Enhanching RSSI-based Positioning Accuracy using Advanced Filtering Techniques

Master Thesis

International Software Systems Science

Faculty of Information Systems and Applied Computer Science

Otto-Friedrich-Universität Bamberg

Kamrul Hasan (Matr. No. 2029063)

July 26, 2023

Supervised By: Prof. Dr. Daniela Nicklas
&
Simon Steuer

Abstract

Indoor Positioning Systems (IPS) offer an advanced approach for precisely determining the location of objects and individuals within the indoor environment. The emergence of a Bluetooth Low Energy (BLE)-based IPS represents a promising technological advancement, holding practical implications in various use cases, namely asset tracking and indoor navigation.

BLE beacons, as cost-effective and energy-efficient devices, transmit wireless signals that can be utilized to estimate the location of a target in conjunction with an IPS. Despite their utility, the Received Signal Strength Indicator (RSSI) from BLE beacons can be subject to environmental noise and variations, leading to considerable inaccuracies and uncertainties in location estimation. This study commences with a focus on enhancing BLE based IPS through advanced filtering techniques.

The investigation explores the efficacy of Particle filter and Kalman filter algorithms in noise reduction and error mitigation in RSSI measurements, subsequently facilitating accurate indoor positioning. Additionally, the requirement analysis process for the proposed system is detailed, an exercise that includes the elicitation, analysis, categorization, and prioritization of requirements from diverse stakeholders.

The proposed system's architecture includes crucial modular components that are necessary for constructing an IPS based on BLE. To implement these components, a number of commonly used technologies have been utilized. Evaluation procedures involve collecting ground truth data and beacon data within a controlled environment, employing specific tools and configurations. Metrics for evaluation encompass noise reduction, distance accuracy, and performance analysis of the suggested system.

Results showed a substantial improvement in positioning accuracy when advanced filtering techniques were applied to RSSI data. In comparison to traditional IPS claims, the location estimation of the implemented system achieved sub-1.5-meter accuracy, demonstrating a significant improvement.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Bluetooth Low Energy offers an advantage in localization accuracy at a relatively low cost of development.Commercial off-the-shelf (COTS) devices can deploy BLE-based Indoor Positioning Systems, making it an accessible technology for various industries. However, while High Precision IPS exist, they often come with higher costs, presenting a barrier to their widespread adoption Liu et al. (2007). An RSSI measurement of BLE device is particularly useful in IPS because it allows proximity estimation between devices without requiring additional hardware. The accuracy of such an IPS depends on the quality of the RSSI measurements, which can be affected by various factors such as signal attenuation, interference, and multipath propagation.
The motivation for this research is to address the inherent inaccuracies and inconsistencies associated with using BLE beacon data for positioning and to explore new techniques to improve the accuracy of BLE RSSI-based IPS. Additionally, investigate how the accuracy and cost-effectiveness of BLE RSSI can be leveraged for optimal use in IPS. Another key consideration in the application of any technology is its energy efficiency. BLE beacons designed for low power consumption present a more energy-efficient solution than technologies like Ultra-Wideband (UWB), which require higher energy throughput Gomez et al. (2012). This factor is especially critical in large-scale deployment contexts where the energy consumption of devices can significantly impact operational costs and environmental sustainability.

1.2 Problem Statement

The use of BLE beacon data for positioning is confronted with significant challenges due to inherent inaccuracies and inconsistencies. A primary issue lies in the noise associated with the RSSI information from BLE beacons. The RSSI is a crucial measure used by BLE devices to estimate the distance between them, forming the basis for indoor positioning systems Faragher and Harle (2015). An RSSI-based IPS must address the following factors

1. RSSI can be very noisy, leading to significant errors in the computed positions.

2. Environmental factors further complicate this problem. Signal interference caused by peripheral objects, particularly metal objects and reflecting surfaces, can lead to inconsistencies in RSSI Zhou and Chen (2017).

3. Living beings, due to their water content, can also interfere with signal propagation, causing further inaccuracies.

These issues combined lead to a less accurate positioning system, undermining the potential benefits of BLE RSSI-based IPS. This problem is especially pronounced in complex environments such as industrial settings or farms, where a large number of peripheral objects and living beings can be present. To address these challenges, this research proposes a solution with advanced filtering techniques for BLE-based IPS. Filtering techniques such as Kalman filter, Particle filter , etc can potentially improve the IPS's performance Wan and Balakrishnan (2002), contributing to higher localization accuracy, cost-effectiveness, and energy efficiency. By exploring these techniques, this research will contribute to the broader scientific understanding of BLE RSSI-based IPS and provide practical insights into their applications.

1.3 Research Questions

To specify proposed IPS requirements, this work attempts to answer the following question:

What filtration method can reduce BLE beacon signal noise, consequently improving the accuracy in a highly reflecting and noisy area?

In order to address the research question, the study aims to gather data, specify critical design choices, define system architecture, and implement the proposed solution to enhance the accuracy and reliability of BLE-based IPS.

