Indoor sensor localization using RSSI and building map

P10 Master Thesis

Frederik Huse Jensen, Marcus Bisgaard Jensen

AALBORG UNIVERSITY

STUDENT REPORT

Aalborg University
Communication technologies
Networks and Distributed Systems

Communication technologies
Aalborg University
http://www.aau.dk/

AALBORG UNIVERSITY
STUDENT REPORT

Title:
Indoor sensor localization using RSSI and building map

Theme:
Communication Systems

Project Period:
Spring 2022

Project Group:
Group 1020

Participant(s):
Frederik Huse Jensen
Marcus Bisgaard Jensen

Supervisor(s):
Jimmy Jessen Nielsen
Tatiana Kozlova Madsen

Page Numbers: 60

Date of Completion:
June 2, 2022

Abstract:

In this project, the objective was to develop an algorithm that could determine sensor locations in an indoor environment based on Transmit Power (TxP), Received Signal Strength Indicator (RSSI) and the building map. Initially, the problem is analysed and current methods within localization determination are examined. This analysis lead to the choice to use RSSI to determine distance with a path-loss model, and an explanation of how the building map is used to improve the model. Based on this, an algorithm was designed to estimate distance, location and room in the building. To assess the algorithms performance, a simulation tool was developed, and the proposed algorithm was compared with other localization methods. It was concluded that the proposed algorithm "Multilateration with filtration" had increased the performance compared to other methods, however, further improvements can be done.

Contents

1 Introduction

The technological world has certainly gained momentum in recent years. Due to both new efficient communication possibilities, the proliferation of wireless networks and super cheap chips, the possibilities within Internet of Things (IoT) have exclusively become greater.

The ability to interconnect different objects and share sensor data between them adds a level of digital intelligence. This eliminates the need for human interaction and aims to make the world around us smarter and more responsive.

There are many examples, where the ability to collect data from various sensors in the industry has had a positive effect. Wireless Sensor Network (WSN) is a term for a series of scattered sensors that monitor the physical environment. These networks can vary from different network topologies such as a simple star network or a large multi-hop mesh network. Regardless of type, the network is tasked with distributing the data from the nodes to a central hub wirelessly. Some examples of applications, where WSNs are used, are; water quality monitoring, natural disaster prevention, farmers' soil monitoring, and more. WSNs are not only used in outdoor environments but have, in fact, become increasingly popular to embed in IoT sensors in indoor environments for a Building Management System (BMS). BMSs can be especially suitable in locations such as airports, supermarkets, hospitals and office buildings. these situations could eg. benefit from different monitoring techniques like people-counting and indoor environment management.

A study from 2014 indicates that roughly 1.5 hours of productivity are lost every day, where discomfort acts as one of the reasons for this[12]. Moreover, it is stated that a big portion of commercial building energy costs are made up of Heating, Ventilation, and Air Conditioning (HVAC) and lighting, which could be significantly decreased with the use of proper HVAC equipment and a BMS[3]. However, upgrading old buildings can be a costly affair, this includes expenses for new HVAC equipment, wiring for sensors and configuration. This is where WSN comes into play, eliminating the wiring cost and maintenance connected with this. It is said that even without new HVAC equipment, a WSN would significantly decrease the operating costs. When adding that indoor climate and productivity are also improving, it is difficult to find negative factors for these systems.

However, indoor WSNs do not come without complications, this is because the location of the wireless sensor must be known to avoid incorrect data.

On Figure 1.1 a simplified example can be seen with three rooms.

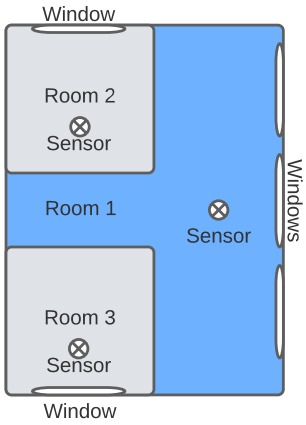


Figure 1.1: Simple example of three rooms with sensors and windows.

Now assume that all windows have an electric open and close mechanism, which is controlled by a BMS. The windows will therefore operate reactively based on the temperature values given by the sensor in the room. If the temperature is high, the windows will open, and if the temperature drops to a set threshold, the windows will close.

Even with a very simple example like this, it is crucial that the sensor values are correct, and that the sensor measurement corresponds to the room, in which it is placed. One can imagine, a scenario where the sensor in room 2 thinks that it is located in room 1, an increase in temperature in room 2 would then cause the sensor to inform the BMS to open the windows in the wrong room. This would quickly become annoying and unpleasant for the people in room 1, and diminish the advantages of a BMS.

Now imagine the same problem, but this time it is scaled to an entire office building with multiple sensors that measure both temperature, CO2 levels, and humidity, as well as a complete HVAC system controlled by the BMS.

1.1 Problem

In the previous section, it was emphasized how a modern BMS can both improve the indoor climate and thereby increase productivity, and at the same time reduce the energy consumption of the building. Furthermore, WSN was introduced as a method that eliminates the great expense that is wired connections.

WSN provides several advantages, some of these are listed below[14].

- Scaling with new nodes
- Flexible with physical partitions

- Can be centrally controlled
- Does not require wires or cables

Given the nature of wireless equipment, the advantages are relatively self-explanatory. Scaling with new nodes or devices should be possible at any time, by simply installing the new node, and letting the system configure it. In Addition, all hassles with cables and wires are removed, and wireless signals offer the possibility to pass through physical objects.

However, some disadvantages are also connected with a WSN, some of these are listed below.

- Prone to hacking
- Lower communication speed
- Battery driven operation
- Affected by noise
- Localization of sensors

With the use of wireless communication, there is always the threat of hackers, as it is not possible to control the propagation of waves. Moreover, given the nature of being wireless, the sensors must operate on a battery, it is therefore important to consider power usage or include some kind of energy harvesting in order to prolong the lifetime of the sensors. Likewise, wireless signals will inevitably be affected by the surroundings e.g. other signals.

However, when working with wireless sensors, and especially indoor wireless sensors, one major complication is the localization problem, as sensors that are not located correctly will cause the system to perform poorly.

In this project, the problems behind localization will be investigated, which involves analysing current as well as new methods to locate sensors in an indoor environment. Figure 1.1 shows a small example of this problem with an explanation of the consequence of bad location prediction. Some of the aspects that make it especially challenging are; Varying structure of buildings, different materials in walls, various objects in the room and many more. It can therefore be very hard to understand the surrounding environment, however, problems do not always occur only due to cumbersome obstacles.

Problems like the one shown on Figure 1.1 can also occur when a person has to manually configure and pair the sensors to their controllers. In this scenario, a controller is a stationary local device, that is assigned to a specific room.

A typical approach to installing a BMS involves having a large number of sensors installed in all rooms and then pairing the individual sensors with their respective controller. Currently, this process is performed manually, which will inevitably lead to errors, and then lead to extra expense when the error needs to

be corrected. It is therefore essential to outsource human interaction and have the system itself determine the placement of sensors and connections between sensors and controllers. This leads to the following problem statement, which will form the basis of the study that follows in this report.

"How can wireless sensors in an indoor environment be located using knowledge of the transmission power and the RSSI?"

Indoor localization can be challenging, as more complex models need to be developed to accommodate the indoor aspects. In the forthcoming analysis, the various parts of the system will be examined. Furthermore, different techniques for localization will be investigated, finally, leading to a proposal for a localization method for wireless sensors.

2 Analysis

This chapter will start with a detailed use case of an existing system. This includes what devices are involved, a description of their functionality, and what the current solution for localization is.

Next the State of Art (SoA) within localization will be investigated. This section will also include an analysis of how various necessary parameters will be estimated.

Lastly, the problem statement will be addressed, and a proposed solution will be analysed, including highlighting what the main challenges are.

2.1 Use case

Creating a fully functional BMS is a challenging task with many obstacles. To elaborate on the issues connected with this, a meeting was held with the department of Build, who works to incorporate a BMS in an office building at Aalborg University.

2.1.1 Current setup

As mentioned in Chapter 1 the usage of a BMS requires a wide range of sensors and controllers, which is also what is implemented in their setup. Each room is indicated as a so-called Intelligent Building Installation (IBI) zone with a local controller. In addition, there are sensors for windows, temperature, motion, ventilation, etc., respectively. All these sensors are connected to the local controller, which is connected to a central controller and finally the server. On Figure 2.1 the current format can be seen.

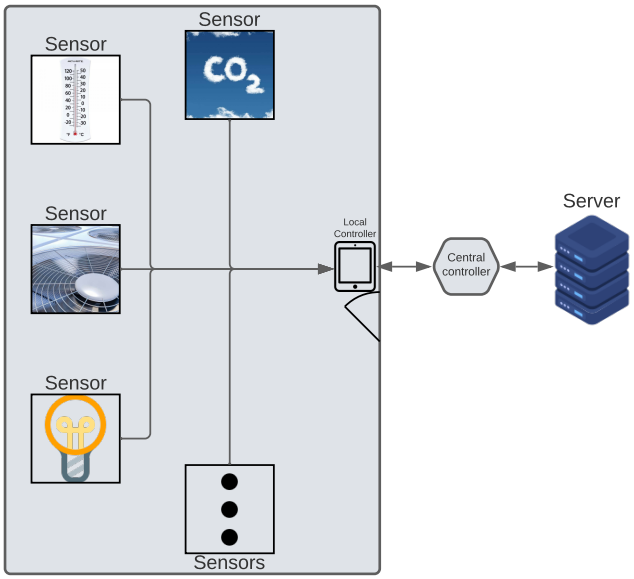


Figure 2.1: The current setup with wired sensors communicating with a local controller.

Figure 2.1 indicates that all sensors in the room communicate to the local controller. However, they are not directly wired to their respective controller. According to the department of Build, the current approach to setting up the BMS is.

1. Set up the sensors and connect them to the router (wired)
2. Set up the local controller and connect it to the router (wired)
3. Pair sensors and local controllers (manually)

In a system like this, the specific sensors must be connected to the correct controllers, otherwise, the system will malfunction.

It was emphasized that due to everything being wired, the cost of correcting errors is both expensive and impractical. Moreover, it is estimated that adding a new sensor to a room cost approximately 3000-4000 DKK or more, depending on the manual work involved in the installation.

2.1.2 Desired setup

The goal for Build is to be able to set up and scale their BMS as easily and as cheaply as possible. However, due to the potential size of the buildings, as well as their dynamic, it is far from optimal to connect all devices by wires. It is therefore a focal point to establish wireless connections between sensors and controllers, as this will lower the amount of manual work needed. On Figure 2.2 the desirable system with a wireless link can be seen.

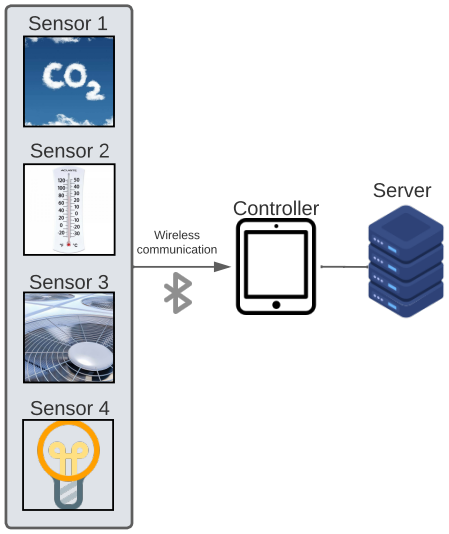


Figure 2.2: The desirable setup with wireless communication between sensors and their controller.

To make the process even cheaper, it is desirable that the system can find the room location of the sensors itself, and thereby make the pairing to the correct controller. In the figure above, the controller will receive information from the sensors wirelessly. Based on the received signal, the system will estimate the physical location of the individual sensors, thus assigning them to a room. The new approach to setting up the BMS would be.

1. Set up the sensors and connect them to the router (wireless)
2. Set up the local controller and connect it to the router (wired/wireless)
3. The system locates and pairs the sensors with their respective controller (no manual work)

The following section will outline a description of the devices used in the BMS, as well as how each of them should function during setup and room assignation.

2.1.3 BMS devices

Sensors

The sensors are, as explained shortly in Chapter 1, a set of devices measuring different relevant conditions in their environment. These devices will obviously have to communicate with the controllers, and must therefore include either a WiFi or a Bluetooth module. This module will be used to transmit the data, but will also be used in the efforts of localizing and assigning the correct room to each

device. The sensors must relay some information to the controllers in order for the controllers to correctly assign a room to the devices. This information will include their transmit power level, Transmit Power (TxP), since this value is essential for the calculation of the distance between the devices.

Controllers

The task of the controllers is to act on the information provided by the sensors. Both in terms of the indoor environment, but also in terms of information during setup. The controller must gather the information sent from the sensors and add some additional information, before relaying the data to the central server. The information added by the controllers should include the Received Signal Strength Indicator (RSSI) since this will be used in combination with the TxP from the sensor device to identify in which room the sensor is located.

Central server

The server is in charge of the setup and assignment of the sensors in each room. When receiving the information from all controllers and sensors, the server has to calculate and estimate the correct rooms for the devices, by determining the most probable location of each sensor and relating that to the building layout.

2.1.4 Communication Technology

As mentioned, to facilitate a system like this, the need for wireless technology is evident. In the current development, Bluetooth is used for communication. Therefore, it also makes sense to use Bluetooth in this project. This section will shortly present and evaluate some of the advantages of Bluetooth

Bluetooth

This paragraph will highlight a few of the possibilities and properties that Bluetooth provides in a wireless setup. Bluetooth has many advantages when considering wireless communications protocols for devices that should be power efficient. When focussing on the energy consumption the Bluetooth protocol provides a version called Bluetooth Low Energy (BLE), which was first implemented in Bluetooth 4.0. BLE has, as the name implies, been developed specifically with power usage in mind. The implementation of BLE lowers the power consumption, by turning off communication during idle periods. A feature which was not possible in Bluetooth Classic.

This low power consumption property makes Bluetooth an extremely attractive technology for a BMS. This means that the battery-powered sensors can operate for

a longer time period, thereby reducing the need for maintenance. While Bluetooth is very power efficient it does not provide the same amount of bandwidth as other technologies, such as Wi-Fi. Bluetooth 5 only has a maximum bitrate of up to 2 Mbps when using high-speed mode [1], whereas the newest Wi-Fi 6 provides a much higher bitrate of up to 2.4 Gbps [6]. However, this obviously will have a much higher power demand and is not deemed necessary for a system with relatively low data exchange.

The Bluetooth packets differ slightly depending on whether the devices are connected or unconnected. If the devices are unconnected, the Bluetooth Low Energy protocol has a type of packet called advertisement packets. These packets are broadcasted, and all controllers nearby will be able to receive the transmission. Since the RSSI is measured on the receiving side of a transmission, each of the receivers will be able to measure the current RSSI from the transmission regardless if they are connected or unconnected. The RSSI will of course depend on the transmission power, a value that can differ from sensor to sensor. This transmission power may be transmitted along with the advertisements and can be read by the receiver.

