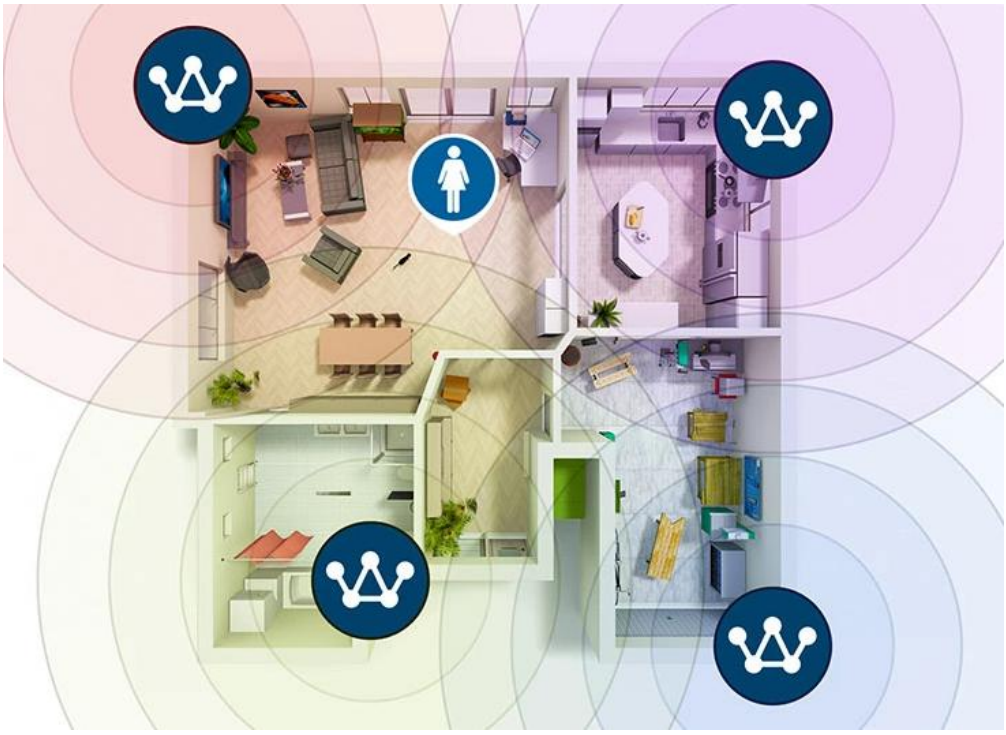
Signal Processing Systems
Mekelweg 4,
2628 CD Delft
The Netherlands
https://sps.tudelft.nl/

M.Sc. Thesis

Indoor in-network asset localization using Crownstone network

Vishakha K. Marathe B.Sc.

5242029



https://sps.ewi.tudelft.nl/shared/SPS/Thesisprojects/almende_crownstone_localization_2.jpg

Faculty of Electrical Engineering, Mathematics and Computer Science **Delft University of Technology**

Indoor in-network asset localization using Crownstone network

Thesis

submitted in partial fulfillment of the
requirements for the degree of

Master of Science

in

Electrical Engineering

by

Vishakha K. Marathe B.Sc.
born in Mumbai, India

This work was performed in:

Signal Processing Systems Group
Department of Microelectronics
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

Delft University of Technology

Delft University of Technology
Department of
Microelectronics

The undersigned hereby certify that they have read and recommend to the Faculty of Electrical Engineering, Mathematics and Computer Science for acceptance a thesis entitled **"Indoor in-network asset localization using Crownstone network"** by **Vishakha K. Marathe B.Sc.** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: September 13, 2023.

Chairman: _____
dr. R.T. Rajan

Advisor: _____
dr. R.T. Rajan

Committee Members: _____
ir.Martijn van der Marel

_____
ir.Ludo Stellingwerff

_____
dr.ir.C.J.M Verhoeven

Abstract

This thesis project has been done with Crownstone, a subsidiary company of Almende. One of their project goals was to develop indoor localization algorithms to determine room-level location of an asset within the Crownstone network for smart-building, home automation and healthcare applications. An asset is a wireless device transmitting Bluetooth messages that are heard by Crownstones(sensors) and measure the strength of the received signal(RSSI). The terms position and location are used to denote two different concepts in this thesis. Position refers to estimating the specific coordinates, whereas a location refers to a much wider space like a room. In this thesis, we are interested in getting a location estimate.

There are two most widely used and researched localization techniques to determine the location of the asset. First, a model-based(MB) method (e.g. Trilateration algorithm) which uses a mathematical model based on distance and second is a data-driven(DD) method (e.g. Fingerprinting algorithm) that relies on existing data, like RSSI to directly get the location. Algorithms are tested on real data collected by the Crownstones at the Almende office (test environment) divided into a finite number of locations or rooms. Metrics are defined based on the requirements of Almende to compare the MB algorithms with the DD algorithms. In this thesis, firstly, a centralized multilateration(MB-C) algorithm is implemented taking into account distances from N Crownstones at the office. Since one of the requirements was to perform in-network localization, a simple averaging consensus based distributed(MB-D) algorithm was selected and compared against the MB-C algorithm. Results show that the MB-D algorithm is faster, scalable and robust against single-point of failure than the MB-C but is less accurate and does not converge to the centralized solution for a noise variance greater than 10dB.

The MB algorithms have limitations in terms of selecting a model, learning the model parameters and an additional step of mapping the position output of the implemented MB algorithms to a location is also required. To deal with these challenges, a Machine-learning(ML) based data-driven algorithm is proposed. In this, training datasets were iteratively improved with different features. Then, an Ensemble based centralized ML algorithm (DD-C) is implemented, giving a classification accuracy of 65%. Algorithm is further improved by distributed data handling leading to a classification accuracy of 77%. There has been very little to no study on finding the room-level location of an asset in an indoor setting using a distributed ML based data-driven algorithm. A consensus based distributed ML algorithm (DD-D) is proposed that performs local predictions within the Crownstone network using the same globally trained model giving a classification accuracy of 73%.

The results show that the proposed DD algorithms perform better than the MB algorithms in terms of accuracy and are comparable in terms of prediction time. Results also indicate the proposed DD algorithms are more scalable, robust against noise but are computationally expensive.

Acknowledgments

The complete journey of this thesis has not only improved my academic knowledge but has also contributed significantly to my personal growth. I would like to extend my heartfelt gratitude to my mentor, dr. R.T. Rajan, for his invaluable guidance and insights that helped me improve my thesis. My appreciation extends to Almende for giving me the opportunity to work and for all the support and resources they provided. Furthermore, I am deeply thankful to my company supervisor Martijn van der Marel and appreciate the consistent discussions, advice and support throughout the project. I would also like to thank Ludo Stellingwerff for giving valuable inputs which proved instrumental in refining both my code and research methodologies.

I also wish to express my gratitude to the entire Almende team for fostering a sense of inclusivity during my time there. Additionally, I want to acknowledge and appreciate the DAS students for their constructive feedback, which played a significant role in enhancing the quality of my presentations. Furthermore, my heartfelt thanks go out to my fellow colleagues at TU Delft, with whom I was able to exchange and share my thesis experience.

This research enabled me to explore into a novel topic of Machine Learning. The entire journey has taught me to be patient and this would not have been possible without the guidance and blessings of my family and friends. Over the past year, the unconditional support received from Sagar, Nishanth, Jai and Deepali has been immensely important.

Vishakha K. Marathe B.Sc.
Delft, The Netherlands
September 13, 2023.

Contents

List of Figures

List of Tables

Symbols & Notations

$\hat{\mathbf{p}}$	Position estimate
$\hat{l}$	Location estimate
$\mathbf{C}$	Crownstones vector
$\mathbf{G}$	Ground truth matrix
$\mathbf{L}$	Output
$\mathbf{M}$	Machine learning models
$\mathbf{N}_e$	Neighbor matrix
$\mathbf{w}$	Noise vector
$\mathbf{X}$	RSSI measurements
$\mathcal{D}$	RSSI-distance dataset
$\mathcal{D}_g$	Global rssi dataset
$\mathcal{D}_{test}$	Test data set
$\mathcal{D}_{Tr}$	Training data set
n	Path loss exponent
R_{d_0}	RSSI at reference distance d_0
R_d	RSSI at distance d
r_{min}	Lowest RSSI value
T	Total time period

Notations:

a	Scalar
$\mathbf{a}$	Column vector
$\mathbf{A}$	Matrix
$\mathbf{A}_{ij}$	Value on i^{th} row and j^{th} column of matrix $\mathbf{A}$
$\mathbf{A}^T$	Transpose of matrix $\mathbf{A}$
$\mathbf{A}_{N\times M}$	Matrix with N rows and M columns
$\hat{\mathbf{a}}$	Estimated value of vector $\mathbf{a}$
$\bar{\mathbf{a}}$	Average value of vector $\mathbf{a}$
$\mathcal{A}$	Set
$\mathcal{N}(\mu, \sigma^2)$	Gaussian distribution with mean μ and variance σ^2
.	Dot product operator
$\|\|.\|\|$	Euclidean norm operator
diag($\mathbf{A}$)	Diagonal matrix $\mathbf{A}$
∇	Gradient operator
∇^2	Hessian operator

Table 0.1: Standard notations used in the algorithms and in Thesis

1 Introduction

This master thesis is a part of the graduation project done in the Faculty of EEMCS at Delft University of Technology. It is completed in collaboration with Almende, a company in Rotterdam, specializing in developing ICT solutions based around self-organizing networks. The thesis topic is based on indoor localization using sensor networks.

A sensor is a device which measures a physical input from the environment and converts it into data that can be interpreted by either a machine or a human. A wireless sensor network(WSN) consists of such a group of spatially dispersed sensors used for monitoring and recording physical conditions in the environment. Localization in WSN has been a major issue for applications that require the information received from the sensor nodes with accurate positional information of the event occurred. In some applications, sensor networks need to be deployed in inaccessible terrains and the position of sensor nodes may not be known from the beginning [1]. Therefore, a localization system is required in order to provide this positional information. All these examples lead to the fact that localization in WSN has a lot of importance for node addressing, geographic routing, object tracking and so on. Another term that is interchangeably used with localization, is positioning, defined as the process of determining the position of an object in space. Navigation and other location-aware services can be thought of as having their roots in localization. Finding someone or something's location is a long-standing difficulty and it has seen a lot of developments and innovations to localize and navigate. The most commonly used for this purpose is the Global positioning system(GPS). GPS is a space-based navigation system that relies on satellite for analyzing outdoor activities [1]. But, the non-availability of the GPS radio waves to penetrate the indoor space, makes it difficult to apply GPS technology to indoor applications.

A location positioning system(LPS) on the other hand, is a navigation system providing location information within a coverage area of the network. Such systems do not provide a global coverage, instead use a set of sensors that have a limited range leading to indoor positioning systems(IPS), a network of devices used to locate people or objects with technology other than GPS like, Wireless Fidelity(WiFi) and Bluetooth Low Energy(BLE) [1]. Indoor environments essentially refer to closed, fixed space which are physically bounded by building components. Overall, the use of WSN to determine where an object is within the confined space (indoor), has been an open problem that has been previously explored yet has a lot of room for innovative, better solutions.

1.1 Indoor Localization

Indoor localization stems from the ability to bring the power of outdoor localization into indoor environments. WiFi and Bluetooth are the two most ubiquitous technologies that can be used for indoor localization as mentioned before. However, WiFi access point (AP) sightings are buffered and reported via a single aggregate point, smearing the radio fingerprints and WiFi scans increasing the network traffic [2]. Also, not all the mobile platforms enable WiFi scans, but BLE does not suffer from these problems and advertisement packets are reported immediately. Bluetooth(BT) is a wireless technology standard for exchanging data over short distances operating within the 2.4GHz band [2], [3]. Another advantage of BT is that is is relatively cheap and has a relatively low power consumption. BLE devices are small, inexpensive and designed to run on batteries for many months to years that a lot of building will contain in the near future.

Indoor localization has two components, namely, Localization systems and Localization algorithms which represent the signal technologies and types of algorithms respectively that can be used for localization as seen in 1.1. In the range-based algorithms, nodes are able to measure the distance to their neighbors using received signal strength(RSSI) [3]. The other type of classification is based on model-based and data-driven methods. The text highlighted in blue indicates the chosen technology or methods. There are many algorithms that are widely researched and used, which is explained in detail in chapter 2.

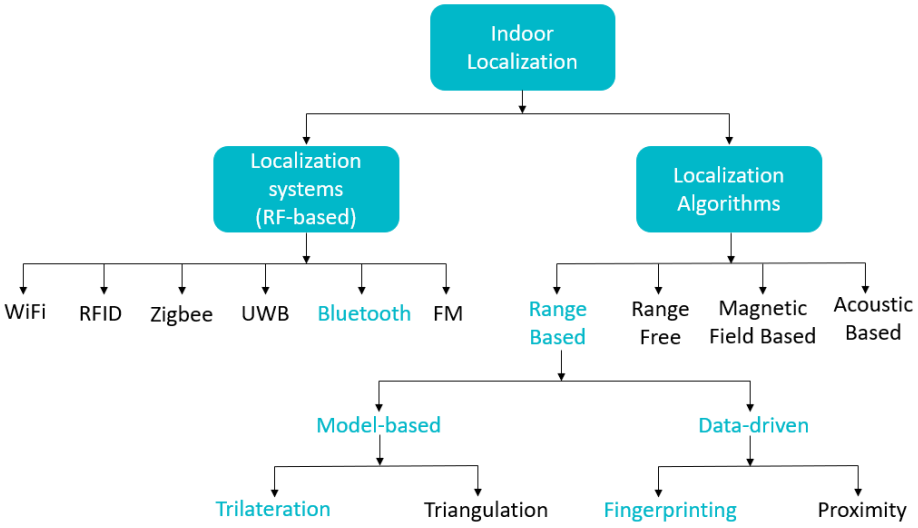


Figure 1.1: Classification of the components used in Indoor Localization. Text highlighted in Blue represents chosen method and technology.

1.2 Application Areas of Indoor Localization

The application areas of indoor localization in different industries are mentioned below, also focusing on finding location of an object or person is presented.

1.2.1 Healthcare

Indoor localization has several applications in the health care industry, it can be utilized for Patient Monitoring, to track the movement of patients. This can make sure of patients safety, especially for the patients who require continuous monitoring. Another application lies in asset tracking to help locate medical equipments, ensure efficient resource allocation [4]. Positioning and condition monitoring of hospital beds, digital call based on location can be done. Investigations in indoor scenarios where localization is performed by placing wearable sensors or beacon tags on human body.

1.2.2 Location-Based Services

Indoor localization can be leveraged to provide location-based services(LBS) and experience to users within indoor spaces [4]. LBS sends data dependent on context accessible by mobile device by knowing the geographical location, can be used to obtain safety information, or up-to-date data on events in the vicinity. Also, navigation applications to guide a user towards the target. LBS can achieve high level of personalized experience for the user. The positioning provider get added value by making use of localization by resource tracking, user statistics. These applications require an indoor positioning accuracy of couple of meters [5].

1.2.3 Smart building systems

Indoor localization also plays a vital role in enabling the development and implementation of smart building systems by tracking and monitoring the occupancy of various areas within the building. By utilizing BT and WiFi technology, system can detect presence and movement of individuals. The basic application of indoor navigation and way finding can be implemented for visitors, digital mapping and many more [5]. Also, proximity based services can be implemented to ensure security and access control for the employees within the indoor environment. Indoor localization can work with the WSN of different environmental sensors for analyzing temperature, air quality conditions as well as energy efficiency.

Apart from these, there are a lot of applications such as inventory management in warehouses to classify assets based on their locations to improve inventory control, even in hospitality for tracking essential equipments and improving asset utilization, reducing response time [5]. These are the applications that make use of the predicted location of an important asset and use it to their benefits in numerous ways, even extend it to tracking for continuous monitoring.

1.3 Motivation

Almende as a company provides ICT solutions in domains of healthcare, energy, mobility, manufacturing and security. They have a lot of subsidiaries, one of them is Crownstone, that combines indoor localization with building automation for homes and offices. One of the indoor localization goal is to determine the location of an asset(object/beacon) in an indoor environment. Although the terms position and location can be used interchangeably, they are used to denote two different concepts in the context of this thesis. Position can be expressed quantitatively as a set of numerical coordinates, whereas a location can be expressed qualitatively, such as a city, building, or room.

Localization can also be used to describe a technique that can constrain the position solution to a particular area, like a room instead of determining coordinates. Positioning is a subset of localization. There are a lot of applications where exact position is not required but a room-level location is sufficient, such as in health care for asset tracking, safety and security, travel, transportation, occupancy applications and many more [5]. An asset such as a BT beacon transmits advertisement packets that can be detected by the sensors. Aside from its payload, the receivers can also measure the signal strength that is RSSI which is readily available. RSSI signals are suitable for indoor Non-line of sight(NLOS) environments since they are quite noisy. Range-based localiztion is challenging because of the environmental effects like reflection and signal attenuation due to the indoor obstacles.

In indoor environments to accommodate human activities, it is more likely to have a higher density of wireless-capabilities and so for predicting a room-level location of an asset, RSSI signals from the BLE technology can be a good combination. Indoor localization using RSSI signals from BT is widely being used in a centralized manner, but implementing this for getting a location of an asset in a distributed way can be another approach to be examined. Moreover, the RSSI signals can be modelled using various physical equations or a more data-driven method can also be explored.

1.4 Problem Statement

Crownstone's technology can be used for indoor positioning, asset tracking where Crownstones are used as sensor nodes. In a Crownstone-equipped building, outlets have Crownstones installed that communicate over Bluetooth mesh and an asset transmits BT messages at regular interval. The indoor positioning systems are very well studied, but predicting the room-level location of an asset in an indoor environment using Crownstone network with comparable accuracy and simple techniques can be a challenge.

1.4.1 Objectives

The main objective of the thesis is to develop an algorithm to determine the location of an asset using the Crownstone network in an indoor environment. The objectives can be listed as:

- To develop indoor localization algorithms to estimate the location of an asset with room-level accuracy.
- Emphasis is to perform in-network localization algorithms that run within the Crownstone network, taking into account the strict hardware constraints of the Crownstone and available communication sources.
- A special focus is on implementing a Machine learning based data-driven approach that can be compared to the model-based methods.
- Performance comparison is based on two main metrics, the accuracy and prediction time of the algorithms.
- Apart from two main metrics, the model-based algorithms will be compared with the data-driven algorithms in terms of scalability, complexity and robustness of algorithms.

1.5 Thesis Outline

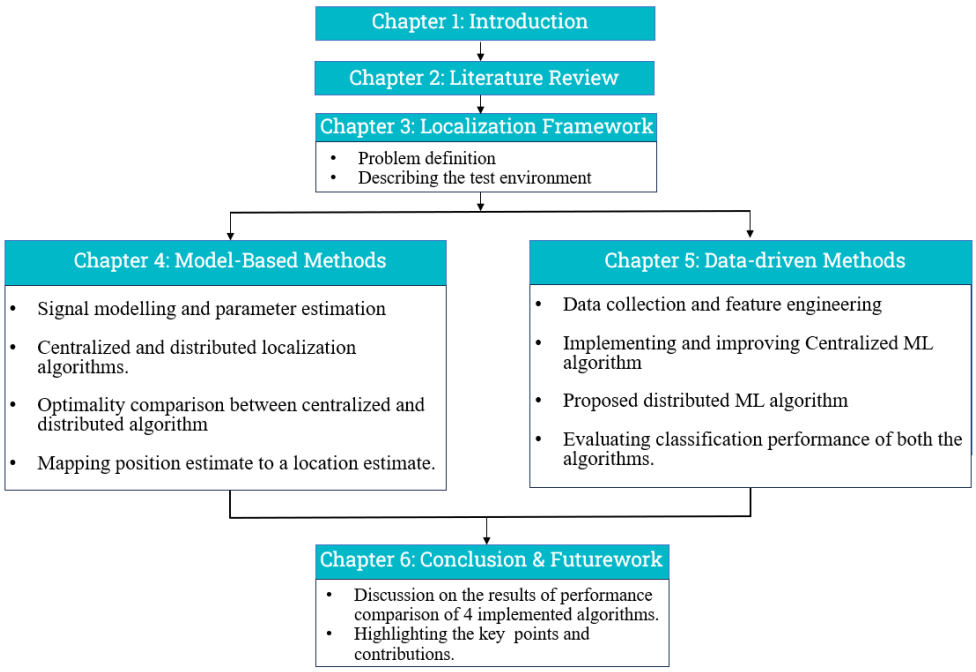


Figure 1.2: Block diagram representing the overview of the chapters in this Thesis.

The thesis is structured into various key chapters providing a clear framework starting with chapter 2 that examines related work and a literature review of previous research in the field of indoor localization. Chapter 3 defines the complete localization framework required to setup the problem along with the assumptions. The localization algorithm using model-based approach is described in detail in chapter 4. Chapter 5 deals with the machine learning based data-driven methods with results. In the end, chapter 6 compares the performance of two methods, followed by conclusion and future work as seen in figure 1.2

2 Literature Review

In this chapter, the related work with respect to indoor localization is presented in detail starting with the basic positioning techniques combined with the most commonly used signal technologies. More importance is given to exploring BLE based methods by using RSSI signals for indoor localizaton for a static case. Then, special focus lies on location based techniques in combination with the latest technology. Apart from that, the comparison of the previously used data-driven techniques for localization are discussed. Overall, the most widely researched and implemented model-based approach is presented along with the machine learning approach especially for asset localization.

Signal technologies for indoor localization
Techniques for indoor localization have received extensive research. The concept of localization using RF beacons is not new; sensor network groups and the robotics industry both make heavy use of it. Trilateration, Triangulation, Scene Analysis (Fingerprinting), Proximity, and Dead Reckoning are the main methods for indoor localization. These algorithms employ a variety of measurement techniques to estimate position. WiFi, BT, Zigbee, UWB, RFID, and other technologies are the main indoor localisation technologies [6]. Time of arrival (TOA), time difference of arrival (TDOA), direction of arrival (DOA), and received signal strength (RSSI) are common range-based techniques for localization. Numerous heuristic algorithms exist. Citation [7] states that although the TOA, TDOA, and DOA methods are more accurate than the RSSI approach, they are also more complicated and not always practicable, resulting in significant energy usage in networks with limited resources, such as the one in our example. Since distances can be effectively determined using RSSI, it is thought of as the essential indication data for indoor location. An effective option for low power, low complexity signal processing in a WSN is RSSI-based localization algorithms. The least square trilateration algorithm is one of the most straightforward and popular techniques for RSSI-based localization [7]. However, this algorithm heavily depends on ranging accuracy.