1.4 Thesis Structure

The next Chapter 2 reviews previous research on the open RSSI dataset for IPS, focusing on data collection methods and characteristics. In addition to that, it discusses the use of Kalman filters in mitigating RSSI variation and the application of Particle filters in complex positioning systems.

Chapter 3 undertakes a detailed requirement analysis using various techniques such as collaborative brainstorming, document analysis, interviews, and observation. The requirements thus obtained are categorized, prioritized, and validated.

The research's implementation phase in chapter 4 includes designing and developing a system that encompasses a backend architecture, data collection framework, pre-processing modules, and position estimation algorithms.

[0]`https://en.wikipedia.org/wiki/Kalman_filter`
[0]`https://en.wikipedia.org/wiki/Particle_filter`

The performance of the implemented system is then evaluated in chapter 5 concerning metrics such as noise reduction, distance accuracy, and overall system performance. The results indicate a substantial improvement in positioning accuracy when advanced filtering techniques are employed.

Lastly, chapter 6 of the thesis concludes by discussing the results, the implications of these findings and suggesting potential avenues for future research in RSSI-based indoor positioning systems.

Chapter 2

Related Work

This chapter aims to provide an overview of the literature on different Indoor Positioning Systems and their filtering techniques that rely on RSSI measurements. The discussion begins with examining the RSSI measurement dataset and the methods used for collecting data in indoor environments. Subsequently, two probabilistic approaches for indoor positioning estimation are addressed, including the Kalman filter and Particle filter, two widely adopted Bayesian estimation methods in indoor localization. The chapter also reviews several studies conducted on indoor positioning systems that utilize these filtering techniques.

2.1 Indoor Localization Dataset

The significance of indoor positioning and the demand for Location-based Services (LBS) has been driving the interest in indoor localization. The current methods used for determining the location of a person or object are not very accurate, especially in buildings with multiple levels. These methods do not perform well when given only a small amount of data to work with Shu et al. (2021). However, leveraging the RSSI dataset, combined with advanced filtering algorithms, can help address these challenges and further enhance indoor positioning accuracy. By effectively processing the RSSI data, the impact of signal fluctuations, multipath effects, and interference can be minimized.

2.1.1 Open RSSI Dataset for IPS

Fingerprinting dataset

RSSI fingerprinting is a commonly used technique that involves the measurement of the RSSI of wireless signals transmitted by Acess Points (APs) and received by mobile devices. A fingerprint database is created by recording RSSI values at different locations in an indoor area. These fingerprints are then compared with the current RSSI values of the mobile device to estimate its location. The accuracy of the location estimate depends upon the quality and density of fingerprints in the database and the characteristics of signal propagation in the

[0]https://en.wikipedia.org/wiki/Bayes_estimator
[0]https://en.wikipedia.org/wiki/Fingerprint_(computing)

parameter Chatzimichail et al. (2020). It can achieve sub-meter-level accuracy in indoor positioning, making them suitable for various applications such as navigation, asset tracking, and other LBS. Torres-Sospedra et al. (2014)

One such dataset for the fingerprinting method is from Shu et al. (2021), which has been widely used and studied in the research community. Several recent studies have utilized the dataset from the Microsoft Indoor Localization Competition. For example, Agrawal (2020) presents implementations from the competition that demonstrated an accuracy of 10 cm using beacons and encoded communication protocols. Barsocchi (2018) contributes to covering the research gap by presenting a new BLE dataset and discussing its applications in localization, tracking, occupancy, and social interaction. This dataset has served as a valuable resource for developing and refining indoor localization algorithms, enabling researchers to tackle challenges such as multipath interference, signal attenuation, and dynamic environmental conditions.

Another dataset focused on Wireless Local Area Networ (WLAN) fingerprint positioning technologies and methodologies is the UjiIndoorLoc dataset Torres-Sospedra et al. (2014), collected at the Universitat Jaume I (Spain), which serves as a valuable resource for indoor localization research. It comprises Wi-Fi fingerprint measurements obtained from multiple buildings with different floor plans, encompassing diverse environmental conditions and spatial complexities. The dataset has been applied explicitly to develop novel techniques for indoor navigation, context-aware services, and location-based applications. It has been used in studies on indoor localization accuracy estimation from fingerprint data Nikitin (2017), also been specifically used in studies on Recurrent Neural Networks (RNN) for accurate RSSI indoor localization Hoang (2019). Its diverse collection of RSSI fingerprints and ground truth labels has enabled significant advancements in indoor positioning techniques. Recently, a supplementary open dataset was created for UjiIndoorLoc called SODIndoorLoc Bi et al. (2022) offering denser and uniformly distributed Reference Points (RPs).