With the wireless technology examined, the challenges of indoor localization and distance estimation will be investigated including a look into Line of Sight (LoS) vs non LoS and how this affects the communication. Moreover, the next section will analyse different methods for localization and compare these to the challenges of indoor environments.

2.2 Localization SoA

As mentioned in Section 1.1, localization of wireless sensors can be a difficult task, especially in an environment that does not provide clear LoS and can have a lot of interference for the signal. In many localization scenarios Global positioning system (GPS) is a popular solution, however, GPS signals are weak and are therefore not applicable for indoor systems. Indoor systems must therefore utilise methods to estimate the location of units based on limited knowledge, such as RSSI, Time of Arrival (TOA) and Angle of Arrival (AOA). In this section, some of the localization detection techniques, that utilise this knowledge, will be presented.

2.2.1 Triangulation

Triangulation is a well-known localization method which, as the name implies, use AOA measurements to determine a point. Figure 2.3 shows a simple example of triangulation with two Access Point (AP) (point 1 and point 2). These points have known position and with knowledge of the length between them and their measured angles it is possible to calculate the position of point 3.

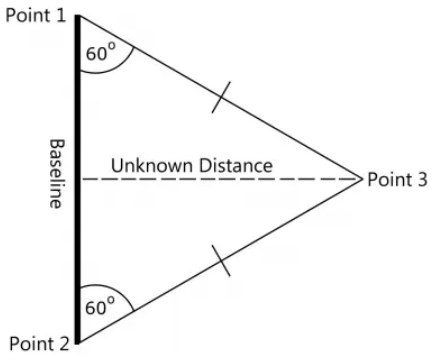


Figure 2.3: Simple triangulation to determine the location of point 3[4].

The calculations are relatively simple, as it is possible to calculate the unknowns in an arbitrary triangle, with knowledge of two angles and one side length. The formulas used are given by the law of sines[13].

$$\frac{a}{sin(A)} = \frac{b}{sin(B)} = \frac{c}{sin(C)} \tag{2.1}$$

Taken the example from Figure 2.3 the parameters are point 1 = A, point 2 = B and point 3 = C, and the opposite lengths are given as a, b and c. Using Equation (2.1) and setting the baseline to 10 meters, the calculations would be.

$$\begin{aligned} \angle C &= 180 - 60 - 60 \\ \frac{a}{sin(60)} &= \frac{10m}{sin(60)} \\ a &= 10m \end{aligned}$$

The example given is an equilateral triangle, which makes the calculations obvious, but the procedure is the same for any triangle.

Triangulation has the advantage that only two reference APs are needed to determine the desired location. Furthermore, AOA does not have any requirement for time synchronization between APs, making the setup easier.

However, being dependent on AOA measurements also include challenges, which are highlighted in the paper [15]. This is due to AOA measurements being highly affected by multi-path and non LoS. This property is obviously not beneficial for indoor localization systems with poor conditions. Moreover, the receiver requires an additional antenna to be able to measure AOA, and since signals travel at the speed of light, the devices must be very accurate, which entails an extra expense.

With triangulation examined, its sibling, namely trilateration, will be analysed.

2.2.2 Tri-/multilateration

Trilateration uses knowledge of the location of three fixed non-collinear APs to calculate the physical position of the target. This is accomplished by calculating the distance from the target node to each AP, which can be done using the RSSI or using TOA measurements. After calculating the distance, each AP draws a circle indicating the possible locations for the target node. The intersection of the three circles will then indicate the true location of the node, which can be seen on Figure 2.4.

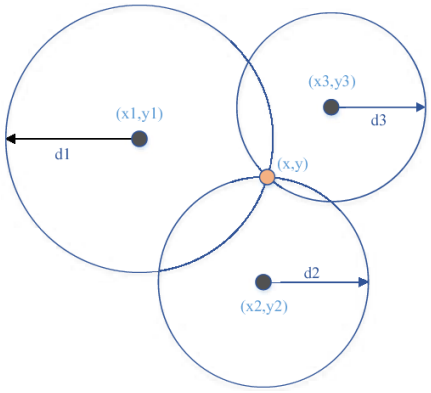


Figure 2.4: Example of trilateration used to determine the point (x,y) in 2D [9].

The formulas used to determine the coordinates (x,y), based on the circles are as follows.

$$(x - x_1)^2 + (y - y_1)^2 = d_1^2 \tag{2.2}$$

$$(x - x_2)^2 + (y - y_2)^2 = d_2^2 \tag{2.3}$$

$$(x - x_3)^2 + (y - y_3)^2 = d_3^2 \tag{2.4}$$

Where, x_n and y_n are the known coordinates to the n^{th} AP and d_n is the calculated distance from the node to that AP.

Multilateration is a method used to increase the number of APs used to determine the location of a node, and thereby increase the accuracy of the prediction. This will also increase the complexity, as multiple APs have to perform calculations for each node.

These methods do depend on knowing the distance between the AP and the node, which will be examined in the following section.

2.2.3 Distance estimating

As mentioned, the distance can be calculated using a method to convert RSSI values to distance. This is achieved by using a path-loss model, which describes how the signal decreases over distance. For now, the path-loss model that will be used is given in Equation (2.5), where RSSI is given based on the TxP, n and the distance d.

$$\text{RSSI} = \text{TxP} - 10 \cdot n \cdot \log(d) \tag{2.5}$$

Where n represents an environmental factor and ranges from 2 to 4, where $n = 2$ is given as the LoS scenario, while $n = 4$ is given as worst case scenario. Since it is an indoor environment with many obstacles, it is assumed that the n value will be in the high end, therefore for simplicity this will be set to worst case $n = 4$ for now. TxP will be set to 0 dBm, since this is the standard value for transmissions from small devices. Furthermore the value is well under the allowed 20 dBm stated by the European Telecommunications Standards Institute (ETSI) in [5]. This limit applies for applications were frequency hopping is utilised, but the limit without frequency hopping, of 10 dBm is likewise complied with. With the model and the values set, it is possible to determine a distance from an RSSI measurements. Figure 2.5 shows the correlation between these parameters.

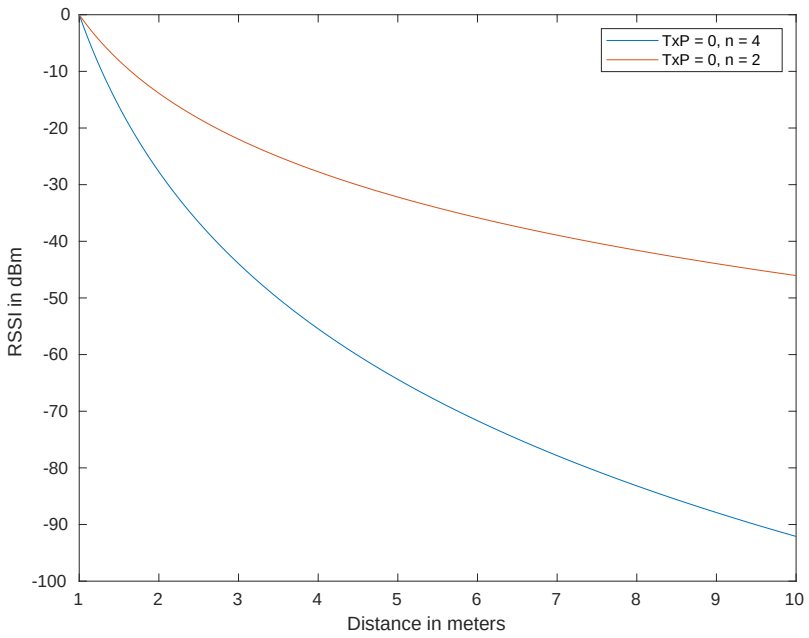


Figure 2.5: RSSI values and the corresponding distance using Equation (2.5).

The distance can also be determined by calculating the time it takes for a signal to travel from the node to the AP, also known as TOA measurements. The formula

used to calculate the distance using TOA is:

$$\mathrm{d} = c \cdot t_{travel} \tag{2.6}$$

Where c is the speed of light: $3 \cdot 10^8 m/s$, and t_{travel} is the time it takes for the signal to travel to the receiver.

One of the drawbacks of using TOA is the strict need for synchronization between units. A common way to achieve synchronization is to use Round Trip Time (RTT), which measure the total time it takes to send from the AP to the node and back, including the processing time.

$$t_{RTT} = \frac{(t_{back} - t_{send}) - (t_{process} - t_{receive})}{2} \cdot c \tag{2.7}$$

Here the $(t_{back} - t_{send})$ represent the total time from sending to receiving at the AP, $(t_{process} - t_{receive})$ is the time it takes to receive and process the signal at the node. However, this method requires the node to be able to both transmit and receive signals.

To overcome this, a method called Time Difference of Arrival (TDoA) can be used. In this method, the node can remain only as a transmitter, and synchronization between the node and AP is not necessary. Localization determination is instead calculated based on TDoA in the APs. This does, however, come with an additional requirement for the APs, since clock synchronization between them now becomes extremely necessary. In this scenario, having two APs being out of sync, would result in miscalculation of location, this can be seen using Equation (2.6) with the APs being $2 \cdot 10^{-9} s$ out of sync.

$$\mathrm{d}_{sync-err} = 3 \cdot 10^8 m/s \cdot 2 \cdot 10^{-9} s$$
$$\mathrm{d}_{sync-err} = 0.6m$$

It is therefore clear that a small offset in the synchronization can lead to errors in the localization prediction. This will once again require the equipment to be very accurate, as the environment does not provide perfect conditions, which will make it more expensive.

2.2.4 Fingerprinting

Fingerprinting is a method that, based on several details, tries to describe an overall picture or "fingerprint" of a scenario. It is achieved by mapping data to a bit string, which can be used to identify that specific data. Fingerprinting can be used for various purposes such as retargeting advertisements and tracking online, but it can also be utilised for localization problems.

For indoor localization, fingerprinting is a process in which an appropriate amount of data is collected in a database. This approach is generally referred to

as the offline phase[7][10][2]. This data would consist of a series of measurements, such as RSSI values from a specific location to the different AP. This can be done simply and manually by physically walking around in the environment to collect the data. Another approach involves using a path-loss model to simulate the environment and build a theoretical database from this.

Subsequently, the online phase is handled where new data with unknown locations will be compared with data in the database and, based on similarities, classified to a location.

In [7] fingerprinting is used in collaboration with different machine learning algorithms to evaluate this approach for indoor localization. In this paper, the authors use fingerprinting to feed the machine learning algorithms and compare the results. It was concluded that estimating location using these algorithms based on fingerprint observations provided good results in terms of accuracy given the computational cost required. Moreover, it was emphasized that using this method for indoor environments is favourable due to its efficiency in highly variable environments, which is one of the challenges associated with indoor localization.

In [2] a hybrid approach between fingerprinting and trilateration is proposed. Here it is stated that fingerprinting is the superior indoor positioning technique of the two in terms of accuracy. However, it is also mentioned that fingerprinting has a disadvantage in the time-consuming phase, namely the offline phase. This is also indicated in [10], where localization determination with trilateration and fingerprinting is compared. In this paper, it is concluded that it is advantageous to determine a conversion from signal strength to distance and apply simple trilateration, rather than using resources to build a database with fingerprinting.

With an analysis of different localization determination techniques, the next section will analyse the approach used in this project.

2.3 Localization Approach

This work aims to develop an algorithm that can be used in a BMS to locate sensors. As analysed in the previous section, there exist many methods of indoor location determination.

In this project, the RSSI value will be utilised to estimate the distance between the sensor and controllers. Using the path-loss model given in Equation (2.5), and isolating the distance yields.

$$d = 10^{\frac{\mathrm{TxP}-\mathrm{RSSI}}{10 \cdot n}} \tag{2.8}$$

Evidently, a few different elements will be necessary to calculate the distance from the measured RSSI, and a few important things need to be considered.

First and foremost, it is necessary to know the transmission power from the sensor device, since this must be used in the calculations along with the received

signal strength at each of the in-range controllers. The transmission power is initially defined and set by the factory, and can be read or changed on the Bluetooth chip, with AT commands or through the Host Controller Interface (HCI). The necessity of knowing the transmission power becomes evident when considering the Bluetooth technology specifications. The Bluetooth technology supports a transmit power from -20 dBm (0.01 mW) to +20 dBm (100 mW), Which in terms of range is a rather large span[8]. As mentioned in Section 2.2.3, the standard TxP for small devices is 0 dBm, which will be assumed as the value in this setup.

Since the surrounding environment will not always provide clear LoS between two devices, the obtained RSSI will vary in different situations. It would therefore be beneficial to include extra information into the algorithm concerning whether the signal has had LoS passage or not. In this way, the algorithm can take into account whether the signal has been affected by obstacles, and possibly improve the estimation of the location. This will be the focus of the next section.

2.4 Refined problem statement

Arguably one of the biggest challenges with indoor localization is the unknown parameters of the environment. Localization is therefore performed with limited knowledge, and the current methods will always suffer from these unknowns. Given an example where trilateration is used with distance calculation using RSSI measurements, and the signal is received with a known TxP. Here it is important to know whether the signal has had a free passage or whether it has been affected by walls or other objects. Likewise, locations relatively close to each other can have a significant difference in RSSI, because the signal propagation can vary based on the surroundings

In this project, additional information is available to perform the localization, which consists of a building map. It is not inconceivable to have a building map available, as buildings most often have some kind of map that shows the room distribution and walls. Figure 2.6 will represent a section of the total building map, which will be used going forward for tests and evaluations.

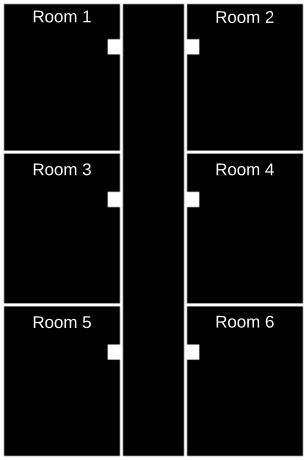


Figure 2.6: Building map indicating different rooms with their individual controller depicted as white squares.

With the building map, it is possible to analyse the possible scenarios that can occur when a sensor is placed in a room. On Figure 2.7 an example can be seen with two randomly placed sensors, who have two and three walls respectively between them and the controller in room 2.

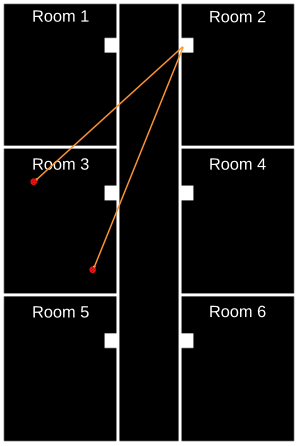


Figure 2.7: Example of sensors in room 3 and the number of walls from them to the controller in room 2.

Assuming the signal will always travel in a straight line between a sensor and a controller, the following Table 2.1 will indicate the maximum and the minimum number of walls between two units.