According to references [8], [9], and [10], there are a number of proposed location estimation protocols for sensor networks. All of these protocols follow a similar approach, in which certain network nodes are aware of their location and utilise that knowledge to estimate their location using data from other nodes. These details may include beacon coordinates and beacon signal characteristics like the RSSI. Convex optimization, minimum mean square error (MMSE), and Kalman filters are a few of the computationally intensive methods used in [8]. Citations [8] and [9] use the most realistic model for sensor network communication-RSSI, to predict the position of the beacon. The

position and location of the sensor node are often considered to be known in all of the proposed methods [10]. In model-based localization systems, the spatial propagation of raw RSSI signals is modelled using a variety of methods, including the Gaussian process and the Friis space model [11]. When dealing with numerous physical restrictions, such as localizing to an inaccessible location, RF models frequently fall short. Citation [11] points out that many methods, which produce misleading results, do not account for fundamental RSSI model elements such the standard deviation and path loss exponent.

Multiple environmental conditions, such as reflection, fluctuating humidity levels, obstructions, and multi-path fading, can weaken and affect signals. Traditional trilateration and signal propagation techniques are unable to provide precise position estimates as a result of the interaction of these affecting elements [12, 13]. In order to provide precise indoor positioning with a noisy WiFi channel, Microsoft's RADAR project proposed the first location fingerprinting technology. With a median accuracy of 2.94 metres, [14]'s deterministic closest neighbour technique was used to match the online fingerprints. Research on obtaining positional accuracy using several classification techniques, such as Support Vector Machines (SVM), Neural Networks (NN), and Bayesian Inference, has been done in [12, 14].

Machine-learning based localization
A further recent development in machine learning-based localization is the application of fingerprinting to RSSI indicator data [15]. In static measurements, the trilateration approach offers respectable accuracy, but it solely depends on the input RSSI data. According to [15], fingerprint-based approaches can outperform trilateration ones in complex-based circumstances. It is usual practise to fingerprint the environment in order to create a signature map for RSSI signals. In [2], Faragher and Harle employed a fingerprinting methodology to evaluate the effectiveness of BLE-based localization methods based on K-NN and proximity. These procedures, however, necessitate the time-consuming task of mapping the RSSI values in various environments. Additionally, it is mentioned in [2] that the probability map is non-adaptive and ignores environmental dynamics like RSSI reflection and multipath problems. Model-based approaches gather RSSI data to estimate target and AP separations, then employ fundamental positioning strategies to determine the target location [16].

Authors B.Yang et.al., proposed two-weighted least squares method aimed at improving the accuracy and robustness of node location [7]. To address the issue of accuracy loss in the conventional least square implementation, [17] also suggested another weighted non-linear least square method.

Distributed indoor localization
The essential characteristics of a WSN are a large number of low-cost nodes with constrained computing and power resources [18]. WSNs must be scalable, resilient to topology changes and node failures, as well as energy-efficient. As a result, a large number of distributed localization algorithms are being constructed by [18] based on node localization convex optimization ideas. As more nodes are added,

it is demonstrated that a distributed algorithm can solve SDP using interior point approaches and reach the maximum likelihood (ML) estimate. However, neither of the techniques covered in [18, 19] imposes any restrictions on communication between the nodes that could materially reduce the battery life of the sensors.

Direct location estimation of an asset
The position estimation is the end result of the most popular procedures previously described. But in our situation, a definite location is required. [11, 20] offers mapping the position coordinates found by trilateration method to a grid-based fingerprinting location. A floorplan can be used as an input for relocating positions based on a reference, according to [20]. In addition, [20] implemented location of interest based categorization, where the sensors are installed in predetermined sites whose positions and locations are known a priori, which may not be true in real scenarios.

Other significant work on positioning prediction directly from RSSI readings without extracting the distance parameter is done by [21, 22]. This strategy is more data-driven. For obtaining the positions, [21] implemented a machine learning (ML) method using decision trees (DT), SVM, and other methods. It should be noted that this still has to be mapped to a location. [21, 22] and [23] also recommend doing it manually, which is time-consuming. There is, however, not a great deal of research on using ML algorithms to directly estimate an object's room-level location based on RSSI readings. Additionally, only limited study has been done on the distributed implementation.

References	**Signal technology**	**Principle**	**Output**	**Method**	**Implementation**
[1],[3]	BT	Trilateration	$\hat{p}$	Model based	Centralized
[7],[8], [9]	WiFi	Proximity	$\hat{p}$	Data driven	Centralized
[13], [15]	BT,WiFi	Fingerprinting	$\hat{l}$	Model based	Centralized
[14]	BT	Fingerprinting	$\hat{l}$	Data driven	Centralized
[10], [16]	BT	Trilateration	$\hat{p}$	Model based	Distributed
[18]	WiFi	Trilateration	$\hat{p}$	Model based	Distributed
[19],[20], [22]	WiFi, BT	Fingerprinting	$\hat{l}$	Data driven	Distributed

Table 2.1: Summary of relevant research using model-based and data-driven techniques.

As seen from 2.1, there is a lot of research on indoor localization using BT and RSSI for estimating the position of an asset. Most of the implementation is centralized and model-based. A lot of references are there on fingerprinting principle used to determine the location in a distributed way. But, a lot less study is done on data-driven methods in general and even less in estimating a location directly.

3 Localization Framework

This chapter provides a formal definition of the localization problem implemented in the thesis. It gives a general structure of the underlying problem to be solved that can be modified according to the approach that is being implemented later. Section 3.2 lists all the assumptions for the entire experiment that are used in the entire report. This chapter ends with section 3.3 giving an overview of the test environment in which measurements were taken to perform asset localization followed by specifications of the devices used as sensors and the asset.

3.1 Modelling the problem

Consider an asynchronous Wireless Sensor Network(WSN) consisting of $\mathbf{N}$ Crownstones(sensor nodes), $\mathbf{C} = \{c_1, c_2, ..., c_N\}$ deployed in a given area or space denoted by $S \in \mathbb{R}^n, n \in \{2, 3\}$, that has finite number of different locations $\mathbf{L} = \{l_1, l_2, ..., l_M\}$ such that $l_i \subset S, \forall i$. Crownstones are not aware of their own location but are able to communicate with their neighbors $c_j \in \mathcal{N}_i$ that lie within their communication radius(R) i.e. $d(c_i, c_j) \leq R$, where d is the Euclidean distance. For simplicity, assume there is a single asset a at location $l_a \in \mathbf{L}$ emitting beacon frames heard by a subset of N Crownstones. Assume that the Cronwstones can only measure RSSI signals broadcasted by the asset. Given a set of observations or measurements $\mathbf{X}_{t,c}$ with $c \in \mathbf{C}$ and $t \in \mathbf{T}$ of the received RSSI values in the given time period $\mathbf{T} = \{t_1, t_2, ..., t_k\}$, the localization process can be defined as the repetitive evaluation of the following equation for a sequence of time periods:

$$\hat{l} = f(\mathbf{X}_{t,c}, \mathbf{T}) \tag{3.1}$$

Here, the $\hat{l}$ is the location estimate at each time period and it satisfies that each $\hat{l_i} \in \mathbf{L}$. The function f is called the localization function, which maps the asset and the measurements to a location estimate. The localization function is a mathematical model to estimate the location of an asset for a given set of observations. In some cases, the output could be a position estimate $\hat{\mathbf{p}} \in S$ if $\mathcal{M}(\hat{\mathbf{p}}) = \hat{l}$ is known, where $\mathcal{M}$ is a mapping function which maps $\hat{\mathbf{p}}$ to a $\hat{l}$. For this approach, we will call the localization function $\hat{\mathbf{p}} = g(\mathbf{X}, \mathbf{T})$ and implicitly $\hat{l} = f(\mathbf{X}, \mathbf{T}) = \mathcal{M}(g(\mathbf{X}, \mathbf{T}))$. Apart from $\mathbf{X}$ and $\mathbf{T}$, the function f can have additional parameters which depend on the localization algorithm used.

3.2 Assumptions

The indoor localization methods and algorithms implemented in the next chapters are based on certain assumptions about the test bed, hardware used and much more. All the assumptions are listed here in points. These assumptions are important for implementation and if not true can affect the results. The assumptions are:

A1. Asset is transmitting at regular intervals i.e. every second.

A2. Asset is stationary for a time period T.

A3. Asset always lies within the given space S.

A4. Position and location of the Crownstones are known in the office.

A5. Crownstone network is connected and can communicate with their neighbors.

A6. Crownstones can communicate with the central hub

A7. The position estimates can be mapped one-to-one to a unique location estimate.

A8. Crownstone has r_{min} as the RSSI value incase of no measurement.

3.3 Test Environment

This section describes about the test environment and its components and contains information about the setting in which the measurements and localization of the asset will be performed. Test environment consists of a setup for conducting the experiments or simulations in a particular setting which can be extended to different setups for a similar application. Test environment has a main component called the localization system made up of hardware and software. The hardware part is basically the Crownstones used as sensors and the software is an algorithm to process the Crownstone data to estimate the location of the asset. In this Thesis, Almende office is the testing bed or space denoted by $S \in \mathbb{R}^2$ which can be seen below in fig 3.1.

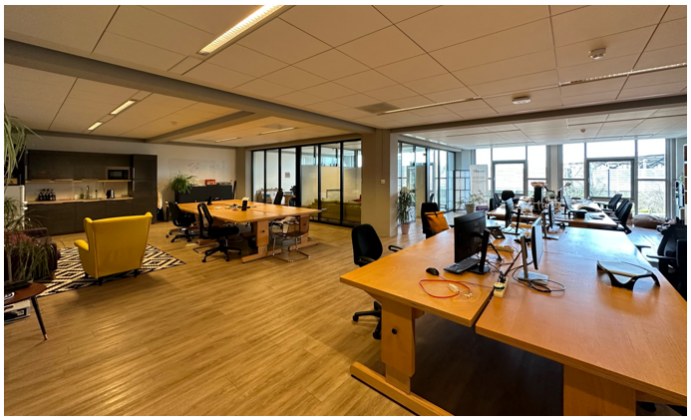


Figure 3.1: Image of Almende office to visualize the test environment

The office space has a total area of 14x14m^2 approximately divided into finite number of locations $L = \{l_1, l_2, ..., l_M\}$ such that $l_i \subset S, \forall i.$ of different area where every location or room is highlighted with different color. The red dots represent the sensors located in the given space with the unique IDs as shown in the figure 3.2 below. The asset is assumed to be present somewhere within this indoor space.

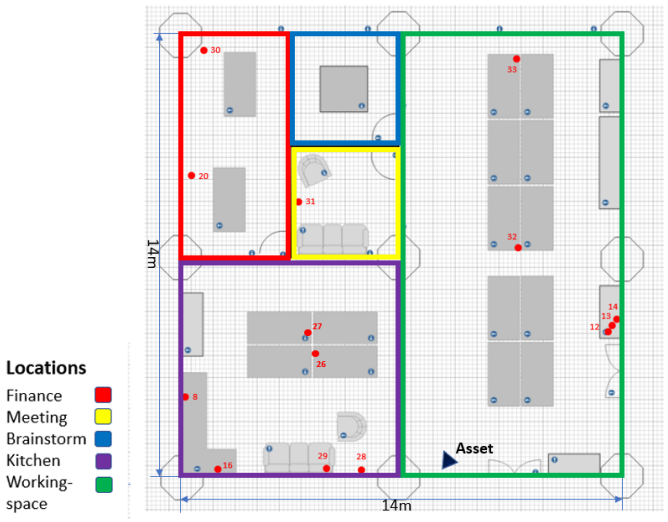


Figure 3.2: Floorplan of Almende office divided into finite number of locations. Crownstones locations are indicated by red dots along with their IDs.

3.3.1 Hardware setup

Figure 3.3: Two types of Crownstones are represented. The left image is built-in One Crownstone and right one is a plug type.

Crownstones are sensors nodes which receive RSSI beacon frames broadcasted by the Bluetooth Asset. They are basically used to collect the data and their exact positions

and locations are assumed to be known to us from the floorplan. The Crownstones can be placed behind single or double power outlets. In Almende office, the Crownstones are pre-installed. For Crownstones to perform localization, Bluetooth asset must be switched on and emitting beacon frames. Crownstone can also broadcast from 5m to upto 50m while still contributing to indoor positioning [24]. It uses BLE 4.0 and above communication protocol, can track smartphones with specified accuracy 1.5-4m. For room-level presence detection it requires 4 Crownstones in sight, not necessarily per room. Bluetooth mesh connects Crownstone with each other. It has a Nordic nRF52832 chip, 64kB RAM,513kB flash. The device weighs 40grams and is supported by both android and iOS [24]. The transmission power can be configured and has a resolution of 1dBm.

Figure 3.4: FeasyBeacon FSC-BP103 Bluetooth beacon used as an asset.

The asset 3.4 used in this thesis as transmitter is a FeasyBeacon product [25]. The asset(a) to be localized lies in this given test environment at a location $l_a \in L$, where $L \in S$ that emits beacon frames at a regular interval. The asset 3.4 used is a BT 5.1 proximity low energy beacon. It can work with android and iOs, configurable via the FeasyBeacon app. The major parameters like device name, transmission power, advertising interval can be configured. It has a battery life of 3 years. It is lightweight device weighing 0.02kgs made up of plastic material. The size of the asset is also small with dimensions 37.8mm x 33.8mm x 7.9mm [25].

4 Model-Based Methods

This chapter describes the model-based methods used to solve the localization problem and explain each phase of the solution in detail. The section 4.1 talks about the measurements for implementing the algorithms. The RSSI signal modelling using the log-distance model is described in detail in section 4.2. Section 4.3 consists of both the centralized and distributed algorithms and they are compared based on different criterias in the last section. A mapping function to map asset's $\hat{p}$ to a $\hat{l}$ is given in section 4.5 followed by conclusion.

A model-based approach for indoor asset localization using RSSI basically involves mathematically modelling the problem as accurately as possible using equations from physics. It involves leveraging the RSSI measurements to estimate the location of an asset within the indoor environment. The problem of indoor localization based on signal measurements and model-based method can be divided into **three** major parts:

- Data Collection
- Signal Modelling
- Localization

The first two phases are important before implementing the actual algorithms. The major phases of the model-based method can be seen below in the figure 4.1.

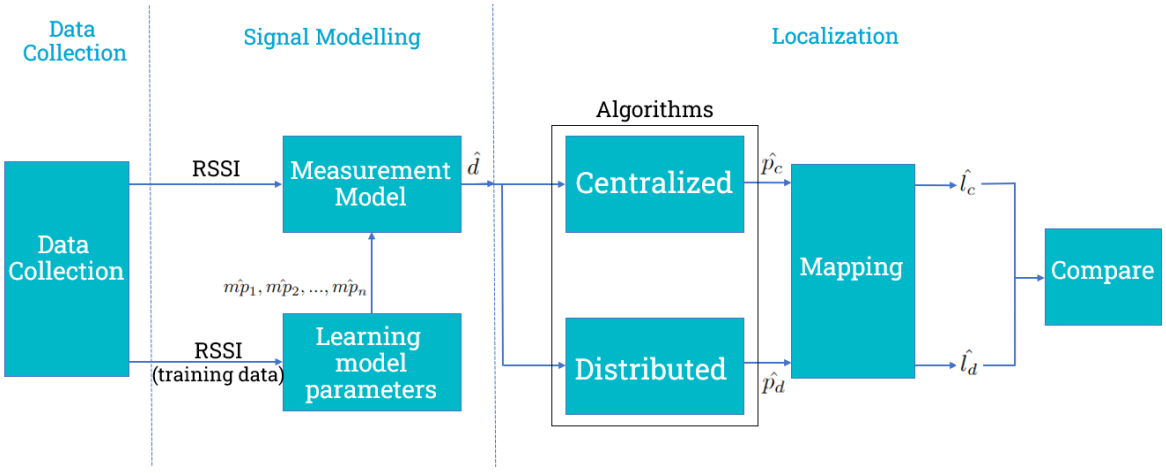


Figure 4.1: Block diagram representing the three phases of the model-based approach

Each block in the block diagram is explained in detail which eventually leads to a location estimate of the asset. In the end, performance of both the localization algorithms are compared after mapping as seen in figure 4.1.

4.1 Data Collection

The data collection is an important block of the model-based process. This section explains how the measurement system is used to collect a large indoor dataset in detail. Data collection is necessary for various reasons such as to understand the relationship between RSSI and distance, for parameter estimation, training phase and to process and analyze data to estimate the location of the asset. For this thesis, all the data is collected at the test environment with the help of Crownstones as sensor nodes and a bluetooth beacon(asset). Data is collected for different purposes:

- To calibrate the model
- To perform temporal analysis
- To test localization algorithms
- For training purpose (offline phase)

In this chapter, we only look at the first three points for which data is collected and data collection for training purposes is briefly explained in the next chapter.

4.1.1 To calibrate the model

Out of the 14 Crownstones present in the office, one of them was selected at random for data collection. An inbuilt iOs application called "measure app" on iPhone was used to measure the distance between the asset and the Crownstones. The measure app has an accuracy of 95% and since it is used for calibration purpose, it was verified using a measuring tape. The asset is kept at a distance d from the Crownstones as shown in the figure 4.2 below

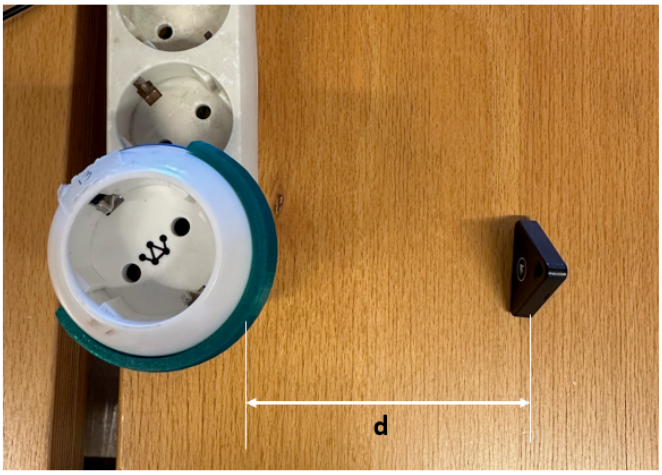


Figure 4.2: Image depicting the measurement setup with Crownstone kept at a distance d from the asset

The asset is placed at 0.1,0.2,0.5,1,2 and 4 meters distance from the sensor. At each distance, RSSI measurements were gathered by the Crownstone for a time period of 5 minutes bu noting down the Unix timestamps. Based on the timestamps, the data was exported from the Almende data base. It was made sure that there were no obstacles present in between the sensor and the asset. Therefore, the Crownstones made all Line-of-Sight(LOS) measurements. The use of this data is explained in the subsequent sections.

4.1.2 To perform temporal analysis

An experiment was conducted at the office on two different days for a period of approximately 5 hours. In this, the asset was placed at a fixed spot facing the kitchen area, transmitting at regular intervals. The RSSI values received by all the Crownstones continuously with Non-line of sight(NLOS) measurements. The goal of this experiment was to understand how the obstacles affect the RSSI values in the real world scenario.

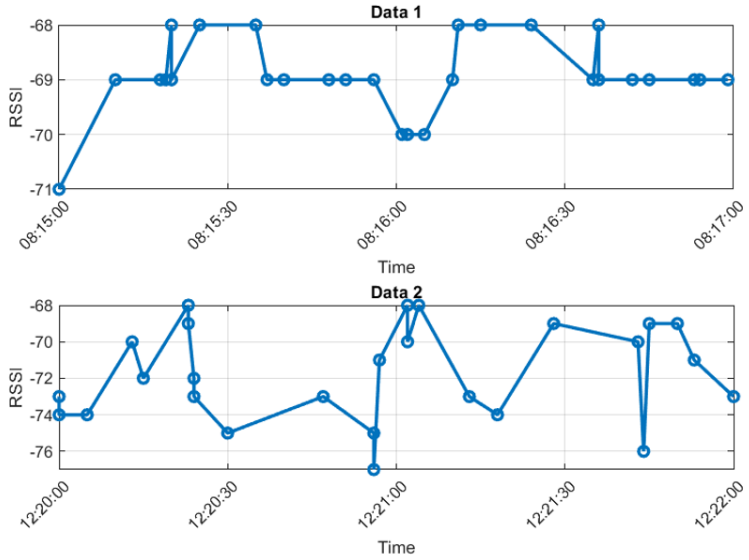


Figure 4.3: RSSI measurements recorded by C12 at 2 different timeslots in one day. The top plot shows RSSI measurements before the office timings and bottom plot during lunch time.

The figure 4.3 shows how the RSSI values fluctuate with time. All the measurements can be extracted from the Almende dashboard with Crownstone IDs and timestamps. The plot shown below has more fluctuations because that is usually when there are more people walking around that room blocking the signals than around 8am. 4.3 also shows how RSSI values are affected by the dynamic environment during different time of the day. This dataset is used later for further analysis, parameter estimation described in detail.

4.1.3 To test localization algorithms

It is important to build a dataset in order to test the localization algorithms. This dataset should consist of RSSI values measured from a subset of Crownstones whose positions are assumed to be known. These set of measurements are taken by keeping the asset at a fixed position and then recording measurements from 3 Crownstones and also multiple Crownstones in the kitchen room. The representation of this is shown in the fig 4.4.

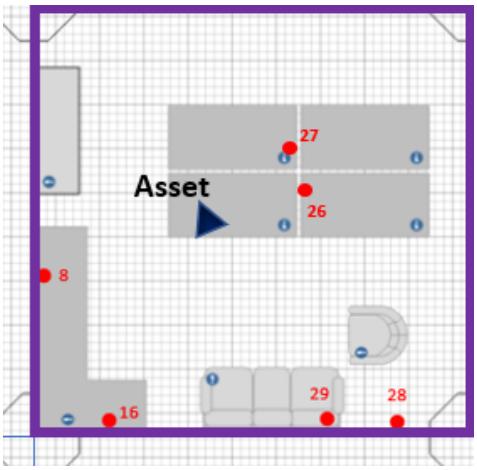


Figure 4.4: Image depicting an example of data measurement setup in one of the rooms

From the fig 4.4, the asset is kept at a known position for a fixed time period T. The measurements are extracted from C8,C16 and C26 for implementing trilateration. For multi-lateration case, data from all the Crownstone given are considered. The same steps apply for all the other rooms while collecting data from the Crownstones present. The implementation is explained later.

