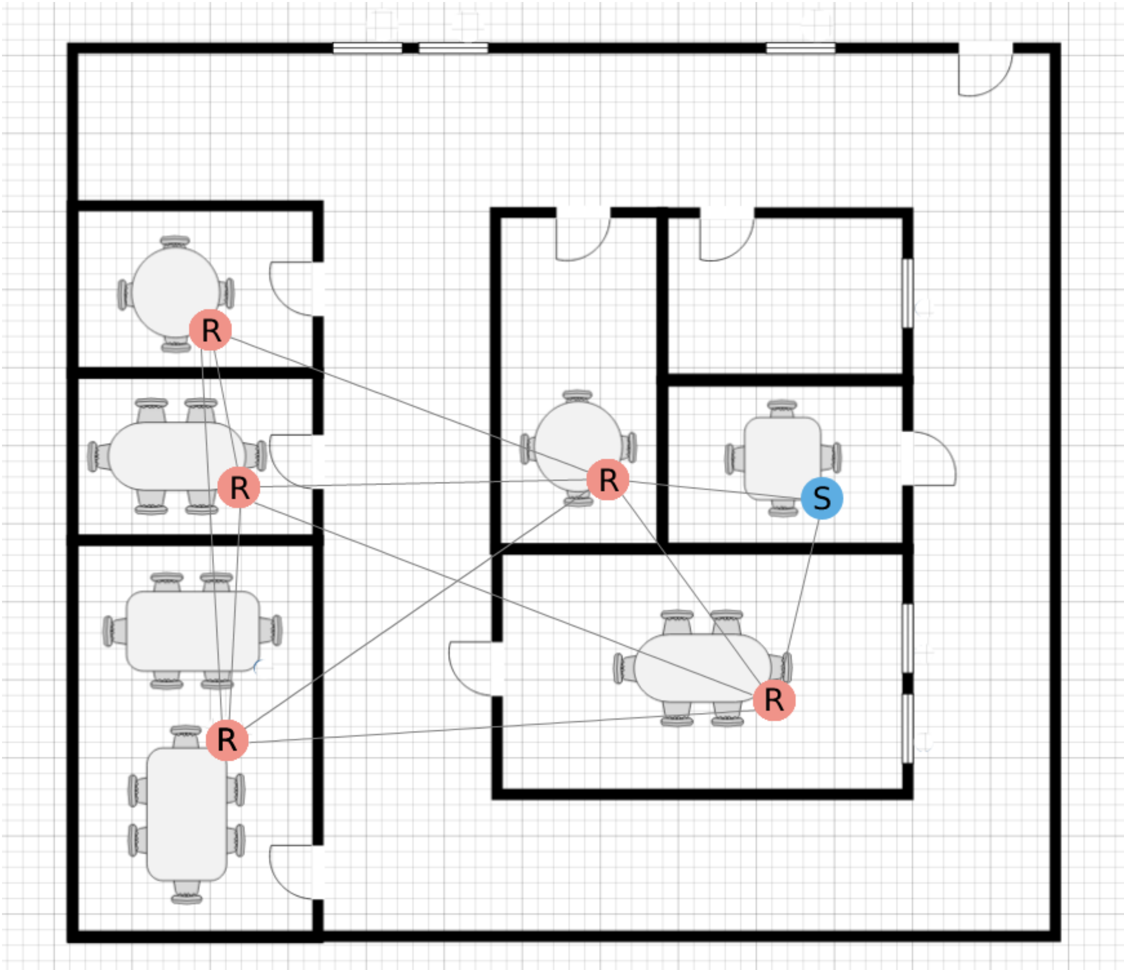
CHALMERS
UNIVERSITY OF TECHNOLOGY



Design and implementation of an Ad-hoc Indoor Positioning System based on a Distributed Routing Algorithm

Master's thesis in the System, Control & Mechatronics master programme

Borg Pontus & Wall Arvid

Department of Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2019

Master's thesis 2019:EENX30

Pontus Borg, Arvid Wall

Department of Electrical Engineering
Systems and Control
Chalmers University of Technology
Gothenburg, Sweden 2019

Design and Implementation of an Ad-hoc Indoor Positioning System based on a Distributed Routing Algorithm

PONTUS BORG
ARVID WALL

Supervisor: Robert Gustafsson, Combitech
Examiner: Martin Fabian, Full Professor of Automation at the Department of Electrical Engineering.

Master's Thesis 2019:EENX30
Department of Electrical Engineering
Systems and Control
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Bird perspective of an office environment with our created positioning system (sketched with www.planyourroom.com).

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2019

Design and Implementation of an Ad-hoc Indoor Positioning System based on a Distributed Routing Algorithm

PONTUS BORG
ARVID WALL
Department of Electrical Engineering
Chalmers University of Technology

Abstract

Compared to positioning outdoors, which to a large degree has been resolved with the use of GPS, positioning indoors suffers from multipath and attenuation effects, mainly due to building structures causing bad line of sight (LOS). With this aspect in mind, the thesis investigates the possibility of implementing a relative positioning system with mobile nodes to locate conference, team or meeting rooms using Zigbee technology.

A wireless sensor network (WSN) is constructed using the received signal strength indicator (RSSI) as a measurement of distance. Results show that RSSI used to represent distance is highly dependant on a number of factors: The direction of the antenna for both the transmitter and the receiver, the channel on which the sensor network is operating on and fabrication differences between devices have all been noted to affect measurements. Tests show that variance of transmission power affect the measurements to a degree such that it is not possible to relatively sort devices based on signal strength located at a distance shorter than three meters apart. In order to locate rooms based on signal strength the signals need to be processed with a Kalman filter before the positioning can take place. Zigbee devices rely on hop-count as a metric of distance and nodes implementing Zigbee does not store the shortest path to every other node in the network. To create a network better suited for locating remote nodes in the network a shortest path algorithm called DV-distance is implemented. It relies on the Bellman-Ford shortest path algorithm and uses RSSI measurements instead of hop-count as a measurement of distance. Successful results, devices sorted in the correct order, are achieved when the network is set-up in the correct way.

Keywords: Indoor Positioning, WSN, Routing, Algorithm, Zigbee, localization, 802.15.4, Kalman, RSSI, Distance Vector .

Acknowledgements

The authors of this report would first of all like to thank Combitech for providing us with this interesting subject for a master thesis, without their support this project would not have been performed. Robert Gustafsson, our supervisor from Combitech, should have a great thank for helping and encouraging us throughout the project. Martin Fabian, professor at Chalmers should also be thanked for being our supervisor and examiner.

Pontus Borg & Arvid Wall, Gothenburg, May 2019

Contents

List of Figures

List of Tables

Abbreviations

AoA	Angle of Arrival
AODV	Ad-hoc On-Demand Distance Vectoring
API	Application Programming Interface
AT-Commands	Attention-Commands
BFS	Breadth First Search
CCA	Clear Channel Assessment
CSI	Channel State Information
dBm	Decibel milliwatts
DFS	Depth First Search
DV-HOP	Distance Vector HOP
GPS	Global Positioning System
IDE	Integrated Developing Environment
IPS	Indoor Positioning System
LBS	Location Based Service
LOS	Line of Sight
LQI	Link Quality Index
MAC	Medium Access Control
OSI model	Open Systems Interconnection model
PCB	Printed Circuit Board
PHY	Physical
RSSI	Received Signal Strength Indicator
RTOF	Return Time of Flight
SBC	Single Board Computer
SCP	Secure Copy
SSH	Secure Shell
TDoA	Time Difference of Arrival
ToA	Time of Arrival
UART	Universal Asynchronous Receiver-Transmitter
UWB	Ultra Wide Band
WLAN	Wireless Area Network
WSN	Wireless Sensor Network
CSMA-CA	Carrier Sense Multiple Access - Collision Avoidance

1
Introduction

This chapter is intended to provide a background to the subject of indoor positioning, introduce the purpose of this thesis in light of the background and state research questions that will be answered throughout the report. It will also present the main scope of the report as well as an outline for what the following chapters include.

1.1 Background

Localization of objects is something that mankind have been interested in forever. It is crucial for us to be able to have a reference system and describe where things are located while having an everyday conversation. Today people rely on location based services (LBS) like smartphone apps that need location information on an everyday basis. Map services like Google Maps[1] and Waze[2] have become standard navigation tools, accessible in everyone's pocket, for providing location information and instructions about how to reach a desired position. These utilities are usually designed for outdoor positioning and navigation which to a large degree today is resolved with the use of global positioning system (GPS) technology. GPS however, has not been able to provide reliable and accurate positioning while navigating indoors, for that purpose other solutions must be used [3].

Navigation indoors has during the last decades popped up as an interesting and promising topic and research field. Many companies are developing their own solutions and until now several solutions have been suggested and implemented using a variety of different sensors, algorithms and wireless communication standards. Indoor environments introduce a more complex task of positioning than outdoor environments, due to attenuation and multipath effects of radio waves caused by building structures. Wireless sensor networks (WSN) have previously been used to position objects indoors, where Zigbee together with Wi-Fi, Bluetooth and ultra wide band (UWB) are some of the techniques used. Among these, Zigbee is a wireless standard specially developed for data rate (up to about 250kbit/s), low power, monitoring, control and automation tasks [4]. The Zigbee Alliance[3], maintainers of the Zigbee standard, provides a set of protocols for communication open for anyone to use. Known tech-giants like Philips, Huawei and lately also IKEA are some companies that have started to develop their own applications and products built

[1] Web based map service developed by Google.
[2] Another web based map service.
[3] Maintainers of Zigbee https://www.zigbee.org/.

on top of the Zigbee-standard, this opens up multiple areas where the technology could be used and it has never been easier for a hobby enthusiast to create a smart home[4] at a low cost.

1.2 Purpose

This master thesis will be executed at the company Combitech [5] and is closely related to some of their main field areas: network stacks and wireless communication. The purpose is to create an efficient Zigbee network that is able to locate objects indoors. Locating objects with an sufficient precision could be beneficial for companies and institutions with large building complexes. For example imagine an employee who wants to book a conference room. The employee wants to book an available room that is located as close as possible to his current location. Implementing a wireless sensor network combined with a graph search algorithm with nodes placed throughout the office building can help cut the time of finding a room which instead can be spent on something more productive. If the indoor positioning system (IPS) is designed in a mobile way, meaning that it easy to set up, relocate and extend, the opportunities and usages for it are countless. Another area where IPS:s will play a large role will be within automated warehouses, where it is of high use to know the position of an automated truck. This problem is today solved with combinations of techniques that utilize expensive equipment for positioning. If a Zigbee network could replace the existing techniques used within this field, the costs for the IPS would most likely be reduced. This thesis will also evaluate how high precision the Zigbee technology can deliver.

1.3 Objective

The main objective of this master thesis will be to answer the following research questions with the main one of them being:

- **How should a Zigbee network topology be structured in order to be able to locate indoor objects (e.g conference rooms) in an office environment in the most efficient way?**

Other research questions are:

- **What graph search algorithm's are most suitable, given the selected structure?**
- **Is the precision high enough for an everyday use in an office in order to locate the right conference room?**
- **What are the practical limitations?**

[4] A home with electronic devices that can be controlled remotely by smartphone or computer.

1.4 Scope

The thesis consists of both a theoretical part as well as a practical part involving the design and construction of a wireless network that will work as an IPS. The practical part of the thesis comes with a choice of how many nodes should be in the network. Since the aim of the thesis is to test the feasibility of an indoor positioning system's ability to locate and sort meeting rooms based on distance, no more than 10 nodes will be necessary. The IPS will not provide turn by turn navigation to a room only an approximate distance. Further the implementation of the wireless sensor network (WSN) will only make use of off-the-shelf Zigbee devices and no additional antennas or measurement equipment will be used with the aim of keeping the cost down. Another aspect that restricts the project is that the thesis has a time limit of 20 weeks, with each of them being 40 hours. Additionally for the thesis, the office space where the tests will be setup is considered a typical office space. An example of how the experiment might be set up is shown in Figure 1.1, the example consists of 5 mobile router nodes and one mobile start node (symbolized by the computer). Mobile in this case means that the nodes must have the ability to be relocated without errors occurring.

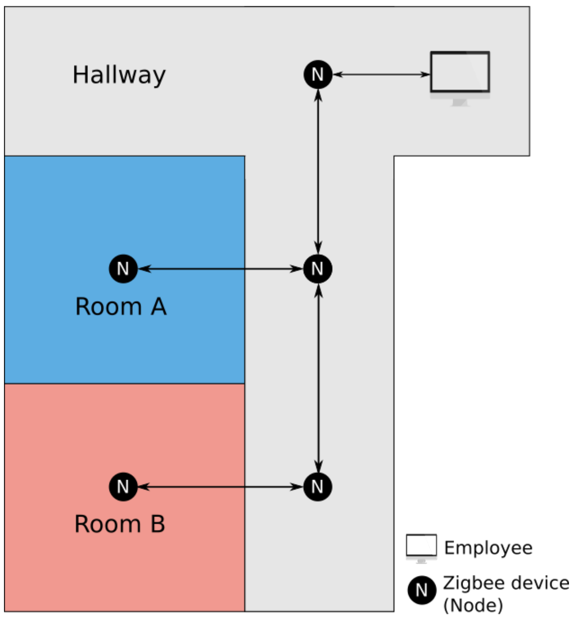


Figure 1.1: An example where an employee can book both Room A and Room B

1.5 Report Outline

This report will in a detailed way describe the process of work related to designing an Indoor Positioning System. Chapter 2 consists of some fundamental theory and explanations required for understanding parts of the thesis, here graph search theory, control structures, positioning techniques and also provide an introduction to Zigbee. Chapter 3 digs deeper into the actual implementation of the IPS. It also describes the programming approach more in detail and how the IPS are intended to work. In chapter 4 different tests that have been performed can be found. Chapter 5 brings a discussion about the actual performance of the IPS, it also mentions some of the issues encountered during the project. Chapter 5 ends with a discussion of what could have been done differently as well as some possible future improvements. The last chapter, is a conclusion and a summary of the project.

Throughout this report some technical words are used in their abbreviated form, these can be found at page 1. Another thing that should be clarified is the use of the word Zigbee compared to XBee, where Zigbee is a wireless communication standard and a set of protocols. There are many devices that implement the Zigbee standard, a XBee-device on the other hand is physical device manufactured by Digi International Inc. that meets Zigbee's (IEEE 802.15.4) requirements and implements one of the specified protocols.

2

Theoretical Background

This chapter provides some necessary background theory in order to follow this report. First, some Zigbee related topics will be explained, this includes: firmwares, messaging modes and already existing functionalities. This is followed by an introduction to graphs and how they can be represented. Different positioning techniques are also described in this chapter as well as some work that has been done in the past that is related to our scope.

2.1 IEEE 802.15.4, Zigbee and DigiMesh

Zigbee is a set of wireless communication protocols that were specified back in 2004 with the first commercial product being available in 2006 [6]. Zigbee Alliance [7] is the owner that regulates the standard. Lately the number of devices that implement the protocol have grown a lot, one reason for this is due to the introduction of smart and connected homes [8]. The standard is designed to be cheaper and require less power [9] compared to other standards for wireless private area networks (WPAN), like Wi-Fi and Bluetooth, which makes it ideal for these types of applications. Another standard which is proprietary of Digi International Inc. is DigiMesh, which is similar to Zigbee, but designed to be easier to set up and have even lower power consumption. The lower protocol layers in both Zigbee and DigiMesh work according to the IEEE 802.15.4 standard for low power and low rate WPAN.

2.1.1 IEEE 802.15.4

According to the specification [10]: IEEE 802.15.4 is a wireless standard specially developed for data rate (up to about 250kbit/s) and low power use. The main attraction of the standard has been the use in tasks such as monitoring, control and automation. IEEE 802.15.4 defines the physical (PHY) and medium access control (MAC) layers in the open systems interconnection model (OSI). The PHY-layer handles modulation, demodulation and physical transmission of signals. The MAC layer handles sending and receiving of packets using the 64-bit MAC address. The MAC address is assigned to the unit during manufacturing and does not change for the entire lifetime of a device. The standard has support for two topologies, star topology where all nodes communicate with a central coordinator and peer to peer topology where every node can communicate with any other node within range.

IEEE 802.15.4 enables communication in the 868-868.8 MHz band, mostly for European use, 902-928 MHz which can only be used in the US and a few other countries and 2.400-2.4835 GHz industrial scientific and medical (ISM) band, which is the most widely used since it is open in most countries. The 2.4 GHz band has up to 16 channels, with all of them, which the standard specifies, separated by 5MHz. The channels in the 2.400-2.4835 GHz band can be seen in Figure 2.1, where each signal is 2 MHz wide.

Raw IEEE 802.15.4 defines functions that are able to handle tasks such as discovering, forming and joining a network, this includes choosing personal area network-ID (PAN-ID), choosing channel, scan channels to detect interference and noise and also single hop broadcasting. It is capable of single hop communication using a technique called carrier sense multiple access with collision avoidance (CSMA-CA) to avoid collisions when two devices want to send a packet over the same channel. CSMA-CA is a function implemented at the MAC layer that receives information about clear channel assessment (CCA) from the PHY layer. CCA can be in either one of three modes. The first mode uses energy detection (ED) to determine the energy level on the channel. The second mode utilizes carrier sense (CS) and modulates the signal to evaluate if it is compliant with 802.15.4 PHY. The third mode uses both CS and ED. The PHY layer uses link quality indicator (LQI) to judge the quality of a link between two nodes. A link quality measurement should have at least eight different levels to evaluate the link against. LQI can be a measurement of RSS but could also be an estimation of signal to noise ratio.

What 802.15.4 does not specify is things that have to do with multi-hop communication, a requirement for mesh networking. The 802.15.4 standard is also not able to handle address assignment, instead it relies on the fixed 64-bit MAC addresses for communication and do not support any higher level application functionalities or encryption. The Digi XBee radios used in this masters thesis do not require a coordinator as opposed to the original 802.15.4 standard. The 802.15.4 standard is faster and suffers from less latency than networks using communication protocols built on top of 802.15.4 such as DigiMesh or Zigbee.