	Room 1: max/min	Room 2: max/min	Room 3: max/min	Room 4: max/min	Room 5: max/min	Room 6: max/min
Room 1	0 walls/0 walls	2 walls/2 walls	1 wall/1 wall	3 walls/2 walls	2 walls/2 walls	3 walls/2 walls
Room 2	2 walls/2 walls	0 walls/0 walls	3 walls/2 walls	1 wall/1 wall	3 walls/2 walls	2 walls/2 walls
Room 3	1 wall/1 wall	3 walls/2 walls	0 walls/0 walls	2 walls/2 walls	1 wall/1 wall	3 walls/2 walls
Room 4	3 walls/2 walls	1 wall/1 wall	2 walls/2 walls	0 walls/0 walls	3 walls/2 walls	1 wall/1 wall
Room 5	2 walls/2 walls	3 walls/2 walls	1 wall/1 wall	3 walls/2 walls	0 walls/0 walls	2 walls/2 walls
Room 6	3 walls/2 walls	2 walls/2 walls	3 walls/2 walls	1 wall/1 wall	2 walls/2 walls	0 walls/0 walls

Table 2.1: Table indicating the maximum and minimum number of walls between a node placed in room x to a controller in room y.

From the table it can be seen that in this section the maximum number of walls between two units are three. With this additional information, the path-loss model can become more customized to fit the specific scenario, which is included in Equation (2.9).

$$\text{RSSI} = \text{TxP} - 10 \cdot n \cdot \log(d) - k \cdot att + \omega \tag{2.9}$$

Where k is given as the number of walls between a sensor and a controller, att is the wall attenuation and ω is the added noise given as $\omega \sim \mathcal{N}(0, \sigma^2)$. As mentioned earlier in Section 2.2.3 $n = 4$ could be set to compensate for the fact that the surroundings are not well defined. However, with the building map and the added $k \cdot att$ to the model, it can be argued that the non LoS aspect is covered in this expansion and should therefore not be included in the n value. Therefore, the value is scaled down to $n = 2$.

With this model, as well as knowledge of the building map, each controller can now calculate four lengths, indicating that the signal has passed zero, one, two and three walls, respectively. From Section 1.1, it was asked how localization in an indoor environment could be performed based on TxP and RSSI knowledge, which is a well-known and popular topic. However, in this project, the building map will be added to the prior knowledge, which redefines the problem statement to:

"How can wireless sensors in an indoor environment be located using knowledge of the TxP, the RSSI and the building map?"

With this new addition to the problem statement, the following Chapter 3 will outline the approach to the design of the algorithm, including what considerations and choices have been made during the development.

3 Design of Algorithm

This chapter will present how the algorithm is designed and which elements were considered when designing it. Firstly the considerations regarding the use case, this solution fits into, are outlined in the following section.

3.1 Considerations

As explained in Chapter 1 and Chapter 2 the purpose of this project is to reliably and automatically locate wireless devices in an indoor environment. The project is made at the instigation of the Build Department at Aalborg University, who furthermore have provided a set of considerations in regards to what features the algorithm can include.

- Automatic location estimation of wireless sensors.
- Assignment of room number to the located sensors, based on an estimated location.
- Easy scaling of system, by adding additional sensors.
- Automatic reconfiguration of relocated sensors.
- Easy reconfiguration of sensor subset if errors are suspected.
- Low power consumption.

The focal point of this project is the algorithm to determine the location estimates, which narrows the relevant considerations down to the following.

- Automatic location estimation of wireless sensors.
- Assignment of room number to the located sensors, based on an estimated location.

These points are seen as the main elements in the overall system, while the other considerations rely more on the specific implementation than the algorithm. For this reason, the remaining points are not further considered in this project.

3.2 Algorithm iterations

The development of the algorithm was split into iterations, to develop a basis, and then improve on this through each iteration. Each iteration introduced new considerations and improved the algorithm. Each algorithm consists of a few different parts, which will vary throughout the development. Common for most iterations will be.

- Distance estimation
- Location estimation
- Room estimation

Each part will be explained separately concerning its respective iteration.

3.2.1 Best RSSI (Control)

As a control algorithm, the "Best RSSI" was chosen. The essence of this algorithm is to measure which controller receives the highest RSSI value from a sensor and then assign the sensor to the corresponding room. A valid option, but also an option where no further information than the room number is obtained. Furthermore, in some situations, this algorithm will assign the wrong room, depending on the location of the sensor, the walls, and the controllers.

The next iteration increases the complexity slightly, utilising RSSI information from a range of controllers, to pinpoint a closer location.

3.2.2 Trilateration (Control)

The first intuition to locate the sensors was to implement trilateration since the controllers' locations are known. This approach is likewise used as a control, to verify whether the algorithm proposed in this project outperforms the simple trilateration. Using the RSSI at the three controllers with the best RSSIs to estimate the distance from the sensor to the controllers, the trilateration calculations provided an estimate of the coordinates.

3.2.2.1 Distance estimation

To determine the distance between the sensors and the controllers, the algorithm utilises the TxP and the RSSI measured at the controller. As described in Equation (2.8) the estimated distance can then be calculated from the path-loss formula. This accuracy of this distance will again depend on a few different parameters like environment and other conditions. But this trilateration does not take these factors into account.

3.2.2.2 Location estimation

Essentially the trilateration is a "three equations two unknowns" problem, which is explained in Section 2.2.2. The trilateration was implemented using Equation (2.2), Equation (2.3) and Equation (2.4).

The equations are expanded to get rid of the parentheses.

$$x^2 - 2x_1x + x_1^2 + y^2 - 2y_1y + y_1^2 = d_1^2 \tag{3.1}$$

$$x^2 - 2x_2x + x_2^2 + y^2 - 2y_2y + y_2^2 = d_2^2 \tag{3.2}$$

$$x^2 - 2x_3x + x_3^2 + y^2 - 2y_3y + y_3^2 = d_3^2 \tag{3.3}$$

To reduce these three equations into two equations with two unknowns, the second equation, Equation (3.2) is subtracted from Equation (3.1) to give Equation (3.4).

$$(-2x_1 + 2x_2)x + (-2y_1 + 2y_2)y = d_1^2 - d_2^2 - x_1^2 + x_2^2 - y_1^2 + y_2^2 \tag{3.4}$$

The third equation, Equation (3.3) is then subtracted from the second equation, Equation (3.2) to give Equation (3.5).

$$(-2x_2 + 2x_3)x + (-2y_2 + 2y_3)y = d_2^2 - d_3^2 - x_2^2 + x_3^2 - y_2^2 + y_3^2 \tag{3.5}$$

Using substitution to isolate x and y the two equations become:

$$x = \frac{(d_1^2 - d_2^2 - x_1^2 + x_2^2 - y_1^2 + y_2^2) \cdot (-2y_2 + 2y_3) - (d_2^2 - d_3^2 - x_2^2 + x_3^2 - y_2^2 + y_3^2) \cdot (-2y_1 + 2y_2)}{(-2x_1 + 2x_2) \cdot (-2y_2 + 2y_3) - (-2y_2 + 2y_3) \cdot (-2x_2 + 2x_3)} \tag{3.6}$$

$$y = \frac{(-2x_1 + 2x_2) \cdot (d_2^2 - d_3^2 - x_2^2 + x_3^2 - y_2^2 + y_3^2) - (d_1^2 - d_2^2 - x_1^2 + x_2^2 - y_1^2 + y_2^2) \cdot (-2x_2 + 2x_3)}{(-2x_1 + 2x_2) \cdot (-2y_2 + 2y_3) - (-2y_2 + 2y_3) \cdot (-2x_2 + 2x_3)} \tag{3.7}$$

This algorithm utilises the distance estimations directly from the RSSI and does thereby not consider the attenuation through walls and furniture. With the indoor environment, this means that the location estimates become rather inaccurate. To mitigate the effects of the wall attenuation, the algorithm had to be improved, to include a decision based on the average expected wall attenuation.

3.2.3 Trilateration with wall information

For the first iteration of the algorithm, the consideration regarding the wall attenuation was added to the trilateration. Thereby changing the distance estimation.

3.2.3.1 Distance estimation

Like with traditional trilateration, the distance is determined by utilising the TxP and the RSSI measured at the controller. The uncertainties in the distance estimation, are mitigated by calculating a set of distances based on the expected average wall attenuation. The wall attenuation is estimated by analysing the internal communication between the controllers, since their locations, their TxP and the building layout are known.

$$ExpectedRSSI = TxP - (10 \cdot n \cdot log(d)) \tag{3.8}$$

For this wall attenuation calibration, the TxP from the controllers are set to $20dBm$, as it is assumed that they are connected to a direct power source. A numerical example of the expected RSSI is given below, where the known distance between two controllers is $8m$.

$$ExpectedRSSI = 20 - (10 \cdot 2 \cdot log(8)) = 1.938dBm$$

By calculating the expected RSSI from the path-loss model, and measuring the actual RSSI, the difference between these are found. Then using the building layout to determine the number of walls between two controllers to estimate the wall attenuation, here it is assumed that the signal is only affected by walls.

$$WallAttenuation = (ExpectedRSSI - MeasuredRSSI)/(Walls) \tag{3.9}$$

This *WallAttenuation* is calculated for every controller pair and the average wall attenuation is calculated.

Each controller will estimate a set of distances, with considerations about the expected amount of walls. In the trilateration algorithm, each controller calculates four distances with the assumption, that there are zero, one, two and three walls interfering with the signal. Each of these distances is utilised when estimating the most likely location of the sensor device.

3.2.3.2 Location estimation

To determine the location estimate the algorithm utilises the estimated distances and the controllers' locations. Using an example, this will be further explained.

Consider a scenario such as in Figure 3.1, here a setup can be seen with multiple controllers and a single sensor. The sensor will send a signal to the controllers, and each controller will estimate the distances to the sensor. But instead of using one estimate of the distance, each controller provides the aforementioned set of estimations based on the possible amount of walls between them and the sensor. The four coloured rings around each controller represent these distance estimations, the outermost ring, is the distance estimate where it is assumed that no walls are

interfering with the signal. The second ring represents the estimate with one wall blocking the signal, the third ring represents two walls blocking the signal, and the inner ring is with three walls. The algorithm will then compare the distances from the different controllers, to find the most likely location of the sensor.

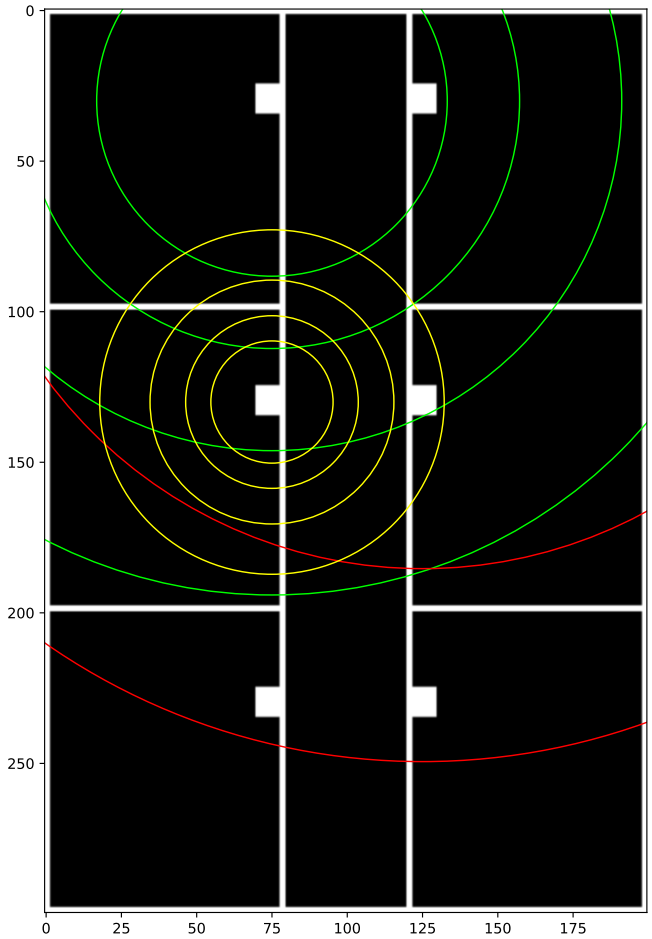


Figure 3.1: Building example with one sensor and distance estimates from three controllers.

With distance information from the different controllers, the algorithm will use the ring distance information. Since the controllers make four different rings, it will be possible to estimate which of the ring pairs are most likely to be correct. In the figure, three controllers can be seen with rings, which will amount too $x^c = 4^3 = 64$ different ring pairs where $x = \textit{amount of rings}$ and $c = \textit{amount of controllers}$. The pairs will look like the following:

- C1-R1 C2-R1 C3-R1
- C1-R1 C2-R1 C3-R2

- C1-R1 C2-R1 C3-R3
- C1-R1 C2-R1 C3-R4
- etc. for the last 60 different pairs

These calculations can again be done in multiple different ways. The approach chosen is finding the intersections between the different rings, with the least mean squared error. These will be the coordinates that minimise the squared distance between the point and the rings.

```
Error:
    sums = []
    For x,y in controller locations:
        sums.append(((x_est-x)^2+(y_est-y)^2-ring_radius)^2)
MinError = Min(sums)
```

In the example from Figure 3.1 it can be seen that there is no point with a perfect intersection between three rings, but there are multiple points where the circles are relatively close, such as.

- C1-R2 C2-R4 C3-R1
- C1-R1 C2-R4 C3-R1

Having multiple points, where the error is very small, is not necessarily good for the algorithm, as it will not be able to pinpoint the true location. In this example the sensor is located in the coordinates (140,20), which makes the correct rings to use.

- C1-R2 C2-R4 C3-R1

This combination was one of the options with low error, however, it is not necessarily the combination with the lowest error. Therefore, if the algorithm simply chooses the best fit according to least squares, the system might eventually guess wrong. This shows the necessity of evaluating the reliability of each of the estimations. This issue will be discussed further in Section 3.2.5.

By using the information about the wall attenuation, it was possible to improve the trilateration estimation, but some different errors and problems occurred. If the three controllers with the best RSSI which is chosen to do the trilateration on, is located on a straight line through the building, the trilateration will be under defined and will fail to provide a single set of coordinates, which of cause is undesirable. Therefore, it was chosen to add more controllers to the algorithm to turn trilateration into multilateration.

3.2.4 Multilateration with wall information

The transition to multilateration did not influence the procedure regarding distance estimation. This is still done by calculating four different distances for each controller. By calculating locations with additional controllers the issue with the controllers being on the same line will be mitigated since the risk of this happening is much lower. Multilateration utilises a bigger set of controllers to estimate the location of the sensor, in the case of this project, the amount of controllers for the multilateration is six controllers, since the emulated building map that is used, consists of six rooms and a hallway, where each room has one controller.