Finally, Hisham et al. (2022) presents a dataset that combines Wi-Fi and Bluetooth Low Energy (BLE) fingerprinting techniques for indoor localization in multi-floor environments with various layouts. There seems to be a scarcity of datasets that offer measurements of (RSS) for indoor environments with varying layouts. The authors of this study have successfully created a dataset, the first of its kind, which focuses on location fingerprinting for different layouts within the same indoor area. This dataset includes fingerprints for both Wi-Fi and BLE signals enabling the development of reliable positioning techniques based on fingerprinting. Additionally, the authors have analyzed to examine how layout changes affect signal variations and the performance of various fingerprinting-based positioning techniques. In previous fingerprinting datasets intended for indoor positioning, such as those listed in table 2.1, RSS measurements were not included for different layouts within the same indoor area.

Raw RSSI Dataset

One potential application of BLE beacons is in indoor localization. Using the raw RSSI data and Trilateration or Multilateration techniques, BLE beacons can be used to estimate the position of Bluetooth-enabled devices within an indoor environment. Additionally, by analyzing the RSSI values from BLE devices, it is possible to infer the presence or absence of individuals in a given space. This has implications for navigation systems, asset tracking, energy management, and security systems Abade (2018).

The BLEBeacon Sikeridis et al. (2018) dataset is a collection of BLE advertisement packets created from Bluetooth beacons carried by individuals as part of their regular routine in a university building. It can be used to understand network behavior and reliability detection in similar sensing environments, extract user mobility patterns, occupancy clustering with group monitoring, and provide insights for facility management in real-world conditions. This dataset has the potential to aid a wide range of research domains, particularly those involved in the development of advanced smart infrastructures and location-aware sensing platforms. The BLEBeacon dataset has been included in Dartmouth's Community Resource for Archiving Wireless Data from 2019. Sikeridis et al. (2022).

Similar to the preceding dataset, the dataset from Mohammadi and Al-Fuqaha

Table 2.1: An Analysis of Indoor Fingerprint Datasets Recently Published Hisham et al. (2022)

Dataset	**Signal Types**	**Access Points**	**Multi Building**	**Multi Floor**	**Changing Layouts**
UJI IndoorLoc Torres-Sospedra et al. (2014)	Wi-Fi	520	Yes	Yes	No
IPIN2016 Tutorial IPI (2016)	Wi-Fi	168	No	No	No
Tampere University Lohan et al. (2017)	Wi-Fi	309 354	Yes	Yes	No
ALCALA2017 Mendoza-Silva et al. (2018)	Wi-Fi	152	No	No	No
Database Torres-Sospedra et al. (2016)	Magnetic field Wi-Fi	9 indoor 88 outdoor	No	No	No
BLE RSS Mendoza-Silva et al. (2019)	BLE	22, 24	No	No	No

(2018) encompasses a compilation of RSSI readings obtained from an array of (BLE) iBeacons deployed in a real-world operational indoor environment. The primary purpose of this dataset is to facilitate academic research endeavors focusing on localization and navigation. The dataset used in this study Mohammadi et al. (2018) involved the utilization of RSSI readings from a collection of 13 iBeacons located on the first floor of Waldo Library at Western Michigan University. Additionally, a dedicated mobile app was developed specifically for capturing training data. It was used to train and test a semi-supervised deep reinforcement learning model for indoor localization. Initial experiments highlighted the need to enrich the features, so the authors added two sets of features to the original ones.

While there numerous datasets exist for both fingerprinting-based and raw RSSI data, it is worth noting that fingerprinting datasets possess inherent advantages due to their superior collection methodologies and frameworks. In contrast, raw RSSI datasets are typically a subset of fingerprinting datasets and are limited in scale, available only in smaller quantities.

2.1.2 Data Collection and Characteristics

The Indoor Location Competition 2.0 Dataset Shu et al. (2021) was collected by 1446 participants from over 60 countries, forming 1170 teams. The competition provided a unique indoor location benchmark dataset, including dense indoor WiFi, geomagnetic field, and iBeacon advertisements, in addition to ground-truth for waypoint locations from hundreds of buildings across Chinese cities. Site surveyors used an Android smartphone to collect data, including Inertial Measurement Unit (IMU) and geomagnetic fields using sensor readings, as well as WiFi and Bluetooth iBeacon scanning results. Training and test path files were organized by site and floor, as well as metadata files that included floor images, floor information, and geojson maps .

Hisham et al. (2022) authors collected the dataset in a university building equipped with Wi-Fi access points and BLE beacons. They used smartphones with Wi-Fi and BLE receivers to collect data, and developed a software tools for data collection and preprocessing. The data collection setup involved a trained individual standing at predetermined reference points in a multi-floor building. The RPs were arranged in a grid with a 1-meter distance between adjacent RPs, resulting in a total of 384 RPs throughout the building, the figure 2.1 depicts a fragment of it. An Android smartphone device was used for fingerprint measurement. The device could detect both BLE and Wi-Fi signals. Throughout the experiment, the mobile device was held consistently in the same orientation at a height of 1.5 meters. Two applications were used for signal measurement and collection. The Sensoro application was used for BLE beacons, while the All Router Setup application was used for Wi-Fi access points (APs). RPs were measured and recorded into multiple Comma-separated Values (CSV) files using 30 samples each.