2.1.2 Zigbee

Zigbee uses the upper layers to define additional features, some of which are, authentication of nodes, encryption and data routing capabilities enabling mesh networking topology [11]. These upper layers include: the network layer, security layers, application support layers and application framework layers [12]. The Zigbee network stack with all layers can be seen in Figure 2.2. The network layer extends the networking capabilities of raw 802.15.4. When a Zigbee network is formed each device is assigned a 16-bit address. This address is unique for the device in the network. If a device rejoins the network it is assigned a new 16-bit address [13]. Forming a network using Zigbee is similar to IEEE 802.15.4 in that a coordinator chooses a channel and PAN-ID by first performing an energy scan to determine which of the

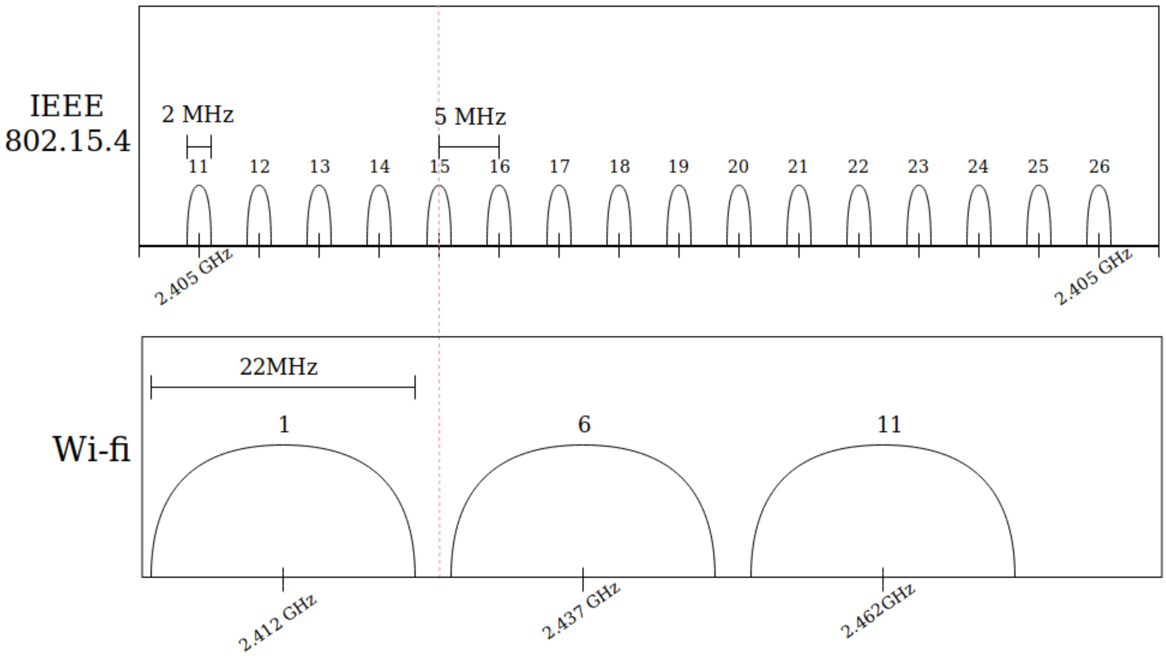


Figure 2.1: Channels in the 2.4 GHz frequency band for Wi-Fi and IEEE 802.15.4

16 IEEE 802.15.4 channels to choose based on the level of noise on each [14]. In Figure 2.1 the 16 IEEE 802.15.4 channels in the ISM band can be seen together with the three non interfering Wi-Fi channels. It can be seen that IEEE 802.15.4 channel 15 (or F in hexadecimal representation) is situated between Wi-Fi channel 1 and 6 in the frequency spectrum.

Mesh capability is a necessary feature whenever communication needs to be done between two nodes that are out of each others range. To enable mesh networking the Zigbee networking layer enables routing of packets to send unicast messages between devices. The routing algorithm used by Zigbee is Advanced Ad-hoc On-Demand Distance Vectoring (AODV) and is supposed to be resource efficient and require less memory for large sensor networks. Zigbee mesh networking uses LQI to assess if a link between two nodes is good or not [13]. One node might have to retry several times to establish communication, resulting in lower quality link. The Zigbee protocol is concerned with finding the shortest path with regards to time mostly by preventing retries and is thus not interested in the physical distance between nodes. The Zigbee network is self healing in that it has the ability to find a new route when a link breaks down. When a node is no longer able to send a packet to the next node along the shortest path it will report a route error back to the originating node, which in turn starts a new route request. Zigbee does not try to find the shortest path for every unicast message, only when a node becomes unresponsive or when a new path search is requested. The Zigbee stack comes in two variants, the first referred to simply as Zigbee supports up to 10 hops and is primarily used when cost is a deciding factor. The second, called Zigbee Pro is used when cost is less of a deciding factor and large networks are important [13].

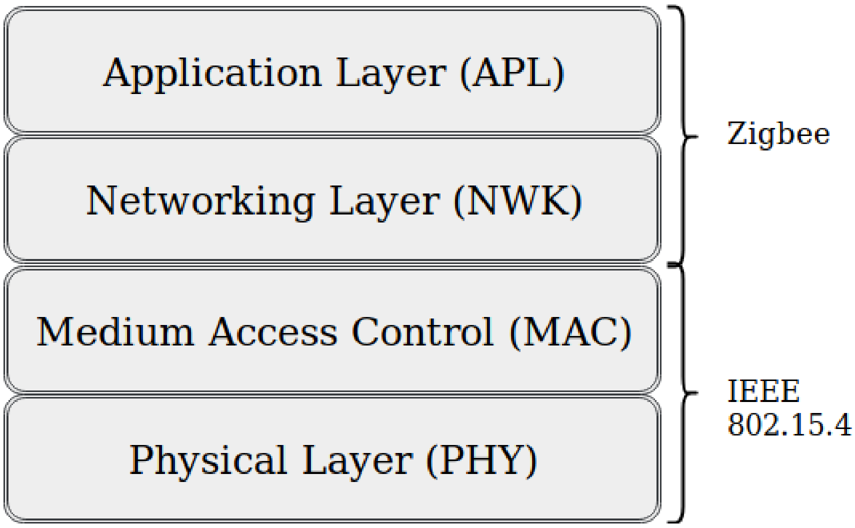


Figure 2.2: Zigbee and IEEE 802.15.4 layers

All additional functionality means devices configured as Zigbee have higher latency and lower bandwidth than raw 802.15.4 devices. Most of the Zigbee-protocols come with the eligibility to choose between different messaging modes [15], the most common are:

End-device: An end-device is a device that resembles a one to one communication scheme. The communication is between a parent host and the end-device itself. XBees configured in this mode can sleep according to a cyclic schedule and are usually woken up only to poll new messages from their host. These devices can run on battery since they consume very little power.

Router: A router is a device that is capable of routing messages between devices that are out of reach to each other. A router is always on and can not enter a sleep mode, its power consumption is therefore larger in comparison to end-devices and hence preferably connected to the main power supply.

Coordinator: A coordinator is a device that manages the network in the initialization phase. It selects the PAN-ID and the physical radio channel of the network and when that is done it works exactly like a router. Each network that is based on the protocols that support these modes must have exactly one coordinator in order to work. Routers and end-devices will ask the coordinator for permission before they join the network [15].

2.1.3 DigiMesh-Firmware

DigiMesh is a protocol available for Digi XBee devices. It has support for mesh networking, the major difference compared to other protocols is that in a DigiMesh network all devices are of the same type [16]. DigiMesh networks are very easy to setup due to all devices being configured in the same way. And owing to the ability for all devices to periodically sleep, a property limited to end-devices in Zigbee networks, all devices in a DigiMesh network can be battery powered. DigiMesh like Zigbee uses AODV for routing and discovery of nodes [17].

2.1.4 Alternative Wireless Technologies for Indoor Positioning

1. **Wi-Fi:** Uses the same frequency band as Zigbee, namely the 2.4 GHz band. It utilizes 22 MHz channels and has a higher bit rate than Zigbee. The three non interfering channels 1, 6 and 11 can be seen in Figure 2.1. The drawback of using Wi-Fi is that it does not support mesh capabilities and is more power consuming than Zigbee which makes it less suitable for battery power usage.
2. **Bluetooth:** Is a technology that has become a standard in mobile phones today, the frequencies range from 2.4 to 2.485 GHz. It utilizes something called a frequency-hopping spread spectrum (FHSS) where it at a pseudo random sequence switches between different channels. The sequence is known by both the transmitter and receiver which enables a lot of Bluetooth devices to work without interference.

2.1.5 XBee Java API

For the programming of the XBee devices, an application programming interface (API) created by Digi [18] is partially used. The API comes with a great extendability and makes it easier for users to start designing their own programs. Some files could be used directly out of the box while others require some minor changes. Here follows a brief description and listing of the files:

- **XBeeDevice.java**: A class that represent the serial connected XBee device. The class has methods that makes it possible to open and close the serial interface, retrieve a number of parameters from the local device and a also methods for sending data to desired devices.
- **RemoteXBeeDevice.java**: A class representing a remote XBee device on the same network as the local XBee device. It is used for establishing connections with the remote XBees through the local XBeeDevice. Contains the 64-bit MAC address, 16-bit NWK address and a string called Node-ID with the name of the device (assigned by the user).
- **XBeeNetwork.java**: A class representing all devices in the same network as the local device. The class has the ability to start a discovery process to create common network for XBee devices.
- **XBeeMessage.java**: A class representing a XBee message. Messages consists of a 64-bit address to a remote XBee as well as a byte array with data. The messages always contains a RSSI-value which is possible to retrieve.

2.1.6 AT-Command Mode

XBee devices support a certain programming mode called AT-Command Mode. AT stands for attention and the mode is entered if the user gives '+++' (three plus signs) as input in a XBee connected terminal. When a XBee is in the AT-Command Mode its configuration can be changed or read. Remote XBee devices can be configured through a local XBee if their AT-Command Mode is accessed.

2.2 Graph Theory

Networks are often represented by graphs, this representation gives a good overview and the information flow can be easily monitored. When XBee devices are used to communicate with each other a network is formed, in this network different graph structures can usually be identified. This section will give the reader a basic introduction to graph theory that will be useful for understanding the structures used while implementing the indoor positioning system. A graph, $\mathbf{G}$, is a data structure consisting of ordered pairs, the mathematical formulation is $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, where $\mathbf{V}$ symbolizes the vertices (or nodes) and $\mathbf{E}$ stands for edges (or transitions). For simplicity a graph can be thought of as a map, where the nodes represents cities and the edges represent paths. Each edge has a sink state and a source state, i.e. the edges tie the nodes together. In most cases the edges have costs associated with them, in the case where it is visualized as a map the costs can be thought of as the distance one have to travel from a city to another or the ticket costs for public transportation between cities. A graph can be both directed (the transitions have a direction) or undirected (no direction exists). This means that the possibility to revisit states in a directed graph might be limited. Graphs may have a start state as well as a goal state, Figure 2.3 shows an example graph with an initial state (A) and a goal state (E), marked with a circle. The graph in the example can be represented as:

As one moves along the edges the costs accumulate. The target is usually to reach the goal state with an as low cost as possible. Some transitions may lead to a sink state which is a state that has no outgoing transitions. It is usually desirable to avoid reaching dead-lock states since problems are often related to these. In the example graph the goal state E can be reached in two ways. It might seem trivial that both the paths: A $\rightarrow$ C $\rightarrow$ E and: A $\rightarrow$ E end up in the final state and that the best of these in terms of cost is the latter one (with a cost of 5), but when more states and transitions are added the problems can become very complex. Multiple ways to reach the goal state might exist making the graphs so complex that it is desired to use some kind of strategy to traverse them, with one being the utilization of graph search techniques or algorithms (see Section 2.6). Since the computers of today are very powerful they can search graphs and return the proven to be optimal path within fractions of a second. Imagine yourself requesting a LBS, for example

Google Maps, for the directions to a specific location, Google can return the optimal path in terms of either distance or travel time thanks to smart algorithms that are working in the background, totally hidden from the user. This is a result of that they have both satellites that triangulate the user as well smart algorithms that compute the shortest path to the specific location.

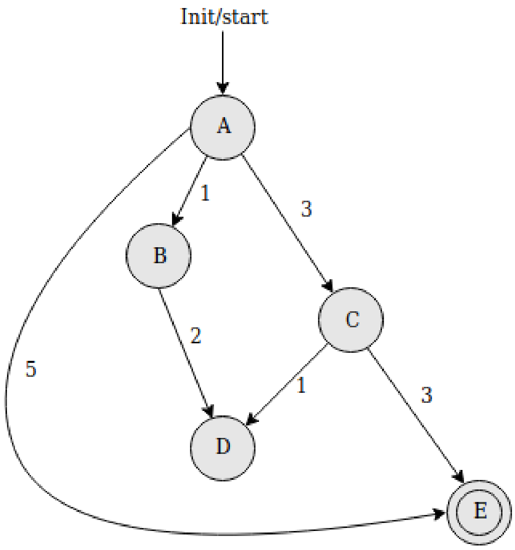


Figure 2.3: An example of a graph with node E as the goal state

2.3 Network Topologies

Different graph structures exists, these are mainly used to describe how objects relate to each other, a network where devices interact with each other is an example where the structures can be used to provide a good representation. The structures are sometimes called topologies and some of them can be identified in almost every network. A star (shown in Figure 2.4) is a simple structure where one device acts as the coordinator in the network, i.e. it talks directly to all other devices which are configured as end devices (only one connection). The star structure can also be seen as a centralized control scheme where one unit controls other sub-units.

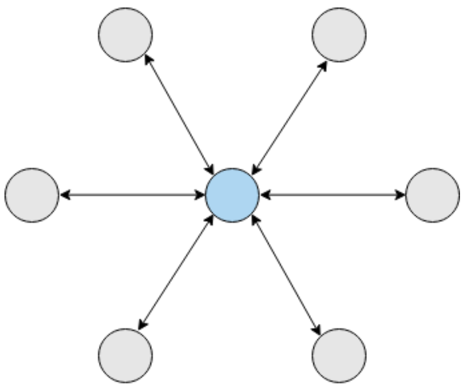
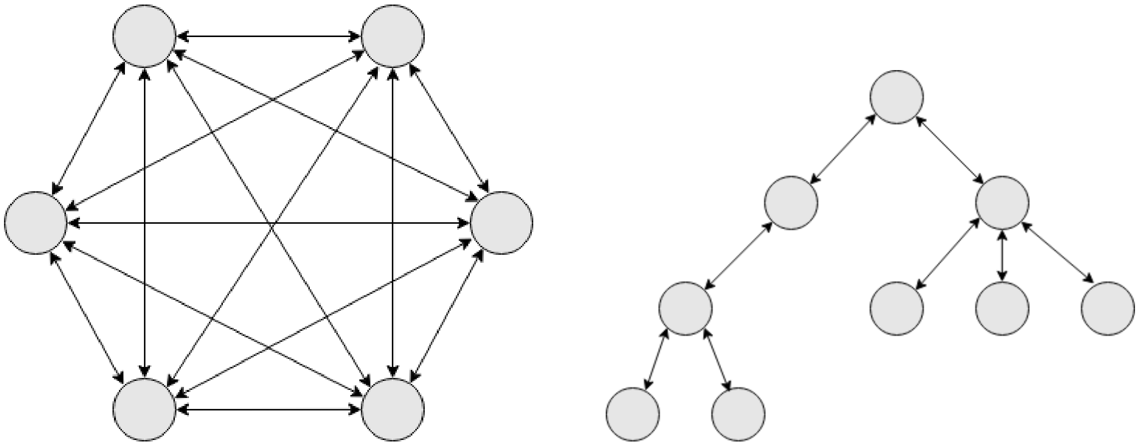


Figure 2.4: A star topology, where the blue node is the central unit

Another data structure is the tree, where one node is the root of the tree and the other nodes are visualized as tree branches. A further topology is the mesh structure (also called a meshnet) where nodes connect to each other automatically and spontaneously. All nodes that are within range to each other connect resulting in that edges cross each other, see Figure 2.5 (a). Nodes in a mesh topology also cooperate to route packets through the network. In reality networks are usually combinations of topologies and only when parts of the global network are isolated these structures can be identified.



(a) A fully connected mesh **(b)** A tree structure

Figure 2.5: Two more topologies

2.4 Positioning Techniques

In order to determine the position of an object different techniques can be used, most of them can be visually represented with a model. This model is explained in *RSSI-based Indoor Localization using Antenna Diversity and Plausibility Filter* [19] and gives a good overview about three separate phases, data acquisition, data preprocessing and localization (see Figure 2.6). The first phase (Phase I) is the data acquisition phase, where devices or users make estimations based on their surroundings. These estimations can be RSSI, infrared-light or ultrasonic sound etc. The data is gathered and used as input to Phase II where the data preprocessing will take place. If the data is fluctuating (i.e has high diversity in measurements) it is important to filter it and get a representation that is as close to the reality as possible. For example a Kalman filter (see 2.5.1) can be used to reduce some of the measured noise originated from Phase I. The better the data is processed the more accurate the actual localization will be. The data is preferably processed with respect to the specific algorithm that is used in the third phase. Sometimes it might be useful to remove some outliers (data points that are distinctly separated from most of the other points), whereas in other scenarios it might be better to take the mean value of all data points. If errors are added early on, the second phase gives the last opportunity to reduce these. When this phase is finished the processed data is sent to the last step which is where the actual localization takes place. Here smart algorithms operate in order to determine an exact position of the given object. The algorithms can be mathematical calculations or graph search algorithms (see section 2.6). Some algorithms rely on range based estimations, within these some of the methods estimate the actual distance from a point to the object (distance estimations/relative position estimation) and others (absolute position estimations) measure the exact position related to some sort of coordinate system. Depending on the purpose different algorithms may be more or less suitable, sometimes a high precision is desired and then techniques like RSSI may be insufficient (it is reported to have between 1.5 and 3.0 m precision [20] [21]).

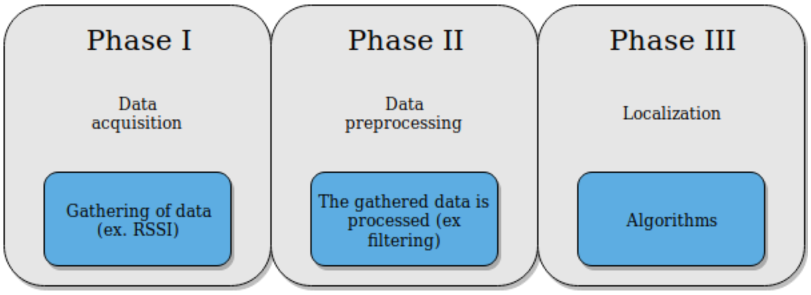


Figure 2.6: Visual representation of the three steps in a localization algorithm

2.4.1 RSSI

Radio frequency signals have energy that decreases proportional to the distance that the signal travels. RSSI is a measure of the relative energy or quality of a received signal sent from a transmitting to a receiving device. The measurement is derived from the received signal strength and describes how the receiver interprets the quality of the received signal. RSSI has no absolute value and different chipset manufacturers define their own max value that can be reached. A standardization has however emerged where the strength is measured in a logarithmic scale, decibel milliwatts (-dBm), a value closer to zero gives a better signal which can be seen and described in Table 2.1. In the same table one can observe that all RSSI-values are negative (these values are usually negated for practical reasons), the higher the value is, the better the RSSI is. The received signal is highly dependant on the environment in which it propagates, even the temperature affects how the signal travels. In [22] describes some practical experiments that investigates how RSSI and other RF-transmissions are affected by a change of temperature. The outcome is that less energy is required to send and receive packets with a 100% success rate if the temperature is low. A simple model relating the received and transmitted power with the distance is:

$$P_r(dB) = P_t(dB) - 10\alpha \log_{10} d \tag{2.1}$$

Where P_r and P_t are the received and transmitted power, d is the distance between transmitter and receiver. The constant α is the distance power gradient and is depending on the environment in which the signal propagates, typical values for α can be seen in Table 2.2. Additional uncertainties are introduced by propagation effects of radio signals and power level differences between devices. The RSSI-measurement has previously been evaluated based on its performance when it comes to positioning with varying results. Some [23] reports that it fluctuates to much in order to be a reliable measurement, while others [24] consider it sufficient for applications that do not require very precise positions. The objective of this thesis can be reached with sufficient location data and if the data varies to much smart filtering techniques can be used. It is not necessary to convert the RSSI values to meters, a graph search algorithm with edge costs based on the RSSI measurements between each and every device might be a feasible solution. RSSI is often used in combination with fingerprinting (see section 2.4.4).