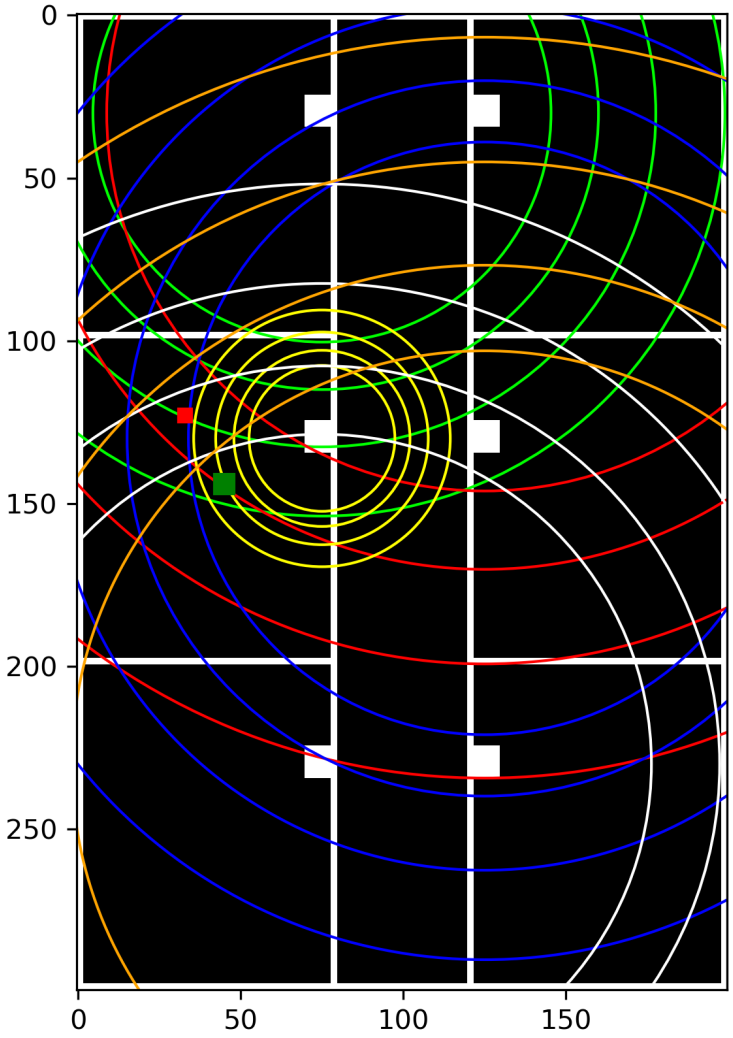


Figure 3.2: Building map with circles visualized and estimated location plotted as the green square, the red square represents the true location.

Using the Algebraic approach constructed in the article "An Algebraic Solution to the Multilateration Problem" [11], the set of formulas for the circles, is rewritten to the following matrices.

$$\begin{bmatrix} 1 & -2x_1 & -2y_1 \\ 1 & -2x_2 & -2y_2 \\ \vdots & \vdots & \vdots \\ 1 & -2x_n & -2y_n \end{bmatrix} \cdot \begin{bmatrix} x^2+y^2 \\ x \\ y \end{bmatrix} = \begin{bmatrix} d_1^2 - x_1^2 - y_1^2 \\ d_2^2 - x_2^2 - y_2^2 \\ \vdots \\ d_n^2 - x_n^2 - y_n^2 \end{bmatrix}$$

With the following structure the matrices can be named $A \cdot x = B$. Since the result of the estimation is located in the vector x. The equation can again be rewritten to the following, to solve for $\hat{x}$.

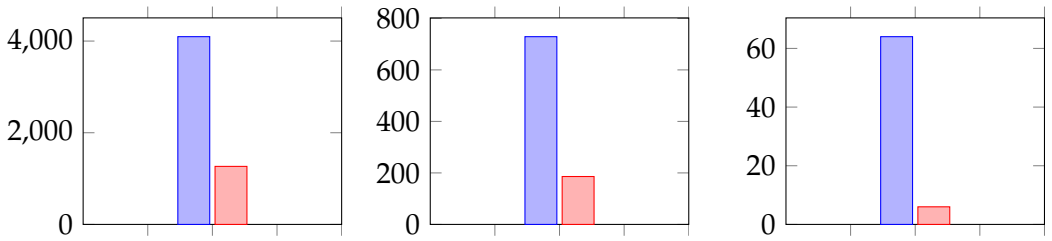
$$\hat{x} = (A^T A)^{-1} A^T B \tag{3.10}$$

For each pair of distances, these calculations are conducted, and the results are compared to each other to decide which is the most accurate. The most accurate result is just as it was with triangulation, the one with the lowest mean squared distance error between the location estimation and the estimated distance. With the increase in the number of controllers the probability that multiple locations will provide low errors increases.

Therefore, it will be necessary to evaluate the reliability of each of the estimations and filter out solutions that are impossible. An example of an impossible solution could be if multiple controllers use the estimated distance with the zero walls assumption since the sensor can not be located in multiple rooms at once. Since the accuracy of the distance estimation decreases when the distance increases, it would also make sense to evaluate the estimations based on their RSSI where the controllers with higher RSSI are valued more reliable.

3.2.5 Filtered multilateration with wall information

To improve the accuracy of the algorithm further the impossible room/wall combinations should be sorted away, for two reasons. To improve the running time of the algorithm, and to sort away possible mistakes. Looking at the emulated building map from Figure 2.6, an analysis was made, to understand which combinations were possible, and which were not. With a single controller in each room, it makes sense to exclude all combinations using more than one direct distance (distance with no wall factor). All combinations with more than one of these distances are therefore discarded. Furthermore, it was found that in each situation there will be at least one neighbouring room, meaning at least one controller with a single wall between the sensor and the respective controller. These filtering rules decreased the number of combinations from 4096 to 1266 combinations, when using four distances and 6 controllers. This decrease drastically improves the running time and also ensures that fewer impossible combinations can influence the result.



(a) Six controllers each drawing 4 circles for 4 distances.

(b) Six controllers each drawing 3 circles for 3 distances.

(c) Six controllers each drawing 2 circles for 2 distances.

Figure 3.3: Relation between combinations in multilateration(Blue) and filtered multilateration(Red).

In Figure 3.3 it can be seen that the improvement in amount of combinations is eminent even with fewer distances as well.

3.3 Room estimation

When the location estimation has been conducted, the room estimation can begin. For each of the different location estimations, the same method for room estimation has been used, except for the "Best RSSI" algorithm, which assigns rooms corresponding to the controllers' room number.

There could be several different ways of determining the room of the current sensor. But the most obvious one is plotting the location on the building map and "see/read" which room it is in. In all the different iterations of the algorithm, this corresponds exactly to what was done.

With the room number assigned each of the algorithms can be evaluated against each other, to determine if the proposed algorithm increases the accuracy of indoor localization. Since the algorithm is only implemented in this specific environment the coordinates and measurements of the rooms are known in advance. Therefore a simple check with a series of if-statements, checking the placement in relation to the walls, was conducted, to determine the room number, of the sensors.

With the room number assigned each of the algorithms can be evaluated against each other, to determine if the proposed algorithm increases the accuracy of indoor localization.

4 Performance evaluation approach

To test the performance of the developed algorithm a choice had to be made, on whether to make a physical testbed or a simulation. For this project, it was chosen to develop a simulation tool, to assess the performance of the algorithm on a few different parameters. The simulation tool will enable the testing of the algorithm, and a comparison between the developed algorithm's different iterations. Each iteration was evaluated to determine how it acted and to investigate which measures could be taken to further improve the estimations.

4.1 Simulation tool

The need for a test environment is evident when developing an algorithm of this kind. The tool helps ensure that the algorithm behaves in a manner acceptable for the use case, and to provide evidence to prove that the algorithm is working. For this project, a simple simulation environment was developed.

The simulations are based on a building map that is created initially, and a fixed set of controllers with fixed coordinates. Since the purpose of the simulation is to assess the algorithm compared to other localization methods, the tool must be able to run simulations on different algorithms, with different parameters. And since the important factor is only the differences between these algorithms, the finer details of the map can be abstracted and the building map is therefore as seen on Figure 4.1, simplified compared to a regular map. As seen on Figure 4.1 there are seven rooms on the map, six of these are "office" rooms and the middle room is a hallway. The controllers only populate the office rooms, and therefore add up, to six controllers.

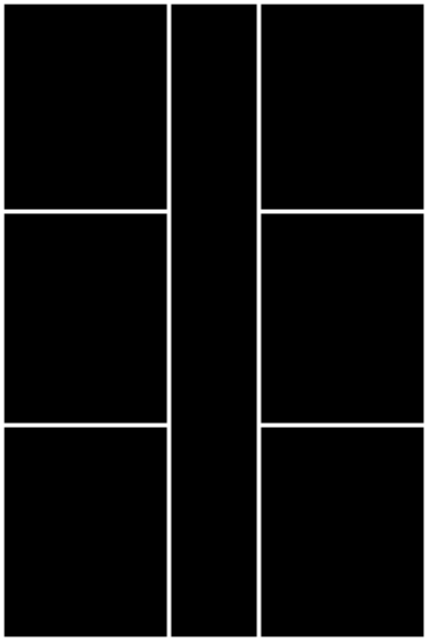


Figure 4.1: Initial building map, consisting of 6 rooms and a hallway.

Using the building map as a basis, a series of calculations are done. To simulate a wireless setup, a path-loss model must be chosen.

4.1.1 Path-loss

The purpose of the path-loss model is to emulate a simplified real-life scenario. In the environment, the distances from each controller to either sensor or the other controllers are known. This distance can then be used to calculate an expected RSSI which in turn, in the algorithm, is used to estimate the location of the given sensors. For the project, the log distance path-loss model was chosen, due to the simplicity and the accuracy as explained in Section 2.2. This was altered due to the environment setting not being clear line of sight, and to enable further control. The altered model is as also seen in Equation (2.9):

$$\text{RSSI} = \text{TxP} - 10 \cdot n \cdot \log(d) - k \cdot att + \omega \tag{4.1}$$

Here, k is the number of walls between the controller and the sensor, att is an estimated average value of how much the walls in the building attenuate the signal and ω is the noise that represents fluctuations in the signal strength, given as $\omega \sim \mathcal{N}(0, \sigma^2)$. Regarding the number of walls and the wall attenuation, each of these are calculated/counted, by utilising the building map.

4.1.2 Wall information

To count how many walls the signal travels through, the building map is loaded, and each of the paths are traced. A path is the route from a controller to either a sensor or another controller, which looks similar to the paths shown on Figure 2.7. A function for tracing the route of the path will count the number of walls in the path and will then save this number, which is used to calculate the RSSI.

The wall attenuation is set manually in the simulation tool and can be varied according to the needs of the operator. The real-life value can be determined by conducting an experiment in a real office setting, where the signal strength of a fixed distance is noted when the signal has clear line of sight and with a wall blocking the path. These values can then be compared to see how much the signal strength has decreased.

4.1.3 Simulation structure

The simulation tool consists of a few different elements, working together to provide the necessary information. Each of the different elements serves a different purpose in the simulation tool, but all contribute to enabling the testing of the algorithm. Firstly an explanation of the sensors and the controllers will be presented.

Sensors and Controllers

To emulate the office setting, it is necessary to be able to include each of the devices, that would belong in a physical setup. The sensor objects and the controllers are similar, but comes with a few differences. The sensors store an ID, a coordinate set used as the ground truth, a set of paths to all the controllers and a TxP variable that as explained in Section 2.2 is set to 0 dBm.

```
class sensor:
def __init__(self,id,coords,controllers,variables):
    self.id = id
    self.Coords = coords
    self.TxPower_dBm = 0
    self.paths = self.getPaths(controllers,variables)
```

Code-block 4.1: Sensor object.

The controller object likewise consists of a coordinate set and a TxP, but along with these they also store the room number of the room they are placed in.

Paths

The paths in the building map are calculated for each sensor-controller pair and again from each controller to the others. Each path stores the object of both devices

in the path, respectively the sensors and the controllers. The path objects store all information relevant to the simulation and calculation of the expected RSSI. The values calculated and stored are:

- Distance
- Walls (amount)
- TxP
- Path
- RSSI

The *Path* mentioned in the above list is the actual steps travelled through the building map and is utilised to count the number of walls between the devices. Each step in a *Path* consist of a (x, y) coordinate set. Thereby a *Path* becomes a list of coordinates. This can be seen in Code-block 4.2

```
def CalcPath(device1, device2):
    path = []
    X1 = device1.Coords.X
    Y1 = device1.Coords.Y
    X2 = device2.Coords.X
    Y2 = device2.Coords.Y
    direcX = 1
    if(X1>X2):
        direcX = -1
    direcY = 1
    if(Y1>Y2):
        direcY = -1
    lenX = (X2-X1)*direcX
    lenY = (Y2-Y1)*direcY

    if (lenY<=lenX):
        j = Y1
        for i in range(X1,X2, direcX):
            j += lenY/lenX * direcY
            path.append([i,math.floor(j)])
        return path
    else:
        j = X1
        for i in range(Y1,Y2, direcY):
            j += lenX/lenY * direcX
            path.append([math.floor(j),i])
        return path
```

Code-block 4.2: CalcPath function which returns the path travelled from one device to another.

All the information in a path is used to calculate the RSSI value. This RSSI will then be utilised later in *Estimates*, where the estimated location of the sensors, is calculated.

Environment

The environment object stores the information of the environment. These values are all controllers, all sensors, and every set of controller paths (paths from controller to controller) in the simulation.

```
class environment:
def __init__(self,variables):
    self.controllers = InitControllers(variables)
    self.sensors = InitSensors(self.controllers, variables)
    self.CPaths = getCPaths(self.controllers,variables)
    self.wallatt = CalcWallatt(self.CPaths, variables)
```

Code-block 4.3: class environment

These values are stored along with the estimation of the wall attenuation calculated between the controllers. The calculation of the wall attenuation is done by utilising the controllers' knowledge about the environment. As explained in Section 3.2.3.1, there are a few different variables used in this calculation.

- Measured RSSI
- Expected RSSI
- Number of walls

The controllers get a measured RSSI between them when transmitting packets. This value is then compared to the expected RSSI between these devices, which is calculated using the path-loss model and the knowledge of the controllers location.

$$\text{RSSI} = \text{TxP} - 10 \cdot n \cdot \log(d)$$

This scenario should represent the expected RSSI at direct LoS without interference. The environmental factor is therefore $n = 2$. The difference between these two values is then divided by the number of walls in the specific path to give an estimation of the wall attenuation.

$$WallAttenuation = (ExpectedRSSI - MeasuredRSSI)/(Walls)$$

It is thus assumed that all the difference in the RSSI values is due solely to the wall or walls that the signal has passed. This is the estimated wall attenuation that is utilised in the estimation of the locations of the sensors for the calibrated scenarios.

Estimates

The Estimate object consists of only four elements, used to store and evaluate the results. The four elements are:

- Estimated Coordinates
- Estimated Room
- Ground Truth Coordinates
- Ground Truth Room

As the names imply the coordinates calculated in the algorithm are stored and can then be checked against the ground truth, to evaluate how well the algorithm performed. The estimates are stored once for each sensor in the environment, and can then afterwards be evaluated together by counting the number of correct estimations in relation to failures, to give an overall accuracy.

$$Accuracy = \frac{correct}{fail + correct} \cdot 100\%$$

4.2 Simulation tests

To evaluate the algorithm the simulations will have to be run for different scenarios with different variables changing for each run. To ensure a uniform test environment it was chosen to structure the sensor placement. The simulations were structured around the middle room on the left side.