[0]https://en.wikipedia.org/wiki/IBeacon
[0]https://en.wikipedia.org/wiki/Android_(operating_system)
[0]https://en.wikipedia.org/wiki/GeoJSON

2.2 Kalman Filter

The Kalman filter is a widely used algorithm in various fields, including computer science, data assimilation, control theory, and signal processing, to estimate the state of a system over time Hidayat (2011). It combines measurements and predictions while considering noise to make estimates. The algorithm assumes that the system follows a Markov process with Gaussian noise and uses measurements observed over time to produce more precise estimates compared to relying on a single measurement Welch and Bishop (2006).

Understanding the Kalman filter requires basic statistical concepts. This includes the mean, which's the average of all values in a dataset, and the expected value representing the long-term average value of a variable, especially when dealing with hidden variables. Additionally, variance and standard deviation are essential as they measure how spread out values are from the mean in a dataset.

In indoor positioning applications, this filter utilizes sensor measurements such

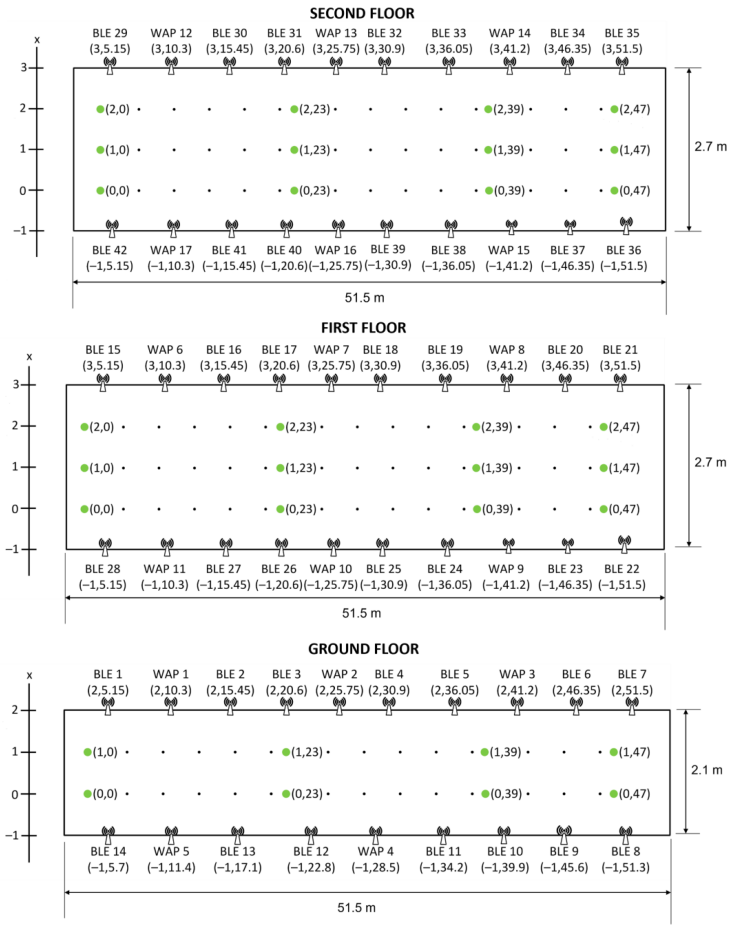


Figure 2.1: Ground, First, and Second Level Floor Plans Hisham et al. (2022)

as Wi-Fi, Bluetooth, RFID, or IMU sensors to estimate the position of an object within an environment. By considering how the target moves and acknowledging the uncertainties in sensor measurements Li et al. (2019), it keeps adjusting its position estimation and how certain it is based on new measurements and system dynamics. This helps to minimize any errors caused by noise.

2.2.1 RSS Variation Mitigation

The (RSS) variation is a major challenge in indoor localization systems. Researchers have proposed various filtering techniques to mitigate the effect of RSS variation, including Low Pass Filter, Frequency Analysis and Kalman Filter. Qureshi (2019) investigated the use of Low Pass Filter and Kalman Filter to mitigate random variations from BLE signals. They observed a significant improvement in localization accuracy and precision with a filter-based approach. Lee et al. (2015) provided the mapping between offline and online RSS measurements by applying a recursive least squares estimation-based self-calibration algorithm. The proposed algorithm achieved higher accuracy even in severe RSS variation conditions by combining the Kalman filter-based tracking algorithm with self-calibration.

Jose et al. (2023) introduces a two-step approach. The first step involves applying a frequency analysis technique to identify the dominant frequency components in the RSSI signal. The method aims to eliminate the noise and variations associated with other frequency components by extracting these dominant frequencies. In the second step, the Kalman filter algorithm is employed to further enhance the RSSI measurements' accuracy, smooth the RSSI measurements, reduce the effects of noise and fluctuations, and provide a more reliable signal strength estimate. A block diagram 2.2 of their system design shows how they used a combination of frequency analysis and Kalman filter alternating sequential manner. The results demonstrate that the approach effectively mitigates the RSSI variations and improves the accuracy of indoor positioning systems.