Signal Strength:	**Quality:**	**Description:**
-30 dBm	Amazing	Devices are very close to each other
-67 dBm	Very good	Reliable signal strength for packet delivery
-70 dBm	Okay	Minimum strength for packet delivery
-80 dBm	Not good	Minimum signal for basic connectivity
-90 dBm	Unusable	To far distance for connection

Table 2.1: Classification [1] of the different levels of RSSI

Environment:	**Distance power gradient (α):**
Free space	2.0
Retail store	2.2
Grocery store	1.8
Office, hard partitions	3.0
Office, soft partitions	2.6

Table 2.2: Table with the environmental coefficient [2]

The book *Localization algorithms and strategies for wireless sensor networks* [25], brings up some applicable measurements for localization of objects. RSSI is one of the techniques mentioned and used for an IPS, compared to other techniques used in IPS (ToA and AoA) RSSI has an advantage since it does not require an accurate time reference. According to the authors of [25] that accurate time references can only be achieved with special designed beacons and antennas. Off-the-shelf Zigbee-devices do not come configured in this fashion which discourages the usage of techniques like ToA and AoA.

2.4.2 Angle of Arrival/Departure

Angle of arrival/departure (AoA/AoD) are two techniques that can be used for positioning. By having antennas that can determine the physical angle of an incoming transmission it is possible to calculate an absolute position. Several antennas (two or more) that are mounted at a fixed distance to each other in the receiver record the received angle of the packet. With this data it is possible to calculate an absolute position in relation to the transmitter. The more antennas the receiver is equipped with the better precision is achieved. The technique is in some sense a triangulation method and is widely used in the latest Bluetooth-firmware (Bluetooth 5.1) [26] where the administrators claim that the technique is capable of providing an accuracy of under 1m [27]. The only difference between AoA and AoD is where the angles are measured, in AoD multiple antennas are instead located on the transmitter side while in AoA they are placed at the receiver side.

2.4.3 Time of Arrival

Time of arrival (ToA) [28], sometimes referred to as time of flight (ToF) is another positioning technique used in triangulation. This positioning is made possible by knowing the time difference from when the signal was sent ($t_{transmitter}$) by the transmitter until it arrived ($t_{receiver}$) to the receiver. By also knowing the velocity (v) of the signal the distance (d) can be easily calculated with the following formula:

$$d = v * (t_{receiver} - t_{transmitter}) \tag{2.2}$$

This technique is used in GPS and does also require more than just one transmission, since more than one position can be derived by only knowing the distance. In equation 2.3 a known reference position must be used, usually the placement of the transmitter ($x_{transmitter}$, $y_{transmitter}$) is a known position. In the 2-dimensional plane the set of solutions is a circle and in 3-dimensions the feasible set yields the surface of a sphere. The exact coordinates are derived by having multiple transmissions being sent from different transmitters and calculating the intersection between these spheres or circles. In the 2-dimensional plane x- and y-coordinates are solved from the following formula:

$$d = \sqrt{(x_{transmitter} - x)^2 - (y_{transmitter} - y)^2} \tag{2.3}$$

2.4.4 Fingerprinting

Fingerprinting is a technique where several data points are gathered for different positions. These data points, will if recorded correctly, span a grid that other signals can be compared to. The grid is used as an overlay to match the measured data points with the the closest recorded one. The more data points that are recorded the higher the resolution of the grid will be and a more accurate localization can be achieved. Figure 2.7 shows a visual representation about how the method works. Each intersection of the lines in the graph is a known position, in order to determine the position of the red and blue measurements they are evaluated against the rest of the points.

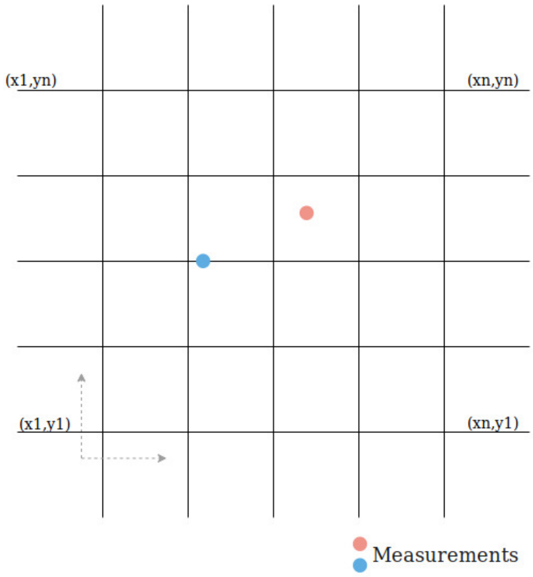


Figure 2.7: Visual representation of how the technique fingerprinting works

2.4.5 Time Difference of Arrival

Time difference of arrival (TDoA) is yet another positioning technique, it does not rely directly on the signal's propagation time, instead the time difference is created by comparing when the signal has arrived at different receivers:

$$\Delta d = v * \Delta t \tag{2.4}$$

Here is v once again the velocity that the signal travels with. Δd in equation 2.4 can be rewritten into an expression where once again several points satisfy the equation:

$$\Delta d = \sqrt{(x_2 - x)^2 - (y_2 - y)^2} - \sqrt{(x_1 - x)^2 - (y_1 - y)^2} \tag{2.5}$$

Both (x_1, y_1) and (x_2, y_2) are known positions of the communicating devices. With the help of linear regression the expression can be formulated as a hyperbola [28] and the intersection between hyperbolas yields an absolute position.

2.4.6 Channel State Information

Channel state information (CSI) is in some sense much like RSSI, in the way that it describes how the signal propagates from the transmitter. Compared to RSSI, CSI is fine-granular and a more sensitive measurement that takes into account how the signal fades, scatters and decays with an increased distance [29]. CSI is usually used in combination with fingerprinting in order to achieve a high precision.

2.5 Data Processing

To further improve measurements collected during the data acquisition phase, data processing methods can be used. The intention of these methods is to extract information from noisy data. And give a better approximation of the quantity being measured.

2.5.1 Kalman Filter

A Kalman filter is used to predict the state described by the stochastic difference equation:

$$x_k = A_k x_{k-1} + B_k u_k + w_k \tag{2.6}$$

and to use the measurement:

$$z_t = H_k x_k + v_k \tag{2.7}$$

to get additional information about the state. Equation 2.6 is usually referred to as the time update step since the state is estimated based on information about previous states. The variable u_k is the current input to the system and matrices A and B describe the system model. The variable w_k is called process noise. Equation 2.7 is called the measurement update step, here the latest measurement is taken into account. Matrix H_k describes the measurement model and v_k is called measurement noise. A Kalman filter is a recursive data processing method which means that it does not need knowledge about all previous data points to estimate the next state, it only needs information about the previous state and the latest measurement [30]. In order for a Kalman filter to perform a state estimation, a few conditions has to be fulfilled, both the process noise and measurement noise have to be white and Gaussian [30].

Updating the filter is done in two steps (2.6). The first step is the time update of the filter and the second step is the measurement update. The time update is done in the following way:

$$\bar{x}_k = A_k x_{k-1} + B u_k \tag{2.8}$$

$$\bar{P}_k = A_k P_{k-1} A_k^T + R_k \tag{2.9}$$

The measurement update step is calculated using:

$$K_k = \bar{P}_k H_k^T (H_k \bar{P}_k H_k^T + Q_k)^{-1} \tag{2.10}$$

$$x_k = \bar{x}_k + K(z_k - H_k \bar{x}_k) \tag{2.11}$$

$$P_k = (I - K_k H_k)\bar{P}_k \tag{2.12}$$

Where R_k is the process noise covariance, Q_k is the measurement noise covariance and K_k is the Kalman gain. P_k is the state covariance.

2.6 Graph search algorithms

A graph search algorithm is an algorithm that is used to traverse states in a graph. As previously mentioned, graphs are usually used to represent real world scenarios in a more mathematical context. Graph search algorithms can be categorized into different categories, breadth first search (BFS) and depth first search (DFS). The main difference between these two is mostly related to how they pick and store the states in their queues (or stacks). BFS visits the states at the same level before it advances to further depths, while DFS follows the full pathways from the start- to the end node.

Graph search algorithms can also be categorized into centralized- or decentralized methods. In a centralized graph search approach there is one device that is in control of the execution of the algorithm, some of the processes can still be distributed to host devices, but a main central processing unit (CPU) is responsible for the scheduling of these. Host devices need to ask the CPU about permission when they want to enter a critical section (CS) and utilize shared resources [31]. This CS is usually controlled by mutual exclusion, which means that only one process can enter it at the same time. Decentralized algorithms on the other hand make use of several processors in order to get the work done and no central unit is used to schedule the tasks. Instead each host make decisions based on its own gathered information and which means that hosts almost never store information about the complete system behaviour. Beyond these, an additional control structure called distributed control exists, in such a control structure a CPU is used to mainly keep track of the complete system information and outsource some of the task to other controllers. Each of the structures have their own advantages and depending on the specific application some may be more suitable [32].

Some famous graph search algorithms are for instance Dijkstra's algorithm, which was invented by Edsger Dijkstra back in 1959 have become well known and utilized for planning transportation routes. Many of today's navigation apps utilizes Dijkstra's algorithm in combination with other smart techniques. According to [33], Dijkstra's algorithm are not in any way optimal, in terms about how fast it finds a solution, but it delivers the correct result after a certain amount of states have been visited. A*, is a another well known algorithm that is an extension of Dijkstra's algorithm. A* is proven to be optimal [34] if the used underlying heuristic is admissible. A heuristic is a feature that supplies an estimate of what the cost to travel from a arbitrary node to another node will be. Optimal in this sense means that the algorithm will expand as few nodes as possible in its search for the optimal goal. A* is in its original issue a centralized control algorithm which is processed in one main CPU only, lately some decentralized versions been published [35].

2.6.1 Bellman Ford

The Bellman-Ford is another algorithm and as described in [36] it is a graph search algorithm commonly used in sensor networks and can be implemented in a decentralized way. The algorithm finds the shortest path between a start node and all other nodes in the network and can thus get the shortest path from the start node to all other nodes. Bellman-Ford has a greater time complexity than Dijkstra's algorithm, but can handle more types of graphs, namely graphs with negative edge costs. The initial step of the Bellman-Ford algorithm is to select one start node and initialize the distance to this start node to zero. The distances to all the other nodes in the network will for the start node be initialized to infinity. The algorithm then proceeds by relaxing the edges which means that it compares the distance to a target node with the distance to a source node plus the cost of the edge from the source node to the target node. If the proposed new distance to the target node is smaller than the previously shortest distance the proposed distance replaces the previous best stored distance. Compared to Dijkstra's algorithm which prioritizes vertices with known lowest cost and relaxes these edges first, the Bellman-Ford algorithm relaxes all edges in the graph with no priority. The algorithm first relaxes all edges of the graph and then starts over and relaxes all edges again, it does this |V| times. This means that the worst case performance of the algorithm is $\mathcal{O}(|V||E|)$ and the best case performance is $\mathcal{O}(|E|)$ compared to Dijkstra that has worst case performance $\mathcal{O}(|V||V|)$. This means that if a graph contains n vertices, the maximal number of edges will be n^2 resulting in a worst case computational complexity of $\mathcal{O}(n^3)$.

Algorithm 1 describes in pseudo-code how Bellman-Ford works

Algorithm 1 Bellman Ford

```
for v in V : do                                   ▷ Initiates all distances to infinity
    v.distance = ∞
    v.p = Null                                    ▷ Pointer used to keep track of the optimal path
source.distance = 0
for i from 1 to |V| − 1 : do
    for (u, v) in E : do
        relax (u, v)
procedure Relax(u,v)                              ▷ Relax all the edges
    if v.distance > u.distance + weight(u, v): then   ▷ Check if better exist
        v.distance = u.distance + weight(u, v)    ▷ Save the best distance
        v.p = u
```

2.7 Routing Protocols

There exist several protocols for routing. These have a number of different characteristics. For one, protocols can be either static routing protocols in which routing tables are fixed for each node and not shared between neighbors. A static routing protocol requires a lot of time to set up and does not scale well, or a dynamic routing protocol which is a routing protocol that updates the distance vectors of each node by sharing them between neighbors.

Routing algorithms can also be divided into proactive and reactive routing algorithms [37]. Proactive routing keeps all the information about shortest paths even if this information is not used. DV-distance is an example of proactive routing.

Reactive routing only keeps routes that are currently in use. A routing protocol can also be a hybrid which is a combination of the two. Other differences between routing protocols include whether they use sequence numbers for distance vectors or not. Routing protocols that use reactive routing are often called on demand routing protocols. On demand routing also often has the benefit of route recovery when a node breaks down and a new route needs to be established.

2.7.1 Distance Vector Routing

A known algorithm [38] that is simple, but at the same time efficient in the right environment is called DV-HOP (Distance Vector-HOP). It calculates the number of nodes that a signal passes (or the number of hops). If the network is constructed in a way such that every node is located approximately with the same distance this algorithm performs good. A version of the algorithm exists called DV-Distance where every node keep track of the distance or a metric representing distance like RSSI and not hop count to all other nodes, and stores it in a routing table. This is not totally applicable for this thesis but parts of the algorithm can be used in combination with RSSI-measurements. Both of these vector routing algorithms use Bellman-Ford to determine shortest paths. Both protocols are proactive routing protocols.

2.7.2 Link state routing

Other than distance vector routing, another common proactive routing protocol is link state routing [39]. Link state routing differs from distance vector routing in that the only information sent between devices is information about the links in the network and not information about distances like in distance vector routing. The information is distributed using a technique called flooding where every node forwards a message to every neighbor node except to the one from which it got the message. In link state routing every node has complete information about the network, in the form of a graph. To find the shortest path to all other nodes every node uses Dijkstra's algorithm.

2.7.3 Destination sequenced distance vectoring

Destionation sequenced distance vectoring (DSDV) [59] is an extension of the distance vector routing protocol and also relies on Bellman- Ford and uses the same routing tables. The two protocols differ in that DSDV uses sequence numbers for every entry in the routing table. The protocol uses frequent broadcast of the distance vectors to determine the shortest path in a changing environment.

2.7.4 Ad -hoc on demand distance vectoring

AODV is another common routing protocol [40] which is reactive and specially developed for mobile ad-hoc networks. The basic principle is that no routes exist until they need to be established between two nodes. When a source node needs to communicate with a remote node the source node initiates a route discovery

procedure. This property is what makes it a so called on demand distance vector protocols.
No routes exist in the network until a route discovery procedure is started by one of the nodes in the network. The route finding procedure starts by a request from a source node for communication with a desired destination node. The request is broadcast to neighboring nodes and each node the request passes records the hop count to the source node and stores the neighbor node, from which it received the request and thus a backward route is established. If a node receives more than one request then the second one is discarded.
If the broadcasting of the request reaches a node that already has a known route to the destination node then that route is used only if the sequence number of the node that made the request is lower than the sequence number of the node with the established path otherwise it is discarded and the broadcast continues until the destination node or a valid route to the destination node is found.
Subsequently when a path to the destination node is found a unicast message is sent back to the source node and the route is established. If during this phase a node on the path receives a route with fewer hop counts then the old route is discarded and the new route is propagated back to the source node. All nodes not on the established route drop the backtracking information obtained during the route finding procedure after a specified timeout period.
The AODV protocol also maintains its routing tables by using a timeout for each entry that is reset every time a specific route is used. Each entry also has a sequence number associated with it. When a node breaks down then a hop count of infinity sent to every source node that relied on the broken node for communication.

2.8 Related Work

Many relevant reports and articles exists related to the design and implementation of an indoor positioning system. The majority of these focus on other technologies than Zigbee and mostly aim to deliver the exact position of an object, usually with as high precision as possible. It is of high relevance to study different techniques that are used in indoor positioning today. *Survey of Wireless Indoor Positioning Techniques and Systems* [4] gives a comprehensive background to indoor positioning and compares positioning techniques like triangulation, scene analysis and proximity based techniques. The paper also compares distance measurement techniques like TOA, TDOA, AOA, RSS and RTOF and different wireless technologies including WLAN, Bluetooth and UWB. It is emphasized that accuracy, robustness, cost and scalability are important factors to consider when designing an IPS. The survey also raises the importance of large, low power sensor networks and brings up IEEE 802.15.4 and Zigbee as the primary standards. *A Survey on Localization for Mobile Wireless Sensor Networks* [41] also surveys various positioning techniques and additionally introduces challenges that comes with introducing mobile nodes to a WSN. Another survey *A Survey of Indoor Positioning Systems for Wireless Personal Networks* [42] compares existing indoor positioning systems implemented using different mediums and wireless technologies. The most relevant and appropriate techniques are listed and explained further in section 2.4. The survey [4] presents RSSI as a measurement for distance that is susceptible to multipath effects and noise . Various other documentations [43] [20] [21] have also reported that this measurement is somewhat questionable and its precision in an everyday use is investigated to be between 1.5 and 3.0m [20] [21]. The article [44] states that RSS is a measurement whose error is proportional to the distance, making RSS a measurement most suitable for sensor networks with a high density of sensors.

Localization systems for wireless sensor networks [45] is a report that describes some relevant techniques for the given purpose. These can be categorized into distributed- or centralized positioning techniques, absolute- or relative positioning techniques and one- or multihop techniques. As previously mentioned, when a Zigbee-network is setup, a coordinator chooses the most appropriate parameters and packets that are sent through the network are automatically routed a specific way. *Routing Protocols in Zigbee Based networks:A Survey* [46] is another survey that explains some of the routing algorithms that the different Zigbee firmwares are utilizing. The drawback is that these does not take the actual distance into account [13] instead they use a measurement called LQI. The ideal case would be to utilize this and extend it with a distance metric.