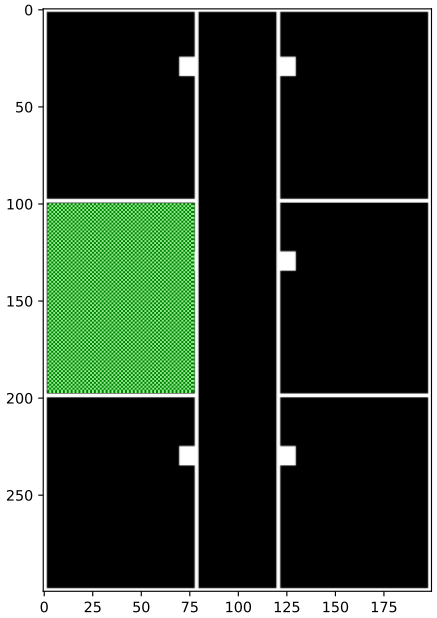


Figure 4.2: Building map showing the sensor grid in the chosen test room.

It was chosen to focus the simulation on this one room, for a couple of reasons.

Due to the simplicity of the building map, the two sides of the map are symmetrical. This means that running simulations on both sides would give the same results. Since the building map is only emulating a section of a larger building, it was important to make sure that there were neighbouring rooms on both sides of the test room, as this would be the case for most rooms in the actual office setting. Thereby the choice fell on testing only for the middle room.

For each run, sensors were placed in a grid in the chosen room, as to ensure that each part of the room was tested. The sensors were placed in a grid, of $97 \cdot 75$ to fit into the emulated room. This means that each run was tested with 7275 sensors. Figure 4.3 shows a zoomed view of the building map where the squares can be seen.

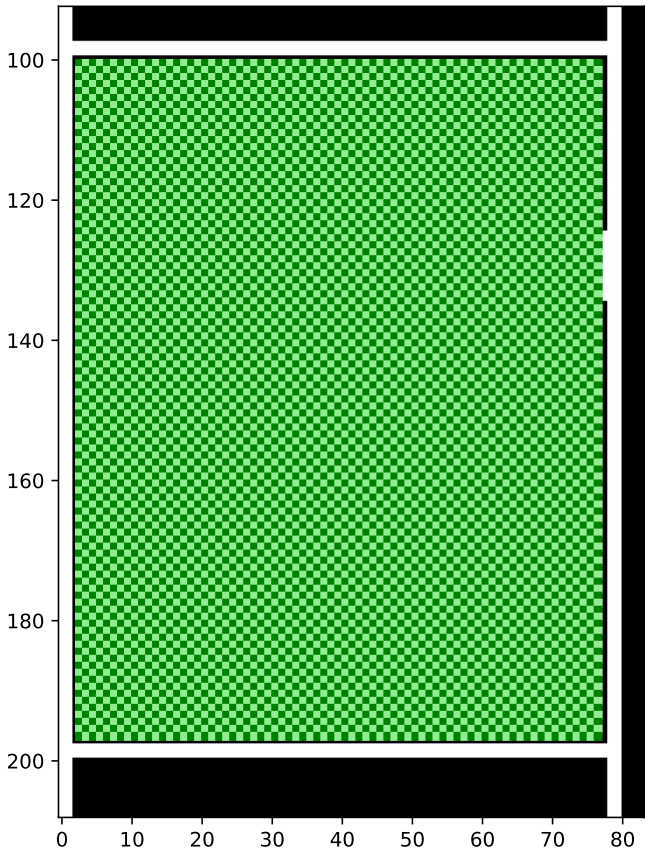


Figure 4.3: Zoomed room view with grid illustration - Each square is one sensor location.

Each scenario is tested with different variables and will be elaborated in Chapter 5.

5 Performance evaluation results

This chapter will begin with a description of the different scenarios that are used to evaluate the performance of the algorithm. Next, the results found by testing the algorithm in these scenarios will be presented, and the proposed algorithm, **Multilateration with filtration**, will be assessed in relation to two other methods being **Trilateration** and **Best RSSI** described in Section 3.2.

5.1 Simulation scenarios

The tests are all performed using the model shown in Equation (2.9) to simulate realistic data, where as mentioned in Chapter 2 the parameters are set to $TxP = 0dBm$ and $n = 2$.

The model used to estimate the distance between sensors and controllers, which will be fed to the localization algorithms is.

$$d = 10^{\frac{((Txp-RSSI)-(k\cdot\widetilde{att}))}{10\cdot(\tilde{n})}} \tag{5.1}$$

Where $\widetilde{att} = att + err_{att}$ and $\tilde{n} = n + err_n$. The RSSI used in this model is the value given from the path loss model shown in Equation (4.1) when generating the data. k represents the number of walls the signal has passed, which is found by taking the *path* from the sensor location to the controller, which is described in Section 4.1.3

The test scenarios were split into different segments. The first "Ideal case" was established to gain a foundation for testing the validity of the algorithm's results.

Secondly "Ideal environment" is presented, introducing noise to the test. As mentioned in Section 3.2.3.1, the wall attenuation is estimated with a prior calibration between the controllers. This test will therefore have two separate runs, one with the calibration of wall attenuation and one with a fixed wall attenuation, where it is assumed that the fixed value is provided from a datasheet, from measurements or tests, in this case, the value is assumed to be $att = 3dB$.

The last scenario "Realistic case" is used to evaluate how well the algorithm handles errors, introducing a variation in the errors for both environmental factor and wall attenuation.

$$err_n = [min \dots max]$$
$$err_{att} = [min \dots max]$$

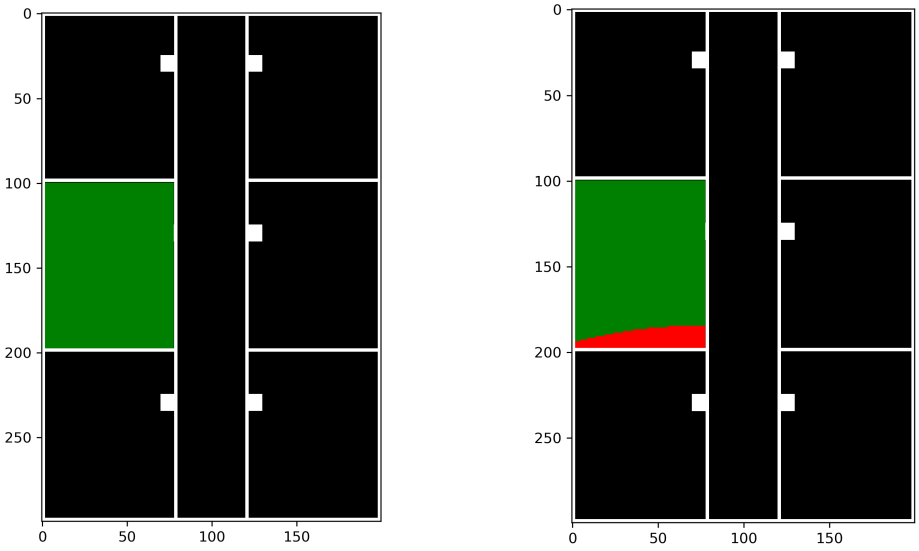
Where the minimum and maximum values will be given in an appropriate interval for the various parameters. Like the previous tests, separate runs with calibrated and fixed wall attenuation are also taken into account here.

5.1.1 Ideal case

The ideal case describes an environment without noise and with perfect conditions.

$$\omega = 0$$
$$err_n = 0$$
$$err_{att} = 0$$

The expectation for the tests, in this case, is a 100% success rate. Since there is no noise interfering with the signals, the algorithm should be able to determine the locations of the sensors, to the exact coordinates. For each sensor placement the location estimation will be calculated and evaluated. When evaluated the results will be illustrated on the building map where for each location (x,y) the square will be coloured either green indicating correct or red indicating wrong, depending on the outcome, which is seen on Figure 5.1.



(a) Multilateration with filtration and trilateration.

(b) Best RSSI.

Figure 5.1: Ideal case for the different algorithms.

From these heatmaps, it can be seen that the accuracy of both **Multilateration with filtration** and **Trilateration** is, as expected, 100%.

However, for the **Best RSSI** it can be seen that the accuracy is not 100%. Some errors occur at the bottom of the room as the sensors have better RSSI to the controller in the neighbouring room. This percentage and border will vary greatly

depending on the layout of the building, the placement of the controllers and the attenuation through walls etc. Due to the simplicity of the algorithm, changes in these factors, have substantial impact on the outcome and accuracy.

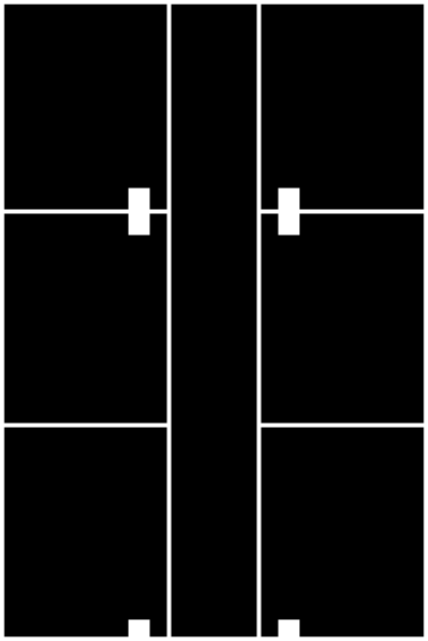


Figure 5.2: Alternative controller placement.

Looking at the placement of the controllers in Figure 5.2 it can be seen that the controllers are placed differently with some located much closer to each other. With the controllers placed closer, naturally, it will require less error to mistakenly assign the wrong controller to a sensor and thus identify the wrong room. If the controllers were placed in the centre of the room, or even just the centre on the walls adjacent to the hallway, the results would likely show closer to or hit 100% accuracy when tested without noise.

With the ideal test performed, the next test with added noise will be completed.

5.1.2 Ideal environment

The second case describes the same environment but with one structural difference. A Gaussian distributed random noise is added on the RSSI.

$$\begin{aligned} \omega = \mathcal{N}(0, \sigma^2) \rightarrow \sigma = [0 \dots 5] \\ err_n = 0 \\ err_{att} = 0 \end{aligned}$$

The standard deviation σ of the noise will be varied in a range and the tests will run for all coordinate points in the room, for each value of σ. The overall accuracy for each run is saved and plottet, to enable a comparison between each value of σ, and to see how fast the accuracy decreases. Figure 5.3 shows the results from both the calibrated wall attenuation and the fixed wall attenuation for both **Trilateration** and the proposed **Multilateration with filtration**.

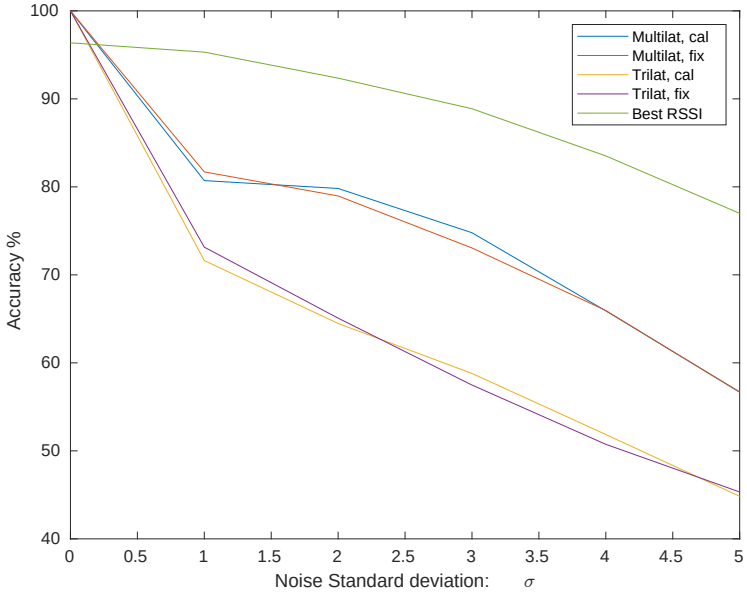


Figure 5.3: Significance of noise in relation to accuracy in room determination with calibrated and fixed wall attenuation.

The tests indicate how accurate in terms of room determination the algorithms are when adding noise to the signal.

From the graph in Figure 5.3 it can be seen that the performance of the algorithm **Multilateration with filtration** is outperforming the **Trilateration** with around 10 percentage points at a noise standard deviation of one, where the difference increases slightly towards two and three before it again converges towards each other. With an accuracy of around 80% at a noise standard deviation of $2dB$, the **Multilateration with filtration** algorithm does not outperform the **Best RSSI**. However, the **Best RSSI** does not provide any additional information about the location other than the room number, whereas the **Multilateration with filtration** provides a more accurate location.

The results show that noise has a very rapid effect on all versions of **Trilateration** and **Multilateration with filtration**, where even small values of σ make the accuracy significantly worse. To visualize where the errors occur, a heatmap is generated for $\sigma = 2dB$ for the **Multilateration with filtration** with calibrated wall attenuation.

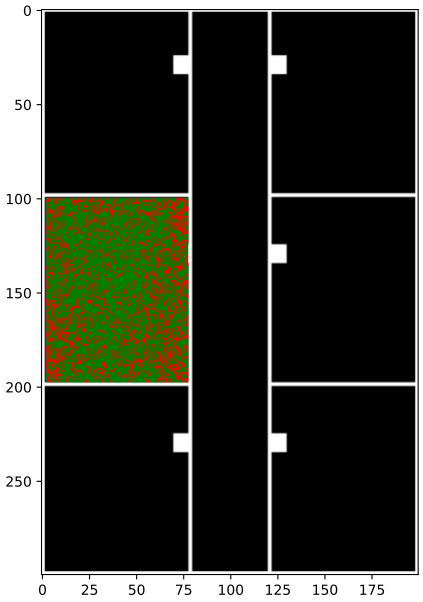


Figure 5.4: Heatmap showing **Multilateration with filtration** for calibrated wall attenuation and $\sigma = 2dB$.

From the heatmap it can be seen that the errors are spread across the entire room, however, it can be seen that along the walls more errors occur than in the middle of the room, which is expected.

A weakness in this test is that each sensor location is only tested once for each value of σ, and an "unfortunate" guess will therefore result in an error. To verify whether this could be an explanation, a test was conducted with a single sensor located at a fixed (140,20) coordinate with same σ and wall attenuation, but tested 100 times.

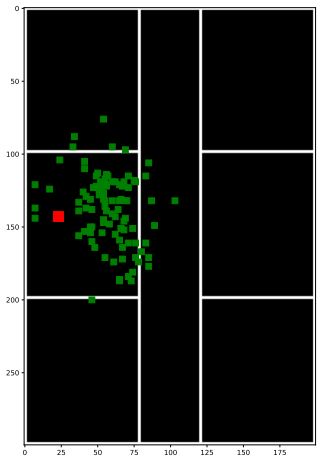


Figure 5.5: A single sensor tested 100 times at a fixed location.