4.2 Signal modelling

In this section, we will model the RSSI measurements to extract important parameter out of it for localization. The observed signal needs to be modelled using mathematical equations in model based approach. For all range-based measurements, such as TOA, TDOA,RSS and DOA, the signal models can be generalized as:

$$\mathbf{x} = f(\mathbf{p}) + \mathbf{w} \tag{4.1}$$

where x is the measurement vector, p is asset position to be determined, $f(p)$ is a known non-linear function in p and w is additive noise vector.

4.2.1 Measurement model

RSSI as the name suggests is a metric that indicates the strength of the received signal. The bluetooth RSSI is a measure that represents the relative quality level of a BT signal received on a device. The values are always on a logarithmic scale, negative and measured in decibels(dBm). It is a distance related parameter. The signal power or RSSI observed over a wireless channel changes in a random and unpredictable manner due to factors such as interference, obstacles, distance and other environmental conditions. Therefore, it is not possible to determine the distance between a transmitter and receiver based on a single measurement of RSS, thus can only be characterized statistically [8, 26]. Therefore, a statistical model for RSS is employed to estimate a transmitter-receiver distance or range d_i, which can be used to infer position coordinates later .

The most commonly used propagation model is the $Friis$ radio propagation model that is used in free space [1]. However, In wireless transmission media, a signal transmitted by a mobile device travels along a number of different paths of varying lengths, referred to as multipaths. This radio propagation causes signal distortion and fades, which are attributed to reflection, scattering, diffraction from buildings, trees, furniture and other obstructions. The overall loss in the signal is typically characterized as a product of three factors, local mean propagation loss, long term or slow fading, short term or fast fading. First two factors are large scale effects, and the last is a small scale effect. This shows that indoor environment is not free space, in which case another model mentioned in [8, 27, 28] used in wireless communication systems is the log-distance path loss model as shown below:

$$R_d = R_{d_0} - 10n \log\left(d/d_0\right) + \mathbf{w}(\sigma) \tag{4.2}$$

where, R_d is the RSSI measured at distance d, R_{d_0} is the RSSI measured at reference distance d_0, n is the path-loss exponent and $\mathbf{w}$ is the noise vector with σ as standard deviation.

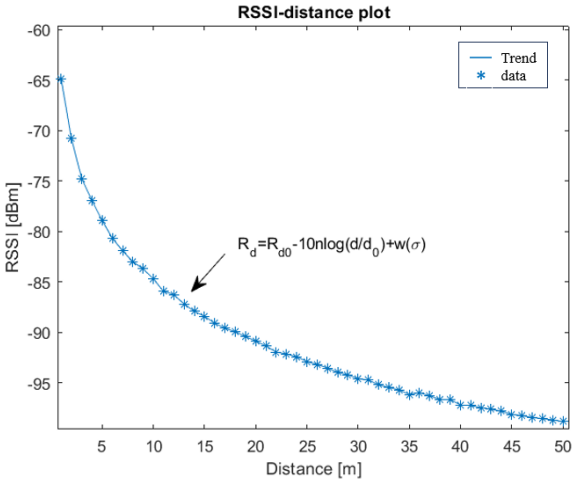


Figure 4.5: Plot of RSSI vs distance using the log-distance path loss model.

The path-loss model is typically modeled as a function of the path loss. The relationship between the path loss and distance is captured using the log-distance equation 4.2. The log-distance path loss model is a modification of the $Friis$ model. This model assumes that the RSSI decreases logarithmcally as the distance between the Crownstone and asset increases. This behaviour can be seen from the plot 4.5.

The model equation 4.2 states that the RSSI at a certain distance d can be estimated by subtracting the attenuation factor from the reference RSSI value R_{d_0}. The attenuation factor is dependent on the path loss exponent n and the logarithm of the distance. The higher the value of n, the faster the RSSI decreases with distance.

4.2.1.1 Maximum Likelihood Estimator for Distance estimation

The distance estimate can be derived from the path loss model 4.2 using Maximum Likelihood Estimator(MLE). MLE is a statistical method used to estimate the parameters of a model by maximizing the likelihood of observing the given data. In the case of the path loss model, the distance estimate can be obtained by maximizing the likelihood of the observed RSSI measurements given the model parameters [7].

To derive the distance estimate using MLE, goal is to determine that value of d that maximizes the likelihood of observing the given RSSI measurements. It is assumed that the RSSI measurements are independent and identically distributed (iid). Usually, the log-likelihood function(LLF) is taken to simplify the calculation as shown below:

$$logL(d; R_{d_0}, n) = \sum logf(R_d|d; R_{d_0}, n) \tag{4.3}$$

where: $logL(d; R_{d_0}, n)$ is the LLF given the model parameters R_{d_0} and n and $f(R_d|d; R_{d_0}, n)$ is the probability density function(PDF) of RSSI measurements at distance d.

To find the distance estimate, the LLF is maximized with respect to d, setting the derivative equal to zero and solving, finally we get:

$$\hat{d} = d_0.10^{(\hat{R}_{d_0} - R_d)/10n} \tag{4.4}$$

The derivation assumes a simplified scenario where the RSSI measurements error follow a Gaussian distribution with mean zero and variance σ^2 and does not account for other factors like measurement noise, interference or multipath effects. In real-world cases, this is not the case and requires additional adaptations to the model and estimation of the parameters. To use the path loss-log distance model for RSSI-based localization, the model parameters R_{d_0}, n need to be calibrated or estimated.

4.2.2 Learning model parameters

The path loss-log distance model provides a simplified representation of the signal propagation characteristics, and the actual RSSI measurements can be affected by various

factors such as interference, multipath effects, and environmental changes. Therefore, to get the best possible estimate of the distance, learning the model parameters is essential. This section describes the methods used for estimating the three main parameters, R_{d_0}, n and σ of noise vector $\mathbf{w}$.

4.2.2.1 RSSI at reference distance (R_{d_0})

In the path loss model, R_{d_0} is a parameter that represents the RSSI at a reference distance from the transmitter. To estimate R_{d_0} from 4.2, typically a calibration or measurement is performed at a known reference distance. In many literatures like [7, 28], the reference distance(d_0) used is 1m, which is also used in our case. This is because, the signal strength is expected to be high and unaffected by significant path loss or interference. There are two ways to find R_{d_0}.

Algorithm 1 To estimate R_{d_0} from measurements [29]

Input: RSSI measurements $\mathbf{X}^{M \times \eta_s}$

Output: $\hat{R}_{d_0}$

Initialize: M, η_s.
for $i = 1$ to η_s **do**
  for $j = 1$ to M **do**
    $\bar{R}_{d_{0_i}} = \frac{1}{M} \sum \mathbf{X}_{j,i}$
  end for
  $\hat{R}_{d_0} = \frac{1}{\eta_s} \sum \bar{R}_{d_{0_i}}$
end for

The first one is a simple method to estimate R_{d_0} directly from the measurements as seen from the algorithm 1. As mentioned, for d_0 =1m, RSSI measurements are collected using one Crownstone. The asset is kept at 1m distance from the Crownstone. Measurements are collected three times with the same setting to ensure repeatability and to account for variations and average out the noise. This is stored in a matrix $\mathbf{X}$. A simple averaging algorithm is implemented to get the final value of R_{d_0} as **-64.5dBm**.

Since, we are using the log-distance model to map RSSI values to distance and to make sure R_{d_0} does not vary too much for different values of d_0, second method is based on the equation 4.2. The equation 4.2 can be rearranged to find R_{d_0} as:

$$\hat{R}_{d_0} = R_d + 10n \log\left(d/d_0\right) - \mathbf{w}(\sigma) \tag{4.5}$$

For simplicity, it is assumed that the noise is Gaussian distributed $\mathcal{N}(\mu, \sigma^2)$ and path loss exponent n =2 which is the case in free space. The pseudocode 2 is given

below:

Algorithm 2 To estimate R_{d_0} using path-loss model [29]

Input: RSSI-distance dataset $\mathcal{D} = \{(d_1, R_{d_1}), (d_2, R_{d_2}), ..., (d_N, R_{d_N})\}$.

Output: $\hat{R}_{d_0}$

1: Known inputs: n, $\mathbf{w}$, d_0, N

2: **for** $i = 1$ to N **do**

3: $\quad \hat{R}_{d_{0_i}} = R_{d_i} + 10nlog\frac{d_i}{d_0} - \mathbf{w}(\sigma)$

4: $\quad \hat{R}_{d_0} = \frac{1}{N}\sum \hat{R}_{d_{0_i}}$

5: **end for**

In this method, the data is collected as per section 4.1.1 and stored as dataset $\mathcal{D}$ which contains paris of distance and RSSI values. The measurements were taken at $N = 5$ different distances. Using the path-loss model, the R_{d_0} is calculated for each distance. By performing this experiment, the $\hat{R}_{d_0}$ was found to be **-69.1dBm**. Moreover, the R_{d_0} parameter can also be adjusted and calibrated by configuring the transmission power and sensor gain in the Crownstone application.

4.2.2.2 Path loss exponent (n)

Path loss exponent is a parameter used to characterize the attenuation of radio signals as they propagate through space. It is a measure of the rate at which the signal strength decreases with distance from the transmitter. n can vary depending on the environment in which the signal is being transmitted [26]. For example, in free space, n is 2, which means that the signal strength decreases as the inverse square of the distance from the transmitter [27]. In other environments, such as urban areas or indoor spaces, n can be higher due to the presence of obstacles that cause additional attenuation of the signal. Accurate estimation of the path loss exponent is important in the design of wireless communication systems, as it affects the range and coverage area of the system.

Algorithm 3 To estimate n using MMSE estimator [29]

Input: RSSI-distance dataset $\mathcal{D} = \{(d_1, R_{m_1}), (d_2, R_{m_2}), ..., (d_N, R_{m_N})\}$.

Output: Path loss exponent $\hat{n}$

1: Known constants: d_0, R_{d_0}, N

2: $\hat{R}_d = \hat{R}_{d_0} - 10nlog\frac{d}{d_0}$

3: **Calculate** $MSE = \frac{1}{N}\sum_{i=1}^{N}(\hat{R}_d - R_{m_i})^2$

4: **Calculate** $\hat{n}_{MMSE} = \frac{\sum_{i=1}^{N} log\frac{d_i}{d_0}(R_{m_i} - \hat{R}_{d_0})}{10\sum_{i=1}^{N}(log(\frac{d_i}{d_0}))^2}$

The psuedocode 3, gives the algorithm to estimate n from the model. Again, it is assumed that, R_{d_0} is known from 4.2.2.1 and noise is Gaussian. The path loss exponent was estimated from equation 4.2 using Minimum mean square error(MMSE) estimator. Assuming the log-distance model, we need to provide an equation determining the value of n that minimizes the mean square error(MSE) between the predicted loss($\hat{R}_d$) and the measurements(R_m), given by the MSE:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\hat{R}_d - R_{m_i})^2 \tag{4.6}$$

Then to find the value of n that minimizes the MSE, equation 4.6 is differentiated and equated to zero and substituting the value of n to get the MMSE:

$$\hat{n} = \frac{\sum_{i=1}^{N} log\frac{d_i}{d_0}(R_{m_i} - \hat{R}_{d_0})}{10\sum_{i=1}^{N}(log(\frac{d_i}{d_0}))^2} \tag{4.7}$$

This minimum MSE represents the variance of error in path loss which can also be accounted for by adding the shadowing effect σ, the squareroot of the variance found. The value of n found from the MMSE estimator was **2.050**.

4.2.2.3 Noise parameter (σ)

The noise vector **w** is a part of the log-distance model. Since, the model is logarithmically normally distributed, **w** is also empirically assumed to be distributed normally. However, the actual characteristics may vary. Therefore, estimating the noise parameter σ is important to allow for a more real and accurate representation of the RSSI measurements and further improve the distance estimate.

Algorithm 4 To estimate σ by curve-fitting [30]

Input: RSSI-distance dataset $\mathcal{D} = \{(d_1, R_{d_1}), (d_2, R_{d_2}), ..., (d_N, R_{d_N})\}$

Output: σ

1: **Initialize** N

2: **Define** the model: $\hat{R}_d = \hat{R}_{d_0} - 10nlog\frac{d}{d_0} + \mathbf{w}(\sigma)$

3: **Set** initial values for: R_{d_0}, n, σ

4: **Estimate** the parameters by curve-fitting.

5: **Extract** $\hat{R}_{d_0}$, $\hat{n}$, $\hat{\sigma}$

6: Known RSSI value: R_d for $d = 0.8m$

7: **Calculate** $\hat{R}_{d_{0.8}}$ for $d = 0.8m$ using model.

8: **Calculate** error: $\hat{e} = \|\hat{R}_{d_{0.8}} - R_{d_{0.8}}\|_2$

As per the algorithm 4, the noise parameter can be estimated by fitting the unknown variables to the model. In order to do so, the initial values of n, R_{d_0} need to be provided,

which can be taken from the previous estimations. This is implemented using a "fit" function. Curve-fitting is another way in which all the parameters of the model can be estimated. This is further validated by calculating the RSSI value for 0.8m distance. Curve-fitting not only can be used to estimate the parameters, but also to predict the future values.

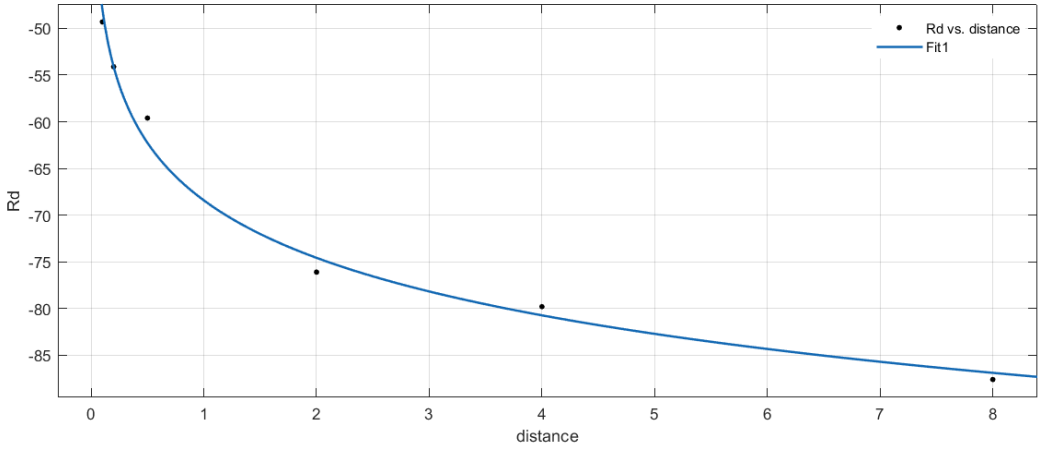


Figure 4.6: Curve-fitted relationship between R_d and d

The fitted curve using the log-distance model is shown in figure 4.6. The trend is a mathematical representation of how the calculated RSSI values decrease as distance to the asset increases. The plot shows that the data aligns well with the theoretical model predictions with low residuals. By implementing algorithm 4, the value of R_{d_0} was found to be **-64.23dBm** similar to the one found using algorithm 1 from the measurements. The value of n is very low **1.4** and σ = **3.33**. Therefore, the noise variance of the test environment is σ^2 = **10.3dB**. The goodness of the fit is determined by R-square value and Root mean square error(RMSE). The R-square value found is **0.9155**, which means that approximately 91% of the variations in the RSSI measurements are accounted by the fitted curve. A RMSE value of **4.114** indicates that on average, residuals between the observed and predicted RSSI is relatively small.

Model Parameters	**Symbol**	**Estimaetd values**
RSSI at reference distacne d_0	R_{d_0}	-69.1dBm
Path loss exponent	n	2.050
Noise variance	σ^2	10.3dB

Table 4.1: Values of the log-distances path loss model parameters estimated using algorithms 1-4.

The parameters estimated using the algorithms 1,2,3, 4 are summarized in the table 4.1. These parameters are estimated for the office environment and are used in the model further for localization using model-based algorithms.

4.3 Localization

This section deals with the third part of the model-based approach, localization, which is actual implementation of the algorithms. It explains about the implemented algorithms for position estimate, along with results and conclusion.

The third phase in the model-based methods is implementing the localization algorithm to estimate the location of the asset. Now that the model parameters are known, localization can be performed by measuring the RSSI of the asset being localized and using the path loss-log distance model to estimate the distance between the asset and the transmitters. The estimated distances can then be used for localization algorithms to determine the asset's position within the indoor environment.

4.3.1 Centralized algorithm

RSSI-based localization algorithms are very well studied as mentioned in the literature review. Lateration-based are the most commonly used algorithms for localization. Out of which, trilateration is the simplest one, which implements the distance between the sensors and the beacons in a two-dimensional plane as shown in the figure 4.7 below:

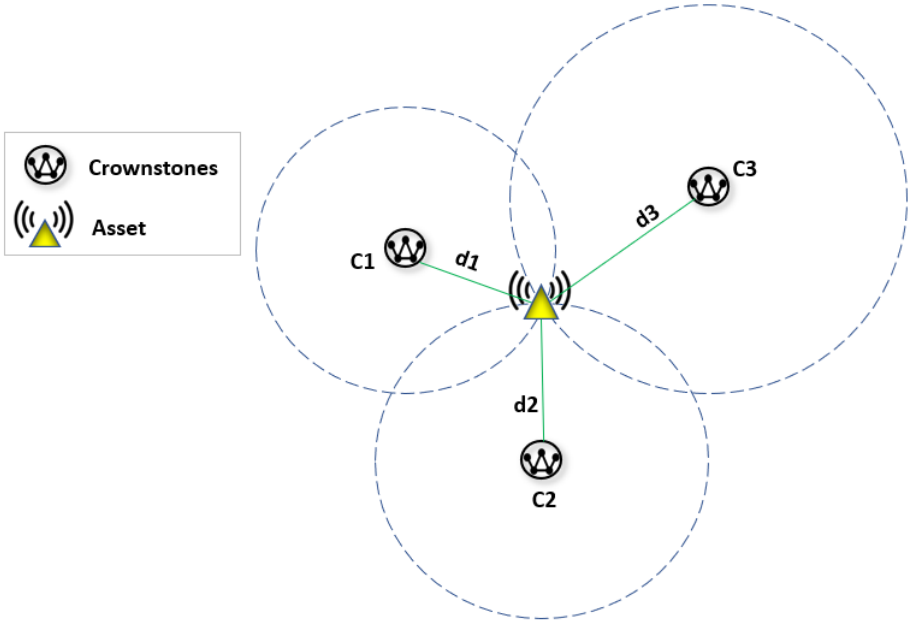


Figure 4.7: A sample figure showing Trilateration based localization.

Trilateration method makes used of the distances(or ranges) to determine the position of the asset with the given propapagation model. The localization method is called as trilateration which makes use of 3 sensors nodes to position the asset [31]. In geometric approach using trilateration, radius of the circles equal their distance from the asset and after drawing the circles, the intersection point of the 3 circles are used to estimate the position of the asset. However, in this we will use the log-distance

model to find out the distances [32].

The position co-ordinates of the Crownstones are denoted as (x_n, y_n) for $n = 1, 2, ...N$ Crownstones. The asset co-ordinates are (x_a, y_a) that need to be estimated. The distance between the asset and the Crownstones can be determined by Euclidean distance using equation 4.5:

$$\begin{aligned} d_1^2 &= (x - x_1)^2 + (y - y_1)^2 \\ d_2^2 &= (x - x_2)^2 + (y - y_2)^2 \\ d_3^2 &= (x - x_3)^2 + (y - y_3)^2 \end{aligned} \tag{4.8}$$

From the equation 4.8, a linear equation of the intersection of the circles can be obtained and expressed as a matrix:

$$\mathbf{Ap} = \mathbf{b}, where \tag{4.9}$$

$$\mathbf{A} = \begin{bmatrix} 2(x_1 - x_2) & 2(y_1 - y_2) \\ 2(x_2 - x_3) & 2(y_2 - y_3) \end{bmatrix}$$

$$\mathbf{p} = \begin{bmatrix} x_a \\ y_a \end{bmatrix}$$

$$\mathbf{b} = \begin{bmatrix} x_1^2 - x_2^2 + y_1^2 - y_2^2 - \hat{d_1}^2 + \hat{d_2}^2 \\ x_2^2 - x_3^2 + y_2^2 - y_3^2 - \hat{d_2}^2 + \hat{d_3}^2 \end{bmatrix}$$

The equation 4.9 can be written as a minimization problem:

$$\hat{\mathbf{p}} = \min_p \|\mathbf{Ap} - \mathbf{b}\|_2 \tag{4.10}$$

where, $\hat{\mathbf{p}}$ is the position estimate of the asset.

The matrix equation 4.10 represents the system of equations that need to be satisfied for trilateration to work. The objective function represents the sum of squared differences between the estimated distances and the actual distances, which needs to be minimized. The solution to this minimization problem gives us the estimated 2D location of the asset based on the observed RSSI and the coordinates of the known locations.

The minimization problem is a non-linear least squares problem which can be solved using various numerical methods such as gradient descent, Newton's method. The objective function is the squared norm of a linear function of the variable $\mathbf{p}$. Therefore, the entire objective function is a convex function and variables x and y appear only linearly, so the optimization problem is convex. Cvx uses quadratic programming to find the optimal position of the asset.

