro, Thomas Watteyne, and Kristofer Pister. OpenMote: Open-source prototyping platform for the industrial IoT. In *International Conference on Ad-Hoc Networks*, pages 211–222. Springer, 2015.

[35] *Atmel AT86RF215 Device Family*. http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-42415-WIRELESS-AT86RF215_Datasheet.pdf. Atmel Corporation.

[36] *CYW43455 Single-Chip 5G WiFi IEEE 802.11n/ac MAC/ Baseband/ Radio with Integrated Bluetooth 5.0*. https://www.cypress.com/documentation/datasheets/cyw43455-single-chip-5g-wifi-ieee-80211nac-macbaseband-radio-integrated. Cypress/Infineon.

[37] Mikrotik rb951g-2hnd. https://mikrotik.com/product/RB951G-2HnD.

[38] Jonathan Muñoz, Xavier Vilajosana, Emmanuel Riou, Paul Muhlethaler, and Thomas Watteyne. Overview of IEEE 802.15.4g OFDM and its Applicability to Smart Building Applications. In *2018 Wireless Days (WD)*, pages 123–130. IEEE, 2018.

[39] Thomas Watteyne, Xavier Vilajosana, Branko Kerkez, Fabien Chraim, Kevin Weekly, Qin Wang, Steven Glaser, and Kris Pister. OpenWSN: a standards-based low-power wireless development environment. *Transactions on Emerging Telecommunications Technologies*, 23(5):480–493, 2012. DOI: 10.1002/ett.2558. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/ett.2558. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.2558.

[40] Sheldon Ross. Chapter 4 - Generating Discrete Random Variables. In Sheldon Ross, editor, *Simulation (Fifth Edition)*, pages 47–68. Academic Press, fifth edition edition, 2013. ISBN: 978-0-12-415825-2. DOI: https://doi.org/10.1016/B978-0-12-415825-2.00004-8. URL: http://www.sciencedirect.com/science/article/pii/B9780124158252000048.

[41] Zach Shelby, Klaus Hartke, and Carsten Bormann. The Constrained Application Protocol (CoAP), June 2014. URL: https://www.rfc-editor.org/info/rfc7252.

[42] Carsten Bormann, Angelo Castellani, and Zach Shelby. CoAP: An Application Protocol for Billions of Tiny Internet Nodes. *IEEE Internet Computing - INTERNET*, 16:62–67, March 2012. DOI: 10.1109/MIC.2012.29.

[43] Matthias Kovatsch, Martin Lanter, and Zach Shelby. Californium: Scalable cloud services for the Internet of Things with CoAP. In pages 1–6, October 2014. DOI: 10.1109/IOT.2014.7030106.

[44] Theodore S Rappaport. *Wireless communications: Principles and practice*. English (US). Prentice Hall communications engineering and emerging technologies series. Prentice Hall, 2nd edition, 2002. ISBN: 0130422320.

[45] Francisco J. Solis and Roger J.-B. Wets. Minimization by Random Search Techniques. *Mathematics of Operations Research*, 6(1):19–30, 1981. DOI: 10.1287/moor.6.1.19. eprint: https://doi.org/10.1287/moor.6.1.19. URL: https://doi.org/10.1287/moor.6.1.19.