As seen in Figure 5.5, many of the results are located in the correct room, more precisely 87% of the 100 runs are correct, indicating that multiple runs for each location could be necessary to fully understand the accuracy.

Next, a series of tests will be performed varying the environmental factor and the wall attenuation parameters.

5.1.3 Realistic cases

For the third case, a more realistic approach is employed. In this test the noise is constant and set to $\sigma = 2dB$, while $\tilde{n}$ and $\widetilde{att}$ are varied. The first test in the realistic case uses:

$$\begin{aligned} \omega &= \mathcal{N}(0, \sigma^2) = \mathcal{N}(0, 2^2 dB) \\ err_n &= 0 \\ err_{att} &= [-3dB \dots 3dB] \end{aligned}$$

Below in Figure 5.6, the accuracy of **Trilateration**, **Multilateration with filtration** and **Best RSSI** can be seen.

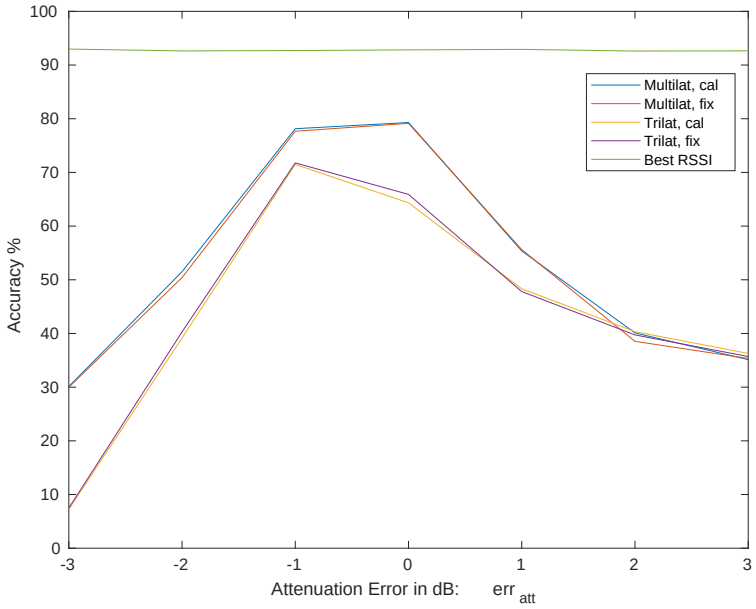


Figure 5.6: Tests run for **Trilateration** and **Multilateration with filtration**, for both calibrated and fixed wall attenuation, respectively. Best RSSI showing no change for varying err_{att}.

From Figure 5.6, it can be seen that the accuracy is very bad for large errors, however, for the edge cases, it is expected that the performance is bad. This becomes clear when looking at the data generation. Here the wall attenuation is set to $3dB$, therefore, by varying the error between $[-3dB \dots 3dB]$, it corresponds to a variation of $[-100\% \dots 100\%]$.

By comparing the **Trilateration** and **Multilateration with filtration** algorithms, it can be seen that the **Multilateration with filtration** has the overall better accuracy. The best point for **Trilateration** is $err_{att} = -1$ with an accuracy of approximately 71.5%, while **Multilateration with filtration** at this point have around 78% accuracy. However, **Multilateration with filtration** increases in accuracy as $err_{att} \to 0$, with 79% accuracy at $err_{att} = 0$, where **Trilateration** falls off and only has an accuracy of roughly 65%.

While it is expected that **Multilateration with filtration** outperforms **Trilateration**, it is not expected that **Trilateration** drops in accuracy between $err_{att} = -1 \dots 0$. Another fact to note is the "lack" of symmetry on either side, for both algorithms. This is expected though due to the relative size of the errors compared to the actual value. This is the case for both the attenuation error and the environmental error as seen in Figure 5.6 and Figure 5.8, respectively, where the decrease in accuracy is steeper on the left, with negative errors than with positive errors. Furthermore, it can be seen that the **Best RSSI** approach has a constant accuracy for varying err_{att}. This is because this method only uses the data that is generated, and does not use Equation (5.1) to determine distances, the results from any of the

realistic runs will therefore always result in a straight line as shown in Figure 5.6.

However, varying the *att* parameter in the data generation will give a picture of how well the **Best RSSI** approach is for different wall attenuations, which is seen in Figure 5.7.

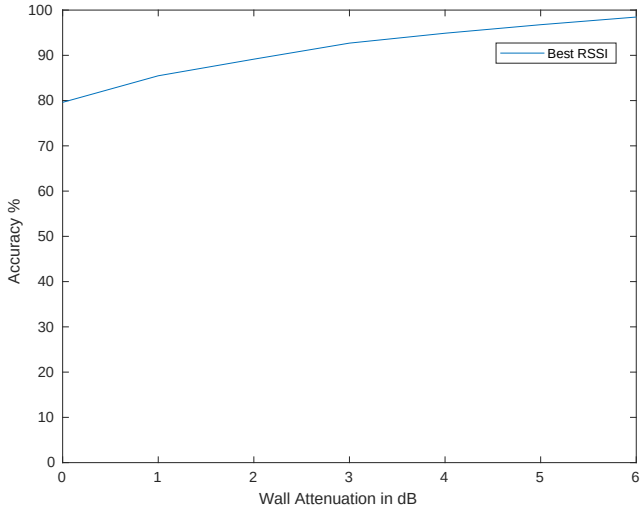


Figure 5.7: Best RSSI plotted for varying *att* in data generation.

This shows that with an increasing wall attenuation, the **Best RSSI** becomes more accurate, which is expected.

The second test in the realistic case uses:

$$\begin{aligned} \omega &= \mathcal{N}(0, \sigma^2) = \mathcal{N}(0, 2^2 dB) \\ err_n &= [-1 \ldots 2] \\ err_{att} &= 0 \end{aligned}$$

Similar to the previous test, Figure 5.8 shows the accuracy of the algorithms, when err_n ranges from -1 to 2.

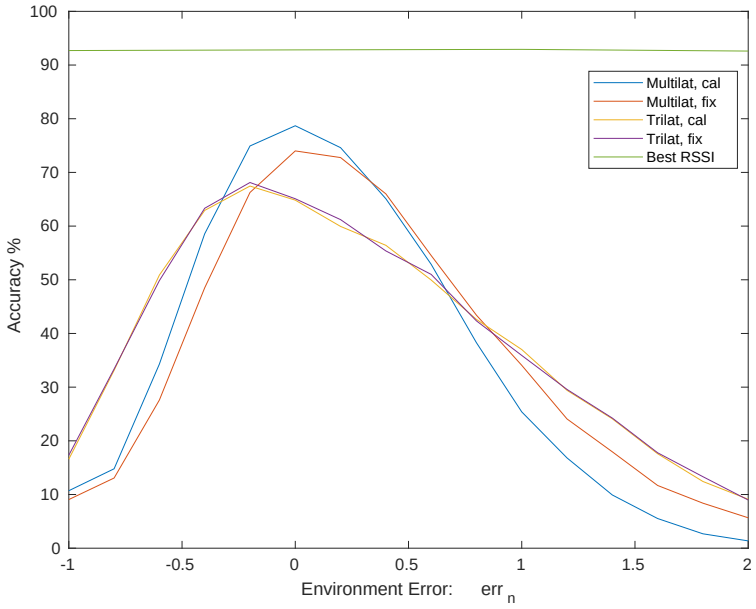


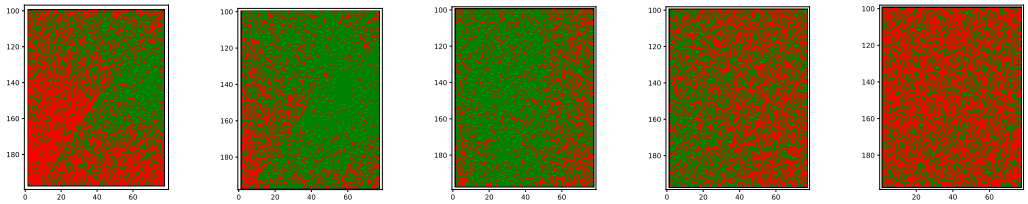
Figure 5.8: Tests run for **Trilateration** and **Multilateration with filtration**, for both calibrated and fixed wall attenuation, respectively. Best RSSI showing no change for varying err_n.

Similar to the previous test, the environmental factor has a large diversity, ranging from -50% to 100% in the err_n compared to the value used in the data generation, and from the graph is it clear that both **Trilateration** and **Multilateration with filtration** are very affected by errors in the n value. From Figure 5.8 it can be seen that the accuracy rapidly decreases even for small variations in n.

However, comparing **Trilateration** and **Multilateration with filtration** with each other, it can be seen that **Multilateration with filtration** in this scenario also outperforms the **Trilateration** method. Furthermore, some symmetry can be seen in the graphs, where for **Multilateration with filtration** it is around $err_n = 0$, while **Trilateration** peaks a little earlier at $err_n = -0.20$.

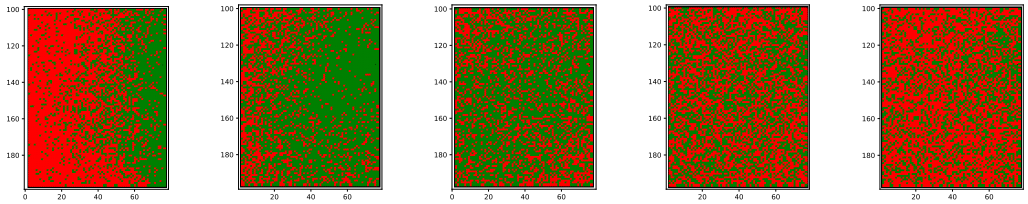
It is also seen that both **Trilateration** approaches peak at around 68% accuracy, while there is a slight difference in the **Multilateration with filtration** approaches. Here, with calibrated wall attenuation, it hits an accuracy of 78.5%, while with fixed wall attenuation, it hits only about 74%.

With results that describe the different scenarios, a series of heatmaps will be presented, where a visual image of the correct and incorrect guesses can be seen. These are given below in Figure 5.9 to Figure 5.12, for different values of err_{att} and with the **Multilateration with filtration** and **Trilateration** algorithms. This gives an insight into where the algorithms are guessing incorrectly, and whether it is always the worst in a particular area.



(a) $err_{att} = -2$ **(b)** $err_{att} = -1$ **(c)** $err_{att} = 0$ **(d)** $err_{att} = 1$ **(e)** $err_{att} = 2$

Figure 5.9: Showing the grid view of a sweep over the attenuation error for multilateration with filtration, where $n = 2$.



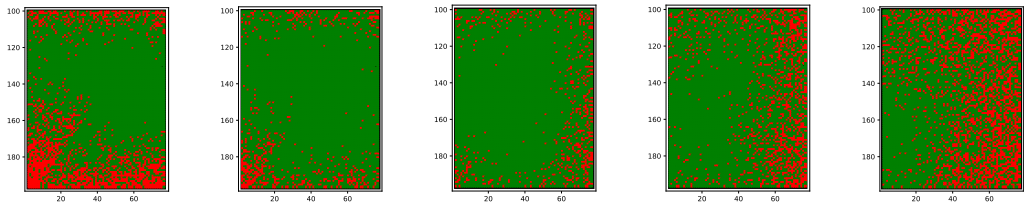
(a) $err_{att} = -2$ **(b)** $err_{att} = -1$ **(c)** $err_{att} = 0$ **(d)** $err_{att} = 1$ **(e)** $err_{att} = 2$

Figure 5.10: Showing the grid view of a sweep over the attenuation error for trilateration, where $n = 2$.

The two Figures above shows the **Multilateration with filtration** and **trilateration** run for an environmental factor of 2. It is clear to see that the algorithms both decrease in accuracy when the magnitude of the attenuation errors increase. However, the **Trilateration** algorithm improves slightly when the error value is -1 - This observation corresponds to the findings from Figure 5.6 which could be due to the layout of the building map. When looking at Figure 5.10a it is obvious that the majority of the errors are located near the left side of the room. This is the farthest side from the controller but also the rest of the rooms. This can be due to the nature of the algorithm's way of determining the different distances. A negative attenuation error will increase the distances but also the difference between the distances. Thereby essentially pushing the estimates out of the room on the far side from the controller. This could explain the overweight of failures on this side. On the other side when the errors are positive, there will be an additional "pull" from the controllers which can pull the estimate out of the room and into the hallway. This effect can also be seen on Figure 5.5 where almost all of the estimates are located to the right of the true location.

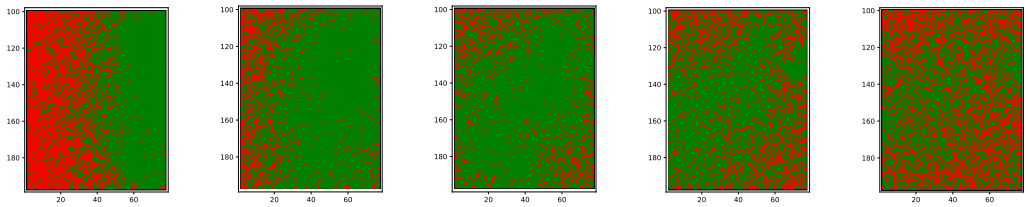
When looking at the **Multilateration with filtration** run in Figure 5.9a, a line can be seen forming almost in line with the diagonal of the room. The reason for this can maybe be found in the location of the controllers. Due to their off-centre placement, the probability of an error changes compared to if the controllers had been placed in the centre of the wall.

Below on Figure 5.11 the heatmaps from the $n = 4$ scenarios are illustrated.



(a) $err_{att} = -2$ **(b)** $err_{att} = -1$ **(c)** $err_{att} = 0$ **(d)** $err_{att} = 1$ **(e)** $err_{att} = 2$

Figure 5.11: Showing the grid view of a sweep over the attenuation error for multilateration with filtration, where $n = 4$.



(a) $err_{att} = -2$ **(b)** $err_{att} = -1$ **(c)** $err_{att} = 0$ **(d)** $err_{att} = 1$ **(e)** $err_{att} = 2$

Figure 5.12: Showing the grid view of a sweep over the attenuation error for trilateration, where $n = 4$.

As with the sweeps in Figure 5.9 and Figure 5.10 the two above sweeps show how the attenuation errors affect the different areas of the room. The only difference between these two and the two prior tests is the environmental factor. With both algorithms, the problematic locations are the same as before, but the influence on both algorithms is quite clear, and the improvement in the accuracy is evident.

To further evaluate the environmental factor n, an additional test was run with varying n from $n = 2$ to $n = 4$. This test was conducted for the **Multilateration with filtration** along with both **Trilateration** and **Best RSSI** algorithm.