Algorithm 5 Trilateration/Multilateration using Path-Loss Model [29], [33]

Input: RSSI meaurements $\mathbf{X}$, $\mathbf{C}^{N\times 2}$, $\mathbf{p}$.
Output: Position estimate $\hat{\mathbf{p}}$

```
Initialize: R_{d_0}, n, d_0, N
for i = 1 to N do
    d̂_i = d_0.10^{(R_{d_0} − R_{d_i})/(10n)}
end for
Define A, b, p
Begin cvx
Minimize ||Ap − b||_2 s.t. constraints
    p ≥ 0
End cvx p̂
Calculate error: ê = ||p̂ − p||_2
```

The algorithm 5 explains the steps to estimate the position of the asset. For this, the dataset is collected as per section 4.1.3. According to the assumption A4, the positions of the Crownstones are known. The true position of the asset was also known. Data from 3 Crownstones were used to simulate the trilateration algorithm to see whether the algorithm converges to the true value or not. The number of iterations required to get to the true estimate was noted and the error was calculated. The same algorithm can be used to implement multi-lateration just by defining $\mathbf{A}$ and $\mathbf{b}$ according to the number of Crownstones used as shown below.

$$A = \begin{bmatrix} 2(x_1 - x_2) & 2(y_1 - y_2) \\ 2(x_1 - x_3) & 2(y_1 - y_3) \\ 2(x_1 - x_4) & 2(y_1 - y_4) \\ \vdots & \vdots \\ 2(x_1 - x_N) & 2(y_1 - y_N) \end{bmatrix}$$

$$b = \begin{bmatrix} x_1^2 - x_2^2 + y_1^2 - y_2^2 - \hat{d_1}^2 + \hat{d_2}^2 \\ x_1^2 - x_3^2 + y_1^2 - y_3^2 - \hat{d_1}^2 + \hat{d_3}^2 \\ x_1^2 - x_4^2 + y_1^2 - y_4^2 - \hat{d_1}^2 + \hat{d_4}^2 \\ \vdots \\ x_1^2 - x_N^2 + y_1^2 - y_N^2 - \hat{d_1}^2 + \hat{d_N}^2 \end{bmatrix}$$

From $\mathbf{A}$ and $\mathbf{b}$, the solution remains the same. $\hat{\mathbf{p}}$ can be found using the equation 4.10. Therefore, multilateration problem is an extension of the trilateration method. Multilateration basically considers N distances that depend upon the number of Crownstones considered for performing localization to determine the asset's position.

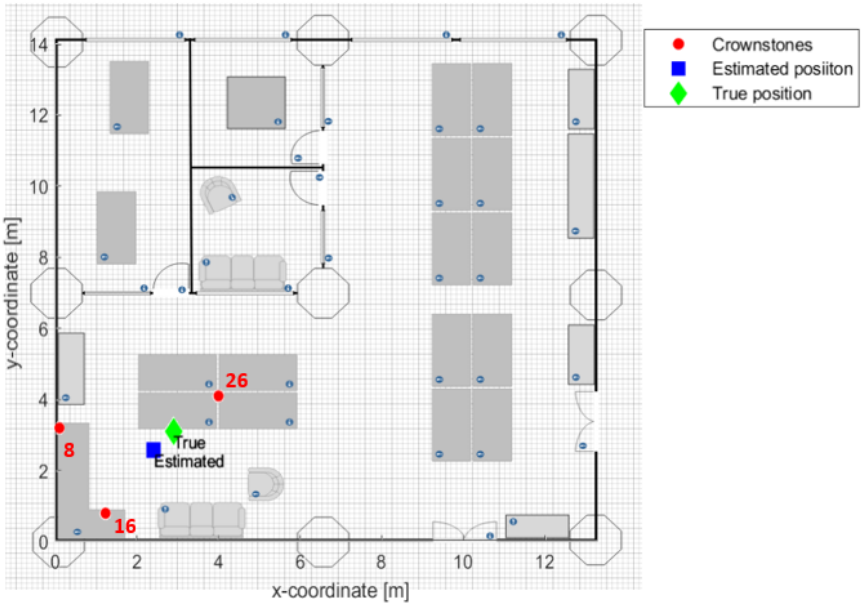


Figure 4.8: Plot of estimated position and true position of the asset bu Trilateration overlayed on the Almende floorplan.

As seen from the plot 4.8, RSSI measurements are collected from 3 Crowsntones and the position is estimated by implementing the trilateration algorithm. The positioning error was found to be 0.71m that is also shown in the figure 4.8. This shows that even though the parameters were modelled and estimated, the path-loss model still lacks and the algorithm cannot predict the output accurately. The algorithm predicts the output in less than a second. Similarly, data from all the Crownstones can be used to estimate the position using multi-lateration.

4.3.2 Centralized vs Distributed

In a centralized approach, the RSSI signals measured by the Crownstones at known positions are sent to a central node or a hub. This node does all the processing and computes the estimated position of the asset using the algorithm stated before. The central node broadcasts the computed position to all the Crownstones. The central node is called as the fusion centre which does all the processing. Also, it only does the computations once it has received all the measurements, similarly for the broadcast. The hub has access to a larger amount of data and computational power to estimate the asset's location with greater accuracy. However, this approach requires more infrastructure and can result in a single point of failure if the central server goes down. A centralized and distributed representation of the configuration can be seen in the figure 4.9 below.

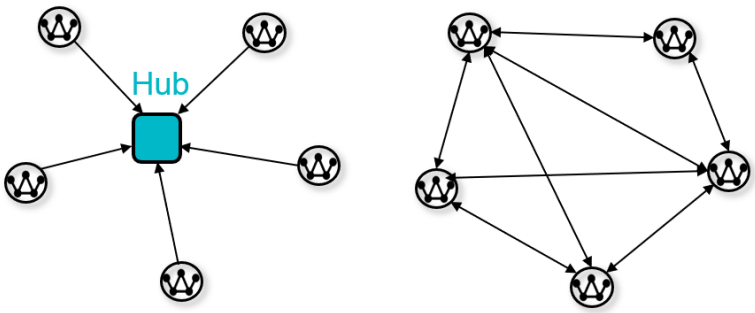


Figure 4.9: Visualization of centralized and distributed configuration

In 4.9, the black arrows represent the links between the Crownstones and they and indicate that they are connected to each other. In a distributed setting, each device or access point collects and processes its RSSI measurements locally to estimate the asset's position. This approach is more resilient to failures and requires less infrastructure, but may result in lower accuracy due to limited data and computational resources available at each local device [18]. All the communication can take place via their 1-hop neighbors.

As mentioned in the objectives, a special focus is given to implementing a distributed localization algorithm, knowing the key differences is essential. In order to implement distributed algorithm, A5 must hold for data transfer to take place amongst the nodes. The links shown in figure 4.9 represent the neighboring links. For implementing a static case, the entire data for a given time period needs to be transmitted to the hub in a centralized setting, whereas in a distributed implementation, Crownstones can start the localization process once local data is available to them. The next section describes a distributed implementation to estimate the asset's position.

4.3.3 Distributed algorithm

Distributed asset localization refers to the process of determining the location of an asset using a distributed network of sensors. In-network localization involves deploying a network of sensors throughout an area and using RSSI measurements from the beacons to determine the location of assets withing the area. Distributed algorithms are used to process the RSSI measurements and estimate the location based on the signal strength information received from multiple sensors [18]. In this, the similar centralized problem is solved in a distributed manner. The distributed Crownstone network of the office is shown in figure 4.10. The connectivity link is based on the link strength data between the Crownstones.

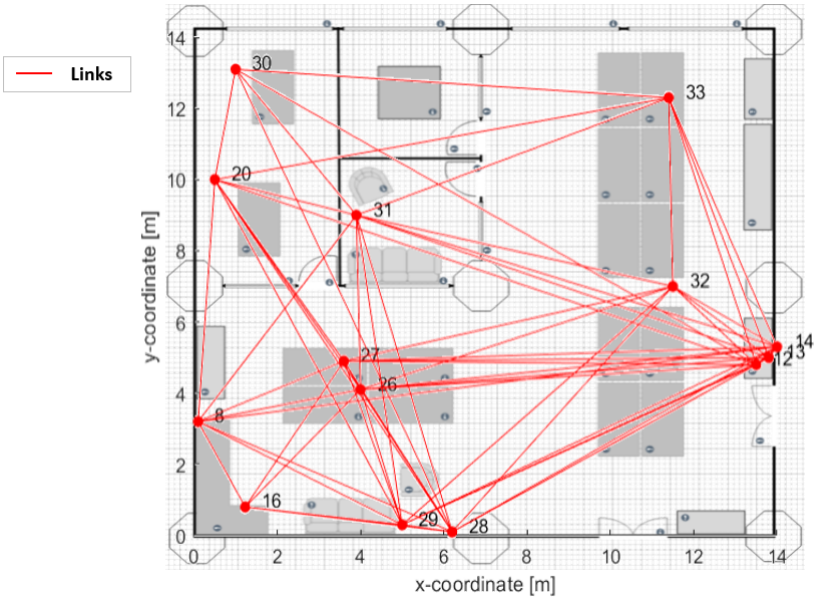


Figure 4.10: Distributed Crownstone network in the office. Links represent that the Crownstones are connected based on the link strengths

Since this is also a model-based approach, we will continue using the log-distance model to calculate the estimated distances through nodes in order to perform joint localization. The position co-ordinates of the Crownstones are denoted as (x_n, y_n) and the asset co-ordinates are (x_a, y_a) that need to be estimated. The distance between the asset and the Crownstones can be determined by Euclidean distance using equation 4.4:

$$\begin{aligned} d_1^2 &= (x_1 - x_a)^2 + (y_1 - y_a)^2 \\ d_2^2 &= (x_2 - x_a)^2 + (y_2 - y_a)^2 \\ &\vdots \\ d_N^2 &= (x_N - x_a)^2 + (y_N - y_a)^2 \end{aligned} \tag{4.11}$$

where $d_n, n = 1, .., N$ is the actual distance between the asset and the n^{th} node.

By expanding the terms in the bracket, the equation 4.11 can be developed as:

$$\begin{aligned} d_1^2 &= x_1^2 + x_a^2 - 2x_a x_a + y_1^2 + y_a^2 - 2y_1 y_a \\ d_2^2 &= x_2^2 + x_a^2 - 2x_2 x_a + y_2^2 + y_a^2 - 2y_2 y_a \\ &\vdots \\ d_N^2 &= x_N^2 + x_a^2 - 2x_N x_a + y_N^2 + y_a^2 - 2y_N y_a \end{aligned} \tag{4.12}$$

By rearranging the terms, equation 4.12 can be represented in a vector-matrix form as:

$$\begin{bmatrix} d_1^2 - (x_1^2 + y_1^2) \\ \vdots \\ d_N^2 - (x_N^2 + y_N^2) \end{bmatrix} = (\mathbf{p}^T\mathbf{p})\mathbf{1} - 2\begin{bmatrix} x_1 & y_1 \\ \vdots & \vdots \\ x_N & y_N \end{bmatrix}\mathbf{p}$$

where $\mathbf{1}$ is a $N \times 1$ vector of all ones and $\mathbf{p} = \begin{bmatrix} x_a & y_a \end{bmatrix}^T$. The distances in the equation are the estimated distances calculated through equation 4.4. If the left hand side vector can be expressed as $\mathbf{b}$, then the vector-valued cost function can be written as:

$$\tilde{\mathbf{f}}(\mathbf{p}) = (\mathbf{p}^T\mathbf{p})\mathbf{1} - 2\mathbf{C}\mathbf{p} - \mathbf{b} \tag{4.13}$$

The cost function in equation 4.13 consists of the estimated distances which are calculated using equation 4.4 and not the true distances based on the positions of the Crownstone and assets. Therefore, the distances depend on the RSSI values measured by the sensors which cause a discrepancy in the final estimate due to noise. The cost function above can be made more robust and applicable to real-world scenarios by adding a weighted vector to it [34]. This is done because some of the Crownstones in the network can have bias in their measurements and their positions are only assumed to be known exactly which might not be the case. Therefore, weights will be used to help reduce the effects of bias-ness as well as uncertainty in the Crownstone placements. The cost function now is defined as:

$$\mathbf{f}(\mathbf{p}) = \Upsilon\tilde{\mathbf{f}}(\mathbf{p}) \tag{4.14}$$

where Υ is the weighted diagonal matrix. The appropriate choice for choosing the weighted matrix would be the inverse of the distances [35]. This is because the lower the RSSI value, larger is the distance which means that the asset is further away from that Crownstone. So, inverse of it would give a smaller weight to that value as:

$$\Upsilon = diag(1/\hat{d_1}, ..., 1/\hat{d_N}) \tag{4.15}$$

The cost function accounts for the difference between the estimated and true asset position. As given in, [34], a weighted non-linear least squared estimate of the target position can be found as a solution to the optimization problem formulated as:

$$\hat{\mathbf{p}} = \min_{\mathbf{x}} \frac{1}{2} \|\mathbf{f}(\mathbf{p})\|^2 \tag{4.16}$$

where $\|.\|$ denotes Euclidean vector norm.

The optimization problem given above can be solved in a distributed way by using using local computations via consensus algorithm. The weighted version of the distributed Gauss-Newton algorithm can be implemented to reach similar solution as

the centralized approach. This is a sub-optimal solution wherein the Crownstones only communicate with their 1-hop neighbours.

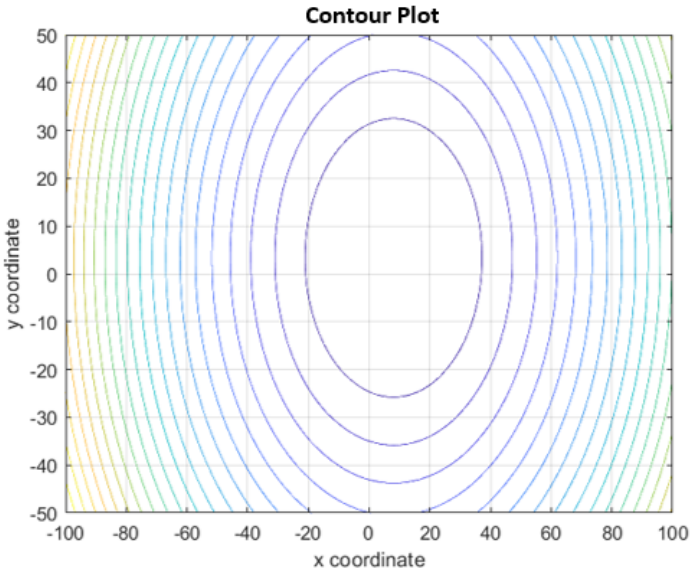


Figure 4.11: Contour Plot of the objective function 4.16

The contour plot 4.11 indicates that the objective function is a smooth function and differentiable. It can also provide information about the gradient which allows the use of gradient-based method to be implemented for optimization such as Gradient descent or Newtons method.

4.3.3.1 Gauss-Newton Method

This subsection explains the basic formulation and implementation of the Gauss-Newton method to compute the step size which is used in the distributed algorithm later. The general Gauss-Newton method for a convex non-linear least squares problem can be considered as a problem similar to our minimization function, in which the function is twice differentiable [36], with its gradient and Hessian with respect to x as:

$$\nabla f(x) = f(x)\nabla f(x) \tag{4.17}$$

$$\nabla^2 f(x) = \nabla f(x) f(x)^T + f(x)\nabla^2 f(x) \tag{4.18}$$

The step-size 'h' would be in descent(negative) direction calculated as:

$$h = \nabla f(x)/\nabla^2 f(x) \tag{4.19}$$

The search direction tends towards the true position of the asset and comprises of the values coming from different Crownstones as consensus. This algorithm is used with embedded consensus algorithm to solve localization problem in a distributed way

as given in algorithm 6.

The Gauss-Newton algorithm is implemented as suggested in [34] because it can be decomposed into local steps to suit the distributed nature. Each node therefore can update its local estimate. Other algorithms such as Distributed Gradient Descent, ADMM and consensus algorithms with Kalman filtering can also be used but as mentioned in [35, 34], Gauss-Newton algorithm in consensus step ensures local estimates converge to a one solution with faster convergence speed.

Algorithm 6 Averaging consensus based Distributed Gauss-Newton Algorithm [18]

Input: RSSI meaurements $\mathbf{X}$
Output: Position estimate $\hat{\mathbf{p}}$
1: Initialize: R_{d_0}, n, d_0, N
2: **for** $i = 1$ to N **do**
3: $\quad \hat{d}_i = d_0.10^{\frac{(R_{d_0} - R_{d_i})}{10n}}$
4: **end for**
5: Initialize $\hat{\mathbf{p}}^{(0)} \leftarrow$ same initial value $\forall\ n \in N$
6: **for** $k = 1$ to K **do**
7: $\quad$ **for** $n = 1$ to N **do**
8: $\quad\quad \mathbf{J}_n^{(k)} \leftarrow \frac{2}{\hat{d}_n}[\ \hat{x}_a^{(k)} - x_n; \hat{y}_a^{(k)} - y_n\ ]$
9: $\quad\quad \mathbf{f}_n(\hat{\mathbf{p}}^{(k)}) \leftarrow \frac{\hat{\mathbf{p}}^{(k)T}\hat{\mathbf{p}}^{(k)} - 2c_n^T\hat{p}^{(k)} + x_n^2 + y_n^2 - \hat{d}_n^2}{\hat{d}_n}$
10: $\quad\quad \mathbf{\Delta}_n^{(k)} \leftarrow \mathbf{J}_n^{(k)T}\mathbf{J}_n^{(k)}$
11: $\quad\quad \gamma_n^{(k)} \leftarrow \mathbf{J}_n^{(k)T}\mathbf{f}_n(\hat{\mathbf{p}}^{(k)})$
12: $\quad$ **end for**
13: $\quad$ **consensus**
14: $\quad \mathbf{\Delta}_*^{(k)} \leftarrow \frac{1}{N}\sum_{n=1}^{N}\mathbf{\Delta}_n^{(k)}$
15: $\quad \gamma_*^{(k)} \leftarrow \frac{1}{N}\sum_{n=1}^{N}\gamma_n^{(k)}$
16: $\quad$ **end consensus**
17: $\quad h^{(k)} \leftarrow \mathbf{\Delta}_*^{(k)^{-1}}\gamma_*^{(k)}$
18: $\quad \hat{\mathbf{p}}^{k+1} \leftarrow \hat{\mathbf{p}}^{(k)} - h^{(k)}$
19: $\quad$ **Calculate** error: $\hat{e} = \|\hat{\mathbf{p}} - \mathbf{p}\|_2$
20: **end for**

The distributed approach does not require a fusion centre for its computations and hence can be computed locally. When a centralized minimization problem is solved in a distributed way, it means that the computation is divided and performed across multiple Crownstones. The algorithm 6 was tested using collected RSSI data from N Crownstones. As seen from steps 7 to 11, each node solves its own sub-problem using local data and communicates its results to the other nodes. Finally, consensus is made to get the final position estimate of the asset. It is a sub-optimal strategy based on local distances and computations are communicated to their 1-hop neighbors and update their own values. The Crownstone network can be seen in fig 4.10, that they are connected well with average degree of 9.

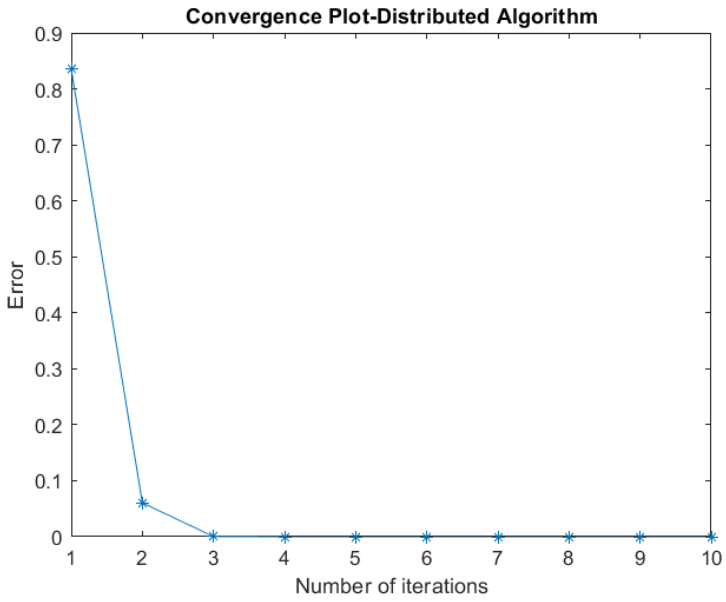


Figure 4.12: Error convergence plot depicting the L2-norm of difference between the true and estimated position as a function of number of iterations.

The distributed algorithm given above is also simulated by randomly selecting Crownstone and their positions, as well as the asset positions to analyze results for more Crownstones. For this, the RSSI measurements are generated randomly and not taken from $\mathbf{X}$ and the distance estimate is calculated based on equation 4.4 again. The algorithm is followed to implement localization in a distributed way for K iterations and error is calculated. From the plot 4.12, it can be seen that the error between $\hat{\mathbf{p}}$ and $\mathbf{p}$ converges in just 4 to 5 iterations, whereas the centralized algorithm takes more than 7 iterations to converge when there is no noise present.

4.4 Optimality Test

In this section, the centralized and distributed algorithms of the model-based method are compared with different performance metrics. The goal of this thesis is to implement a localization algorithm to predict the location of an asset with high accuracy and a low latency. Therefore, an algorithm would be optimal if it gives higher percentage of correct predictions or a lower error and faster computations with few iterations. To compare the optimality of the centralized model-based with the distributed model-based method, the metrics used are accuracy, scalability, latency and single point of failure. Out of these, the 2 most important ones are accuracy and latency. The remaining 2 metrics are explained at the end of this section.