The Kalman filter can be used to estimate the target's position based on the available RSS measurements and a dynamic self-calibration model, the model considers factors such as velocity, acceleration, and direction of movement. The self-calibration process aims to adaptively estimate and compensate for the variations in RSS caused by environmental changes. By continuously monitoring the RSS variations, the system updates the calibration parameters to improve the position tracking accuracy Lee et al. (2015). The authors propose a mathematical formula for the RSS variation model, which includes a state equation and a measurement equation. The state equation represents the dynamics of the RSS variation process, while the measurement equation relates the observed RSS value to the true RSS value.

$$x_k = F_k x_{k-1} + w_k \tag{2.1}$$

where x_k represents the state vector at time k, F_k is the state transition matrix, and w_k is the process noise.

[0]https://en.wikipedia.org/wiki/Low-pass_filter
[0]https://en.wikipedia.org/wiki/Timefrequency_analysis

The measurement equation is given by:

$$z_k = H_k x_k + v_k \tag{2.2}$$

where z_k represents the observed RSS value at time k, H_k is the measurement matrix, and v_k is the measurement noise.

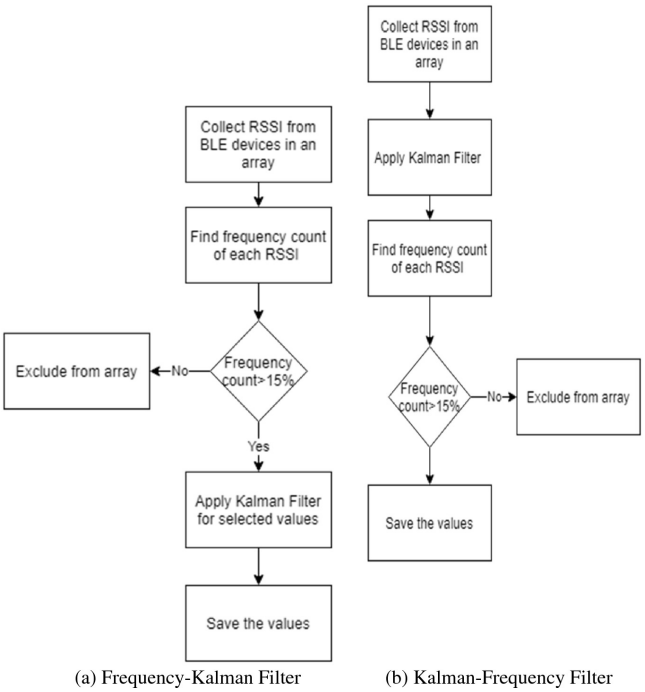
The state transition matrix F_k and measurement matrix H_k are defined as:

$$F_k = 1\Delta t_k\ 01 \tag{2.3}$$

$$H_k = 10 \tag{2.4}$$

where Δt_k is the time between successive measurements.

The process noise w_k and measurement noise v_k are assumed to be Gaussian distributed with mean zero and covariance matrices Q and R, respectively. The Kalman filter is used to estimate the state vector x_k and the covariance matrix P_k of the estimation error. The system continuously monitors the RSS values and estimates the calibration parameters that compensate for the variations. The self-calibration technique ensures that the position tracking remains accurate, even in changing environments where RSS values may fluctuate.



(a) Frequency-Kalman Filter (b) Kalman-Frequency Filter

Figure 2.2: Flowchart Diagram Frequency and Kalman Filtering Jose et al. (2023)

2.2.2 Aid for Hybride IPS

The Kalman filter algorithm enhances indoor positioning techniques like fingerprinting, multi, and trilateration with a dynamic model, improving accuracy and reliability. Studies such as Chen and Wu (2016) demonstrate the Kalman filter's use in fingerprinting-based systems, refining position estimates. Cetin et al. (2016) showcase its effectiveness in multi-trilateration, reducing errors and enhancing accuracy. The integration of the Kalman filter improves the performance of indoor positioning systems in terms of accuracy and robustness.

Kalman Filtering in conjunction with track-to-track fusion, has the potential to enhance hybrid indoor positioning systems that integrate both Multilateration and fingerprinting techniques using (RSSI) measurements Eyng et al. (2020). Three primary parameters influenced the accuracy of the proposed IPS.

- Fingerprinting Grid Size: The researchers observed that the accuracy of the proposed HIPS varies slightly based on the FP grid size, but the variation was comparatively small in relation to standalone FP techniques.
- Number of Access Points: The paper suggests that fewer APs in the HIPS scheme still achieved similar performance levels as standalone FP and MLT schemes with a much larger number of APs. Thus, the number of APs influenced the accuracy.
- Number of Samples: In the study, variations in the error as a function of the number of samples were evident, particularly for the Kalman Filter component of the HIPS scheme.

The results showed significant improvements in accuracy compared to standalone fingerprinting and Multilateration techniques. The hybrid technique showed a probability of 92% accuracy compared to 43% and 47% for the aforementioned techniques when the distance error was less than 2m.