It seems like the field to locate and sort rooms not have been as popular as the aim to achieve a high precision in the past. The common ground is however that a WSN need to be created, *Routing Techniques in Wireless Sensor Networks: A Survey* is a text [47] that describes what a Wireless Sensor Network is and what type of routing methods that can be utilized in such a network. At the same time it categorizes routing techniques into three different groups: Flat-, Hierarchical- and Location-based routing, where Flat-routing is simplest one. In Flat-routing all nodes communicates with all of their neighbours, Hierarchical routing is a way of grouping nodes such that some of them can sleep whilst energy can be saved and Location-based routing, where nodes communicate with each other based on their location and distance to each other. For this category the evaluation of what nodes that will be grouped together is often based on the signal strengths [47].

The report [48] is one of the most related, where localization is performed with the help of the signal strength from Wi-Fi combined with fingerprinting. The result is an accuracy of under 1m. Another project that uses a distributed approach to perform localization is [49], where a relative positioning is performed with the help of an algorithm that computes the nodes' position in a 2-dimensional grid. [50] proposes some other techniques for distributed localization where "DV-distance" is one of the mentioned. How the technique works is mentioned in section 2.7.1, in short it does not calculate the number of hops, instead it uses the the nodes signals to determine the distance between them. The signal could later be transformed from dBm to meters with the help of the previously explained formula 2.1. The RSSI-measurement conversion by itself does not tell where an object is, it only tells how far away it is located. The direction or the exact coordinates is something that can not be extracted from this approach, the same article proposes triangulation techniques like AoA and ToA in order to get an exact absolute position.

3

Method and Implementation of the IPS

This chapter provides detailed instructions about how the experimental part of the master thesis is executed. The main parts will be related to what design choices are made and why. The chapter will also include information about how the selected algorithm is programmed to work, how the Java programs are structured and how the physical IPS is constructed. First of all, all the technical equipment that is used will be presented. This is followed by a detailed description of the different implemented functions in the IPS.

3.1 Experimental Setup

The indoor positioning system will consist of both hardware and software areas where design related decisions have to be made. In some sense the technical equipment make out the boundaries for what kind of software that can be developed. When the literature study was executed many reports stating that RSSI is fluctuating were found, the conclusion is that it might not be the most accurate metric. However in combination with other methods it may still work in the desired way for the purpose to locate a conference room. Just one RSSI measurement does not deliver enough information in order to perform an actual positioning, three or more measurements can be used to triangulate the object but since RSSI is a quite rough scale even this will presumably not meet the needs of an IPS and the technique will not be able to compete with other positioning methods. Instead of triangulation the network can be viewed as a map where each XBee device represents a node and RSSI values from measurements represents the edges. By creating an undirected graph it is possible to utilize graph search algorithms, as described in section 2.6. Since the first research question that was formulated in section 1.3 relates to how the network topology should be structured to be able to locate objects, several network structures need to be investigated. The first step however is to perform some range tests with the hardware and configure the XBee devices in the best possible way (see section 3.2.4). A total of six devices seems like a reasonable amount, that setup will not cost a fortune and at the same time not compromise too much with the functionality. If more units are to be added it could be done without any additional problems occurring since the designed system is distributed. Digi, the manufacturer of the selected XBee devices, provides a useful guide, *Get Started!* [51], which is a good introduction to the technology. It gives the user knowledge about some of the basic functionalities that Zigbee comes with.

The second research question aims to investigate what type of graph search algorithms that perform best and is suitable given the chosen structure. The idea behind the final choice, the Distance vector routing is that it is a distributed algorithm based on the Bellman Ford algorithm that will work with lower latency compared to other centralized versions where a main CPU has to perform all the work on its own and since all devices rely on a Raspberry Pi for power and serial data streams the Pi:s might as well do computations locally. It also guarantees finding a shortest path between all nodes in the network.

Other things that should be investigated is how high precision can that be reached with the final positioning system. Since the scope is more closely related to locate rooms other than small objects in an indoor environment this might be hard to evaluate. The system will also not provide turn by turn navigation to the rooms, instead it will present a list with the rooms and their measured RSSI, more like a first concept of an IPS. In order to improve the functionality an additional map with an overview of the office environment in bird perspective could be used in combination with a list to pair each device with the corresponding rooms. Lastly the practical limitations should be investigated, this is somewhat tied to the previous question. For a system like this it will be impossible to provide navigation since the the XBee messages sometimes are transmitted through walls and windows and even bounced on objects. It is thus not possible to trace the RF signals and base the navigation on these, in such cases other concepts for the navigation feature must be developed.

3.2 Technical Equipment

The actual positioning system consists of the following technical equipment:

- 6 Digi XBee S2C 802.15.4 RF Modules
- 6 Digi XBee Grove Development Boards
- 5 Raspberry Pi 3 Model B+
- 5 MicroSD cards pre-installed with NOOBS
- ASUS RT-AC68U Wireless-AC1900 router
- Laptops, USB-cables Raspberry Pi power cords

Throughout the chapter a setup consisting of one Digi XBee S2C 802.15.4 RF Module, one Digi XBee Grove Development Board and one Raspberry Pi 3 Model B+ with a microSD card will be called a router node. Exactly the same setup without the Raspberry Pi instead connected to a laptop will be called a start node. In total five router nodes and one start node will be used, this setup is able to span a relatively large part of the office environment and give test results that do not differ from the ones expected from larger systems.

3.2.1 RF Modules

Each Digi XBee S2C 802.15.4 RF module is connected to one Digi XBee Grove Development Board via a 20-pin connector. It is the RF module that is responsible for the communication with other XBee devices, therefore it has both a transmitter and a receiver. The module is equipped with an on-board printed circuit board (PCB) antenna which can be replaced by an external one if the range needs to be extended. According to the manufacturer's data sheet [52] the indoor range is approximately 60 meters, this has been tested in a typical office environment and it seems like 60 meters may be little bit of an excessive assertion. The positioning system will however be constructed in a way such that the nodes will not be located with more than 30 meters distance in between them, otherwise packet losses will occur more frequently than desired.

3.2.2 Development Boards

The Grove Development Boards act as the bridges between the Raspberry Pi:s and the RF modules. The board has three buttons named: Reset, Commissioning and User Button. The Reset button does exactly what it sounds like, it resets the hardware. As already mentioned, some of the features that the XBee devices have are that they have very low power consumption and can be configured in different modes. The Commissioning button can be used to alternate between these modes. When a device is configured in sleep mode it will occasionally enter sleep and a two second long press of the button will wake the device up. The User button can be used for specific programming purposes where the user can control external applications with a press of the button. As previously mentioned the Grove Development Boards operate on a bit stream that the Raspberry Pi:s feed to the boards. When the boards receive data a small LED labeled RX lights up and when they send data another LED, TX, flashes. There are also some other LED:s on the board, with one of them labeled RSSI, it is associated to when it the module is involved in a RSSI measurement.

3.2.3 Raspberry Pi:s

The idea behind the Raspberry Pi:s are that they are much more mobile than other desktop computers and can therefore be placed more easily at different spots. Since the positioning system will be built up in a typical office environment it is also preferable if the nodes are as small as possible. All Raspberry Pi:s are connected to a local Wi-Fi network which opens up the possibility to control them wirelessly. This is made possible through a technique called Secure Shell (SSH). The technique lets a user control another computer via a terminal window in a secure way. The only requirement is that both of the computers are connected to the same network and have SSH enabled. With SSH enabled on each and every Raspberry PI they can operate in headless mode, this is when no external monitor is used. Another technique named Secure Copy (SCP), that builds upon SSH, can be used to copy files from one local computer to the Raspberry Pi:s. These two techniques are extremely helpful for this kind of project since most of the routers will be fixed

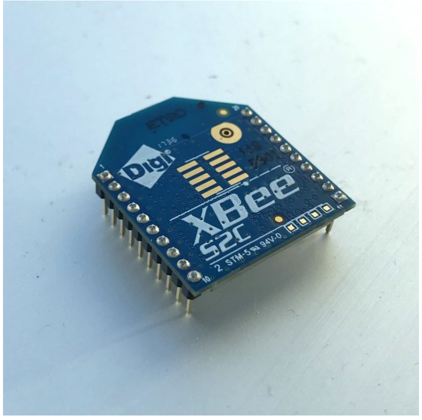


Figure 3.1: The Digi XBee S2C 802.15.4 RF module

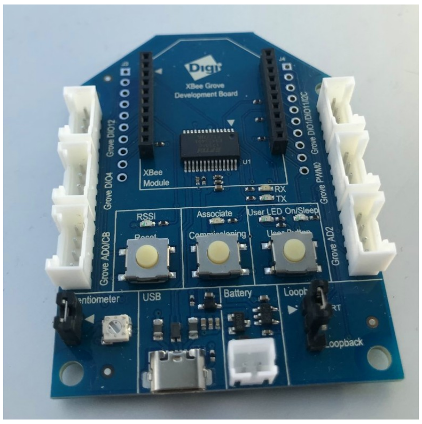


Figure 3.2: The Digi XBee grove development board

at a spot in the office and it would require a lot of effort and time to bring all of the devices in when new stuff is added to the operating firmware. In order for the Raspberry Pi:s to execute the written Java programs a automation tool called Maven [53] is installed. Maven opens up the possibility to run Java programs from the terminal without an IDE (Integrated Developing Environment) like Eclipse or Netbeans. Maven both builds the program and describes how the software should run with all of the dependencies between files. For this particular use the Raspberry Pi's are very easy to setup, they come with a Micro SD-card with a pre-installed OS-handler called NOOBS.

3.2.4 Configuring the XBees in XCTU

XCTU is a configuration platform that enables a user to communicate between devices using a graphical interface which can be seen in Figure 3.3. With XCTU it is possible to configure a set of parameters related to networking, serial communication and addressing among others. The only parameters of interest in the scope of this thesis are the channel that the device is operating on, ranging from 11-26 (B-A1 in hexadecimal values), see Figure 2.1, CCA threshold specified in -dBm, Node-ID, network ID and whether the device should operate in API or transparent mode.

The last mentioned setting concerns serial communication between the local XBee device and the computer. In transparent mode, all serial data received by the XBee will be communicated to any device on the same network (except for data preceded by "+++" which will be interpreted as commands). In API mode serial data is interpreted as packets, this enables more complex communication between devices which includes identifying the device that sent the message and acquiring RSSI values, hence API is the preferred mode.

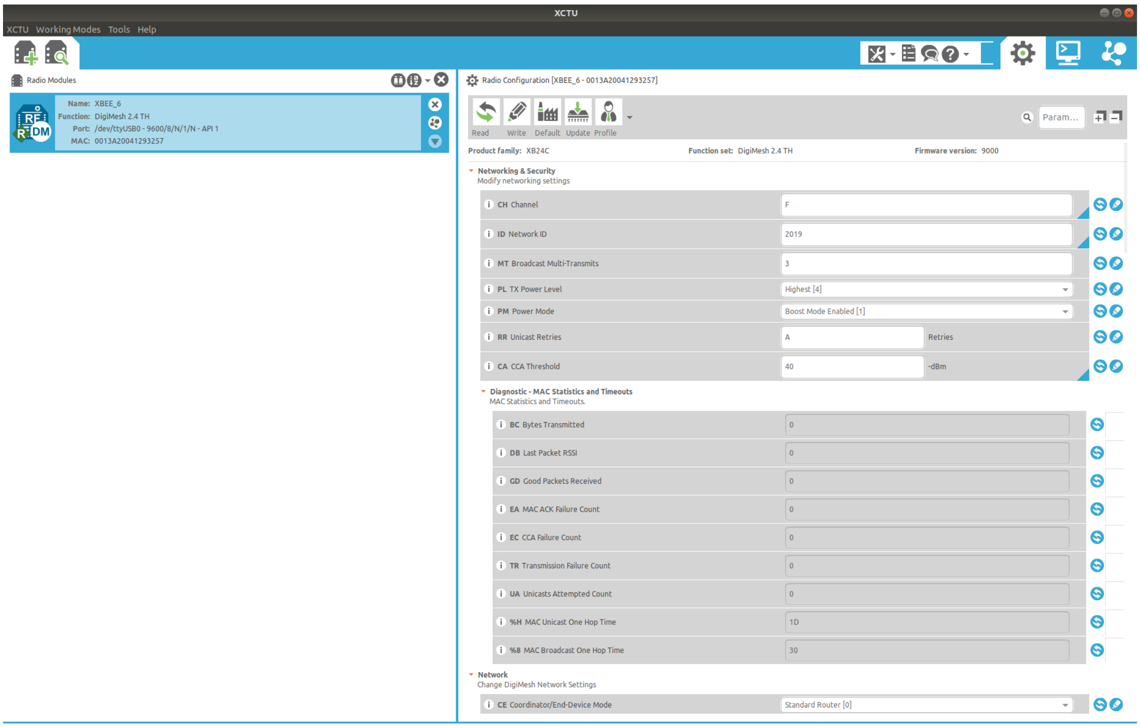


Figure 3.3: XCTU Configuration Software

3.2.5 The Testing Environment

The testing environment will be typical office environment with team and meetings rooms. As already stated in section 2.4.1, RSSI is a measurement that is highly affected by the surroundings. If a human gets in the way of a transmission the receiver will perceive a signal strength that is much lower than it otherwise would have been with a clear line of sight (LOS). Doors and other objects will of course also affect the strength of the signal, hence the system should be constructed in a way such that as few objects as possible will disturb the transmissions. This is best achieved by placing the nodes in smart positions, high up in the roof for instance, then they will not interfere to much with other ongoing activities in the office and be in the way of employees everyday work. As previously mentioned, RSSI is measured in a logarithmic dB scale, meaning that a increase of 10 dBm in the received signal is actually an increase of 10 times in power. When the nodes collect their RSSI information they implement a Kalman-filter (see Section 2.5.1). The filter is used to reduce some of the noise caused by objects that are moving in the surrounding but the filter can of course not remove all of the noise. It is not good if the area that the network is spanning is to small for several reasons. First of all it can be thought of a more economical design choice if the positioning system will be able to span a greater area, imagine a scenario where the positioning system should be extended to cover a much bigger office with more than one hundred conference rooms, then the expenses can be vastly reduced if one node is able to cover a larger area. Other than that there is also no use for a routing algorithm if the area is too small, in that case triangulation would be much more efficient since all 6 nodes would be able to communicate directly with each other without hops.

3.3 Implementation

The functionality of the IPS as specified in section 3.1 is implemented in Java using the XBee Java library briefly described in section 2.1.5. The following section aims to describe this implementation. The Java Code is structured in an object oriented manner and utilizes a hierarchical structure to minimize the amount of code. A unified model language (UML) diagram representing the implemented code is attached in Appendix A.1.

3.3.1 Searching for Neighbors

This kind of positioning is made possible by having several XBee devices that communicate with each other. Each node performs a search where only the closest located neighbors are added to a list, sometimes devices that are visible to a node will be removed since they are located to far away. This limit is set to a certain RSSI value where devices found with an RSSI lower than a specific value will be removed. The list is an array list and consists of RemoteXBeeDevices as described in 2.1.5. This list is later used when packets are transmitted to make sure that all relevant devices will receive the information as desired. This step does not assign a RSSI value to all of the devices, this is later done in the data acquisition step (see section 3.3.2) it only adds them based on their RSSI without recording any data. This neighbor search is only performed once in the initializing step for all the router nodes, it is therefore important that all devices that are intended to be used are up and running when a neighbor search is performed.

3.3.2 Data Acquisition

For every node in the network each neighbor node needs to be associated with an RSSI value. After each node in the network has obtained a list of neighbors the local node subsequently performs a collection of RSSI values related to each neighbor. These values are processed by a Kalman filter and lastly the final value is stored in a hash map, mapping RSSI values to the Node-ID of each neighbor. These RSSI values are later used as costs for edges from neighbor nodes to the local node in the vector routing. RSSI values are obtained by using an AT-command called DB [54]. This AT-command sends a packet and then, from the received packet, obtains the RSSI value using LQI.

3.3.3 Data Filtering

Data filtering is performed by obtaining RSSI values as previously described and use a Kalman filter to filter out outliers. The Kalman filter as described in section 2.5.1 uses a prediction step and an update step. For the case of estimating RSSI between router nodes the prediction and update step can be simplified and written in the following manner:

$$\bar{x}_k = x_{k-1} \tag{3.1}$$

$$\bar{P}_k = P_{k-1} + R_k \tag{3.2}$$

The variable $\bar{x}$ is the estimated RSSI and P is the covariance. According to Equation 3.1 the RSSI value is predicted to stay the same over time and according to Equation 3.2 a small process noise is added to the covariance at each prediction. This is equivalent to saying that we have slight uncertainty about our prediction. The update step is defined as:

$$x_k = \bar{x}_k + K_k(z_k - \bar{x}_k) \tag{3.3}$$

$$P_k = P_k - K_k P_k \tag{3.4}$$

The variable z_k is the latest measurement while K is called the Kalman gain and is calculated in the following way:

$$K_k = \bar{P}_k(\bar{P}_k + Q_k)^{-1} \tag{3.5}$$

By adjusting Q, the Kalman gain K can be tuned. In short a high Kalman gain means the updated value, fx_k, relies more on the latest measurement while a low Kalman gain results in an updated state that relies more on the system model and the previous state.

3.3.4 Communication and Message Handling

The XBee devices communicate with each other by sending packets that are created locally. Accessible to the programmer a XBee packet consists of a RemoteXBeeDevice address as well as some data, also called the payload. However there are some fields hidden in lower level layers that can not be edited or viewed from the network layer, for instance a start delimiter and a cyclic redundancy checksum (CRC). How these lower layer structures are built up varies with the firmware and will not be more explored here. In this particular case the payload is in the form of a byte array, byte arrays are not handy to work with if different data types should be combined as in this case where both a sequence number and each nodes distance vector should be transmitted. This is solved by forming a string and appending it with all the desired information, the different fields in the string are separated by different headers and characters which makes it possible to split it up and handle different parts separately. When a message is transmitted the string is converted to a byte array which will be received at the source nodes. These nodes convert in their turn the byte arrays back to strings again. The data in the string is extracted and depending on what the difference between the locally stored data and the received one will be, the node acts accordingly.