4.4.1 Accuracy

Accuracy of the algorithms is compared under same conditions. It is done by calculating the error between the estimated position and the true position of the asset as:

$$\hat{e} = \|\hat{\mathbf{p}} - \mathbf{p}\|_2 \tag{4.20}$$

Even after learning the model parameters, noise plays an important role in estimating the asset position using these algorithms. So, accuracy of both the algorithms is compared in terms of the error for different noise variances $\sigma^2 = 0, 1, 10$. The noise variances are chosen as 0, representing no noise and $\sigma^2 = 10$ is the estimated variance in the test environment. The plots can be seen below:

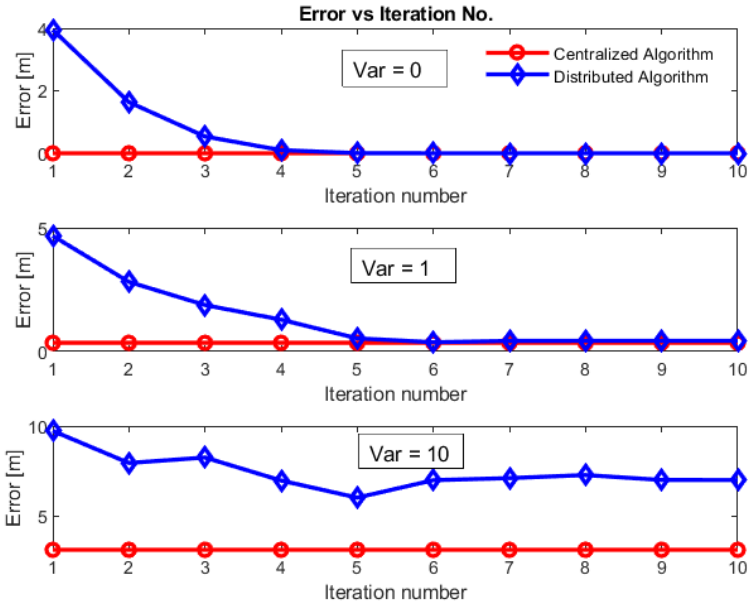


Figure 4.13: Error plot for different σ^2 for both the model-based algorithms. L2-norm of the difference between the true and estimated position is plotted as a function of iteration number.

As seen from the error plot 4.13, for no noise, both the algorithms converge to the true value in fewer iterations. The algorithms still do well for a $\sigma^2 = 1$, but the distributed algorithm takes more iterations to settle to the actual value. However, for increasing noise variance, such as $\sigma^2 = 10$, which is also the noise variance in the office, the distributed algorithm does not converge to the true value and it gives an error of 6m. The centralized one also gives an error of 3m which is still high.

4.4.2 Processing Time

Latency is defined as the time delay between the initial start and reception of an output. For computing latency, its different fragments such as response time, messaging

overhead, network latency etc must be considered. Usually, the latency of a distributed algorithm is higher than a centralized one. But in this section, we are more concerned about how fast the algorithm processes the data and gives the output, which is the processing time as seen from the plot 4.14 below:

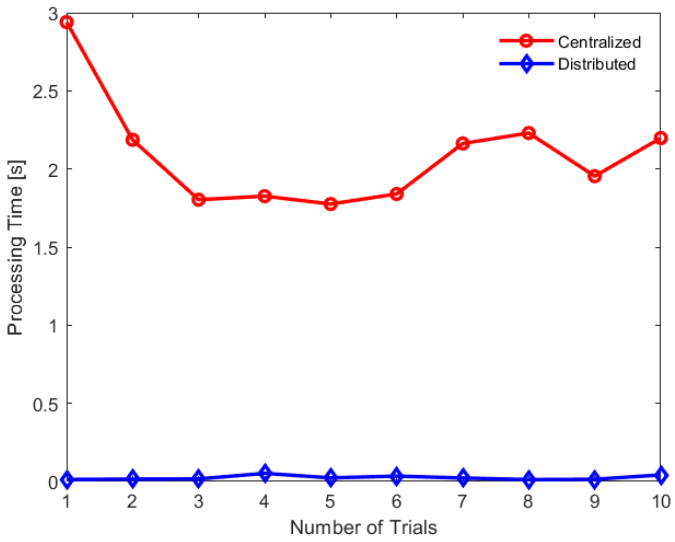


Figure 4.14: Plot for processing time of model-based algorithms calcualted for N trials

The plot shows the processing time taken by the algorithm to get an output for N trials. The centralized algorithm takes longer to compute the output than distributed. This is simulated for 3 Crownstones using the trilateration algorithm. Processing time is corresponds to the time taken by the algorithm to generate a position estimate after it receives the input measurements. By increasing the number of Crownstones, scalability is also checked. Distributed algorithms offfer faster computations by parallelizing the workload across multiple sensors, processing subsets of data. But, a centralized algorithm has to wait for the input data before it starts processing. However, due to inter-communication, the communication overhead can increase for distributed algorithms which is not taken into account.

The **scalability** of the algorithms is checked by increasing the number of Crownstones in the network. Both the algorithms are simulated and the time taken by them to output a position estimate is considered. For a centralized algorithm, the time required for computation scales linearly with the number of Crownstones, whereas the distributed algorithm gives the output faster within 10 seconds. This is logical because, in a centralized method, the hub can perform the algorithm only after it gets data from all the Crownstones, which is not the case in a distributed algorithm. It performs its computations with local data and communication with its 1-hop neighbor.

Centralized algorithms are affected by **single-point of failure** and this is tested by simulating the trilateration algorithm wherein one of the Crownstone is assigned

a RSSI value of r_{min} indicating that it does not make a measurement or has stopped working. The positioning error was found to be 10m even with no noise. However, a distributed algorithm gives a much lower error of 4m and is robust.

Approach / Criteria	Centralized	Distributed
Accuracy	More	Less
Scalability	Less	More
Processing Time	More	Less
Single point of failure	Yes	No

Table 4.2: Performance comparison of centralized and distributed algorithms summarized in terms of 4 different criterias.

The summary of performance criterias is given in the table 4.2. The distributed algorithm has advantage over centralized approach even if its a suboptimal approach. Therefore, the distributed algorithm is now tested by plotting the average error versus the number of iterations for increasing σ^2 values:

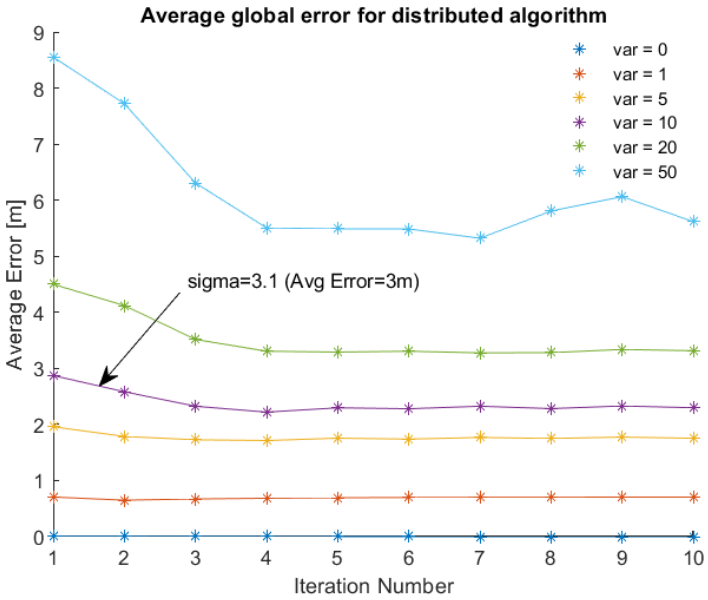


Figure 4.15: Average global positioning error for increasing noise variance σ^2 on RSSI values for distributed algorithm. Average is taken over 1000 different realizations.

The error is calculated by simulating for 1000 realizations, the error converges for lower noise variance, but for more noise the average error is very high above 8m. The estimated office noise variance is 10dB which is marked on the plot and it is still acceptable, but for increasing variance, the distributed algorithm diverges and gives inaccurate results.

4.5 Mapping

Mapping refers to the process of associating one set of data to another set of data based on some criteria or rule [32]. In our case, it refers to mapping position coordinates to location estimates. The output of the model-based approach is a position estimate $\hat{\mathbf{p}}$, but in this thesis, there is a need for room-level accuracy which means, a location label $\hat{l}$ is the requirement. Therefore, a mapping function is defined as:

$$\hat{l} = \mathcal{M}(\hat{\mathbf{p}}) \tag{4.21}$$

where $\mathcal{M}$ is the mapping function.

There are different functions that can be used to map $\hat{\mathbf{p}}$ to $\hat{l}$, but a simple boundary condition is implemented since according to assumptions **A3** and **A4**, the positions of Crownstones are known within the test environment. Also, the divisions of rooms are well known from the floorplan.

Algorithm 7 To map $\hat{\mathbf{p}}$ to $\hat{l}$ [32]

Input: N, $\mathbf{L} = \{l_1, l_2, ..., l_N\}$, $\hat{\mathbf{p}}$, Boundary matrix $\mathbf{B}^{N\times 2}$
Output: $\hat{l}$

```
Set l̂ ← -1 as initial value.
for i = 1 to N do
    if x_a > B_{i,1} & x_a < B_{i,3} & y_a > B_{i,2} & y_a < B_{i,4} then
        l̂ = L_i
    end if
end for
if l̂ = −1 then
    Display: "Asset not found"
else
    Display: "Asset is in room l̂"
end if
```

The algorithm 7 is used to get the location estimate of the predicted position of the asset from the model-based algorithms using a one-to-one mapping function as per the assumption **A5**. It is a simple approach but it is environment specific and depends highly on the assumptions that positions of Crownstones are known. But, this algorithm cannot be generalized to all the other environments. Instead of this, a more complex ML based mapping function can be used. But, this is not within the scope of the thesis, and therefore, the results are compared based on the computed $\hat{l}$ using the algorithm above.

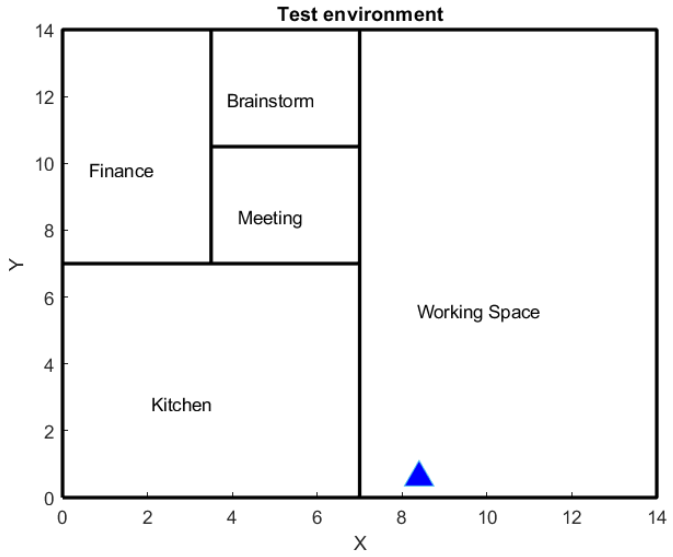


Figure 4.16: Schematic depicting the $\hat{p}$ mapped to a $\hat{l}$ in the test environment

Algorithm 7 is simulated for mapping the asset to one of the rooms as in figure 4.16 above. The position of the asset was (8.4,0.6), mapped to the location *"Working space"*. This algorithm does not work well with the boundary conditions and it gives incorrect output. When a boundary condition occurs, the default output is displayed as *"Asset not found"*.

4.6 Conclusions

- Signal modelling is the most important phase in the localization process before implementing the algorithms.
- RSSI measurements collected from the Crownstones for a time period $\mathbf{T}$ are used for model calibration and testing the algorithms.
- The log-distance path loss model is used which takes into account all the environmental parameters like n and σ.
- To extract the position related parameter, $\hat{\mathbf{d}}$, the values of the model parameters R_{d_0}, n and σ are estimated to be -69.1dBm, 2.050 and 3.33 respectively.
- The simple centralized Trilateration algorithm is implemented giving a positioning error of $\leq$ 3m for $\sigma^2 = 10$dB.
- The averaging consensus based distributed algorithm is faster, scalable and robust against single-point of failure, but does not converge to the centralized solution for $\sigma^2 \geq 10$dB.
- The $\hat{\mathbf{p}}$ are mapped to a $\hat{l}$ using the mapping function $\mathcal{M}$ for both the algorithms.

5 Data-driven Methods

This chapter describes the limitations of the model-based methods implemented in chapter 4. Section 5.2 introduces an alternative approach to get the location estimate. In the subsequent section 5.3, machine learning based data-driven methods are explained in detail with the complete step by step implementations. The different ML classification models and feature engineering is explained in section 5.4 and 5.5 followed by results. Section 5.6 gives the distributed algorithm with results and conclusion.

5.1 Localization Process

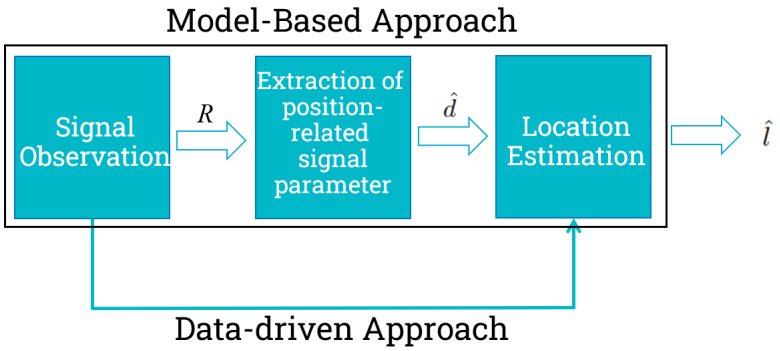


Figure 5.1: Block diagram representing the flow of localization process for both approaches

Localization process using a model-based approach is divided into 3 main steps. Firstly, a signal is observed, a Bluetooth signal in our case. The asset emits beacon frames and the Crownstones record RSSI measurements along aside from its payload. In order to estimate the position of an asset based on the RSSI measurements $\mathbf{X}$, model-based methods use a mathematical model to extract a position-related parameter, distance $\hat{d}$ in our case. Using $\hat{d}$, localization algorithms are implemented to get $\hat{\mathbf{p}}$. Since, we are interested to know the location of an asset, mapping $\hat{\mathbf{p}}$ to a $\hat{l}$ is also needed.

However, as explained in the previous chapter, the middle block contributed to most of the errors in the final output which makes it difficult to implement a model-based method. The idea to predict a location directly from RSSI measurements by skipping all the blocks in-between can be accomplished by a data-driven method. As seen from the block diagram 5.1, data-driven approach is to get $\hat{l}$ from $\mathbf{X}$.

5.1.1 Limitations of model-based methods

1. Selecting a propagation model:
Indoor environments are complex with various obstacles, multipath propagation. The selected path-loss log distanced model cannot accurately capture all the complexities, leading to localization error. Selecting a propagation model that defines the test environment perfectly is difficult.

2. Algorithms depend on $\hat{d}$:
Model-based approach uses a mathematical model to describe the relationship between RSSI and distance. The localization algorithms use distances to estimate the location of the asset. For this, distances estimated from the observed RSSI measurements are not always accurate which adds to the positioning error. Moreover, the implemented distributed algorithm does not converge to the true solution for increasing noise variance.

3. Learning the model parameters:
The selected model has parameters that are dependent on the test environment which need to be estimated. In order to estimate the distances to be used for the localization algorithms, the different model parameters like R_{d_0}, n and σ^2 are learnt. This is time consuming and may require re-calibration.

4. Mapping:
This thesis requires a location estimate of the asset and the model-based approach needs to map $\hat{p}$ to $\hat{l}$, which is an additional step. This can be avoided by implementing a data-driven method, which can directly give location as the output.

Overall, the data-driven method is more flexible, adaptive to changes in the environment or signal propagation characteristics over time. Also, it can learn from new data without the requirement of any prior knowledge. Moreover, the attempt to use model-based methods relies on the accuracy of the underlying assumptions.

5.2 Introduction

A data-driven approach essentially refers to a systematic collection, analysis, interpretation and application of data. For this thesis, a data-driven based algorithm learns to map the RSSI readings to a location label, allowing the algorithm to predict the asset's location based on the RSSI measurements received from the Crownstones. To address some of the aforementioned limitations of the model-based approach, recently a lot of interest has been towards utilizing Machine-learning(ML) based localization algorithms as in [20, 37].

5.2.1 Machine-learning based Localization

Machine learning is a method which focuses on the use of "data" concerned with algorithmically building a statistical model based on that data to solve a practical problem [38]. The learning machine finds a mathematical formula to be applied to a collection of input(training data) to produce the desired output. This formula also generates correct outputs for most other inputs(test data) on the condition that those inputs came from the same or a similar statistical distribution as the one the training data was drawn from.

Classification techniques predict discrete responses. Essentially, these models classify input data into a pre-determined set of categories. Regression techniques predict continuous responses. For our case, this becomes a ML-based multi-classification problem with RSSI values (X) measured from the Crownstones and output location labels (l) for 5 rooms for a static case. Given a dataset $\mathcal{D}$, consisting of samples with features $\mathbf{X}$ and corresponding output labels $\mathbf{L}$, where:

$\mathbf{X}$ represents the RSSI values measured by the Crownstones for time period T.
$\mathbf{L}$ represents the output room location labels for 5 rooms.

The goal is to train a ML-based model $f(\mathbf{X}, \mathbf{T})$ that can predict the room level location of the asset based on the input features. ML model is trained on the dataset to approximate the underlying mapping between the RSSI values and the output locations based on the given features. Once trained, the model can be used to predict the location label for new, unseen RSSI values or test dataset.

5.2.1.1 Types of learning

Learning can be supervised, semi-supervised, unsupervised. In **unsupervised learning**, the dataset is a collection of unlabeled examples and from that it tries to find hidden patterns by creating a model and either transforms it into another vector or into a value that can be used later [38].

In **supervised learning**, dataset is a collection of labeled examples and each element in it is a feature vector [38]. 'Feature vector' is a vector that contains values that can describe the outcome in some way. The goal of supervised learning is to use this dataset to produce a model that take a feature vectors as input and outputs information that allows to deduce the label, so it uses the labeled data along with the classification technique to develop predictive models. It trains a model based on known input and output data so that it can predict the future outputs. The major difference between supervised and unsupervised learning is that the supervised learning requires correctly labeled instances to train the ML model and then uses that trained ML model to label new data.

5.2.1.2 Supervised ML workflow

The process or workflow of the ML can be seen in the block diagram 5.2 shown below. The figure 5.2 describes the phases implemented in this chapter.

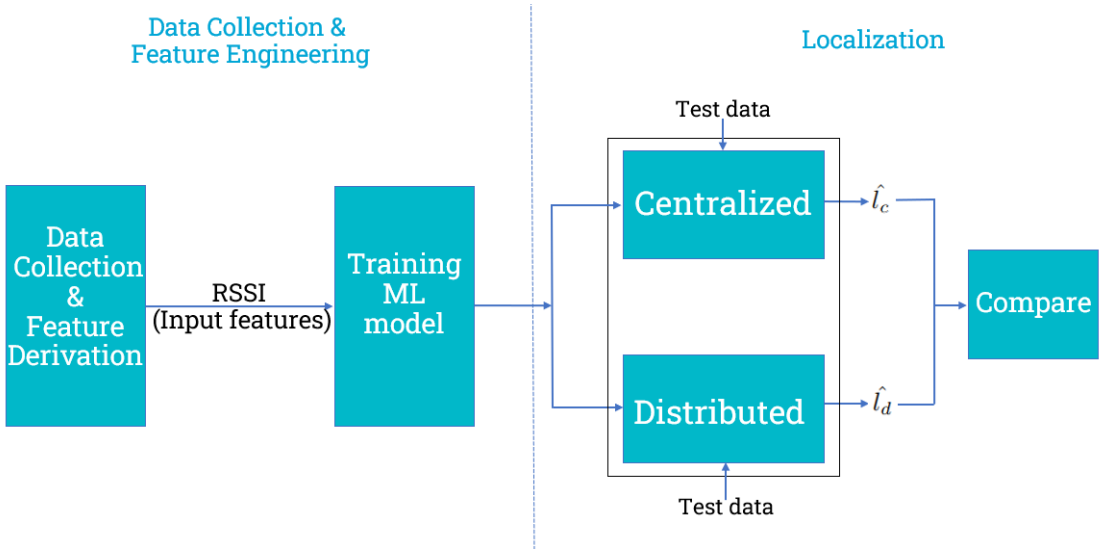


Figure 5.2: Block diagram describing the process of ML used in this chapter

As mentioned before, ML focuses majorly on the use of "data" and that is the reason that the first step in the workflow or process of ML is the "Data Collection" as seen fig 5.2. Most of the data that is measured or collected are raw signals, which is also the case with the RSSI signals. This is raw data that is highly noisy and affected by the environmental factors, obstacles, reflections etc. Due to this, data-preprocessing step is the most obvious thing to do before data analysis. Feature derivation block essentially involves extracting those values that might influence the outcome that we are trying to predict. From 5.2, the process is an iterative one, where in different ML models are trained to find the best model. Again tested with unseen data to cross-validate and iterate again to improve the model.

In the next sections, the entire process of data-driven approach as given in the workflow 5.2 is described. The experiments are done in the test environment 3.2. A supervised ML approach is used because it becomes easier for the model to generalize well for unseen test data and can be implemented in other indoor environments. Implementation of every phase is described as different sections to finally implement the ML based localization algorithms.

5.3 Data Collection

To implement a supervised learning method in this multi-classification problem, the data must be a labelled set. As seen from the figure 3.2, the asset is present in the working space. For data collection, this asset beacon was kept in every room. The asset

was placed at 3 different positions within every room for a period of 10 minutes each. This is done essentially to help capture the variations in the RSSI patterns, discriminative features, environmental factors that could help the ML model to generalize well and make accurate predictions. The placement is random and not dependent on the positions of the Crownstone, since they are assumed to be unknown. The asset is placed approximately 2m distance from one position to cover the entire room. This method also helps to avoid biases that can be made by the ML model. The asset placement is shown in the figure 5.3 below for one of the rooms.

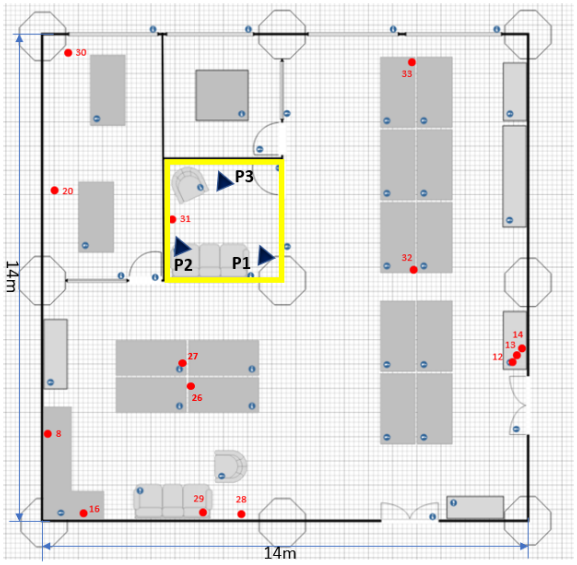


Figure 5.3: Example of asset placement at three different position in one of the rooms. RSSI measurements are collected by keeping the asset in positions p1,p2,p3 for 10mins at each position.