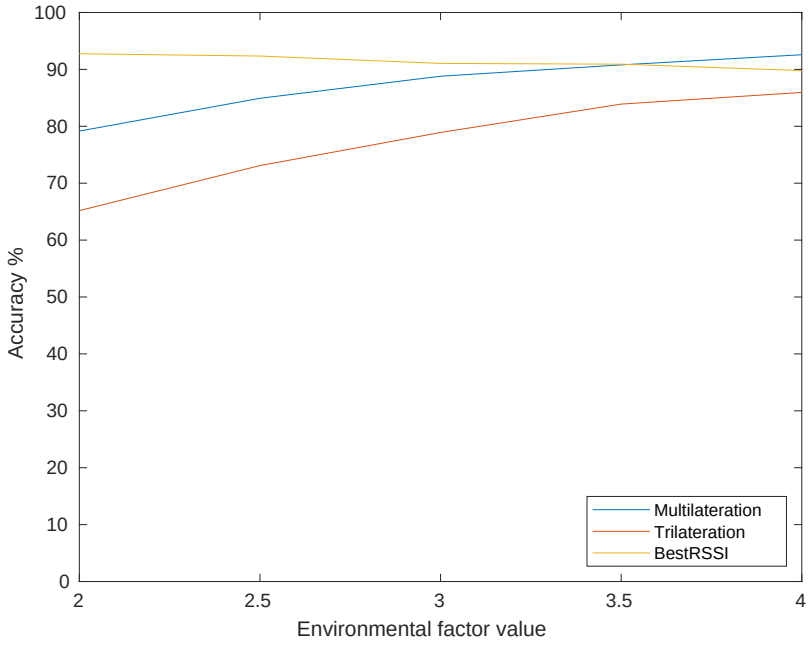


Figure 5.13: Sweep through Environmental factor (n) in path loss model.

As can be seen from Figure 5.13, the algorithms **Multilateration with filtration** and **Trilateration** both have an increase in accuracy, when n increases. Indicating that when the environment attenuates the signal more, the algorithms increase their accuracy compared to **Best RSSI**, which experiences a decrease in accuracy when the environmental factor increases.

In Figure 5.14 a test was conducted for a simple version of multilateration, to enable a comparison between this and the multilateration with wall information.

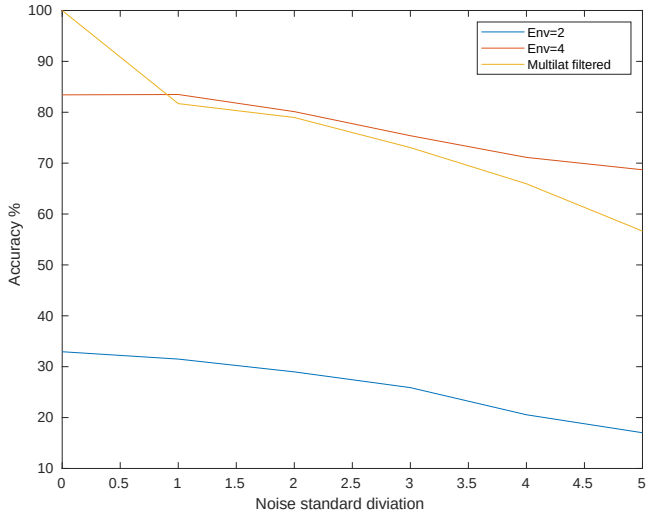


Figure 5.14: Varying noise with simple multilateration.

Two runs were conducted varying the noise standard deviation in both, but a change was made to the Environment factor. In the two different runs, the Environment factor was deviated in the estimation. It can be seen from the figure that the run with $Env = 4$ is substantially better than that of $Env = 2$, and even in this case better than the **Multilateration with filtration**. However, the accuracy of the simple multilateration is highly dependent on the environmental factor and the wall attenuation "cancelling" each other, which is why this result seems good, but in reality, is not reliable. When running a sweep of the wall attenuation it was found that the simple multilateration had the highest accuracy with a wall attenuation of between 6 and $8dB$. In a real office setting, the localisation accuracy must not rely on whether the wall attenuation happens to be between 6 and $8dB$, which shows why the calibration or prior knowledge of the wall attenuations is utilised in the **Multilateration with filtration** algorithm.

Furthermore, the overall performance of the **Multilateration with filtration** can be additionally analysed by using the cumulative distribution function and relating this to that of the other algorithms.

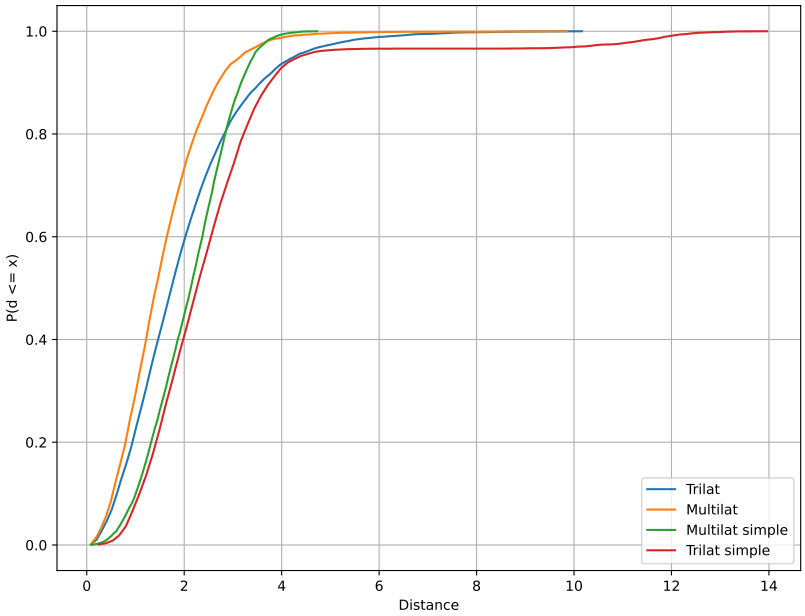


Figure 5.15: Cumulative distribution function for the different algorithms.

The graph in Figure 5.15 shows the cumulative distribution functions of the four algorithms, which indicates the probability that the distance error is less than or equal to the distance on the x-axis. Looking at the **Multilateration** it can be seen that the probability that the error is less than or equal to two meters is around 70-75%. Comparing this to the other algorithms shows that the **Multilateration** algorithm is better than the rest by around one meter when looking at the range from 0.5 to 0.8. Furthermore, when looking at the range from 0.8 to 0.95 the **Multilateration** algorithms are significantly better than their **Trilateration** counterparts. When looking at the progression through the graph it can be seen that the two simple algorithms follow each other when the distance errors are small, and the two algorithms with wall information follow each other. When the error increases the influence of the wall information decreases and the accuracy of the simple multilateration becomes better than that of the **Trilateration**.

With all tests presented, and the results discussed, the next chapter will focus on future work and what should be considered to improve the algorthm.

6 Future work

Throughout the report, it has been investigated how to develop and assess the proposed algorithm **Multilateration with filtration**, and several considerations have been made for how this approach could be improved, as well as what the next step would be. This chapter will outline some of these considerations, and highlight what should be the focus for the future work.

6.1 Better filtration

As mentioned in Section 2.2.2 multilateration is a method where multiple controllers are used to estimate the location of a sensor. It is therefore obvious that more calculations are necessary with this approach. Keeping this in mind, with the proposed algorithm, where each controller essentially draws four circles to indicate whether the signal has been blocked by zero, one, two or three walls, it quickly becomes a lot of calculations to find the one with the smallest error. To be specific, and as mentioned in Section 3.2, for 6 controllers this becomes $4^6 = 4096$ different sets of circle combinations. If this method was to be implemented in a larger building and the amount of in-range controllers were just 2 higher, the number of combinations would be $4^8 = 65536$, which is extremely many possibilities.

As the building map is known in this report, it has been possible to exclude some combinations of circles that are not possible. This is described in Section 3.2.5, where Figure 3.3 shows that with this initial filtration the number of combinations for 6 controllers decreases from 4096 to 1266. While this is a good start, it is clear that additional filtration of incorrect combinations will greatly improve the algorithm, especially if it was scaled to larger buildings.

One suggestion for how this could be done would be to make room determination based on another method, e.g. **Best RSSI**, and then solely calculate the smallest error based on the circle combinations that are in this room. In the event that two or more controllers have an almost identical "best" connection to the sensor, the calculations could be extended to these rooms. However, it is expected that this approach could decrease the number of circle combinations even further.

Furthermore, in the event that the situation is upscaled, an additional improvement is advantageous. With the additional controllers in a larger building with more rooms, the amount of calculations would again increase. Therefore a method for filtering the utilised controllers should be applied - This could be either a threshold on the RSSI, a location based exclusion, or simply a maximum controller

amount in the calculations. Either of these suggestions could potentially improve the algorithm's performance in a larger environment, and should therefore be considered in future work.

6.1.1 After filtration

To further utilise the information provided by the building map, considerations were made in regards to after filtration as well. This approach was thought of as a sort of sanity check for the results. The basic idea of after filtration is to check if the estimated location has the same amount of walls to the different controllers, as the amount used for the estimation. So if the estimation used the combination $\{1, 2, 0, 2, 1, 3\}$ of the distances to give an estimated location, the number of walls from that location to the controllers should correspond to this combination. If the two combination sets are not equal, the result can be discarded, and the second-best estimation can be checked to see if this location fits the requirements. This approach was slightly tested in the project, but with inconclusive results. Some further tests would have to be conducted to ensure that there is a viable solution that uses a fitting pair of combinations, and possibly tweak the algorithm to ensure that it does not discard all solutions.

6.2 Testing

One major challenge when using a simulation tool to evaluate a product, such as the **Multilateration with filtration** algorithm, is to recreate the reality in the simulation. This is highly important in order to verify whether or not the algorithm will work. If the simulated setting is simplified too much, the results provided in the simulation may look promising, however, will not work in real life. There are several possibilities to improve the current test environment, which was developed during this project. Either improve the information provided in the simulation, by using the building map to additionally introduce multipath and fading into the RSSI generation, use a more detailed building map with doors windows etc., use a path-loss model proven to be closer to real-life wave propagation, making measurements in a real-life setting and utilising these in the simulations, or adapt the entire test to an entirely physical setup to ensure real-life wave propagation, fading, multipath etc.

These additional changes may improve the results of the algorithm. In the simulation tool developed in this project, the wall attenuation is set to $3dB$. In a real-life scenario, this value may vary, which makes the wall attenuation calibration a much more usable additional piece of information. However, the accuracy of these calibrations may very well decrease in a real-life scenario, introducing an additional point of error.

6.3 Energy consumption

Looking back at the considerations established by the department of Build at Aalborg University in Section 3.1, it can be seen that energy consumption was also a focal point, since the sensors will operate on battery, and lower power usage means a longer lifespan. Although this has not been incorporated into this project, it is certainly an important point for future work and something that was considered.

Different approaches could be made to address this issue. Some sensors could probably have some kind of energy harvesting possibility, which would be a great solution, however, this might not be possible for all sensors.

To save on power consumption, a sensor that has been located and thus "paired" with its controller could initiate a connection to its controller, where it finds out what the smallest power it can send with, while still ensuring a reliable connection to the controller. This would obviously require an analysis of how much power could be saved and evaluate how the sensors would handle additional room disturbance.

With the considerations for future work, the next chapter will make an overall conclusion on the project as a whole.

7 Conclusion

In Chapter 1, the problem was introduced. Here it was presented that a proper Building Management System (BMS) will be able to increase productivity in a working environment and reduce energy costs for the building. However, the importance of having correct data in a BMS, and the consequence of having a wrongly configured system, was also highlighted. Furthermore, Wireless Sensor Network (WSN) was introduced to reduce the cost of wired connections and make scaling with multiple sensors easier.

In Section 1.1, both advantages and disadvantages of a WSN was described. From these, it was chosen to focus on *Localization of sensors*, which involves many challenges due to the imperfect surroundings in an indoor environment.

Based on the problem presented and the limited information available to determine the location of the sensors, the problem statement became: *"How can wireless sensors in an indoor environment be located using knowledge of the transmission power and the RSSI?"*.

In Chapter 2, the use case with Aalborg University department of Build was introduced, where the focus was on developing a method for wireless localization of sensors in a BMS. This led to an analysis of the different components that make up their setup for the BMS. Moreover, it was chosen to focus solely on Bluetooth for communication between devices, since this was already used in their equipment.

Section 2.2 analysed different approaches for localization techniques, and introduced the path loss model: $RSSI = \text{TxP} - 10 \cdot n \cdot log(d)$, where n is given as an environmental factor. In this analysis, it is clear that the different techniques have both advantages and disadvantages. Furthermore, it showed that indoor localization is a challenging task due to limited knowledge and a complex environment. It was chosen to move forward with a method using Received Signal Strength Indicator (RSSI) and converting this to distance with the formula: $d = 10^{\frac{\text{TxP}-RSSI}{10 \cdot n}}$.

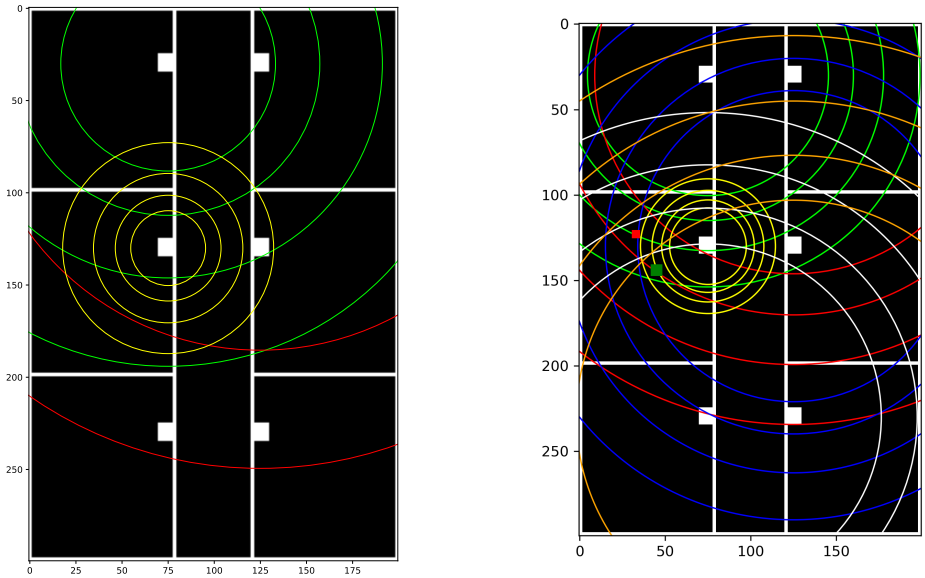
Some additional information was available in this project, namely the building map. In this project, it was chosen to use this information to examine how many walls would block the signal and incorporate this into the path loss model: $RSSI = \text{TxP} - 10 \cdot n \cdot log(d) - k \cdot att$, where k represents the number of walls blocking the path from a sensor, and att the wall attenuation. Based on the findings and the examination of the building map, it was chosen to use a lateral approach to determine location and draw circles for all wall combinations. The problem statement was refined to: *"How can wireless sensors in an indoor environment be located*

using knowledge of the TxP, the RSSI and the building map?".

In Chapter 3 the design approach was presented. This involves algorithm iterations in Section 3.2, where the process of distance and location estimation is described. For **Trilateration** the approach was to choose the three controllers with the best RSSI and use these to estimate the location of the sensor by using Equation (3.6) and Equation (3.7). However, with this approach the algorithm could use three controllers all sharing the same line, which defeats the purpose of trilateration, therefore a check was made to ensure that this was not possible.