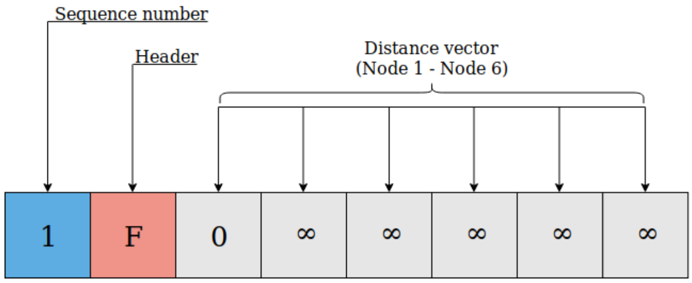


Figure 3.4: The payload of a XBee packet

The payload of a packet can be seen in Figure 3.4 where the first number represents the sequence (further explained in section 3.3.6). The second field is a header that tells the program that the distance vector begins and after that the actual distance vector comes. It contains the data for the different nodes in the distance vector and is also separated by headers which makes it possible for the program to assign the right values to the right node.

3.3.5 Vector Routing

The vector routing follows what is described in section 2.7.1 and follows the Bellman-Ford algorithm described in section 2.6.1. The DV is stored as an array list which is of equal size for all nodes, it corresponds to the size of the network. All XBee devices are manually assigned a node identifier which is a string with the word XBEE_ followed by a number. The DV is initiated by setting the element corresponding to that number to zero (in Java this is shifted by one since indexing starts at zero). This element in the DV will represent the distance from the local node to the local node which is obviously zero. All other elements in the DV are initialized to infinity, representing the known distance to all remote nodes. Figure 3.4 shows an example of a DV. This particular example represents the values that will be given to XBEE_1 during the initialization phase.

When a DV is received from a neighbor, the node identity of the neighbor is used to retrieve the RSSI value associated with it from the hash map stored in the local node. This RSSI is added to each element in the received DV and then each element is compared. If the received distance to a certain node plus the RSSI to the sender node in total is better than the current stored distance to a node, the local DV will be updated and transmitted all neighbors.

3.3.6 Sequences

A sequence is started once the start node performs its first search and transmits its first message. The sequence number is stored locally in every node and included as an integer in the payload (see Figure 3.4). This means that all nodes that receive a XBee message can see what sequence the message belongs to and at the same time compare it against their locally stored number. A new search can only be initiated from the start node and when this is done the sequence number is locally incremented by one. When the start node transmits the next message other nodes will see that a new sequence has begun, since the sequence number of the received message is higher than their local, and therefore their old distance vector (from the previous search) will be replaced by the initial one. The main idea behind the sequences is that consecutive searches should be easy to perform and if the nodes would keep their previous best results the searches would be inaccurate once the start node is moved. There is also a possibility that old messages still will be in transmission when a new sequence is started so messages received with a sequence number lower than the local is always tossed away. The sequence number serves in one way a reset functionality to all of the router nodes.

3.3.7 Mobile Router Nodes

All mobile router nodes in the network have the exact same functionality and run the same code. These nodes are idle and wait for a new message until one is received, they distinguish between searches using the sequence number and communicate its own distance vector to neighbors whenever it is updated according to the DV-routing algorithm. The technical equipment that is referred to as a router router node can be seen in Figure 3.5. All together it consists of one DIGI XBee S2C 802.15.4 RF Module, one Digi XBee Grove Development Board, one Raspberry Pi 3B+, one black chassi, one Raspberry Pi - Power Supply slim and one 16 GB Micro-SDHC memory card.

3.3.8 Mobile Start Node

The start node is programmed in a little bit different way in that it has the functionality to perform several consecutive neighbor searches with incremented sequence numbers as explained in the previous subsection. The basic idea is that the IPS should be mobile and if the possibility to make repeated searches exists it will simplify a lot if none of the nodes have to be restarted. As the system is programmed now, one can make a search from one place, get the resulting distance vector that tells what room is closest, move a little bit and rerun the search procedure with a possible changed result. Compared to the router nodes it uses a timeout while waiting to receive a message, this timeout has the intended use to return the distance vector and make it possible to perform a new search when the previous search has terminated (when no more messages are received).

The start node is connected to a laptop and in order to raise the interpretability of the result and a graphical user interface (GUI) is developed which shows the distance to all of the rooms based on the final distance vector. From button presses in the GUI events are triggered that correspond with the different functions (eg. neighbor search, data acquisition etc.)

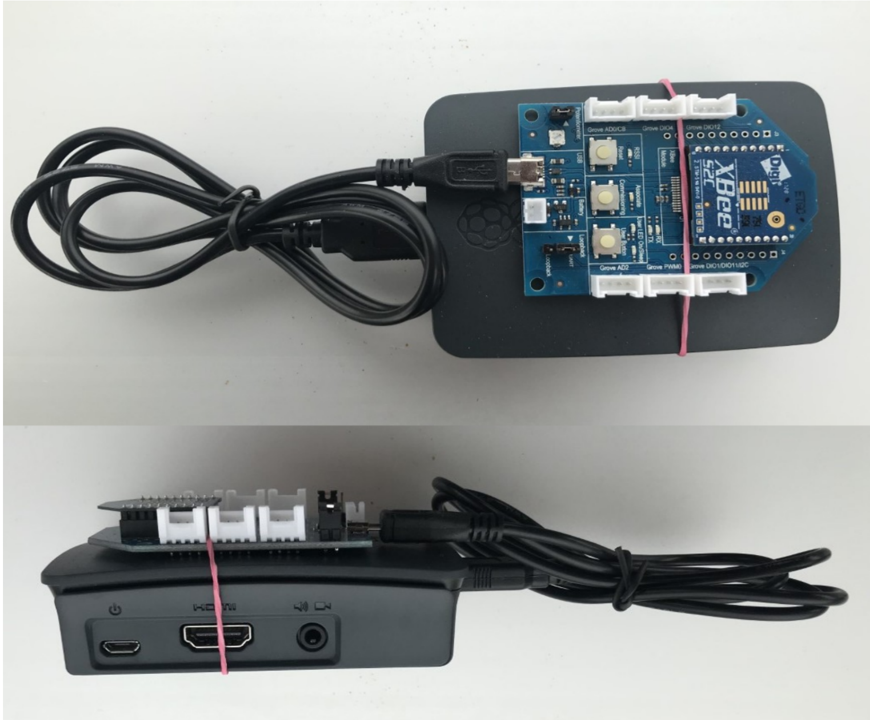


Figure 3.5: The router node configuration

4

Evaluation & Results

This chapter presents all of the results related the performance of the IPS. Various tests are performed while some of the parameters are tuned in order to achieve the best possible performance. The first test aims at determining how the RSSI measurement is affected by noise and interference from other wireless technologies depending on which channel the WSN is operating on. The second test is performed to determine if the direction of the sending and receiving device, i.e the direction of the PCB antenna, effects the signal strength. The variance in output power between devices is also tested by measuring RSSI for each device at one meter distance. To further investigate how precise the technical equipment is, nodes are lined up with one, two and three meters of separation and observed from another node, to see whether the RSSI values to each device can be distinguished from each other. In addition to this, plots from the performed test will be presented. Finally a test where the room localization will be performed while at the same time testing the over all robustness of the WSN as an IPS.

4.1 Channel Noise Level

As previously mentioned the XBees can be configured in an environment called XCTU, 16 different channels (11-26) can be selected, these can be seen in Figure 2.1. With a simple terminal command issued at a computer with a network card supporting Wi-Fi, it is possible to get the Wi-Fi channels used in the surrounding office environment that could possibly interfere with the Zigbee network. The Wi-Fi channels that usually are used are the ones shown in Figure 2.1, these are also the channels in use in the environment where tests are performed with the addition of Wi-Fi channel 7. From this, one would expect that the best channel with the least amount of noise would be either 15, 25 or 26, since IEEE 802.15.4 channel 20 would probably experience interference from Wi-Fi channel 7. Consequently these are the channels that will be tested.

The tests are set up with two devices separated with a distance of one meter. From the results in Figure 4.1 it is possible to see that the RSSI readings vary a lot depending on which channel the network is operating on. It can be observed that channel 26 reports lower RSSI than all the other investigated channels, which could mean that this channel is less affected by noise and therefore a lower RSSI is reported [55]. The RSSI measured by the XBees does not distinguish between noise and signal [56], thus higher noise also means a higher RSSI since signals in the same frequency range are added to each other. It can be argued that it would be better to choose a channel with a low noise level even if the distance is not derived from the RSSI value, since choosing a noisy channel may cause RSSI values to fluctuate more depending on how close different nodes are to Wi-Fi access points (AP). An IEEE 802.15.4 channel with a lot of interference from Wi-Fi would make some measurements accurate while others are incorrect, the final test will highly rely on that every device could be trusted in the same way, with respect to that, channel 26 is chosen and kept constant here after.

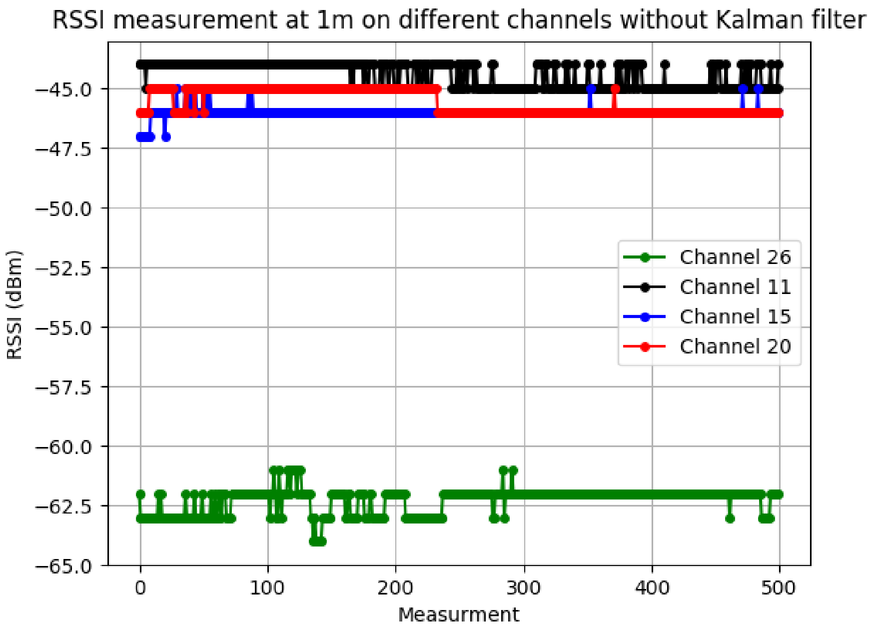


Figure 4.1: RSSI measured at for different IEEE 802.15.4 channels

4.2 Antenna Direction

The second test aims to determine in what direction the XBee devices should be oriented. Since the built in antenna in the RF module is printed to the circuit board the XBees are not equally good transmitters and receivers in every given direction. A test where two XBees are used, pointed in different directions, placed on the floor and separated with a distance of one meter is performed. The four different directions of the two devices are shown in Figure 4.3 - 4.6, where the bottom module is the one connected to the laptop and therefore the device that reads the RSSI for incoming packets, it can in this case be thought of as the receiver while the module at the top of the each picture in Figure 4.3-4.6 can be thought of as the transmitter. The result of the measurements for each direction is shown in Figure 4.2. It is possible to see that the best configuration both in the sense of signal power and consistency of measurements is to direct both devices with the flat side of the module facing each other. The worst results seems to be associated with the direction of the transmitting module (top module in Figure 4.3 - 4.6) since both the blue and red plot in Figure 4.2 corresponds to the RSSI measurements taken when the sending module has its flat side facing away from the receiving module. The measurement for configuration D, the red plot, consists of values between -65 and -89 dBm, which is quite a high variance considering that the devices only are separated by one meter. By performing this test and finding out the best direction the results can be utilized in the final test where the meeting rooms should be localized.

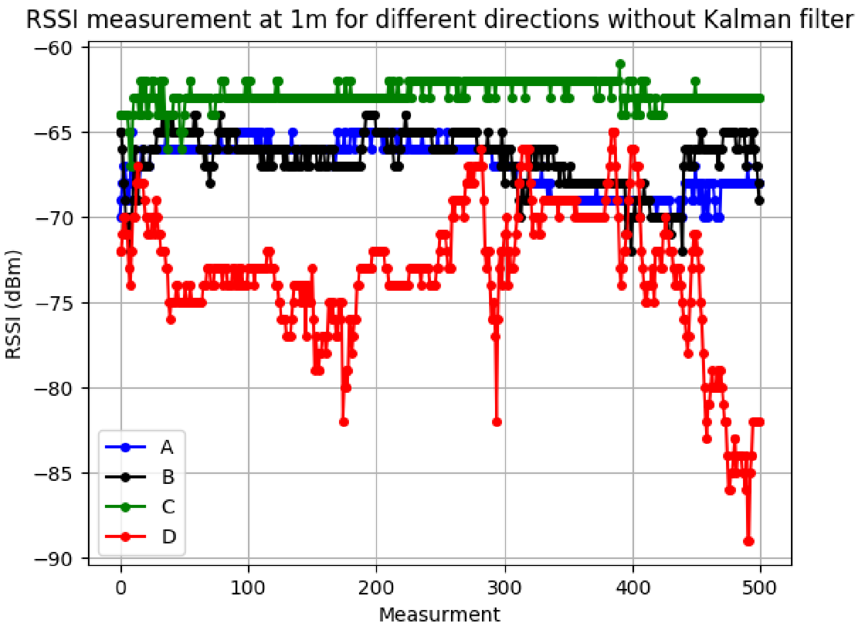


Figure 4.2: RSSI measured for different orientation of a sender and receiver

Figure 4.3: Configuration A

Figure 4.4: Configuration B

Figure 4.5: Configuration C

Figure 4.6: Configuration D

4.3 Kalman Filter Tuning

Now when the channel and the orientation have been investigated the testing procedure can continue with the devices fixed in the direction as in Figure 4.5. The Kalman filter used for data processing can be tuned to make it more or less reliable

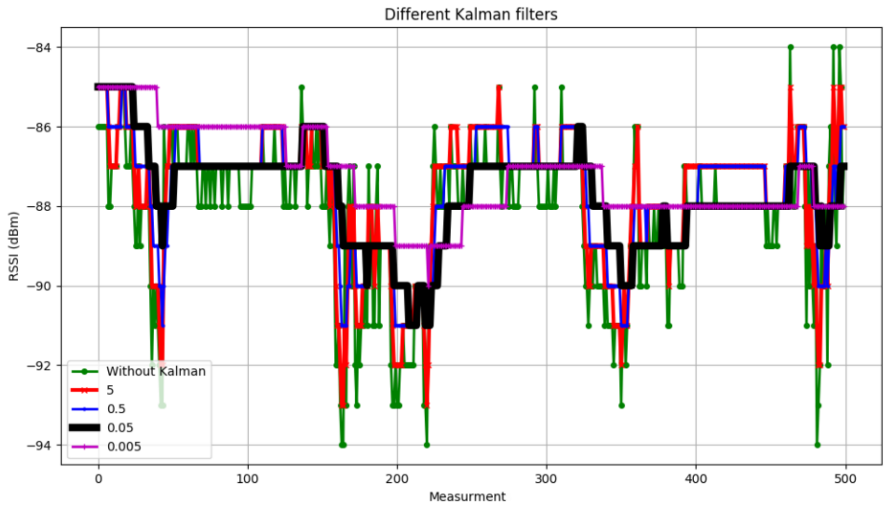


Figure 4.7: Kalman Filters tuned with different parameters

on new measurements. This is done by adjusting the value for the process noise (R). By increasing the process noise the Kalman gain is increased and the filter relies more on new measurement rather than on the model that says every RSSI value is expected to be the same as the last one. A test is performed where different Kalman gains are used to filter the measurements and then comparing how affected they are by noise. It is desirable to filter away most of the noise, but it may also lead to incorrect readings if the filter starts out with a faulty measurement and is too slow to adapt to new more accurate measurements. The test is configured with two devices located in different rooms separated by a corridor and performed with measurements close to the RSSI sensitivity threshold. It is intended to provide measurements with high variance in order to see the effect of applying the Kalman filter to smooth out the noise. The results can be seen in Figure 4.7. The process noise parameter R is selected as 0.05. From a test with 500 measurements the Kalman filter removes outliers and smooths out the resulting curve. The difference between the mean values of the curve with and without a Kalman filter is only about 0.48dBm , which accounts for a 0.5% difference from the unfiltered measurements. A Kalman filter will always result in a more accurate mean value than what would otherwise be obtained, since most of the outliers which affects the mean value are removed. The corresponding values for all tests can be seen in Table 4.1, one can observe that the span of values gets more narrow when the Kalman gain is decreased. The measurement without the Kalman filter fluctuates 10 dBm while the selected filter (with R=0.05) varies with only 6 dBm. The filter with the smallest Kalman gain performs about the same, but it is slower in adapting to changes. For subsequent tests a process noise parameter of 0.05 will be used since it smooths the measurement and the filter could at the same time follow changes that might occur if the initial measurements are faulty.

Type:	Process noise:	Min [dBm]:	Max [dBm]:	Mean [dBm]:
No Kalman	-	-94	-84	-88.174
Kalman	5	-93	-85	-87.638
Kalman	0.5	-91	-85	-87.58
Kalman	0.05	-91	-85	-87.696
Kalman	0.005	-90	-85	-87.234

Table 4.1: Values from different Kalman filters compared against each other

4.3.1 RSSI Positioning Accuracy

Another important aspect to investigate is how accurate position that can be achieved with the selected setup in an optimal environment. Optimal in this sense means that the LOS between every device is good and no objects are in the way of the RF transmissions. Five nodes are placed in an open environment on a straight line with one meter separation between each other. The chosen test location is a corridor placed on the floor in a straight line. Their RSSI:s are then measured from another node placed on the same line, if the tests indicate that it is not possible to distinguish the nodes from each other, the distance between them will be increased until it is possible to determine in which order the devices are placed. Since RSSI previously has been described as a measurement that is uncertain (see Section 2.4.1) the test may very likely show that the starting separation of one meter is not enough. Figure 4.8 shows a illustration of the test scenario where d_2 represents the distance between nodes and d_1 represents the distance from the measuring device to the closest node on the line. It is also interesting to study whether d1 affects the accuracy of the measurement. The test will start with a value of one meter of d_1 it is then adjusted until the tests provide a successful result. This experiment will of course not represent the typical scenario for where an IPS is needed, but it can deliver some wisdom to coming tests.

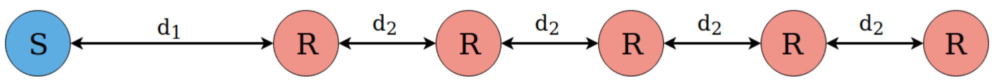


Figure 4.8: A test scenario where nodes are placed on a straight line

The results from the first test with both d_1 and d_2 selected as one meter can be seen in Figure 4.9, where 500 data points have been collected and processed by the Kalman filter. The sampling time is about 60 seconds for each channel. For this test the start node (the node collecting the measurements) is able to communicate directly with all other nodes, which means no multi-hop functionality is involved. The start node is shown in Figure 4.8 as the blue node.
As can be seen in Figure 4.9 separated by a distance of one meter it is for a some of the devices possible to distinguish between them based on RSSI readings specially when the Kalman filter is applied, however the measurements does not correspond to the distance at which the devices are placed. For example the device placed 5

meters from the start node reports higher RSSI than all other nodes placed closer to the start node. The reasons for this result could be many, the output power level of transmissions could vary between devices or small changes in the direction of the receiving and transmitting antenna could have an effect on the received signal strength or maybe the LOS for each device is not equal.