The asset was kept in each room for a total time period of T=30minutes and the start and the end timestamps were noted. Using these timestamps, the data can be retrieved from the Crownstone database. It is important to note the timestamps to collect the correct data and label it with the correct room label later. The data consists of different fields like time, assetID, CrownstoneID, RSSI values. So, the crownIDs are listed in a mixed form as and when any of the Crownstone received a measurement and are arranged according to the timestamps. Even if the asset is kept in one room, the Crownstones other than the ones in the room also make measurements. So, the data is collected globally from all the Crownstones. Data collection took approximately 3 hours in total. The collected data is transformed into a global dataset $\mathcal{D}_g$ consisting of the RSSI measurements combined from all the rooms after data-rearrangement using algorithm 8.

Algorithm 8 To rearrange data as per feature vectors

Input: RSSI measurements $\mathbf{X}$, $\mathbf{C} = \{c_1, c_2, ..., c_N\}$.
Output: Rearranged dataset $\mathcal{D}_{new}$

```
Initialize: timerange T, N.
for i = 1 to T do
    Find unique c_i ∈ C
    r_m ← X_{c_i}
    for j = 1 to N do
        c = C_j
        if c has r_m then
            X_new ← r_{m_j}
        else
            X_new ← r_min
        end if
    end for
    D_{new_i} ← X_new
end for
```

Using the algorithm 8, the data is re-arranged into 14 columns each representing a Crownstone and rows filled with RSSI values at given timestamps. If no measurement was recorded at a particular instant, then the missing gaps are filled with an RSSI value of r_{min}. The values of r_{min} is usually chosen to be -100dBm as it indicates very low signal when the asset is not in the range of a sensor. Finally, rearranged data from each room is combined to form the global dataset $\mathcal{D}_g$.

5.3.1 Datasets

Once, the features are extracted a feature matrix is created with proper labelled output for each input data, the next step is to divide the data set into 3 subsets, namely training data, validation data and test data as shown below 5.4

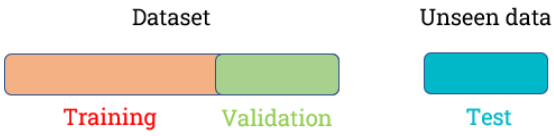


Figure 5.4: Different subsets of a dataset

5.3.2 Training Set

From the data set created as shown above in the figure 5.4, the training set is usually the biggest one and is used to build the ML model. Intuitively, the larger the set is of training examples, the more unlikely it is that the new examples will be dissimilar to the previous examples used for training. The total time period considered was 30minutes in each room which means the sample size is 1800. The training set is 80% of the total data set, so the first 1450 samples were taken approximately. This was done for every

room and then combined together to create one final training set with all the values and output labels.

5.3.3 Validation Set

Validation set is usually the remaining 20% of the remaining data set. Validation set is used to choose the learning algorithm and to find the best value of the hyperparameters. It is also known as the development set.

5.3.4 Test Set

It is important that the test data must also be taken from the similar data set and created in a similar way like the training set. This is because, if slightly changed, it can distort the input and it is highly likely that the prediction would be wrong. The size of the test data is similar to the validation set. The test data only consists of the 14 features at given timestamps and are separated into different rooms. However, the test data can also be combined and shuffled with data from different rooms. The test data is essentially used to assess the model before implementing it. The test data is a completely unseen data.

5.4 Feature Engineering

Feature Engineering essentially talks about extracting relevant information from the collected data [39]. Relevant information refers to those features that might influence the outcome in some way or the other. The problem of transforming raw data into a dataset is called feature engineering. Building a dataset is the first step in feature engineering. Dataset is nothing but a collection of the labelled examples that we collected before. Each element can be used as a feature vector because anything measurable can be used as a feature. Creating an informative feature, it is the one which allows the learning algorithm to build a model that can predict the output labels. These highly informative features are also called as features having high predictive power.

Room Location Labels	Numerical Values
Finance	0
Meeting	1
Brainstorm	2
Kitchen	3
Working Space	4

Table 5.1: Numerical Encoding of room labels used for processing by the supervised ML algorithms

After repeating the same process in every room, the data was exported and the 'location labels' were manually entered to identify what data corresponds to which room. Some of the ML algorithms can only work with numerical feature vectors and therefore the output location labels are converted into numerical values. The five rooms are labelled as shown in the table 5.1 above.

5.4.1 Feature Extraction

RSSI data can be used as a feature in itself, but we can extract various features that capture different aspects of the signal. Here are some common features that can be extracted from RSSI data [39]. They can be divided into different categories such as:

I Statistical Features:

- Mean: Average RSSI value calculated over a specific time period T and provides insights into the proximity of the asset to the Crownstones.
- Variance: Indicates spread or dispersion of the signal strength values. Higher variance indicates more movement or instability in the asset's location.
- Standard deviation(sd): It also measures variability. It is the square root of the variance.
- Minimum and Maximum value: Lowest and the highest RSSI value observed in the given time period.
- Range: The difference between the min and max RSSI value.

II Signal Strength Indicator: Binary feature indicating whether a signal is above or below a threshold value.

III Time-domain features:

- Auto-correlation: It measures similarity between a signal and a delayed version of itself, provides insights into the periodicity or patterns in the RSSI data.
- Cross-correlation: It measures similarity between the RSSI values of different Crownstones.

IV Signal Strength rate: Calculates the rate at which RSSI values are changing over time.

The features listed above can all be used directly or in combination with other features. However, the base features that I used were RSSI measurements based on the timestamps. The RSSI is a relative measure between a transmitter and receiver which means that it already provides an indication of how close the transmitter is to the receiver. Closer the RSSI value to 0, smaller is the distance between the transmitter and receiver. The RSSI values from each Crownstone with the given timestamps can be used as features.

5.4.2 Feature Study

In a data-driven approach data is at the centre of it all. Therefore, understanding and analysing the data is the first step in in building a ML algorithm or model. This section describes analysis of the data features. The data set is divided into training set and two sets-validation and test set. A supervised learning based classification is implemented, therefore, a labelled dataset is used. So, dataset contains RSSI readings of 14 Crownstones and the location label. Along with this, the timestamp is also present on when the RSSI readings were made. Dataset consists of 2900 instances and 14 features.

To get a feel of the data, different attributes of the dataset can be explored such as the shape of the dataset, datatypes of features, distribution of the target variable. The descriptive features can be explored individually and also a mutlivariate exploration is done. Descriptive features are the RSSI signals collected from the 14 Crownstones.

5.4.2.1 Target Feature Exploration

First, the distribution of the target variable is plotted below.

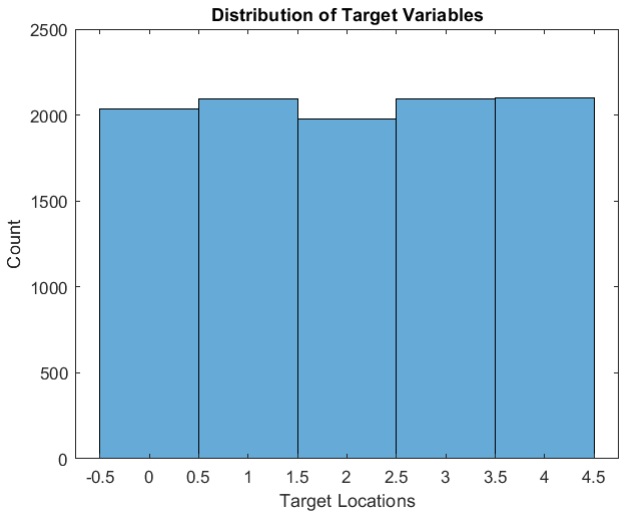


Figure 5.5: Target distribution of the training set. Plot shows the number of instances in the training set that belong to a particular location.

It can be seen from the plot 5.5 that the dataset used is unbalanced, which is important to look at when building a good classifier. Therefore, the unbalanced nature of the set is taken care of at first. In the training data set, there a lot of missing values that are filled with r_{min} and considered as if the Crownstone is out of reach.

5.4.2.2 Distribution of the features

The datatype of the features is 'double'. First, each Crownstone is looked at individually to see if there is anything to be noticed. For each feature, a histogram of the signal strengths is plotted as shown below. This is important to observe how many missing values does a Crownstone have that can be used for assigning weights later.

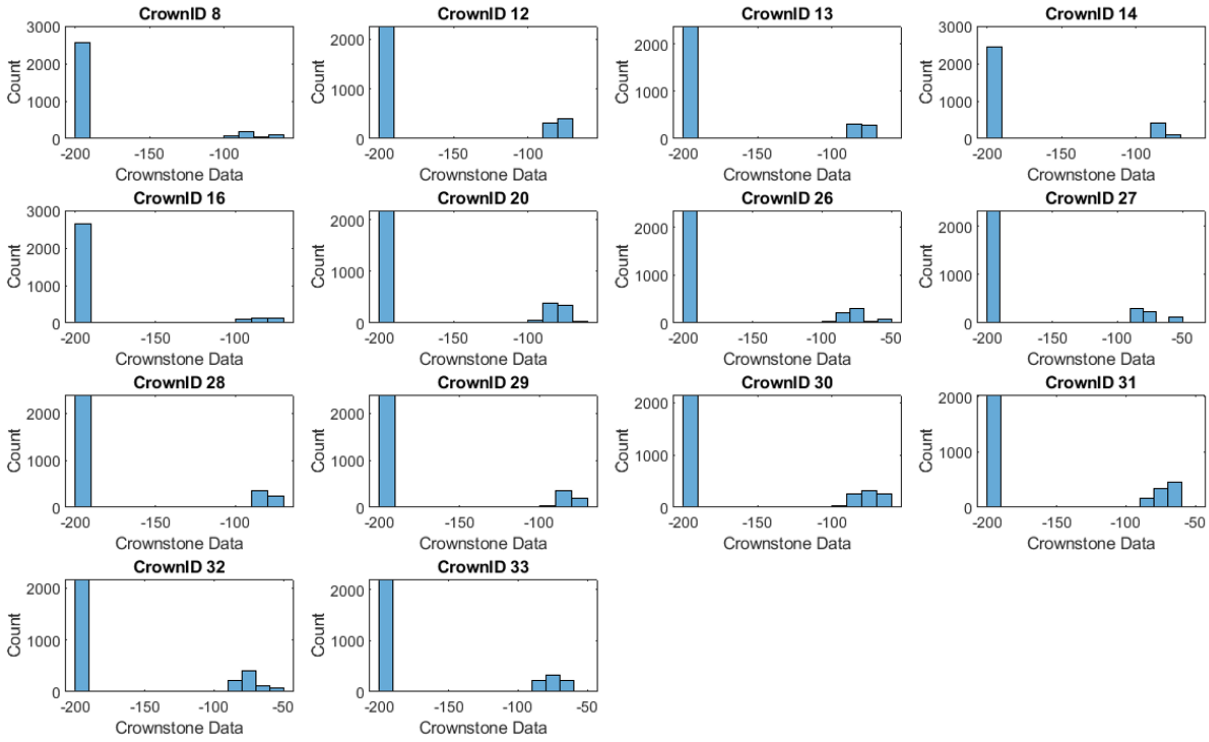


Figure 5.6: Univariate Exploration of the features examining the 14 features individually. Count represents the number of instances the Crownstone measures a unique RSSI value.

The figure 5.6 shows the r_{min} plotted for all the Crownstones. It is important to see the total number of missing values in the given dataset to analyze how they can affect the performance of the classifiers. This is explained later while comparing the performance of the different datasets. From the plot 5.6, it can be seen that the number of missing values or out of reach values are different for each Crownstone and it depends on the location the asset is placed at. All the signals that are detected seem to be distributed around a RSSI signal strength of -70 and the more instances were a signal from a Crownstone is detected, the distribution looks more like a normal distribution as for Crownstone 33.

5.4.2.3 Correlation between features

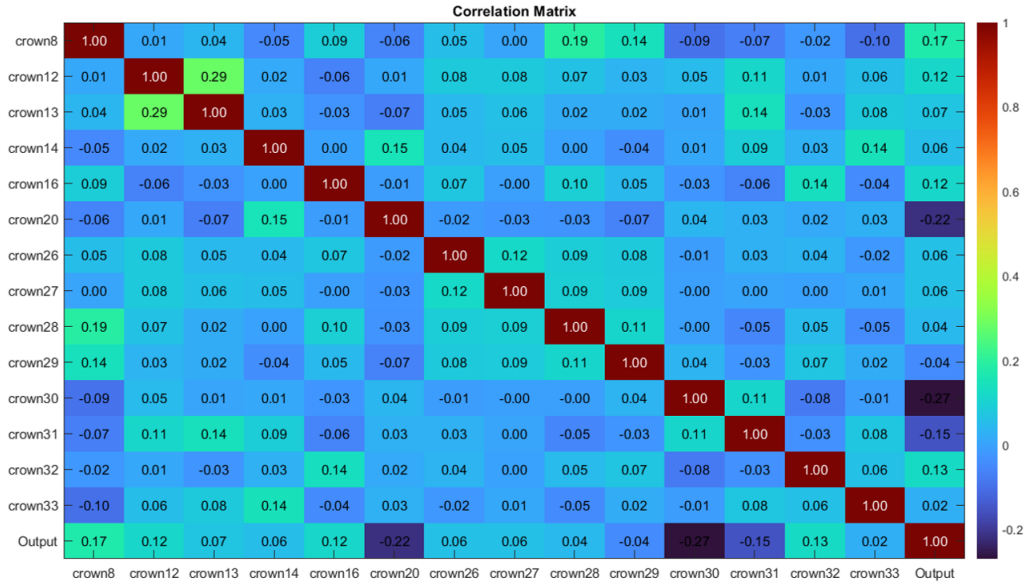


Figure 5.7: Correlation Plot indicating which features are strongly associated with the target variable.

The goal of the classification model is to predict a class based on a set of features, a correlation plot is useful in determining which features are strongly connected to each other and also to the output variable. As per 5.7, there is no or very less correlation between the beacons amongst themselves and also with the target variable because here all the locations were considered. Crownstone 8 is positively correlated and Crownstone 20 and 30 have a negative correlation with the output. Crownstone 12 and 13 have a higher correlation amongst themselves which means that one of them can be dropped.

5.5 ML Classification models

The complete training dataset $\mathcal{D}_{Tr}$ is formed which is a balanced set. This training set will be used to test all the ML algorithms. As per 5.2, building a ML model is an iterative process, that involves training different algorithms and choosing best one amongst them based on their performance [39, 40]. The different ML classification algorithms are compared on different criterias, summary of it is given in the table 5.2

Learning algorithm	Description	Prediction speed	Memory usage	Advantages	Disadvantages
Support Vector Machines(SVM)	Constructs hyperplanes to separate data points of different classes.	Medium for linear, slow for others.	Medium	1.Effective in high-dimensional spaces 2. Handles non-linear decision boundaries	1.Computationally intensive for large datasets 2.Sensitive to parameter tuning
K-Nearest Neighbors(KNN)	Classifies samples based on the majority class amongst its k nearest neighbors.	Medium	Medium	1.Simple, intuitive 2.Handles non-linear boundaries.	1.Sensitive to the choice of k 2.Computationally expensive for large daatsets.
Decision Trees(DT)	Tree-based model that splits data based on features thresholds to make decisions.	Fast	Small	1.Easy to interpret and visualize 2.Handles categorical and numerical features.	1.Prone to overfitting 2.Sensitive to small data variations.
Ensemble Methods	Ensemble model consists of multiple decision trees	Fast to medium	Low to high	1.Robust 2.Handles high-dimensional data 3.Reduces overfitting	1.More complex 2.Computationally expensive
Neural Networks	Deep learning models compose of interconnected nodes(neurons)	Slow	Large	1.Captures complex relationships in data 2.High potential accuracy	1.Requires large amounts of labeled data for training 2.Computationally very expensive

Table 5.2: Summary and overview of classification algorithms in terms of different criterias.

To select one model when implementing this for the first time, it is common practice to test all the algorithms [38]. If a few give comparable training accuracy, then the choice depends on the training time it takes along with the complexity of the model.

5.6 Implementation

In this section, the complete implementation is described in detail, with training, testing, inferences and changes applied to improve the model. This is followed by the results section.

5.6.1 Step I

The training set $\mathcal{D}_{Tr1}$ is used for implementing all the ML algorithms. Out of the 7, the bagged tree algorithm gave the best validation accuracy of 43% which is still poor. When it was evaluated with the test data, the accuracy was close to 40%.

5.6.1.1 Problems:

- The number of missing values are more in the training data, so most of the data points overlap each other. This makes it difficult for the classification algorithm to separate between data.
- Insufficient training data can lead the model to not generalize the data well.
- The selected features may not be relevant enough.

5.6.1.2 Possible Solutions:

- Considering only unique timestamps when a Crownstone records a measurement.
- The training data is sufficient enough and is also balanced for each class.

Reducing the number of missing values by decreasing the sampling frequency to 0.2Hz that is taking timestamps in the interval of 5seconds did not improve the accuracy significantly but only by 2%. This is due to the fact that along with missing values, the useful measurements were also deleted. Plus there were a lot of missing values making the measurements more noisy. This resulted in a new training data set which will be used in the next step.

5.6.2 Step II

As seen in the previous step, validation accuracy did not improve much even after preprocessing and reducing the sampling frequency. Therefore, instead of taking all the timestamps, unique timestamps, those where a Crownstone records a measurement are considered. Due to this, focus is on the measurements.

Using this method, a new training data set $\mathcal{D}_{Tr2}$ was created by considering only unique timestamps and used for training the ML algorithm. The Ensemble algorithm showed a significant increase in the validation accuracy from 43% to 78.6%.

5.6.2.1 Problems:

- Though the training data was predicted well, but poorly predicted for few of the classes. This is called as overfitting.

- For the class labels, Brainstorm room and kitchen, the accuracy was very poor, but for other rooms, it was close to 78%.
- The brainstorm room does not have any Crownstone and in the kitchen room there is no physical boundary to separate two rooms as seen in fig 3.2.

5.6.2.2 Possible Solutions:

- To avoid overfitting, regularization approach can be implemented.
- A simpler model can be used or dimensionality of the examples in the dataset can be reduced.
- Combination of features can be added.

After training the model, the feature importance ranking was calculated from the model using the statistical features listed in previous section. The importance ranking is technique used to identify the most influential feature in a predictive model. Based on this ranking, a new weighted training dataset $\mathcal{D}_{Tr3}$ was created by normalizing the feature values. Higher the value, more important the feature is to predict the output than others. With this weighted data, the model was retrained, however, the accuracy did not improve much and was similar to 78%. The test accuracy of the model did improve to 0.71.

5.6.3 Step III

Even after pre-processing, considering unique timestamps and extracting important features, the model improved but was not able to predict well for atleast 2 classes. To avoid this, additional combinational features are added for specific cases such as the Brainstorm room where there are no Crownstones present. For this, the RSSI values of the nearby Crownstones from fig 3.2: Cr 30,31,33 are combined. The same can be done for test data. After adding the new combinational feature, the model was retrained and accuracy was found to be close to 87% for $\mathcal{D}_{Tr4}$

5.6.3.1 Problems:

- Adding additional feature for a specific room means adding a bias.
- The additional feature is now dependent on the knowledge where and which Crownstone is located near a particular room.
- The ranking for this feature is too high which renders other features redundant.

5.6.3.2 Possible Solutions:

- If combinational features are added for one room, it should be implemented for other rooms as well.

- Knowledge of Crownstone is already inherent in the RSSI value, so adding a combinational feature is unnecessary.

This step resulted in a new training data set called $\mathcal{D}_{Tr4}$ which has 16 features unlike the other training sets.

5.6.4 Results

Once the model is built by the learning algorithm on training datasets, it has to be assessed using the test data. If the model performs well on predicting the labels of the examples from the test set, then it means that the model generalizes well.

5.6.4.1 Performance Metrics

To assess the performance of the model, various metrics are used [38]. For classification, widely used metrics are:

- **Confusion Matrix**: It is a table summarizing how successful the classification model is at predicting the examples belonging to various classes.
- **Accuracy**: Confusion matrix can be used to calculate this metric. It is given by the number of correctly classified examples as:

$$Accuracy = \sum_{i=1}^{noc} \frac{TP_i}{TP_i + TN_i + FP_i + FN_i} \tag{5.1}$$

  where, TP= true positives, TN=true negatives, FP= false positives, FN=false negatives, noc = No. of Classes.

- **Precision**: It is the ratio of correct positive predictions to the overall number of positive predictions:

$$Precision = \frac{1}{noc} \sum_{i=1}^{noc} \frac{TP_i}{TP_i + FP_i} \tag{5.2}$$

- **Kappa coefficient or Kappa score (k)**: In a multi-class classification problem, Kappa score is used more often as a metric. The Kappa score reveals how well the model has predicted data assignments in different classes compared to a random class assignment. The closer it gets to 1, the better the algorithm determines the classes. It determines how much agreement is there between the different classes in the model to predict the correct output given as:

$$\rho_o = \sum_{i=1}^{noc} TP_i$$

$$\rho_e = \frac{1}{noc} \sum_{i=1}^{noc} \frac{(FP_i + TP_i)(FN_i + TP_i)}{TP_i + TN_i + FP_i + FN_i}$$

$$k = \frac{\rho_o - \rho_e}{1 - \rho_e} \tag{5.3}$$

where, ρ_o = observed agreement, ρ_e= expected agreement.

The process of changing the training data and improving the model is described previously and based on that, 4 different training sets are created for which the Ensemble ML model is trained. The first training set $\mathcal{D}_{Tr1}$ is the one with all the timestamps and $\mathcal{D}_{Tr2}$ is with unique timestamps. The third set $\mathcal{D}_{Tr3}$ has the weighted data and $\mathcal{D}_{Tr4}$ has unique, unweighted but combinational feature data. The results on each of the dataset is shown in the figures 5.8, 5.9 and table 5.3 below.

From the first scatter plot, it can be seen that the data points are very close to each other and classifying them by crating a decision boundary is difficult. For the second scatter plot, some of the classes can be distinguished but a lot of the data points coincide with each other making it difficult. The model overfits. In the last 2 scatter plots, the data points can visually be separated and that is why the validation accuracy is high. The scatter plot gives an idea how any ML algorithm would perform just by visualizing the data intuitively.