The Extended Kalman Filter Julier and Uhlmann (2004) is an extension of the traditional Kalman Filter that is specifically designed to handle nonlinear systems. While the original Kalman Filter assumes linear dynamics and Gaussian noise, the EKF allows for nonlinear dynamics and non-Gaussian noise in the system. Lee et al. (2016) utilizes an extended Kalman filter, which considers the nonlinear relationship between the beacon's RSSI-based distance measurement and the smartphone's location on a two-dimensional surface. This study's system 2.3 variables are defined based on the iBeacon RSSI analysis. Standard Deviation of process noise (Q value)

$$Q = 1.0$$

and measurement noise (R value)

$$R = \sigma_1 00 \sigma_m$$

[0] `https://en.wikipedia.org/wiki/Trilateration`

(2.5)

Where σ_1 and σ_m are standard deviation components. However, the paper does not provide specific equations or values for these standard deviation components (σ_1, σ_m). These are calculated based on the observations made in this study. The EKF keeps updating its state estimate to minimize the error and improve the system's positioning accuracy.

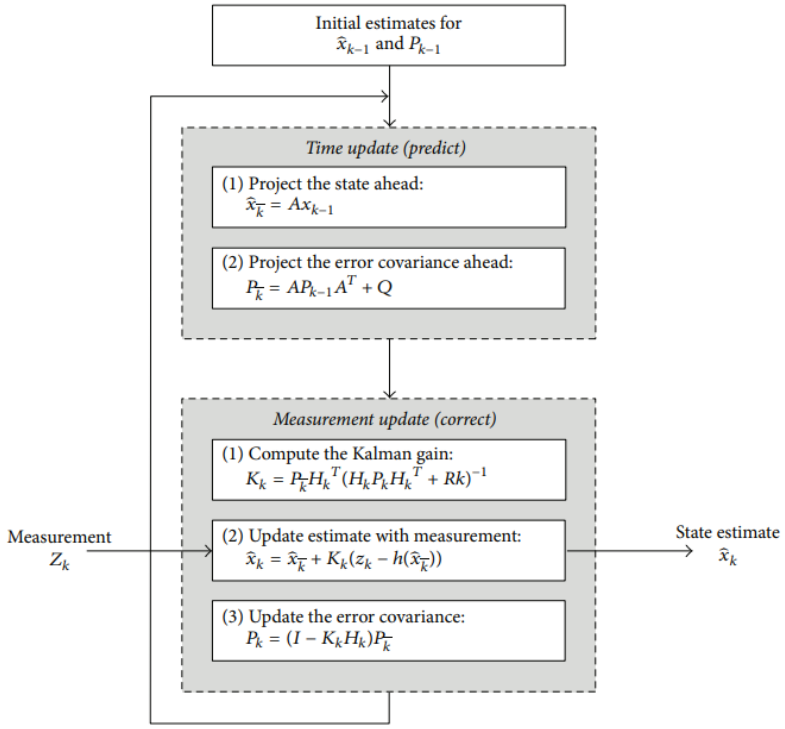


Figure 2.3: Extended Kalman Filter Design Lee et al. (2016)

2.2.3 Kalman-Based Fusion

Fronckova and Prazak (2020) propose an algorithm for smartphone-based indoor localization using (BLE) beacons. The technique includes a channel-separate polynomial regression model, channel-separate fingerprinting, outlier identification, and extended Kalman filtering to estimate the target's location and the distances between the target and BLE beacons.

Additionally, a comparison was made between three filtering techniques for processing RSSI data: Average Filter , Median Filter, and Kalman Filter. The Average Filter performed the function of smoothing by calculating the mean of the data points within a frame size of 20, however, this smoothing technique risks rendering measurements outdated if the frame size is too large. The Median

[0]https://en.wikipedia.org/wiki/Moving_average
[0]https://en.wikipedia.org/wiki/Median_filter

Filter, which also utilizes a frame size of 20, mitigates the impact of outliers by selecting the middle value. Comparatively, the Kalman Filter approaches RSSI value changes as random, one-dimensional data fluctuations and focuses on processing this aspect of the data. Among all the examined filters, the Kalman Filter displayed superior performance, generating the fewest jumps and variations in the RSSI values, as in figure 2.4 shows the filtered distance error by Kalman is the lowest.

2.3 Particle Filter

The particle filter, also called the Sequential Monte Carlo method, is an algorithm used in Bayesian statistics to estimate the state of a system based on a series of observed data Doucet et al. (2001). It is an algorithm for estimating the state in nonlinear and non-Gaussian systems. The particle filter finds applications in robotics and computer vision, such as tracking an object's position and orientation over time using noisy sensor data or determining the location of a robot within a known map based on its sensor readings.

The particle filter's basic idea is to use particles to represent the probability distribution of the system's state. These particles are selected at random from distribution and then updated based on collected data utilizing importance sampling and resampling procedures. Each particle is allocated a weight during importance sampling that indicates how closely it corresponds to the measurements. To preserve the accuracy of the set, particles with weights are copied, while those with lower weights are discarded during the resampling processDong (2020).