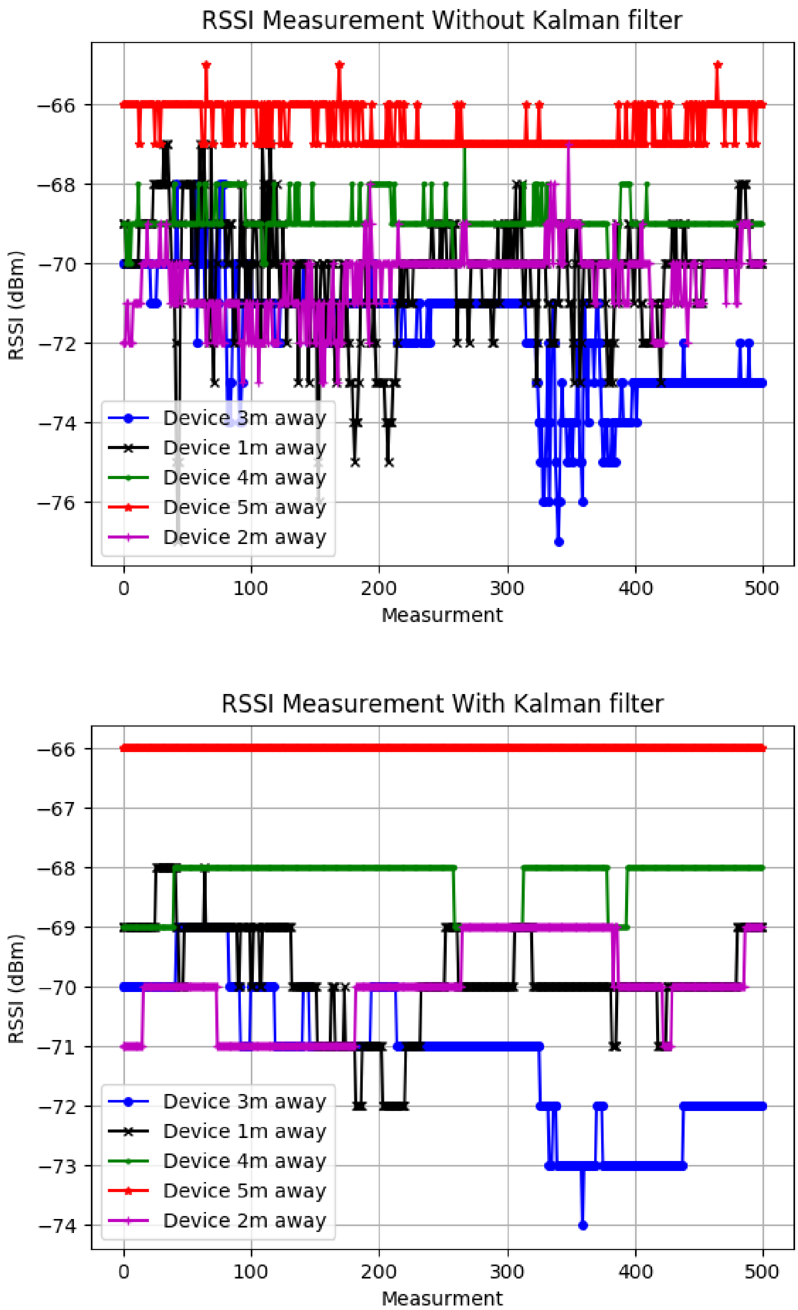


Figure 4.9: Devices placed on a line with d_1 and d_2 selected as 1m

A second test is performed where the distance between the devices is increased to two meters. The result of this test can be seen in Figure 4.10. From the result it seems like a distance of two meters is still not enough separation between devices to distinguish between them using RSSI, although the results show better ability to separate devices based on distance than was shown by the result for the tests performed with one meter separation. The separation must hence be further increased. A separation of three meters results in the measurements is shown in Figure 4.11. It appears from this test that three meters is still not big enough of a separation between devices to determine the order in which they are placed based on RSSI. For this test the configuration of each device was changed from router to end device to make sure that no routing takes place. If a message is routed, then RSSI is measured for the last hop which could lead to a device further away being assigned a RSSI associated with a device closer to the start node. As can be seen in Figure 4.11 the problem still persists with higher RSSI for some devices further away than others. For example XBEE_4, located nine meters away, results in higher RSSI measurements than the device located six meters away. However it seems like the same XBee devices show consistently high RSSI values for all performed test so far and some devices show consistently low RSSI values. For example by examining the graphs in Figure 4.9 to 4.11 XBEE_3 appears to consistently return low RSSI values.

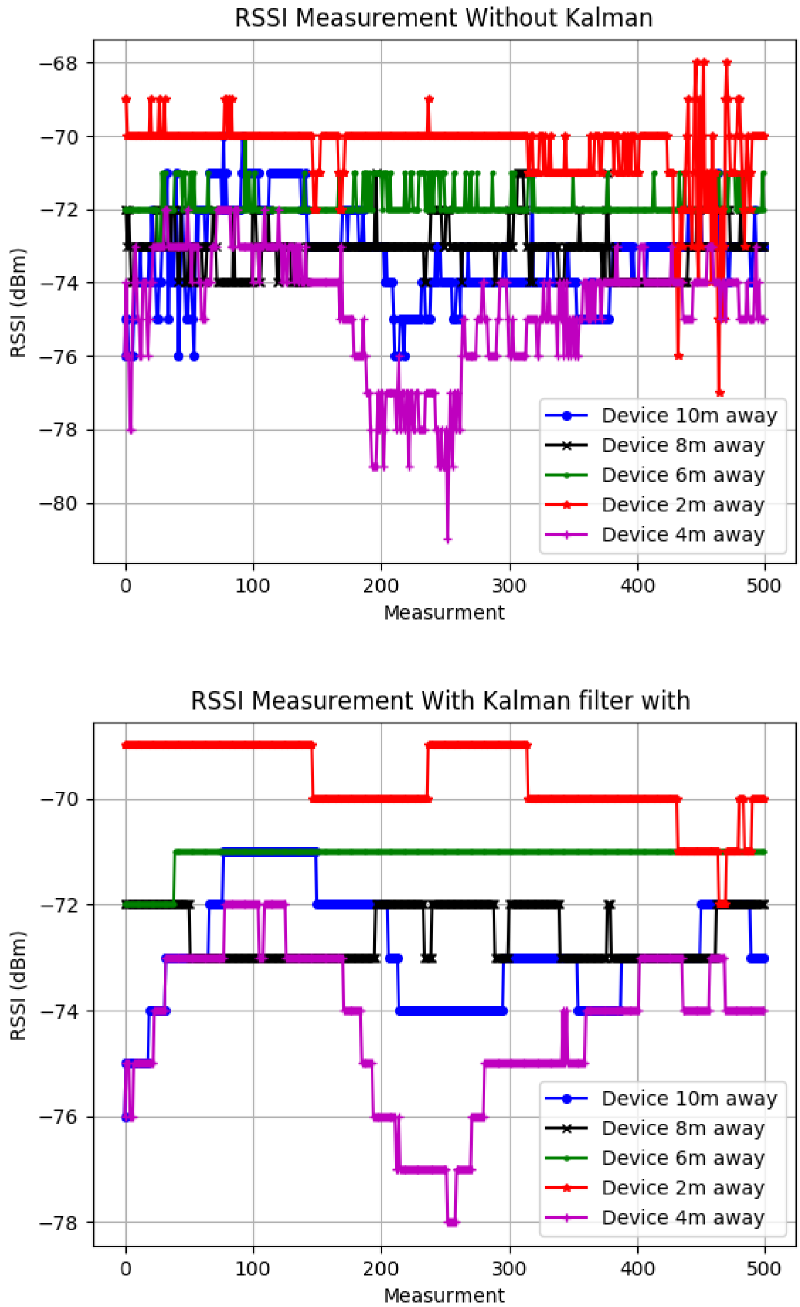


Figure 4.10: Devices placed on a line with d_1 and d_2 selected as 2m

Since these results are unexpected it may happen that the devices have irregularities in performance of receiving and transmitting packets. From the 1m test XBEE_6 appeared to be the device located closest to the measuring device, but in fact it was the complete opposite, it was farthest away. In order to find out if the same device, XBEE_6 have a higher output power than the other devices another test where the devices are isolated and separately tested is performed. The signal strength of each XBee is measured from the same device from a distance of 1m again, but now the

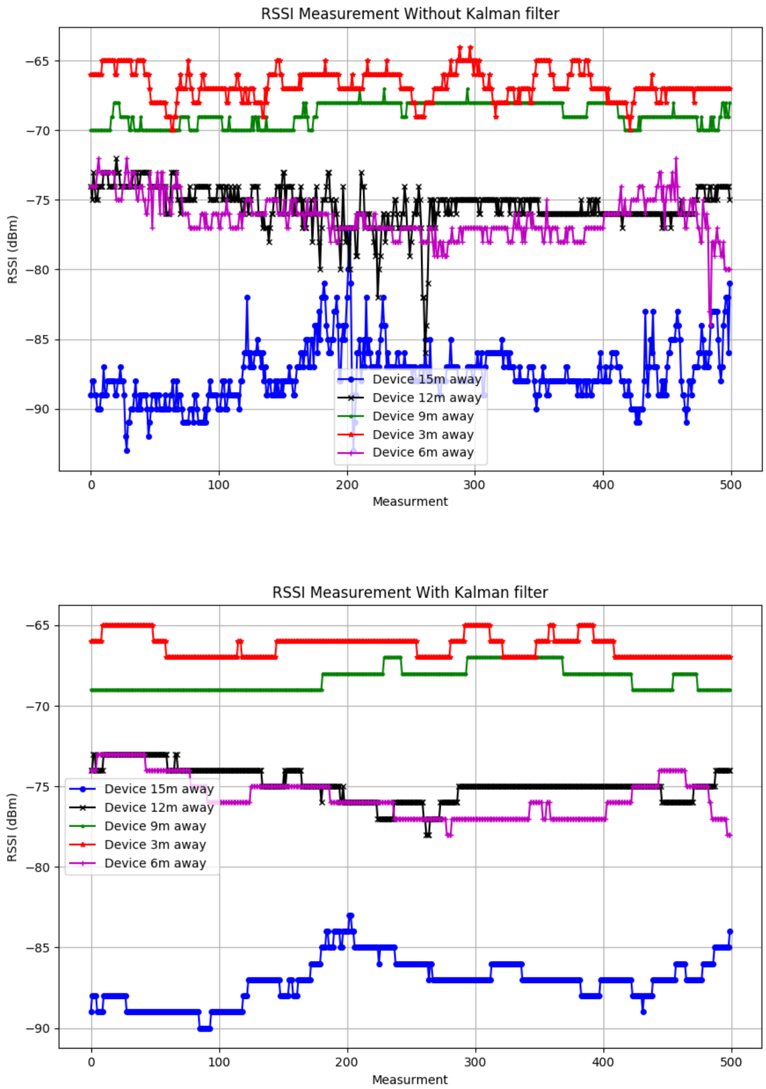
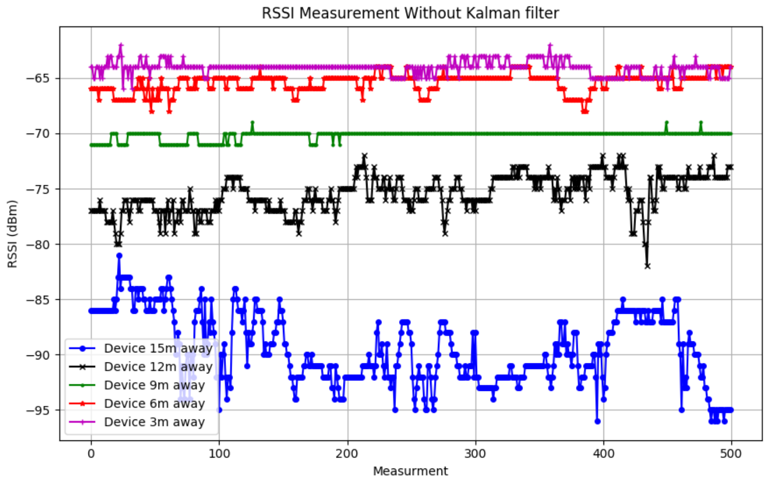


Figure 4.11: Devices placed on a line with d_1 and d_2 selected as 3m

other devices do not disturb the transmissions. Figure 4.13 shows the results with all of the tests for each XBee plotted in the same graph, it can be observed that XBEE_3 reports a lower signal strength compared to the other devices. XBEE_4 seems to be better then XBEE_3, but still a little bit lower RSSI than the rest. This test can therefore be interpreted and taken as a confirmation that not all devices are exactly equal when it comes to output signal power. If a distinct and consistent difference in transmit power is observed this could possibly be compensated for in some way. In Figure 4.13 the radio module representing XBEE_3 has been replaced with a spare radio module since this device reported consistently low RSSI compared with the other devices. The figure shows that the spare module has a higher output power than the old module and its signal level is more similar to the other devices. With the new radio module the tests performed with 3m separation are repeated to see if it is now possible to distinguish between all devices based on their RSSI. The new results can be observed in Figure 4.12. For this set of radio modules, 3m appears to be an enough separation to distinguish between devices. After using a Kalman filter the order in which the devices are placed is easy to observe. From these values an average representative RSSI can be picked and used in the routing phase. From the same graph it is possible to see that values are quite consistent at closer distances, but the farther away they are the more the values fluctuate.

The final test that will be made before the actual room localization test, is related to how far away devices can be placed to one another. We have now seen that one meter separation yields an RSSI of about -60 dBm, according to Table 2.1 the quality of such a signal is very good. If the distance is extended, at a certain threshold the success rate of packet transmission will drop drastically. The same table (Table 2.1) says that a RSSI value lower than -90 dBm will not work for this particular XBee device. This means that if devices are separated by at least one meter the values that are left are the ones between -60 dBm and -90 dBm which yields an absolute value of 30 dBm as the span, if the measurements at the same time are fluctuating a few dBm the resolution will not be especially fine granular. According to the manufacturer of the XBee devices, they have an indoor range of 60m [52], this is an overestimate. If two nodes are placed more than 30m away from each other in a corridor, packet transmissions will start to fail. From a performed test, the signal strength decreases below -90 dBm at a distance of 25m with a decent LOS. In order to achieve the stated range it seems like the surroundings needs to be nearly perfect. In order to have a reliable system it is crucial that packets transmitted also will be received, therefore no risk will be taken when it comes to the distance between units, it is better to place devices with a separation such that packets are guaranteed to be delivered. In the XCTU environment, there exists a possibility to manually configure how many retries that should be made if transmission fails. The default value is set to ten, an even higher value would mean that too much time will be spent on trying to send messages that can not be delivered and a lower value would mean some packets are discarded as they will not be successfully transmitted.



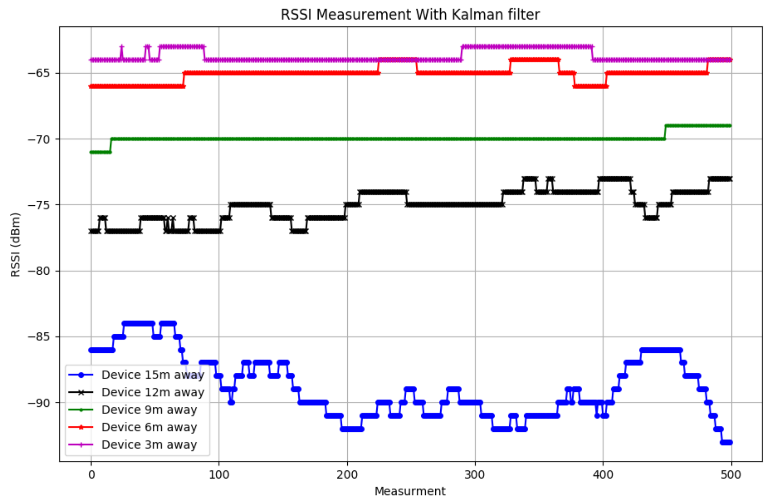


Figure 4.12: Devices placed on a line d_1 and d_2 selected as 3m, with a spare module

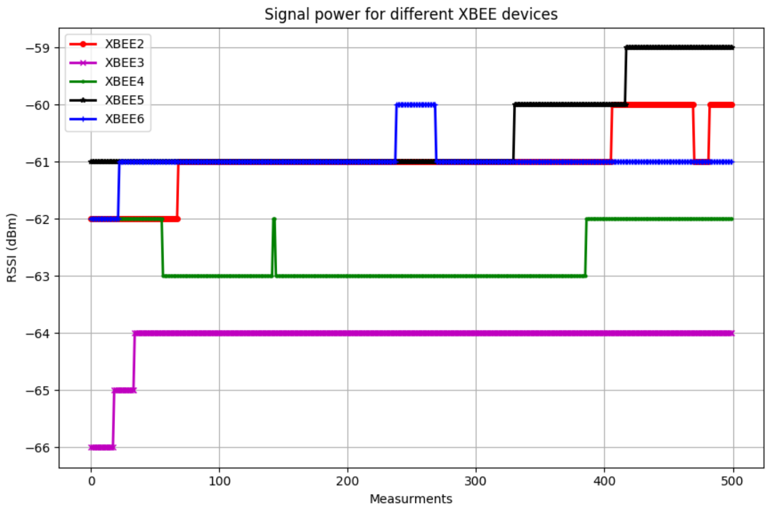


Figure 4.13: Separate measurements for each XBee compared against each other

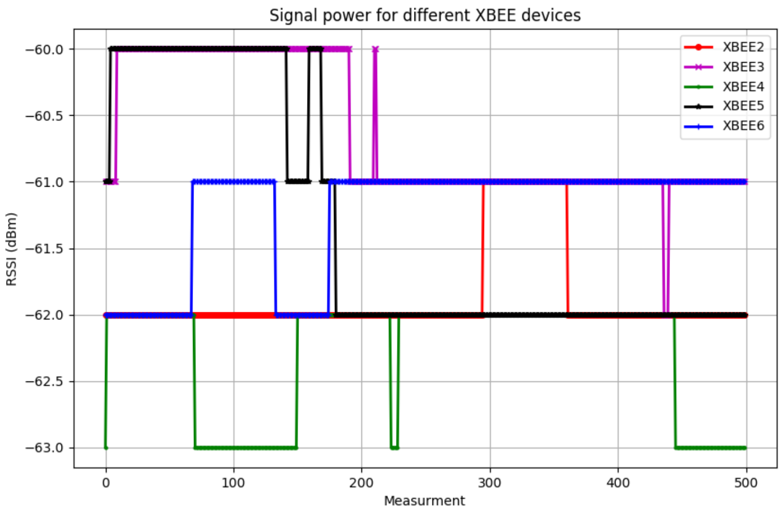


Figure 4.14: RSSI measurements with module for XBEE_3 replaced with a spare radio module

Once all of these tests are performed the results are taken into consideration when creating the final test. That test should be constructed in a way such that it answers the research questions in the best possible way, at the same time the intention is that it should be an as close to real scenario as possible in that way it will represent the true performance of the IPS. After these tests it will be possible to answer if the precision is high enough for this scope and intended application of use.