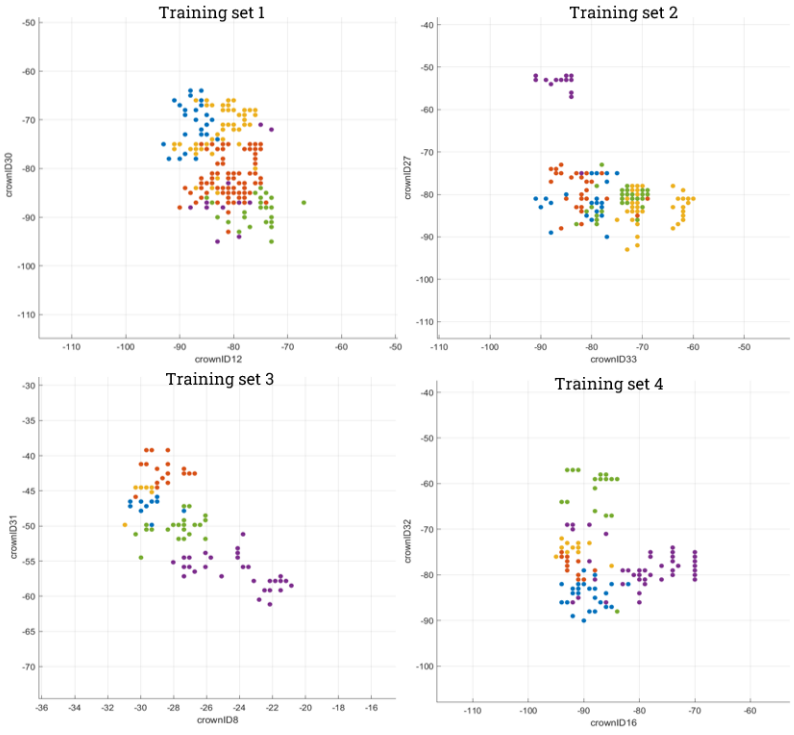


Figure 5.8: Scatter plot for 4 training data for RSSI values measured by 2 different Crownstones.

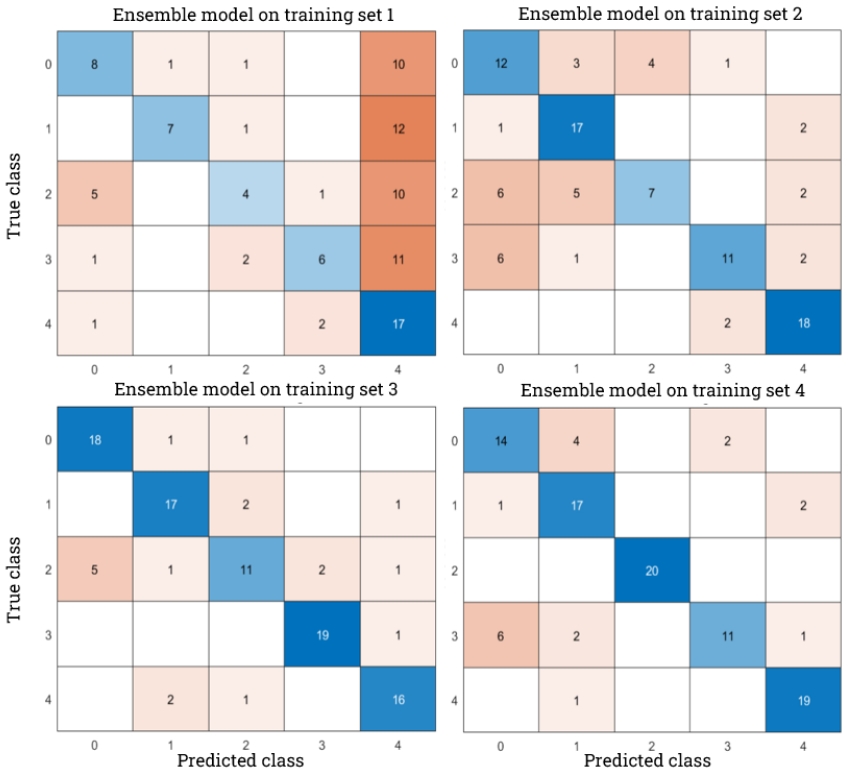


Figure 5.9: Confusion Matrix of Ensemble model trained on 4 different training sets and tested for 4 Test data

The model is assessed using the test data and the confusion matrix for all 4 test sets is seen above. The first confusion matrix consists of many incorrect predictions highlighted in red. In that, most of the predictions move towards class 4 although their true class is something else. The reason could be that there is not much difference between the data points to distinguish between 2 classes. A good change can be seen in the second confusion matrix when only unique timestamps are considered, it leads to lower incorrect predictions. The third and fourth confusion matrices also give good results because of adding weights and additional feature. In the training set 4, I added extra feature for BS room (class2) and from the 4^{th} confusion matrix, for class 2 it predicts all the outputs correctly.

Training Set	**Algorithm**	**Validation Accuracy**	**Test Accuracy**	**Precision**
$\mathcal{D}_{Tr1}$	Ensemble	43.3%	0.39	0.815
$\mathcal{D}_{Tr2}$	Ensemble	78.6%	0.65	0.7213
$\mathcal{D}_{Tr3}$	Ensemble	77.6%	0.71	0.82
$\mathcal{D}_{Tr4}$	Ensemble	86.9%	0.85	0.8783

Table 5.3: Results showing the performance metrics of 4 different datasets

From the table 5.3, the 1^{st} has lowest accuracy and it increases sharply in the successive training sets. Even though the validation accuracy for training set 4 is high, the test accuracy is biased towards one class, therefore does not generalize well for all

the rooms. The weighted set performs well but it overfits for 2 classes. Therefore, after considering different sets, the $\mathcal{D}_{Tr2}$ will be used for building a model and algorithm to perform localization because it has good validation accuracy and test accuracy can be improved further.

5.6.4.2 Performance comparison of models

The training set $\mathcal{D}_{Tr2}$ is the optimal set that gives comparable accuracy and precision as per 5.3. Different ML models like SVM, KNN, Decision Trees, Ensemble method and Neural networks are implemented on $\mathcal{D}_{Tr2}$ and their performances are compared. The models are compared based on the validation accuracy, minimum classification error, prediction speed and the other metrics mentioned in section 5.6.4.1.

Algorithm 9 To evaluate different ML models [41]

Input: Training dataset $\mathcal{D}_{Tr2}^{T\times(N+1)}$, test dataset $\mathcal{D}_{test}^{T\times N}$.
Output: Performance metrics of all models.
1: Initialize: $\mathbf{X}_{Tr}^{T\times N}$, $\mathbf{L}_{Tr} = \{0, 1, 2, 3, 4\}$, $\mathbf{G}^{T\times 1}$, M.
2: **Train** M models on $\mathcal{D}_{Tr2}$
3: **Save** and **Extract** the models.
4: **Predict L** using trained models M on $\mathcal{D}_{test}$.
5: **Calculate** confusion matrix, accuracy, Precision, kappa coeeficient.
6: **Plot** the results.

The different ML models are evaluated using the algorithm 9. First, the models are trained and the results can be seen in table 5.4 below:

ML Models	**Validation Accuracy(VA) [%]**	**Prediction Speed [obs/s]**	**Training time [s]**
SVM	47.6	9700	6.4036
Weighted KNN	50.2	14000	2.2443
Optimized KNN	60	11000	93.087
Tree	67.2	28000	4.1895
Optimized Tree	73.2	88000	77.871
Ensemble (Random Forest/Bagged Trees)	78.5	8300	6.6588
Optimized Ensemble	78.7	750	633.9
Neural Network	62	62000	25.954

Table 5.4: Performance comparison of training metrics of the different ML models

The DT and Ensemble DT models are optimized using hyperparameter tuning to find the parameter configuration that minimizes the classification error and improves the overall accuracy of the classifier. As seen in 5.10, there is a possibility of overfitting because of using a smaller dataset to train the model.

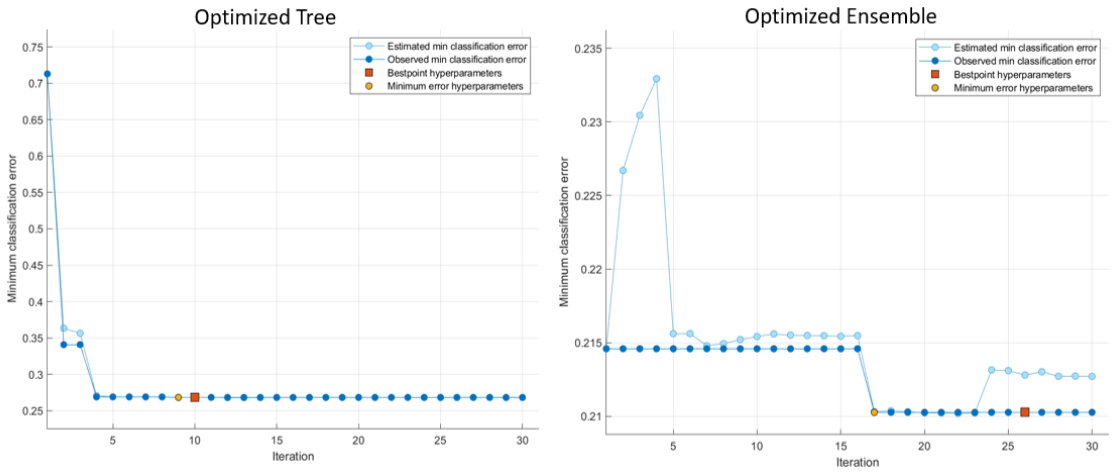


Figure 5.10: Minimum classification error plot for DT and Ensemble DT model where red point indicates the best set of hyperparameters that minimize the classification error.

It can be inferred from the plots that the optimized Ensemble DT performs the best in terms of VA by giving optimal set of parameters which lead to a very low classification error rate of 0.2.

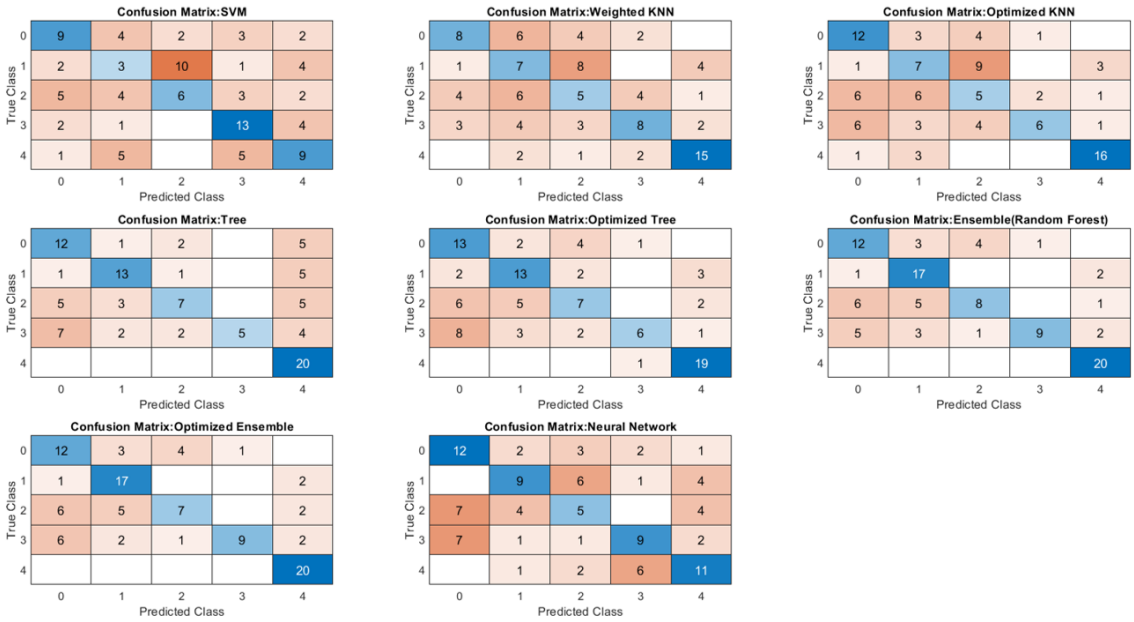


Figure 5.11: Confusion matrices of 8 models for predicting locations on $\mathcal{D}_{test}$.

It can be seen from the confusion matrices in fig 5.11, SVM is the worst model, giving inaccurate predictions and Ensemble DT is the best one which predicts and generalizes well for all the classes. The most commonly implemented KNN algorithm also does not perform well and has a lot of false positive values for class 1,2 and 3. Even a more complex Neural network model predicts poorly. The performance metrics defined in section 5.6.4.1 are calculated for all the 8 models.

The performance of the models can also be visualized from the plot 5.12 which shows test accuracy and precision calculated using the confusion matrix. The Ensemble model has the highest accuracy, but neural network model is more precise. The kappa coefficient is a statistical measure that is used for evaluating the models by providing a measure of agreement between observed and expected. Higher kappa coefficient is desirable, the Ensemble and the DT models have moderate agreement

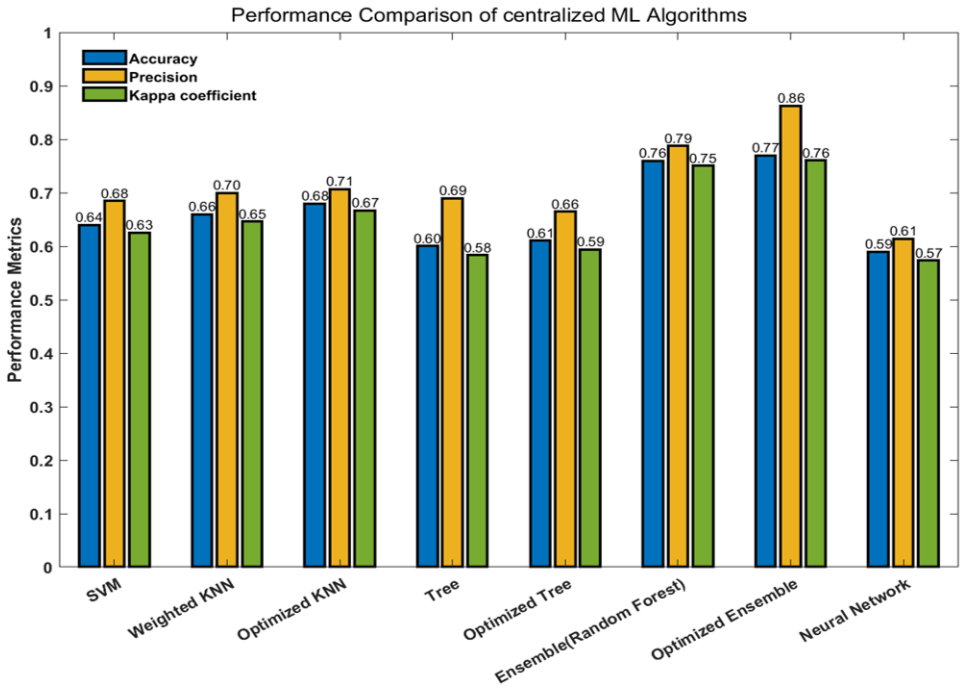


Figure 5.12: Performance comparison of 8 models on $\mathcal{D}_{test}$ for different metrics calculated from the confusion matrices in figure 5.11

5.7 Distributed Machine learning approach

In the previous sections, a complete machine learning based data driven approach was implemented. The training sets are all collected in a global manner and predictions were done centrally at a hub. Till now, the implementation has been centralized, but the same problem can be solved in a distributed way [42]. This section describes the distributed ML approach to predict the location of the asset locally. A complete methodology and execution is explained, followed by results and conclusion.

5.7.1 Methodology

There are two ways in which distributed ML can be done, one is distributing the training part by training separate models for each room or even each Crownstone. This way every sensor would have its own local model. However, a lot of time will be consumed and this scales with the number of sensors in the network. It is also necessary that the models must be synchronized in some way to make the predictions together in the end [43].

The other way is to perform global training, but predictions are done in a distributed way. This involves training a model by considering measurements from every sensor but inference will be made individually by the sensor based on the local data [34, 44].

The centralized ML algorithm performed well but one of its limitations were the presence of missing values. The value r_{min} is filled assuming that a sensor does not record a measurement. Instead of directly assigning r_{min} value to a Crownstone who does not record a measurement, a value combining the measurements from its 1-hop neighbors is assigned. This is done by communicating with the neighbors in a distributed way. The algorithm 10 is explained below:

Algorithm 10 Centralized ML algorithm by handling missing values

Input: Training Data Set $\mathcal{D}_{Tr}$, Test Set $\mathcal{D}_{test}$, neighbor Matrix $\mathbf{N}_e$
Output: $\hat{l}$

Initialize: Time period T, N, $N_{neighbors}$
Initialize Current State $\mathbf{X}_{current}$ matrix to r_{min}.
Train the global Model with $\mathcal{D}_{Tr}$.
for $t = 1$ to T **do**
  Get $\mathbf{X}_t$ from $\mathbf{D}_{test_t}$
  for $s = 1$ to N_{crown} **do**
    if $\mathbf{X}_s \neq r_{min}$ **then**
      $\mathbf{X}_{t,s} \leftarrow \mathbf{X}_s$
      Find neighbors of c_s from $\mathbf{N}_e$
      for $i = 1$ to $N_{neighbors}$ **do**
        $nc = \mathbf{N}_{e_i}$
        if $\mathbf{X}_{nc} = r_{min}$ **then**
          consensus
          $\mathbf{X}_{current_{t,nc}} \leftarrow$mean$[\mathbf{X}_s, \mathbf{X}_{t,nc}]$
          end consensus
        end if
      end for
    end if
  end for
  $\mathcal{D}_{Utest} \leftarrow \mathbf{X}_{current_t}$
end for
Predict locations on $\mathcal{D}_{Utest}$ using global Model.

The trained model is a globally trained ML model. A test set $\mathcal{D}_{test}$ is used as measurement input to the model. In order to implement distributed method, a predefined neighbor matrix is given. The output is a room location using a centralized method described previously, however, the predictions are made on the updated data by handling the missing values in a distributed way. This is done to improve the accuracy of the centralized ML algorithm.

A subset of Crownstones record a measurement at timestamp t and communicate their values to the 1-hop neighbors, who update their values accordingly. Finally, all the sensors and their neighbors come to a consensus and final values for all timestamps are updated. In the end, the global model is used to make the predictions on this updated test set $\mathcal{D}_{Utest}$. This helps to assign values according to the neighbor-sensor communication and not r_{min} directly.

5.7.2 Distributed Inference

The distributed implementation has two parts, first part includes implementing the algorithm 10 to get the updated test set $\mathcal{D}_{Utest}$ by handling the missing values. The second part involves implementing a distributed inference on the $\mathcal{D}_{Utest}$ and the algorithm 11 is used for that.

Algorithm 11 Proposed Distributed ML algorithm

Input: $\mathcal{D}_{Utest}$, neighbor matrix $\mathbf{N}_e$
Output: $\hat{l}$

Initialize: s, M, N
Load the global models $\mathbf{G} = \{g_1, g_2, ..., g_M\}$
Set the threshold to Tr
for $m = 1$ to M **do**
  $C_{model} \leftarrow G_m$
  for $i = 1$ to s **do**
    $X \leftarrow \mathcal{D}_{Utest_i}$
    Select Crownstones based on Tr
    for $j = 1$ to N **do**
      Initialize new table $\mathbf{X}_{new}$ to r_{min}
      Crownstone $c \leftarrow \mathbf{C}_j$
      Find neighbors of c
      Fill columns of c and its neighbors with $\mathbf{X_c}$ and $\mathbf{X}_{N_{e_c}}$ in $\mathbf{X}_{new}$
      Predict $\mathbf{L}_c$ using C_{model}
      Share $\mathbf{L}_c$ with $\mathcal{N}_{e_c}$
    end for
    Begin consensus
    $\hat{l} \leftarrow \text{mode}(\mathbf{L})$
    End consensus
  end for
  Calculate and **plot** confusion matrix, Accuracy, Precision, kappa coefficient for every model
end for

The globally trained 8 models used for central ML method are also used here. The threshold value is selected focus on the Crownstones who make a record a higher RSSI value, indicating that they are closer to the asset. For each set s, measurements $\mathbf{X}$ are extracted from $\mathcal{D}_{Utest}$. Crownstones are selected if they record a RSSI value of greater or equal than the set threshold. Every selected Crownstone makes a prediction based on the local data, consisting of its own measurements and data from its neighbors using a global model. The predictions are then shared again with their neighbors. After this process, each Crownstone has its own predictions and the final prediction or $\hat{l}$ is found after consensus. This algorithm is tested by predicting the asset's location using every model and the results are plotted below.

5.7.2.1 Results

The confusion matrix is plotted by calculating error between the ground truth and the predicted $\hat{l}$. Similar to the centralized implementation, for N trials, the confusion matrix is plotted. From 5.13, with the available partial data, SVM, KNN and Neural network models predict poorly. Though the Tree and Ensemble model have good accuracy, it gives inaccurate results for class 2. The reason is there are no Crownstones present in the Brainstorm room and with only local data available, none of the models predict the class 2 correctly. Amongst the 8, Ensemble gives the best performance.