Particle filters can be effectively adapt to dynamic indoor environments by using techniques such as systematic resampling, stratified resampling, and resampling. These methods aim to maintain or enhance particle diversity, ultimately improving indoor positioning systems' accuracy and reliability.

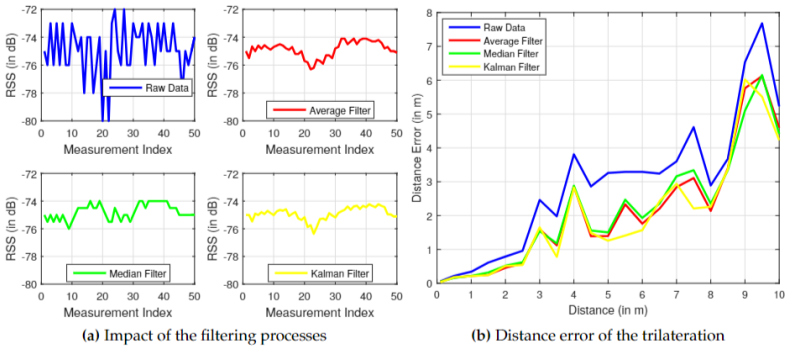


Figure 2.4: A Comparative Analysis of Different Filters Lee et al. (2016)

2.3.1 Directional and MultiFloor

Huang et al. (2022) presents a multi-floor indoor positioning system that combines inertial sensors, such as accelerometers and magnetometers, with a directional particle filter algorithm Zhang (2016). The directional particle filter considers the user's heading direction, which improves the accuracy of the positioning system, especially in multi-floor environments. The system described in the paper utilizes the measurements from the inertial sensors to estimate the user's position and orientation. The particle filter algorithm then updates the estimated position based on the observed sensor measurements. By incorporating the heading direction information, the system is able to handle better the challenges of indoor positioning, such as subtle motions of human movement Kang (2015).

The proposed system, EasiTrack Wu et al. (2020), represents the indoor environment as a graph, where nodes represent landmarks and edges represent the connectivity between landmarks. The system leverages the graph structure to estimate the user's position based on the observed landmarks and their connectivity Xu (2019). Using the graph-based approach, EasiTrack achieves decimeter-level tracking accuracy in indoor environments. The algorithm uses a set of particles to represent the possible positions of the user and iteratively refines the position estimate based on the observed landmarks. The graph structure helps constrain the possible positions and improve the tracking system's accuracy. They emphasized the importance of updating the radio maps in dynamic environments to account for changes in the indoor environment Alshami (2017).

2.3.2 Novel Appraoches

Shen et al. (2020) proposes a Particle Filter-based Indoor Positioning System that localizes and tracks a tag using BLE beacon signal. The system's design utilizes a Gaussian model for state update and an RSSI gradient for estimating target motion in 1D and 2D space. The Kalman Filter is used to smooth the fluctuating RSSI data from the beacon after aggregating it. Furthermore, an effective Particle Filter is developed to approximate the beacon's location, gradually reducing location uncertainty in Gaussian belief space. To validate this design, a series of simulations were conducted in a virtual indoor environment and experiments using COTS devices. The experimental results revealed that the proposed system, outperforms legacy IPS in terms of location accuracy by 24.1%. The system also achieved a median accuracy of 1.16 meters. The authors compared different indoor positioning schemes presented in table 2.2.

A novel framework, STUPEFY proposed by Malekzadeh et al. (2019) that integrates set-valued information within box particle filtering for BLE-based tracking. STUPEFY consists of three integrated modules demonstrated in 2.5: a Smoothing module based on Kalman filtering, a learning-based model referred to as the Cooperation module, and a set-valued box particle filtering approach termed the Micro-Localization module. The Smoothing module applies Kalman filtering to mitigate the fluctuations and facilitates the comparison of Gaussian models of the RSSI values in distribution with the learned ones. The cooperation

module comes into play next, offering an initial rough estimate of the target's location and constructing the smallest axes-aligned box containing the ellipsoid associated with each zone's learned RSSI distribution. The novel set-valued box particle filtering module provides a refined, more accurate estimation of the target's location. Evaluating the proposed framework based on real BLE datasets and shows that it outperforms existing single-model and hybrid solutions in terms of localization accuracy.