4.4 Room Localization Test

The final test will be a setup in order to replicate a typical use case for an IPS. The nodes will be stationed far enough apart to necessitate multi-hop communication and at the same time make use of what have been learned so far from the experiments. This test is intended to show the robustness of the constructed IPS and will be performed in the following way. First, the neighbor search is performed on each node, which results in communication being set up between a mobile router node and all devices within range. Then each node collects RSSI values to each node in within range keeps all remote nodes that reports an RSSI value larger than some set threshold as neighbors. Lastly the graph search algorithm is initiated from the mobile start node from an arbitrary location and the results are stored. This test is repeated from the first step to the last until it can be shown that the system is robust, meaning that it returns the same shortest path to each node in the network repeatedly.For example if two nodes are stationed in separate meeting rooms, then from an arbitrary starting location, the shortest path to each room (minimum spanning tree), returned by the search algorithm should be the same after repeated testing. The accuracy of the shortest path representing the distance, has to be confirmed with the actual distance to each room i.e that the system returns the rooms in the same order (ascending) based on their proximity to the mobile start node. This same test is then performed again with the start node at a different location where another room with is closer located and hence another result is expected, if the result from this test represents the actual scenario it can be taken as proof that the room localization system is working as intended.

In Figure 4.12 when the RSSI readings get below -80 dBm the values start to fluctuate more due to noise. To avoid this, each node will add another node as a neighbor only when the readings are above -80dBm. For communication between nodes farther apart, multiple hops will be necessary. The test will be performed, as previously mentioned by placing nodes with routing capabilities through out an office space. Some of the nodes will be placed at navigation goals of interest i.e meeting or conference rooms. With the number of devices limited to six, the IPS will be set up with three nodes located in different meeting rooms, two nodes located in the corridor and one node used as the mobile start node and connected to a laptop. This scenario is going to represent a small scale version of the typical use case. The system should, if working as intended return the distance to each of the three rooms in the correct order. The robustness of the system will be ensured by making sure the system returns the same result after repeated testing. The number of successful tests will summarized and compared with the number of unsuccessful tests for this scenario. The results are summarized in 4.5.

4.5 Localization Results

This test is setup in the office environment with the purpose to see how robust the IPS is. The test scenario is shown in in Figure 4.15. The total number of tests are 20 and the result of the test are shown in table 4.2. As can be seen the IPS performance varies from time to time and the and the results show that the IPS has a hard time to of giving the same result over and over. The "Correct Position" field in Table 4.2 is related to how many times a device is sorted in the correct order corresponding to the real world scenario. The other field, "Included in the network" is related to how many of the tests that the network is spanning that particular device. These 20 tests were performed with the start node located at location A in Figure 4.15 and the other nodes placed accordingly to the same Figure.

The reason to the varying results could be that either, some node in the network does not add the right nodes as neighbors and unnecessary hops are required and thus the distance to some nodes will not be correct, or that some nodes are not able to distinguish between its neighbors based on RSSI. The last problem is hard to fix as the major cause of it is the sensitivity of RSSI measurements. The first problem could be partly be solved by adjusting the threshold for RSSI that determines which of the remote devices to keep as neighbors. If this parameter is set as a higher value then fewer devices would be added as neighbors and the reach of each device would be lower. This would however mean that the number of nodes would have to increase and that would result in higher equipment costs. One can observe that XBEE_5 is included in the network in all tests and only incorrectly sorted once. This is probably due to that it has better LOS than most other devices. The worst results are related to XBEE_3, it was located in a room where its neighbour had a hard time to find it. For this scenario the IPS would most likely benefit from adding one extra node in the corridor between node 1 and node 5. Other than that the activity in the office where the tests were performed have to be considered as normal. People were moving back and forth without being aware that the tests were performed.

Device:	Correct Position	Included in the network
XBEE_1	14/20	15/20
XBEE_2	13/20	17/20
XBEE_3	11/20	17/20
XBEE_4	14/20	17/20
XBEE_5	19/20	20/20

Table 4.2: Robustness results from 20 performed tests with the nodes located according to Figure 4.15

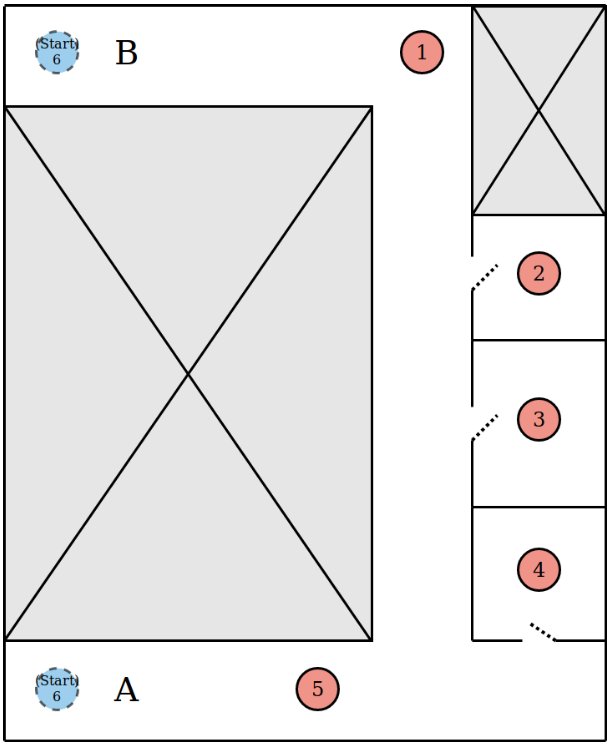


Figure 4.15: A visual representation of the mobility test where the two different locations A and B are marked

In order to test the mobility of the start node two different locations, A and B, are selected as positions (can be seen in figure 4.15) from where the algorithm should be started from. Since the previous test shows that the robustness of the system is varying one has to make sure that all router nodes find their closest neighbours before the algorithm is started. If the IPS works as expected the routing algorithm should return room 4 (XBEE_4) from location A and room 2 (XBEE_2) from location B, the corresponding results can be seen in Table 4.3. The devices are sorted in the correct order for the both locations and the routing consists of several hops. As already mentioned the RSSI itself related to each node are not of high concern, instead the order is the interesting metric to look at. From the table it is possible to see that all nodes are sorted in the correct descending order according to both locations.

Device:	Location A RSSI [-dBm]	Location B RSSI [-dBm]
XBEE_1	303	76
XBEE_2	220	159
XBEE_3	151	226
XBEE_4	82	297
XBEE_5	55	376

Table 4.3: A mobility test where the algorithm is started from two different locations

5
Discussion

This chapter is intended to be a discussion about the performance of the IPS. Some of the problems that were encountered are brought up and analyzed. Ideas about how to avoid these problems are presented. One section is related to how the IPS performs, does it meet the expectations? This is followed by a section revisiting the research questions stated in section 1.3 that are carefully answered. Finally some future improvements for the IPS are brought up with the purpose to inspire further development of the project.

5.1 Encountered problems

Some problems have been encountered during the project, most of them have been solved. A lot of information can be retrieved from these problems and hopefully someone facing the same tasks can avoid stepping into these pitfalls. The ones discussed here are the ones which took a considerable amount of time. Some insights about how some of them can be solved will be given.

5.1.1 The Mutual Neighbor Problem

The RSSI value representing the cost of a transition is collected and stored on the device that receives an incoming distance vector and for this reason communication needs to be mutual. If a node in the network receives a distance vector from a remote node that is not in the list of neighbors then communication is not mutual and no RSSI value is associated with that remote node, hence distance vector routing can not be performed in the correct way. It is thus important to make sure that all communication in the network is mutual. The reason for that two devices are not adding each others as neighbors could be differences in output power between devices or that a change in the environment is occurring between the two RSSI acquisitions. A phase where every device interacts with each other and makes sure that they have added each other as neighbours can be developed and would solve the problem.

5.1.2 Packet Throughput

Only one device can transmit over the same channel at any given time, if two devices are trying to send a packet at the same time one of the devices will have to wait until the channel is available. A fairly good solution to this can be achieved by adjusting the CCA parameter, but some transmissions will still fail. During the neighbor search phase, the devices collect 100 RSSI values to each neighbor and sometimes more than one device tries to access the channel at the same time, after a certain number of retries the developed code will throw an exception resulting in failure to collect a RSSI-value. This has been solved by taking the same value as the most recent successful RSSI request. The number of retries, which can be modified using an AT command, is the number of retries specified in addition to the three retries that are defined in the MAC layer. That means that for every retry specified by the user, the MAC layer will use three retries. For example if ten retries are specified then 30 retries are used. As a result of increasing the number of retries the chance of successful delivery of packets increases, however it also slows down the process of collecting data significantly if the link between two devices is bad and all retries must be used every time a RSSI value is collected.

The problem with packet delivery emerges when devices communicate at a distance corresponding to the RSSI threshold for reliable packet delivery. To make sure all devices are within reach and that all packets are successfully received in the network, a lower limit is set for adding a neighbor. This will result in a network where all nodes have a subset of nodes in the network as neighbours with reliable communication between all nodes.

The packet throughput can affect the scalability in terms of the set up time for the system. If many nodes are placed far away from each other the initial neighbor search will take longer time since more nodes might have to be restarted until a spanning tree is achieved. Messages used in the routing algorithm phase might also take longer time to transmit since retries may have to be made. This can only be solved by placing nodes within better range to each other.

5.2 Research Questions Revisited

This section will revisit the formulated research questions (see section 1.3) and try to answer them as accurately as possible. The results from the performed test together with related literature can be used to support assumptions and draw conclusions. The first research question was:

How should a Zigbee network topology be structured in order to be able to locate indoor objects (e.g conference rooms) in an office environment in the most efficient way?

This has been carefully investigated during the literature study, the implementation as well as during the testing phase. Since both the Zigbee and Digimesh protocols have underlying functionalities that supports the action to automatically form networks, the first idea would be to just use these. In this particular case though, additional features were of high relevance, therefore these inherent functionalities are no option. The developed network is built up based on the signal strengths between every node, nodes that are positioned close to each other will add each other as neighbours and a network map will be formed based on this metric. As a consequence, the same network might not be formed every time. The topology will be formed based on the environment. In this network several of the described network topologies can be identified.

Depending on how the minimum RSSI value that determines which remote nodes are kept as neighbors is selected, the network will look different if initiated in the same environment repeatedly. If some devices are placed far away from others such that they only have one neighbor the star topology can be identified. On the other hand, if they are placed within close range a mesh network will be formed. One important factor is that the network spans all of the nodes, otherwise the routing procedure will not work. Problems can possibly also occur if the environment changes between the initial neighbor search and the routing step, this is because the signal strength is measured in the initial setup phase only. It is of course possible to measure it periodically, but that would also mean a longer run time. The units are not capable of doing several things at the same time, if a neighbor search is initiated in order to collect RSSI values the units are occupied until the search is finished and a new search procedure can be started. To summarize the proposed network topology is therefore an agile mesh network, since the nodes will be stationed close enough to facilitate mesh communication. This network will periodically adapt to the surroundings. It is possible to achieve higher precision and faster a IPS if the system is designed for a specific environment, but with that comes a trade-off in how robust the system is to moving objects and changes in the room layouts.

Another factor that speaks for an agile mesh network is the ability to route packets in such a network. With its flexibility it is possible to add several nodes and just rerun the algorithm and the packets will be routed to each and every node in the network.

What graph search algorithm's is most suitable, given the selected structure?

A network topology is achieved after the neighbor search has been performed and a routing scheme is necessary to determine which route is the shortest and subsequently represents the shortest distance. The IPS is built up by nodes that are configured in exactly the same way, except for one mobile start node. Identical software can run on each of the router nodes and they do not rely on each other in order to work, instead nodes can be removed or added in-between every run of the routing algorithm without any problems occurring. The advantages with such a structure is that if one or several of the nodes are witnessing problems or dropouts from the network the algorithm will still be able to run and deliver a result with the dead node reporting a distance of infinity. The selected algorithm is a distributed version of Bellman-Ford which means that all nodes store and manage their own distance vectors, with each element corresponding to the distances to other nodes in the network. At the same time, from one point of view, the control structure can be seen as centralized, considering that the start node, the node connected to the computer, is responsible to trigger the other router nodes to start. This kind of approach is called "source initiated" [57], router nodes are in their listening state until a message, sent from the start node is received. The source initiated control structure minimizes communication overhead [57] since no communication occurs before the source initiates it.

The choice of routing protocol also means choosing a graph search algorithm for finding the shortest path between nodes . Distributed routing protocols often rely on the distributed Bellman-Ford algorithm. Other protocols where one or several nodes have complete information about the network often use Dijkstra's algorithm.

Link state routing as explained in 2.7.2 uses Dijkstra's algorithm which as already mentioned in section 2.6.1 has a faster convergence time than Bellman-Ford. Link state routing consequently uses less transmissions to converge than distance vector routing but on the other hand uses more memory and computational power. It is also harder to implement than distance vector routing.

Distance vectors as used in our solution also borrows the concept of sequence numbers. Sequence numbers are used to discard old routing information in the network to prevent looping and counting to infinity problems. A sequence number is used in our design but instead of associated with every entry and updated periodically, the sequence number is updated every time the mobile start node initiates a search and is associated with the whole distance vector.

AODV is as already mentioned the networking protocol used in the Zigbee stack. It is designed to be very robust to changes in network topology and is able to handle mobile nodes. Its benefits are in handling networks where only a subset of routes more frequently used than others are stored. AODV also establishes routes with a single source and a single, already known destination. The designed IPS however is concerned with the shortest path from a start node to all nodes in the network thus making AODV unfit for the task. AODV also does not follow any specific graph search algorithm but will use the shortest path found when broadcasting the request.

Is the precision high enough for an everyday use in an office in order to locate the right conference room?

The choice of algorithm does not have a significant impact on the accuracy of the IPS. The accuracy is more closely connected with the placement of nodes and the quality of the signal strength measurements. Looking back at the model described in section 2.4, phase III is where the algorithm runs. Phase I collects the data whereas phase II performs the preprocessing of the data. Both phase I and II, sets the standard for how high accuracy can be achieved and the algorithm subsequently runs based on these values.

Several of the most common routing protocols rely on the use of hop count to determine the shortest path in a network. For the proposed IPS the use of hop count as a metric of distance would amount to a network with much higher density of nodes. In such a network none of the nodes would be able tell which neighbor is the closest and to distinguish between the distance between the two rooms would require that the number of hops between the two rooms differs. On the other hand an algorithm that relies on RSSI as a metric of distance would be able to distinguish between neighbor nodes based on their RSSI value. Additionally the number of hops also has a significant impact since RSSI increases drastically with the number of hops. The use of RSSI as a metric of distance does result in a network which is much more sparse than a network that only relies on the number of hops. One drawback however is that the use RSSI in our IPS requires collection of data while using hop count does not.

As explained in section 4.1, the channel on which communication occurs can to different degrees be effected by background noise, it should preferably be selected as the one which is least affected by noise. Signal strength can be measured on different channels to determine the least affected one. The Zigbee protocol specifies functionality to perform an energy scan for each channel and choose the one with the least amount of noise. This is however only performed one time when the network is setup and the circumstances could change during the lifetime of the network. Since RSSI has been proven to be a measurement very prone to noise, not only due to attenuation and multi-path effects, but also by the orientation of the devices and fabrication inequalities resulting in differences in transmission power. The setup of a WSN intended for IPS based on RSSI would have to be carefully done. Consideration for the specific environment and the distance between devices would have to be taken which makes it a less appealing solution for an IPS. Also the less precise the RSSI measurements are between devices the more the accuracy depends on the placement of sensors in the WSN and the routing of distance vectors. Filtering techniques can to a high level be used to improve the stability and reduce noise to get a higher precision but the technique in combination with signal strength as a metric can still not replace other indoor positioning techniques when it comes to high accuracy.

What are the practical limitations?

The practical limitations of the IPS have already been touched upon. The range that two devices can successfully communicate over without substantial packet loss was after testing confirmed to be only about 25 meters. This could be a drawback for places where sparsity between nodes is desirable. For most cases indoors the range will not be a problem since a higher density of nodes is necessary to perform more accurate localization. As it is of now, the signal strength can be used to distinguish between devices located around 3m away from each other.

The WSN might if the network is not setup properly, route packets between devices in a way that is a bad representation of the indoor environment for example cut through walls. This might lead to the IPS returning the rooms in the wrong order by distance. For this specific purpose, to locate rooms, it might also happen that signals travel between different floors in a building. The height difference can not be read from transmissions which means that problems could likely occur if the IPS spans several floors. It is impossible to see exactly what path signals are traveling, but if triangulation is used such a height difference could be derived from trigonometry.

5.3 Performance

After the tests (described in Chapter 4) have been performed it is finally possible to say something about the performance of the indoor positioning system. It is clearly not as accurate in providing an actual location as other IPS where more expensive hardware is used, LIDAR for example, where some [58] have achieved sub one meter accuracy in their positioning. The accuracy was neither not of a great concern or expected to be high since related projects, [43] [20] [21] reported a precision between 1.5 and 3m. With this in mind it was still interesting to study how high precision that could be achieved. From our tests, a distance of 3m between devices is enough to be able to distinguish between them. The rooms selected for the final and entire room localization test were located with a distance of more than 3m away from each other, but problems still occurred. These are all most likely related to how the signals propagates. Nodes in the IPS are having a hard time to find all their required neighbors in order for the network to span all of the nodes and sort the rooms in the correct order. On the other hand, if the system it set up once in the correct way with all nodes restarted until they have found their correct neighbours the rooms will be sorted in the right way if the mobile start node is moved. As previously mentioned the parameter that determines if neighbors should be added or not is selected the same (-85dBm) for every node, this is mainly due to that the system should be ad-hoc and mobile. There is a trade-off related to this choice, all parameters could instead be selected with the respect to each node's surrounding and a better result is expected. That would mean that the system would become much more stationary and not as scalable as it is today. Adding a device then would mean that not only its parameter would have to be tuned, also the parameter of its neighbors may have to be re-tuned.