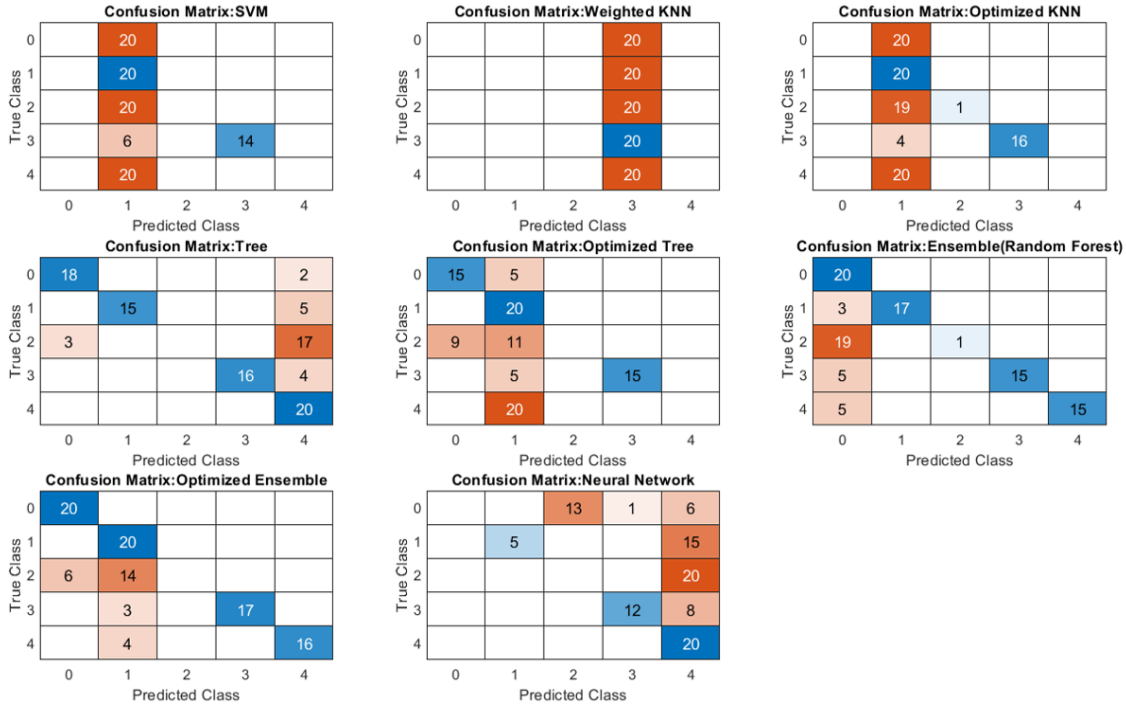


Figure 5.13: Confusion matrices of 8 models using Distributed inference on $\mathcal{D}_{Utest}$

Similarly, other performance metrics such as accuracy, precision and kappa coefficient are calculated using the confusion matrix. The calculations are similar to the

one given in section 5.6.4.1. A plot is made to visualize the performance metrics. The optimized Ensemble model has the highest accuracy of 0.73 and Ensemble model has precision of 0.71. Also the kappa coefficient for Tree and Optimized Ensemble model is high which means that there is more agreement between the outputs as seen in 5.14.

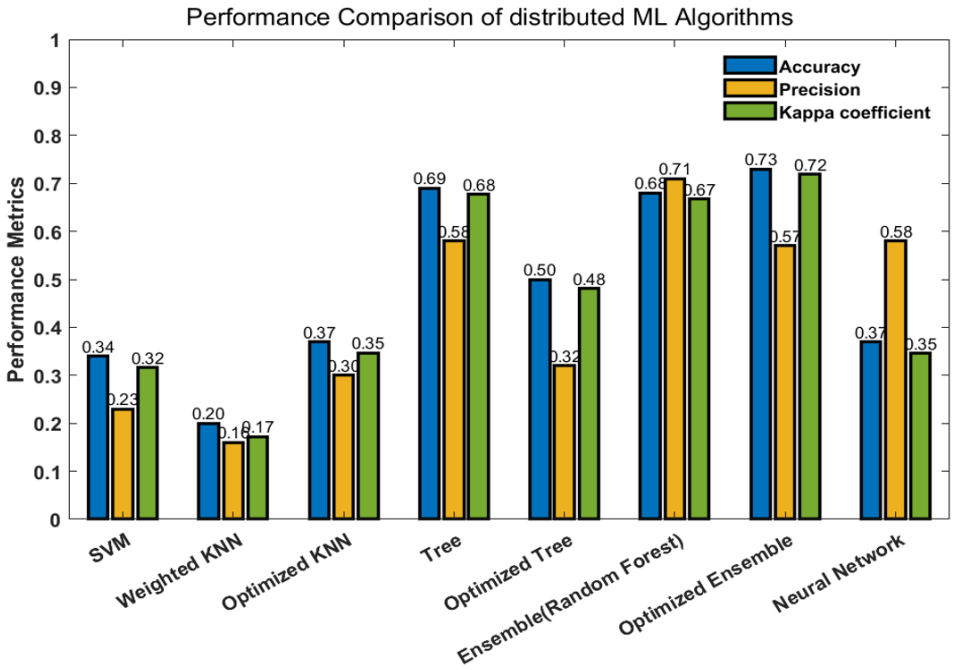


Figure 5.14: Performance comparison based on different metrics calculated from the confusion matrices

Overall, the algorithm implemented in a distributed way does not perform well and gives inaccurate results since the Crownstones make local predictions and do not have access to the entire data, which makes it difficult for the model to generalize the data points well.

5.8 Conclusions

- Machine learning based data-driven method is implemented to predict the $\hat{l}$ directly from the RSSI measurements.
- Data is collected at 5 rooms for a fixed time period $\mathbf{T}$ to develop a training dataset.
- RSSI feature extraction and complete feature study and analysis is done which help to build the ML model for predicting the output.
- Different training sets are created by iteratively modifying the features, out of which $\mathcal{D}_{Tr2}$ is selected for training the ML model.
- Out of the 8 tested models, the Ensemble(Random Forest) model based centralized ML classifier performs the best giving an accuracy of 65%.

- The classification performance of the centralized ML algorithm is improved to 77% by handling the missing values in a distributed way.
- A distributed consensus based ML algorithm is implemented to predict the locations locally at the Crownstones giving a global average accuracy of 75%.

6 Conclusion and Future Work

This chapter concludes the thesis by presenting final results. The first section 6.1 presents a discussion about the 4 algorithms implemented with the two approaches in terms of different performance metrics. Section 6.2 summarizes and highlights the key points of the thesis, followed by future work in section 6.3.

6.1 Discussion

Performance of 4 algorithms are compared in terms of various performance metrics as mentioned in the objectives like scalability, complexity and robustness along with accuracy and prediction time given in table 6.1.

Algorithms	**Accuracy (error %)**	**Prediction time [s]**	**Scalability**	**Computational Complexity**	**Robustness**
MB-C* Multilateration	64.4	1.61	Low	Moderate $\mathcal{O}(KN)$	Moderate
MB-D* Average consensus based distributed algorithm	59.4	0.56	Moderate	High $\mathcal{O}(KNL+KNd)$	Low
DD-C* Centralized Ensemble based ML algorithm	77	1.73	Moderate	High **Training**: $\mathcal{O}(Mlog(M)Nt)$ **Testing**: $\mathcal{O}(Nlog(M))$	Moderate to High
DD-D* Distributed Ensemble based ML algorithm	73	2.3	High	High **Training**: $\mathcal{O}(Mlog(M)Nt)$ **Testing**: $\mathcal{O}(Nlog(M) + MNd)$	Moderate

K= number of iterations, N=number of sensors or features, d=average degree of neighbors, M= Number of data points, t= number of trees. *MB-Model-based, DD-Data driven, C-Centralized, D-Distributed

Table 6.1: Comparison of the 4 implemented algorithms in terms of different performance metrics.

The accuracy metric measures the number of correctly predicted locations of the asset in terms of percentage error and represents the global average accuracy. As seen in

table 6.1, the proposed data-driven algorithms perform much better giving an accuracy higher than 70%. The distributed model based algorithm computes the location output fastest. The ML algorithms are comparitively slower because of the complexity of the global model used. The prediction time corresponds to the time taken by the algorithm to generate a location output after it receives the input measurements data. Scalability of the algorithms assesses its ability to handle an increase in the size of the data, number of sensors or features and also number of locations while maintaining its classification performance. The model-based algorithms are low to moderately scalable because they require precise infrastructure calibration. However, the ML algorithms are scalable in terms of data volume, increase in the number of features or locations and maintain comparable accuracy.

The big $\mathcal{O}$ notation corresponds to the complexity of the algorithms providing an upper bound to the algorithms resource usage as input size grows. From 6.1, the complexity is measured by taking into account the parameters used while implementing the algorithm and the ML algorithms are highly complex. Plus, the computational complexity depends on the number of Crownstones, features and datasize. Robustness metric indicates how well the algorithms perform in presence of noise. The model-based algorithms cannot handle the fluctuations well with increasing noise variance, where as the data-driven algorithms learn from the patterns and handle noise better. The noise is mainly due to the signal interference and obstacles in the dynamic indoor environment.

6.2 Summary

In this thesis, the asset localization algorithms are developed using two different methods with more focus on implementing in-network localization with a motivation to make the indoor localization system more robust to directly get the room-level location of any asset in real-life applications. The summary of the implementations and contributions are given below:

- The most widely studied model based approach-multilateration algorithm is implemented to determine the asset's position. The model parameters are evaluated from measurements and by using estimators to improve algorithm's performance.
- A sub-optimal averaging consensus based distributed algorithm is implemented to perform in-network localization and get the asset's position from the Crownstones only by local computations. Simulations show that distributed algorithm converges to the centralized solution in absence of noise.
- The model-based algorithms are tested using real data collected at the Almende office to predict the asset's position using the Crownstone network to test the effect of noise. As per the results, though the distributed algorithm has several advantages, its performance decreases with increasing noise variance.
- A machine learning based data-driven method is proposed to deal with the real-world uncertainties e.g. changing environment and directly estimate the asset's location from observed data.

- More effort and contribution has been on data collection, preprocessing and feature engineering which are the basic blocks in building a ML model by extracting relevant information from the RSSI measurements.

- Firstly, a centralized Ensemble (Bagged Tree) model is implemented which gives the best performance amongst the 8 tested ML models with classification of 65%. A distributed data handling technique proposed, leads to an improved classification accuracy of 77% of MB-C algorithm.

- Secondly, a distributed ML algorithm is proposed to get the location estimates through local inference by the Crownstones using the globally trained model. Simulations show that this algorithm has better average global accuracy of 73%, but lower local accuracy.

- The model-based algorithms and the proposed data-driven algorithms are tested using real RSSI data. The results show that the proposed algorithms perform better than the base algorithms in terms of accuracy and are comparable in terms of prediction time.

- Results also show that the proposed data-driven algorithms are more scalable and robust against noise than the model-based algorithms, but are computationally complex.

- Lastly, distributed ML based data-driven algorithm has been implemented which gets a room-level location of the asset from RSSI measurements and is tested in the given office environment.

6.3 Future Work

This section addresses the remaining concerns and open possibilities in response to the contributions in this thesis.

There is possibility for future work in this topic not only with the aim of improving the proposed algorithm, but also in terms of extending the scope of the problem itself. Indoor localization using Bluetooth and RSSI signals has been well studied by implementing the basic range-based algorithms, e.g. trilateration [3]. All these algorithms try to model the signal as accurately as possible. So, one of the first challenge was to estimate the model parameters to find the best possible distance estimate and analyze the discrepancies in the model. Though the parameters are well fitted based on the measurements, it is not fixed and varies with the changes in the environment, for e.g. obstacles, presence of other wireless devices.

There are two ways in which the algorithms are compared, first the overall model-based methods are compared with data-driven methods based on the performance metrics mentioned before. The second one is the comparison between centralized and distributed algorithms in terms of accuracy and prediction time. The model-based methods work by assuming that the positions of the Crownstones are known, the

estimated parameters correctly define the environmental conditions, the indoor environment does not change and mapping $\hat{p}$ to $\hat{l}$ is always possible. Even if one of these assumptions do not hold, it can hamper the accuracy of the predictions. The distributed algorithm assumes that the Crownstones can communicate well with their neighbors to reach a consensus. The main problems with these algorithms is their inability to deal with higher noise and limitations of the mapping function to output the correct location. Later, this algorithm can be improved by implementing more advanced consensus algorithms like ADMM which deal with noise better [45],[46].

The proposed data-driven algorithms deal with these uncertainties like noise by training the model to learn the underlying patterns from the RSSI-location data. The ML algorithm better classify the object and it is not even required to know the locations of the Crownstones, which makes this more suitable approach for other real-world applications, like in healthcare or other indoor environments. The proposed ML algorithm uses RSSI as the main features, but they only lead to an accuracy of 77%. Other time-related or measurements from the sensor other than RSSI can be included to improve the model further [47], [48]. Creative combination of ML algorithms with sensor fusion algorithms can help improve the accuracy. Another issue is related to the geometry of the test environment, in which ML algorithms tend to predict better when the asset is in and around the centre of the room, but performance degrades as it moves towards the boundaries. This is because of the multipath effects that become more pronounced near the boundaries, signal interference and limited training data. [49]. To mitigate these challenges, more data collection at various positions, additional features other than RSSI and implementing adaptive algorithms can be used [50]. Moreover, the hardware has its own limitations, like Crownstones hav a tendency to miss some of the advertisements. The ML algorithm deals with the missing values by data imputation in a distributed way. This works on a predefined matrix and a set threshold, so the results are dependent on the network structure.

The data-driven based algorithms can perfectly scale well with the number of Crownstones or data-size, but increases the complexity. The scope of the problem can be extended for predicting the real-time location of the asset which can lead to tracking. Moreover, the results can be improved by making sure that there is atleast one sensor present in every room. Another main possibility is to try a hybrid method, that can be done in two ways. First is to model the received signals to get the distance estimates and then implement a ML model to get $\hat{l}$ based on distance features. This way reduces the data collection efforts for every room and ML model can directly predict the locations. The other way is to implement model-based method to get $\hat{p}$ and then map it to a $\hat{l}$ using ML model. To further improve the performance, Kalman filtering technique can be used that can estimate a state change to predict current location by dealing with temporal variations arising from human movement, change in indoor objects [50].

Bibliography

[1] L. Bai, F. Ciravegna, R. Bond, and M. Mulvenna, "A low cost indoor positioning system using bluetooth low energy," *IEEE Access*, vol. PP, pp. 1–1, 07 2020.

[2] R. Faragher and R. Harle, "Location fingerprinting with bluetooth low energy beacons," *IEEE Journal on Selected Areas in Communications*, vol. 33, pp. 1–1, 11 2015.

[3] S. Puri, "Indoor positioning system using bluetooth," *International Journal of Engineering Research*, vol. 4, pp. 244–247, 05 2015.

[4] S. Hayward, K. van Lopik, C. Hinde, and A. West, "A survey of indoor location technologies, techniques and applications in industry," *Internet of Things*, vol. 20, p. 100608, 08 2022.

[5] A. Yassin, Y. Nasser, M. Awad, A. Al-Dubai, R. Liu, C. Yuen, and R. Raulefs, "Recent advances in indoor localization: A survey on theoretical approaches and applications," *IEEE Communications Surveys Tutorials*, vol. PP, pp. 1–1, 11 2016.

[6] U. Ahmad, "ibeacon localization," 2015.

[7] B. Yang, L. Guo, R. Guo, M. Zhao, and T. Zhao, "A novel trilateration algorithm for rssi-based indoor localization," *IEEE Sensors Journal*, vol. PP, pp. 1–1, 03 2020.

[8] C. Papamanthou, F. Preparata, and R. Tamassia, "Algorithms for location estimation based on rssi sampling," pp. 72–86, 07 2008.

[9] B. Dil, S. Dulman, and P. Havinga, "Range-based localization in mobile sensor networks," pp. 164–179, 02 2006.

[10] R. Nagpal, H. Shrobe, and J. Bachrach, "Organizing a global coordinate system from local information on an ad hoc sensor network," vol. 2634, pp. 333–348, 01 2003.

[11] M. Patel, A. Girgensohn, and J. Biehl, "Fusing map information with a probabilistic sensor model for indoor localization using rf beacons," pp. 1–8, 09 2018.

[12] V. Honkavirta, T. Perälä, S. Ali-Löytty, and R. Piché, "A comparative survey of wlan location fingerprinting methods," pp. 243 – 251, 04 2009.

[13] C. Frost, C. Jensen, K. Luckow, B. Thomsen, and R. Hansen, "Bluetooth indoor positioning system using fingerprinting," vol. 81, pp. 136–150, 01 2012.

[14] P. Bahl and V. Padmanabhan, "Radar: An in-building rf-based user location and tracking system," vol. 2, pp. 775 – 784 vol.2, 02 2000.

[15] J. Torres-Sospedra, R. Montoliu, S. Trilles Oliver, O. Belmonte Fernández, and J. Huerta, "Comprehensive analysis of distance and similarity measures for wi-fi fingerprinting indoor positioning systems," *Expert Systems with Applications*, vol. 42, pp. 9263–9278, 12 2015.

[16] H. Li, X. Shen, J. Zhao, Z. Wang, and Y. Sun, "Inemo: Distributed rf-based indoor location determination with confidence indicator.," *EURASIP J. Adv. Sig. Proc.*, vol. 2008, 01 2008.

[17] F. Xiao, M. Wu, H. Huang, R. Wang, and S. Wang, "Novel node localization algorithm based on nonlinear weighting least square for wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2012, 11 2012.

[18] B. Béjar and S. Zazo, "A practical approach for outdoors distributed target localization in wireless sensor networks," *EURASIP Journal on Advances in Signal Processing*, vol. 2012, 12 2012.

[19] Q. Shi and C. He, "A new incremental optimization algorithm for ml-based source localization in sensor networks," *Signal Processing Letters, IEEE*, vol. 15, pp. 45 – 48, 02 2008.

[20] S.-B. Cho, "Exploiting machine learning techniques for location recognition and prediction with smartphone logs," *Neurocomputing*, vol. 176, 05 2015.

[21] B. Akram, A. Akbar, and O. Shafiq, "Hybloc: Hybrid indoor wi-fi localization using soft clustering-based random decision forest ensembles," *IEEE Access*, vol. PP, pp. 1–1, 07 2018.

[22] T. Blazek, J. Karoliny, F. Ademaj, and H.-P. Bernhard, "Rssi-based location classification using a particle filter to fuse sensor estimates," 04 2021.

[23] I. Marin, I. Bocicor, and A.-J. Molnar, "Indoor localization techniques within a home monitoring platform," 09 2020.

[24] Nordic Semiconductor, *Crownstone chipset*, 2016. Rev. 1.8.

[25] FeasyBeacon, *Bluetooth beacon*, 2021. Rev. 1.2.

[26] Y. Hu, *Signal strength based localization and path-loss exponent self-estimation in wireless networks*. PhD thesis, Delft University of Technology, year = 2017.

[27] R. Priwgharm and P. Chemtanomwong, "A comparative study on indoor localization based on rssi measurement in wireless sensor network," 05 2011.

[28] X. Jiuqiang, W. Liu, F. Lang, Y. Zhang, and C. Wang, "Distance measurement model based on rssi in wsn," *Wireless Sensor Network*, vol. 2, pp. 606–611, 01 2010.

[29] S. Zekavat and R. Buehrer, *Handbook of Position Location: Theory, Practice, and Advances*. 09 2011.

[30] A. I. Sabuj, "Curve fitting." https://www.mathworks.com/matlabcentral/fileexchange/105995-curve-fitting, 2023.

[31] H. Jo and S. Kim, "Indoor smartphone localization based on los and nlos identification," *Sensors*, vol. 18, p. 3987, 11 2018.

[32] A. Booranawong, K. Sengchuai, D. Buranapanichkit, N. Jindapetch, and H. Saito, "Rssi-based indoor localization using multi-lateration with zone selection and virtual position-based compensation methods," *IEEE Access*, vol. PP, pp. 1–1, 03 2021.

[33] A. Norrdine, "Trilateration code." https://www.mathworks.com/matlabcentral/fileexchange/54680-trilateration-code, 2023.

[34] B. Béjar, P. Belanovic, and S. Zazo, "Distributed consensus-based tracking in wireless sensor networks: A practical approach," *Proceedings of the European Signal Processing Con- ference, EUSIPCO, 29 Aug - 1 Sep 2011; Barcelona*, 01 2011.

[35] Y. Ho and H. Chan, "Decentralized adaptive indoor positioning protocol using bluetooth low energy," *Computer Communications*, vol. 159, 04 2020.

[36] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[37] E. Sansano, *Machine-learning based techniques for indoor localization and human activity recognition through wearable devices*. PhD thesis, University Jaume I, 2020.

[38] B. A., "The hundred-page machine learning book," 2019.

[39] A. Alqahtani and N. Choudhury, "Ml for location prediction using rssi on wifi 2.4 ghz frequency band," 10 2022.

[40] L. Calderoni, M. Ferrara, A. Franco, and D. Maio, "Indoor localization in a hospital environment using random forest classifiers," *Expert Systems with Applications*, vol. 42, p. 125–134, 01 2015.

[41] J. Too, "Machine learning toolbox." https://github.com/JingweiToo/Machine-Learning-Toolbox, 2020.

[42] J. Bi, Y. Wang, B. Yu, H. Cao, T. Shi, and L. Huang, "Supplementary open dataset for wifi indoor localization based on received signal strength," *Satellite Navigation*, vol. 3, 11 2022.

[43] N. DIENG, "Distributed localization algorithms in indoor 802.15.4 wireless networks," 01 2014.

[44] M. Jalal Abadi, L. Luceri, M. Hassan, C. T. Chou, and M. Nicoli, "A cooperative machine learning approach for pedestrian navigation in indoor iot," *Sensors*, vol. 19, p. 4609, 10 2019.

[45] G. Soatti, M. Nicoli, S. Savazzi, and U. Spagnolini, "Consensus-based algorithms for distributed network-state estimation and localization," *IEEE Transactions on Signal and Information Processing over Networks*, vol. PP, pp. 1–1, 11 2016.

[46] M. Zhang, Z. Wang, F. Yin, and X. Shen, "Distributed scaled proximal admm algorithms for cooperative localization in wsns," 08 2022.

[47] P. Maduranga and R. Abeysekera, "Supervised machine learning for rssi based indoor localization in iot applications," *International Journal of Computer Applications*, vol. 183, pp. 26–32, 05 2021.

[48] O. Cheikhrouhou, G. Bhatti, and R. Alroobaea, "A hybrid dv-hop algorithm using rssi for localization in large-scale wireless sensor networks," *Sensors*, vol. 18, p. 1469, 05 2018.

[49] P. García-Paterna, A. Martínez-Sala, and J. Sanchez-Aarnoutse, "Empirical study of a room-level localization system based on bluetooth low energy beacons," *Sensors*, vol. 21, p. 3665, 05 2021.

[50] S. Tiku and S. Pasricha, *An Overview of Indoor Localization Techniques*, pp. 3–25. Cham: Springer International Publishing, 2023.