To include the information given from the building map, some wall information was added to the algorithm. This included a calibration of wall attenuation, which was assumed could be done prior by the controllers. This was given as: $WallAttenuation = (ExpectedRSSI - MeasuredRSSI)/Walls$, where the $ExpectedRSSI$ was calculated for a perfect scenario using the path loss model without walls, and the $MeasuredRSSI$ uses the path loss model with walls and noise. It has become clear that this calibration of wall attenuation in a simulated setup is very challenging, as it is hard to emulate a real scenario. Advantageously, this calibration could be made as a testbed in a real environment, but this would require a proper test environment.

In Section 3.2.4 the approach is expanded from **Trilateration** to **Multilateration**. Both algorithms make use of the wall information and draw circles as seen on Figure 7.1.



(a) Trilateration.

(b) Multilateration.

Figure 7.1: Both algorithms drawing four circles representing zero, one, two and three walls blocking the signal.

As can be seen, the **Multilateration** approach has a lot more combinations of circles, which would only scale with a larger building. Therefore, by taking the information from the building map again, it was possible to filter some combinations that could not occur, eg. having no controller use the outer circle, indicating zero walls. From Figure 3.3 it can be seen that filtration greatly decreases the number of combinations (from 4096 to 1266 for 6 controllers), this approach is referred to as: **Multilateration with filtration**. However, it can be concluded that additional filtration would be beneficial to lower the number of combinations, especially if this approach was scaled to a larger building. In addition, as mentioned in Section 6.1 a threshold for the controllers would have to be included such that controllers far away do not influence the estimation, since they will add more calculations and not provide better results.

In Chapter 4 the performance evaluation approach is described, where it was chosen to develop a simulation tool used to evaluate the algorithms.

Section 4.1 describe all parts used in the simulation, which include: Sensors, Controllers, Paths, Environment and Estimates. Each of these parts contributed to the overall simulation tool, but the part that provided most of the building map information was the Paths. These were used to calculate the number of walls between sensors and controllers, which would also be utilized in a real setup to estimate/calibrate wall attenuations.

It is highlighted in Section 4.2 that only one room was chosen for testing. This was chosen since it was assumed that it would provide an overall picture of the situation, as there are neighbouring offices and the building map is symmetrical.

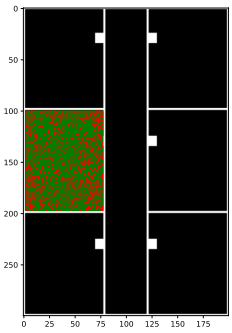
Furthermore, it was chosen to test the algorithms with a sensor located at every point (x,y) in the chosen room. This was done instead of choosing random locations for the sensors, as it would provide a more similar test for the different algorithms, which enabled more direct comparisons. In addition, this would provide results for the entire room, which could be beneficial to know. In turn, this meant that all tests were run with 7275 sensors for each value, which made the algorithms slow.

In Chapter 5 the results for different simulation scenarios were presented. These are categorised as: Ideal case, Ideal environment and Realistic cases. As mentioned in Section 5.1, the tests were performed with $TxP = 0dBm$ and $n = 2$ and for the fixed wall attenuation $att = 3dB$.

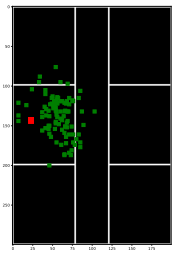
Section 5.1.1 shows the ideal case with 0 noise and no error in the parameters err_n and err_{att}. As can be seen in Figure 5.1, **Trilateration** and **Multilateration with filtration** have 100% accuracy, while the **Best RSSI** approach has some flaws. Previously conducted tests showed that if the controller was located centrally on the wall facing the hallway, then it would also achieve 100% accuracy. It can therefore

be concluded that the placement of the controllers has a substantial impact on the **Best RSSI** approach even with perfect conditions.

The next test included varying the noise, which is shown in Section 5.1.2, here $\sigma = [0 \dots 5]$. As can be seen in Figure 5.3, the **Multilateration with filtration** has a better accuracy than **Trilateration**, however, they are both outperformed by the **Best RSSI**, unless σ is extremely small. The lateral algorithms decrease rapidly in the beginning, which indicates that they deal poorly with noise, even when it is fairly small. At around $\sigma = 1$ they both start decreasing slower all the way to $\sigma = 5$. To analyse these results some heatmaps were created to visualize where the errors occurred, and a test was conducted with a single sensor fixed in location, these are also shown below in Figure 7.2.



(a) Entire heatmap for **Multilateration with filtration** with calibrated att and $\sigma = 2dB$.

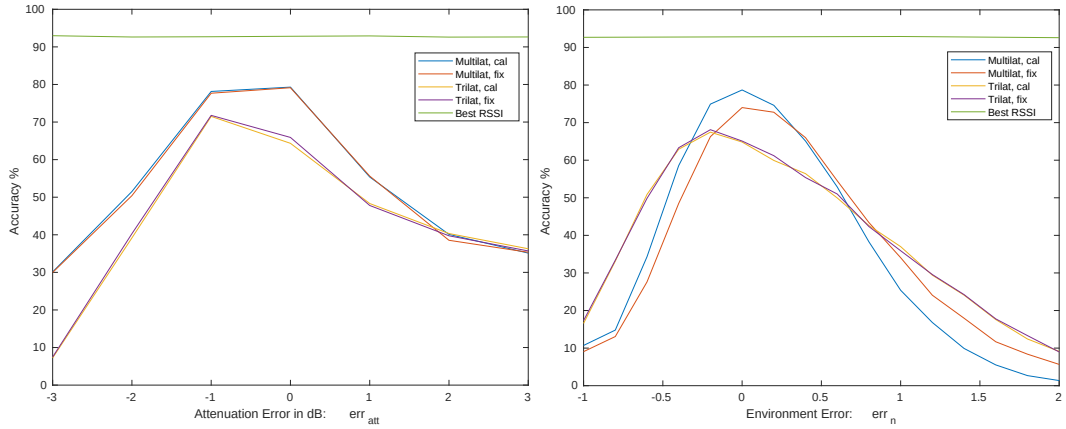


(b) A single point run 100 times with same algorithm and paramters.

Figure 7.2: (a) Shows all points run once each and (b) takes one example and run it 100 times.

Based on this, it can be concluded that it would be beneficial to run every point multiple times and make an average over the runs. However, with more than 7275 sensors, it would quickly become an extreme number of calculations, which could take a very long time to run. It can also be concluded, that Figure 5.3 is not 100% reliable, but it does still give an idea of how the algorithms handle noise. Advantageously, another test could be done with a smaller number of sensors so that each sensor can be tested multiple times for the same values.

Finally, the realistic cases were tested, where $\sigma = 2dB$ is constant and err_{att} and err_n varies. These test can be seen below in Figure 7.3.



(a) err_att varies and $err_n = 0$.

(b) err_n varies and $err_{att} = 0$.

Figure 7.3: Graph for the realistic cases.

From these tests, it can be concluded that **Multilateration with filtration** has a better performance when compared to **Trilateration**. However, for purely room determination and not a specific location, these tests show that the **Best RSSI** has superior accuracy. Looking at Figure 5.3, it can be seen what would be expected around $\sigma = 2dB$, however, both realistic cases show a drastic decrease in accuracy even for small errors.

To gain more information about the algorithms' handling of the errors, it could be an advantage to run these tests without noise and with higher resolution. Furthermore, as for the test in "Ideal environment", a different test setup with fewer sensors but more repetitions could give a better image of the realistic cases.

Throughout this project, it has become evident why indoor localization has been a major challenge for a long time and still is. With very limited knowledge and a highly dynamic environment, it may seem impossible to find a unique solution that solves this issue. In this project, the aim was to develop an algorithm that could locate sensors in a BMS with knowledge of TxP, RSSI and the building map. It can be concluded that an algorithm that utilised the information from the building map was developed.

The proposed algorithm achieved an increase in the accuracy of the location estimation, by incorporating the building information. When compared to the other location methods presented such as simple tri- and multilateration as well as trilateration with building information the proposed algorithm has higher room assignment accuracy as well as lower average distance error.

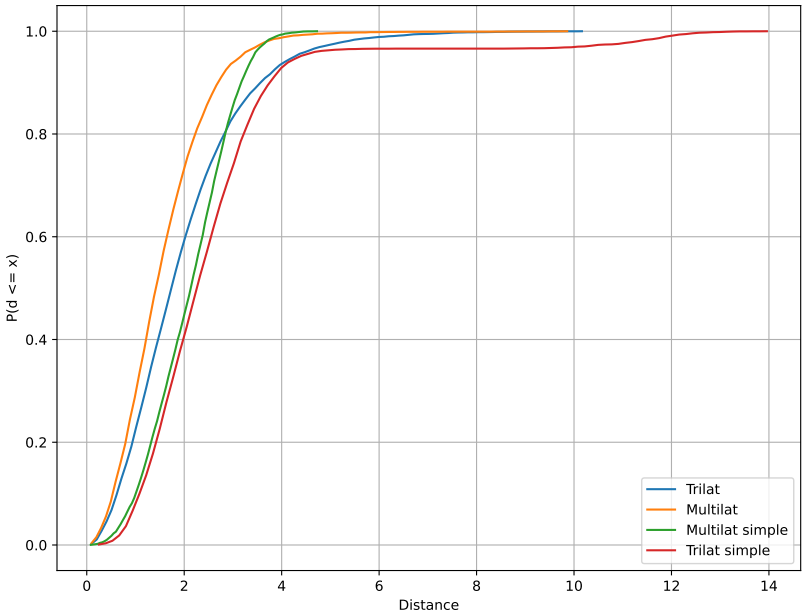


Figure 7.4: Cumulative distribution function for the different algorithms.

These improvements became evident when looking at the cumulative distribution function in Figure 7.4 where the multilateration with wall information and filtration has a higher probability for lower errors than the three others. Furthermore, it was found that the proposed algorithm outperformed the others when looking at the environmental factor as seen in Figure 5.13. When looking at the noise standard deviation in relation to the simple version of multilateration from Figure 5.14 it was found that although the simple version was able to outperform the proposed algorithm, it would only be able to do so if the environment factor chosen, fit correctly with the actual environmental attenuation.

Although better, there is still room for improvement, which could be done by either utilising additional information from the building map and layout or by combining multiple localisation methods into one algorithm with more steps.

Acronyms

AOA	Angle of Arrival	10, 11
AP	Access Point	10–14
BLE	Bluetooth Low Energy	9, 10
BMS	Building Management System	2–4, 6–9, 15, 52, 56
ETSI	European Telecommunications Standards Institute	13
GPS	Global positioning system	10
HCI	Host Controller Interface	15
HVAC	Heating, Ventilation, and Air Conditioning	2, 3
IBI	Intelligent Building Installation	6
IoT	Internet of Things	2
LoS	Line of Sight	10, 11, 13, 16, 18, 32
RSSI	Received Signal Strength Indicator	i, 5, 9, 10, 12–16, 18, 20–22, 24, 26, 29–32, 35–37, 41–43, 49, 52, 53, 56
RTT	Round Trip Time	14
SoA	State of Art	6
TDoA	Time Difference of Arrival	14
TOA	Time of Arrival	10, 12–14
TxP	Transmit Power	i, 9, 13, 16, 18, 20, 22, 56
WSN	Wireless Sensor Network	2–4, 52

Bibliography

[1] SIG bluetooth.com. *Bluetooth - Wireless Technology*. https://www.bluetooth.com/learn-about-bluetooth/tech-overview/. [Visited 11-05-2022]. 2022.

[2] Azat Rozyyev Fazli Subhan Halabi Hasbullah and Sheikh Tahir Bakhsh. "Indoor positioning in Bluetooth networks using fingerprinting and lateration approach". In: (2011). URL: https://ieeexplore.ieee.org/abstract/document/5772436.

[3] Lou Frenzel. *Wireless Sensor Networks Improve Building Efficiency, Security, and Comfort*. https://www.electronicdesign.com/technologies/embedded-revolution/article/21802034/wireless-sensor-networks-improve-building-efficiency-security-and-comfort. [Visited 22-02-2022]. November 17, 2016.

[4] GISGeography. *How GPS Receivers Work - Trilateration vs Triangulation*. https://gisgeography.com/trilateration-triangulation-gps/. [Visited 10-05-2022]. 2021.

[5] European Telecommunications Standards Institute. *ETSI EN 300 328 v2.2.2 (2019-07)*. https://www.etsi.org/deliver/etsi_en/300300_300399/300328/02.02.02_60/en_300328v020202p.pdf. [Visited 16-05-2022]. 2019.

[6] Intel. *Different Wi-Fi Protocols and Data Rates*. https://www.intel.com/content/www/us/en/support/articles/000005725/wireless/legacy-intel-wireless-products.html. [Visited 11-05-2022]. 28-10-2021.

[7] A.Nakib M. Dakkak B. Daachi and P. Siarry. "Multi-Layer Perceptron Neural Network and Nearest Neighbor Approaches for Indoor Localization". In: (2014). URL: https://www.researchgate.net/publication/275853189_Multi-Layer_Perceptron_Neural_Network_and_Nearest_Neighbor_Approaches_for_Indoor_Localization.

[8] Jason Marcel. *3 Key Factors That Determine the Range of Bluetooth*. https://www.bluetooth.com/blog/3-key-factors-that-determinethe-range-of-bluetooth/. [Visited 22-02-2022]. October 17, 2019.

[9] S. T. M. T. R. Mayasari Matsna Nuraini Rahman M. T. I. A. T. Hanuranto. *Trilateration and iterative multilateration algorithm for localization schemes on Wireless Sensor Network*. https://www.semanticscholar.org/paper/Trilateration-and-iterative-multilateration-for-on-Rahman-Hanuranto/e9d9fc073d4b7eaa49adafd079b60123figure/1. [Visited 11-05-2022]. 2017.

[10] E. Mok and G. Retscher. "Location determination using WiFi fingerprinting versus WiFi trilateration". In: (2007). URL: https://www.tandfonline.com/doi/full/10.1080/17489720701781905.

[11] Abdelmoumen Norrdine. *An Algebraic Solution to the Multilateration Problem*. https://www.researchgate.net/publication/275027725_An_Algebraic_Solution_to_the_Multilateration_Problem. [Visited 02-05-2022]. November, 2012.

[12] The Comfy Team. *The Comfort Productivity Connection*. https://comfyapp.com/the-comfort-productivity-connection/. [Visited 22-02-2022]. May 10, 2016.

[13] Wikipedia. *Law of sines*. https://en.wikipedia.org/wiki/Law_of_sines. [Visited 23-05-2022].

[14] rfwireless world. *Advantages of WSN | disadvantages of WSN*. https://www.rfwireless-world.com/Terminology/Advantages-and-Disadvantages-of-WSN.html. [Visited 28-02-2022].

[15] Da Zhang; Feng Xia; Zhuo Yang; Lin Yao and Wenhong Zhao. "Localization Technologies for Indoor Human Tracking". In: (2010). URL: https://ieeexplore.ieee.org/abstract/document/5482731.