5.4 Possible Future Improvements

As it is of now, the IPS does not provide turn by turn navigation to the closest room that is returned after the routing procedure. The algorithm will just return the XBees sorted in an ascending order based on their signal strengths. Imagine an IPS like the designed one, but scaled for a larger office area. It is very likely that people using the IPS for the purpose to book a meeting room are not aware of the office layout, therefore a more descriptive instruction would be useful. As suggested, one could provide a map with rooms paired with the corresponding XBee devices but an even better solution would be to develop a mobile application that could be used to navigate the office environment. In the Digi XBee product catalogue there are some devices that have Wi-Fi embedded in the RF module, if the current modules were replaced by these ones one could extend the IPS and the applications where it can be used a lot. This does however deviate a bit from the scope of the thesis since one of the main tasks was to survey how Zigbee protocols in combination with devices implementing these could be utilized. The current IPS does only sort the XBees based on their added signal strengths from all hops. A system that is updated in real time at a mobile phone could be used exactly as a GPS is used. When the user with a desire to reach a specific room walks in the wrong direction and the RSSI to that specific room decreases he could be notified via an app. Another problem that possibly could occur when someone uses the IPS today is that the localized meeting room may be booked already. As of today, the system can not sense whether the meeting room is occupied or not, this could possibly be solved with a sensor connected to the XBee feeding it with observations from the room. Even better would be to integrate a digital booking system with the designed IPS. If a user friendly front end were developed to the system more people would likely want to use the IPS compared to how it works now.

If the accuracy of the IPS is further improved the system could also be used for other things than locating rooms. As already mentioned the reasons to use Zigbee are mainly its low cost and energy consumption, the technology does not target more advanced corporate applications, but it can still deliver reasonably accurate results. If another metric than RSSI were to be used, even better results could probably have been achieved. This could have been triangulation or even better fingerprinting. The main reason to why a high accuracy can not be delivered from RSSI is because it as mentioned is highly affected by the surroundings. If a sufficient number of measurements were taken in advance for the specific area where the IPS should operate, more exact knowledge would have been gained about how the signals propagates in the same area. The more time that is spent on building up the grid the more accurate positioning can be achieved. With this comes a trade off, it is very time consuming. Imagine if this IPS were scaled up, from six nodes to instead 30 or even more than 100, taking measurements for that many nodes will require a lot of time and measurements and would have to be done simultaneously. For even bigger systems the fingerprinting solution might even be unattainable due to the enormous amounts of measurements that need to be taken and stored [59].

6
Conclusion

To summarize an indoor positioning system that is able to locate meeting rooms and sort them based on distance has been designed. A lot of work has been put into investigating whether Zigbee as a technology is suitable for this specific purpose or not. The bottom line is that it is applicable for some usage areas, if accurate, sub 3m, positioning results are desired other technologies should be investigated. This is mainly because signal strength not is the most accurate metric and highly affected by noise originated from other devices co-existing in the 2.4 GHz band. This results in that filtering techniques are required to stabilize the measurements. In addition to attenuation and multipath effects due to building structures, the designed IPS also suffers from interference, direction of the antenna and variation in output power between device issues. The problem of interference is easily fixed by choosing a channel with less energy, which could be done either manually or automatically. The direction of the antenna is however a larger problem since the sensor network is supposed to be ad-hoc and carefully placing nodes takes time and defeats some of the purpose of using a ad-hoc WSN. The relative output power between devices is an even larger problem since it is infeasible to make sure for every new node that the output power is high enough. One can try to pair devices that show the same behaviours in output power, but that is all that can be done unless special designed antennas and transmitters are added.

When it comes to the algorithm the selected one is a distributed version of Bellman-Ford with dynamic routing tables and sequence numbers. The algorithm is source initiated from a mobile node from where all distances are calculated in relation to. All nodes in the network manage their own distance vector that is updated when a better distance is found. The IPS have an accuracy of about 3m and even though the testing has been done with only six devices it has been designed to be highly scalable. From tests where devices are placed in meeting rooms the results are reliable if the router nodes are setup in the correct way, meaning that they find all the contrived neighbors and the network is spanning all nodes. If placement of the mobile router node is altered, different rooms will be returned by the IPS as the closest ones. In summary the thesis and investigation about Zigbee indicates that it is possible to use Zigbee for the specific purpose to perform an indoor localization, but if high accuracy is desired other technology should be considered.

Bibliography

[1] Understanding rssi. https://www.metageek.com/training/resources/understanding-rssi.html. Accessed: 2019-04-03.

[2] Dr. Rajesh Mehra and Abhishek Singh. Real time rssi error reduction in distance estimation using rls algorithm. pages 661–665, 02 2013.

[3] George Dedes and Andrew G Dempster. Indoor gps positioning-challenges and opportunities. In *VTC-2005-Fall. 2005 IEEE 62nd Vehicular Technology Conference, 2005.*, volume 1, pages 412–415. Citeseer, 2005.

[4] Hui Liu, Houshang Darabi, Pat Banerjee, and Jing Liu. Survey of wireless indoor positioning techniques and systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(6):1067–1080, 2007.

[5] Combitech | digitalisering med människan i fokus. https://www.combitech.se/. Accessed: 2019-04-03.

[6] Zigbee standard timeline. https://www.zigbee.org/the-zigbee-alliance-celebrates-15-years-and-a-decade-of-standards/. Accessed: 2019-04-09.

[7] Zigbee alliance. https://www.zigbee.org/. Accessed: 2019-04-03.

[8] H. J. Patel, M. A. Temple, and R. O. Baldwin. Improving zigbee device network authentication using ensemble decision tree classifiers with radio frequency distinct native attribute fingerprinting. *IEEE Transactions on Reliability*, 64(1):221–233, March 2015.

[9] Vanika Pahwa. 5 reasons zigbee is ideal for smart homes. https://www.einfochips.com/blog/5-reasons-zigbee-is-ideal-for-smart-homes/, Sep 2017. Accessed: 2019-04-02.

[10] IPW Group et al. Part 15.4: Low-rate wireless personal area networks (lr-wpans). *IEEE, IEEE Standard for Local and metropolitan area networks IEEE Std*, 802:4–2011, 2011.

[11] Digi International. *XBEE 802.15.4 PROTOCOL COMPARISON*. Datasheet.

[12] Drew Gislason. *Zigbee wireless networking*. Newnes, 2008. Chapter 1.

[13] Drew Gislason. *Zigbee wireless networking*. Newnes, 2008. Chapter 7.

[14] Drew Gislason. *ZigBee and IEEE 802.15.4 Protocol Layers*, pages 33–135. 12 2008. Chapter 3.

[15] Jean-Philippe Vasseur and Adam Dunkels. Interconnecting smart objects with ip: The next internet. 01 2010.

[16] Wirless mesh networking: Xbee vs. digimesh. https://www.digi.com/pdf/wp_zigbeevsdigimesh.pdf.

[17] Joel K. Young. *Untangling the Mesh The Ins and Outs of Mesh Networking Technologies*. Digi International Inc, 9350 Excelsior Blvd Suite 700, Hopkins, MN 55343, USA, 1 edition, 1 2015.

[18] Brandon Moser, Mohammad Farooqi, and Rubén Moral. Xbeejavalibrary. `https://github.com/charlespwd/project-title`, 2019.

[19] Andreas Fink and Helmut Beikirch. *RSSI-based Indoor Localization using Antenna Diversity and Plausibility Filter*. 2009.

[20] Eladio Martin, Oriol Vinyals, Gerald Friedland, and Ruzena Bajcsy. Precise indoor localization using smart phones. pages 787–790, 10 2010.

[21] Eiman Elnahraway, Xiaoyan Li, and Richard Martin. Poster abstract: The limits of localization using rss. 04 2019.

[22] C. A. Boano, N. Tsiftes, T. Voigt, J. Brown, and U. Roedig. The impact of temperature on outdoor industrial sensornet applications. *IEEE Transactions on Industrial Informatics*, 6(3):451–459, Aug 2010.

[23] Alireza Shojaifar. *Evaluation and Improvement of the RSSI-based Localization Algorithm: Received Signal Strength Indication (RSSI)*. 2015.

[24] U. Bekcibasi and M. Tenruh. *Increasing RSSI Localization Accuracy with Distance Reference Anchor in Wireless Sensor Networks*. 2014.

[25] Guoqiang Mao and Bari Fidan. *Localization algorithms and strategies for wireless sensor networks*. Information Science Reference, 2009.

[26] Martin Woolley. *Bluetooth Core Specification v.5.1*. Bluetooth, 1 edition.

[27] Bluetooth direction finding: Angle of arrival (aoa) and angle of departure (aod). `https://www.silabs.com/products/wireless/learning-center/bluetooth/bluetooth-direction-finding`, Mar 2019. Accessed: 2019-04-03.

[28] Brian O' Keefe. Finding location with time of arrival and time difference of arrival techniques, 2017. Senior Project Tech Notes.

[29] X. Tian, S. Zhu, S. Xiong, B. Jiang, Y. Yang, and X. Wang. Performance analysis of wi-fi indoor localization with channel state information. *IEEE Transactions on Mobile Computing*, pages 1–1, 2018.

[30] Peter S Maybeck. *Stochastic models, estimation, and control*, volume 3. Academic press, 1982.

[31] A. L. Dimeas and N. D. Hatziargyriou. Operation of a multiagent system for microgrid control. *IEEE Transactions on Power Systems*, 20(3):1447–1455, Aug 2005.

[32] D.M. Dilts, N.P. Boyd, and H.H. Whorms. The evolution of control architectures for automated manufacturing systems. *Journal of Manufacturing Systems*, 10(1):79 – 93, 1991.

[33] and and. Map navigation system based on optimal dijkstra algorithm. In *2014 IEEE 3rd International Conference on Cloud Computing and Intelligence Systems*, pages 559–564, Nov 2014.

[34] Jarrell Waggoner and Jimmy Cleveland. The optimality of a*. `https://cse.sc.edu/~mgv/csce580f08/gradPres/clevelandWaggonerAstar080915.pdf`, September 2008.

[35] M. E. Falou, M. Bouzid, and A. I. Mouaddib. Dec-a*: A decentralized multiagent pathfinding algorithm. In *2012 IEEE 24th International Conference on Tools with Artificial Intelligence*, volume 1, pages 516–523, Nov 2012.

[36] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*, pages 588–595. MIT press, 2009.

[37] Martin Mauve, Jorg Widmer, and Hannes Hartenstein. A survey on position-based routing in mobile ad hoc networks. *IEEE network*, 15(6):30–39, 2001.

[38] G. Q. Gao and L. Lei. An improved node localization algorithm based on dv-hop in wsn. In *2010 2nd International Conference on Advanced Computer Control*, volume 4, pages 321–324, March 2010.

[39] John M McQuillan. The birth of link-state routing. *IEEE Annals of the History of Computing*, 31(1):68–71, 2009.

[40] Charles Perkins, Elizabeth Belding-Royer, and Samir Das. Ad hoc on-demand distance vector (aodv) routing. Technical report, 2003.

[41] Isaac Amundson and Xenofon D Koutsoukos. A survey on localization for mobile wireless sensor networks. In *International Workshop on Mobile Entity Localization and Tracking in GPS-less Environments*, pages 235–254. Springer, 2009.

[42] Yanying Gu, Anthony Lo, and Ignas Niemegeers. A survey of indoor positioning systems for wireless personal networks. *IEEE Communications Surveys & Tutorials, 11 (1), 2009*, 2009.

[43] Giovanni Zanca, Francesco Zorzi, Andrea Zanella, and Michele Zorzi. Experimental comparison of rssi-based localization algorithms for indoor wireless sensor networks. In *Proceedings of the workshop on Real-world wireless sensor networks*, pages 1–5. ACM, 2008.

[44] Neal Patwari, Joshua N Ash, Spyros Kyperountas, Alfred O Hero, Randolph L Moses, and Neiyer S Correal. Locating the nodes: cooperative localization in wireless sensor networks. *IEEE Signal processing magazine*, 22(4):54–69, 2005.

[45] Azzedine Boukerche, Horacio ABF Oliveira, Eduardo F Nakamura, and Antonio AF Loureiro. Localization systems for wireless sensor networks. *IEEE wireless Communications*, 14(6):6–12, 2007.

[46] Harmanpreet Kaur and Amol P Bhondekar. *Routing Protocols in Zigbee Based networks: A Survey.*

[47] Jamal N Al-Karaki and Ahmed E Kamal. Routing techniques in wireless sensor networks: a survey. *IEEE wireless communications*, 11(6):6–28, 2004.

[48] C. Yang and H. Shao. Wifi-based indoor positioning. *IEEE Communications Magazine*, 53(3):150–157, March 2015.

[49] Srdjan Čapkun, Maher Hamdi, and Jean-Pierre Hubaux. Gps-free positioning in mobile ad hoc networks. *Cluster Computing*, 5(2):157–167, 2002.

[50] D. Niculescu and B. Nath. Ad hoc positioning system (aps). In *GLOBECOM'01. IEEE Global Telecommunications Conference (Cat. No.01CH37270)*, volume 5, pages 2926–2931 vol.5, Nov 2001.

[51] wireless connectivity kit. `https://www.digi.com/WirelessConnectivityKit`, 2017. Accessed: 2019-04-03.

[52] Digi International. *DIGI XBEE® S2C802.15.4 RF MODULES*. Datasheet.

[53] What is maven? `https://maven.apache.org/what-is-maven.html`.

[54] *XBee/XBee-PRO S2C 802.15.4 RF Module User Guide*. Digi International Inc., 2017.

[55] Karl Benkic, Marko Malajner, P Planinsic, and Z Cucej. Using rssi value for distance estimation in wireless sensor networks based on zigbee. In *2008 15th International Conference on Systems, Signals and Image Processing*, pages 303–306. IEEE, 2008.

[56] Guanbo Zheng, Dong Han, Rong Zheng, Christopher Schmitz, and Xiaojing Yuan. A link quality inference model for ieee 802.15. 4 low-rate wpans. In *2011 IEEE Global Telecommunications Conference-GLOBECOM 2011*, pages 1–6. IEEE, 2011.

[57] V. D. Park and M. S. Corson. A highly adaptive distributed routing algorithm for mobile wireless networks. In *Proceedings of INFOCOM '97*, volume 3, pages 1405–1413 vol.3, April 1997.

[58] Yanbin Gao, Shifei Liu, Mohamed Atia, and Aboelmagd Noureldin. Ins/gps/lidar integrated navigation system for urban and indoor environments using hybrid scan matching algorithm. *Sensors*, 15(9):23286–23302, 2015.

[59] W. Kuo, Y. Chen, G. Jen, and T. Lu. An intelligent positioning approach: Rssi-based indoor and outdoor localization scheme in zigbee networks. In *2010 International Conference on Machine Learning and Cybernetics*, volume 6, pages 2754–2759, July 2010.

A
Appendix 1

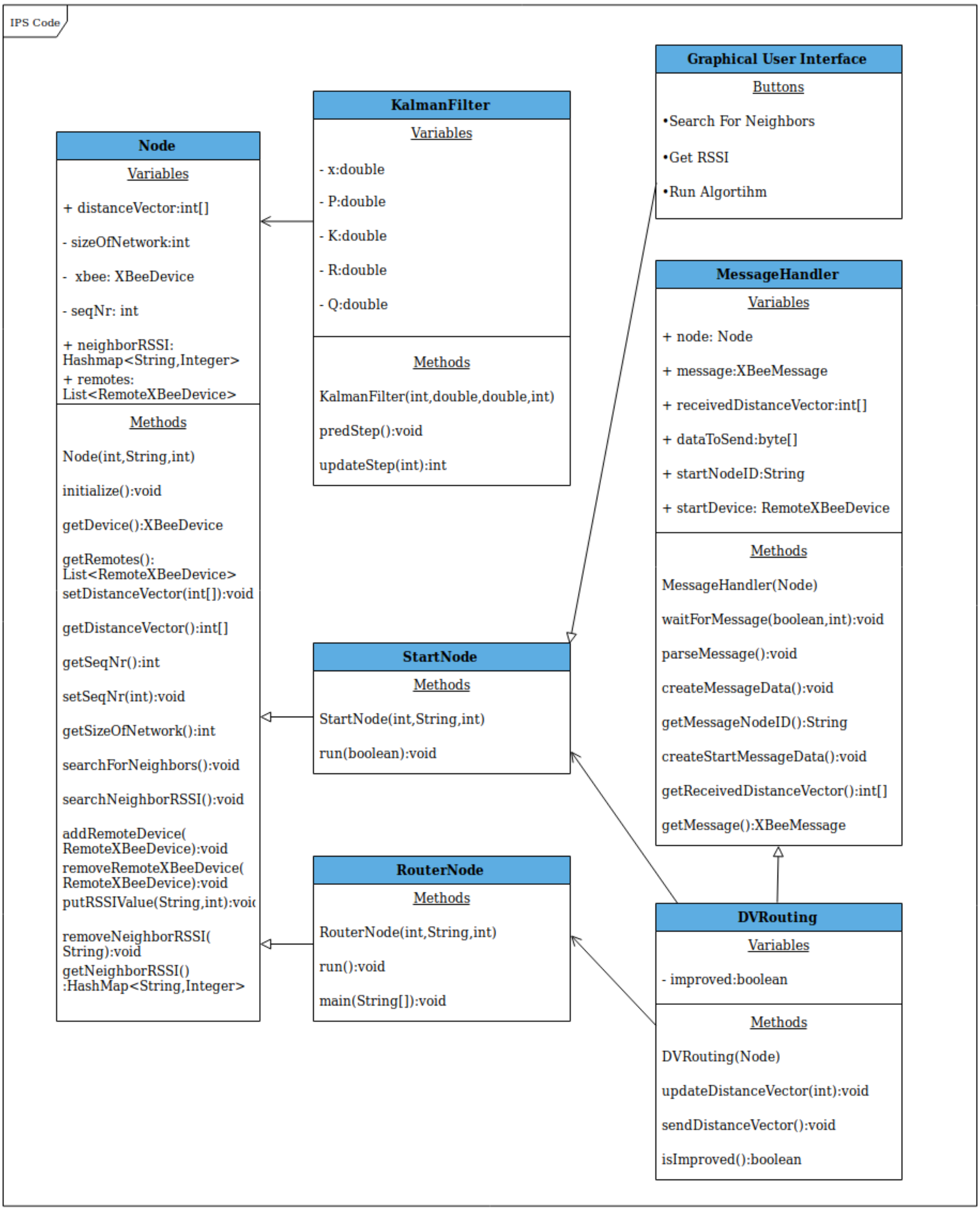


Figure A.1: Unified model language (UML) of the implemented Java code