Table 2.2: Comparative Analysis of Different Indoor Positioning Schemes

Name	**Tech**	**Method**	**Acc**	**Pros**	**Cons**
PLS Kim et al. (2020)	CSI	Fingerprinting & Machine learning	0.5∼1.3 m	NLoS positioning	Complex CSI operations
AngLoc Chen et al. (2020)	CSI RSSI	AoA,ToF & Fingerprinting	0.3∼1.2 m	Angle-delay estimation	Offline angle data set extraction
SPR Mackey et al. (2020)	RSSI	Moving Average, Particle Filter	1∼3 m	Simple system	1-D distance estimation
EasiTrack Wu et al. (2020)	CSI IMU	Multilateration, Particle Filter	0.1∼1 m	High precision; MIMO WiFi device	CSI phase information loss; High processing power

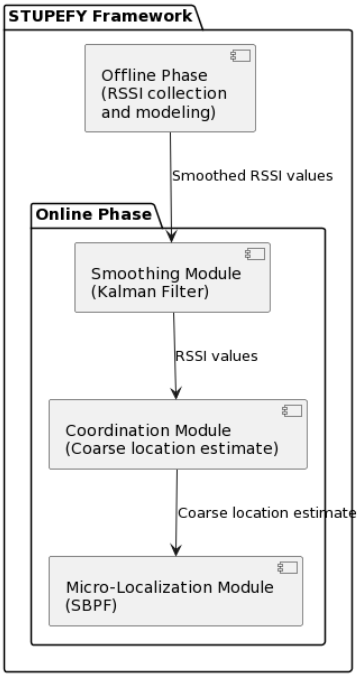


Figure 2.5: STUPEFY Framework Architecture

Chapter 3

Requirement Analysis

This chapter will initially examine the requirement management process for the proposed system. Subsequently, we will explain each phase of the requirement analysis, including how it was employed to obtain a comprehensive understanding of stakeholder requirements, refined requirements, and establish the system's scope.

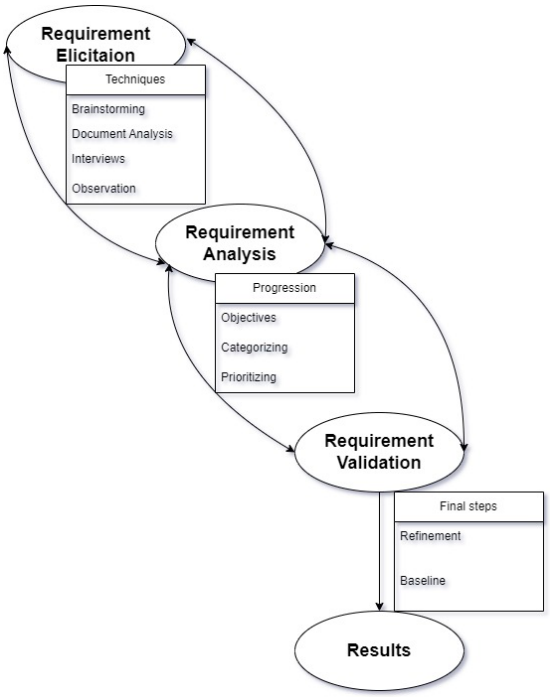


Figure 3.1: Requirement Management Process

Requirement Management

Requirement analysis is a crucial phase in software development that aims to identify, understand, and document the needs and expectations of stakehold-

ers regarding a software system. The principle behind requirement analysis is to gather comprehensive and accurate information about the software's desired functionality, performance, and constraints Sommerville (2016). It involves a systematic approach to analyzing, prioritizing, and refining these requirements, ensuring that they are clear, feasible, and aligned with the overall study objectives, depicted as the requirement management process for this work in figure 3.1

3.1 Techniques for Requirement Elicitation

Various techniques are employed during requirement elicitation to help capture a comprehensive set of requirements, prioritize them, and ensure their accuracy and validity. One commonly used technique is the use of interviews to engage with stakeholders and gather their perspectives directly. According to Sommerville (2016), interviews provide valuable insights into stakeholders' needs, preferences, and expectations. On the other hand, workshops allow for collaborative discussions and brainstorming sessions, facilitating consensus-building and a shared understanding of requirements Kumar and Kumar (2012). Moreover, observing users or stakeholders in their natural environment helps identify implicit requirements and understand the context in which the software will be used Lauesen (2002). Document analysis, conversely, involves reviewing existing documentation to extract relevant requirements and gain insights into the business or technical aspects of the software Karlsson and Ryan (1997).

3.1.1 Collaborative Brainstorming

Regular brainstorming sessions were arranged in conjunction with the supervisor and industry professionals. Initially, a series of collaborative meetings were conducted to establish a clear understanding of the study's scope and objectives, alongside how to incorporate solution providers such as Favendo GmbH effectively without compromising the academic integrity of this work. Recurrent meetings were scheduled with the supervisors during the work, where we generated a wide range of innovative ideas, regardless of their feasibility or initial impression. This approach helped us to explore various possibilities of advanced filtering techniques in indoor positioning systems. Eventually, our iterative brainstorming sessions and evaluation helped us narrow down the ideas.

3.1.2 Document Analysis

We analyzed two existing documents to extract relevant requirements, system processes, and identify gaps or inconsistencies. The first documentation we refer to is from *WeideInsight* project, by reviewing the document, we can conclude that the software system for livestock management should possess functionalities including GPS-based localization, data analysis, third-party system integration, alarm notifications, and historical data access. A user-friendly interface is essential, and the architecture needs to be modular enabling scalable and efficient processing of diverse sensor values. The system must also present real-time

[0] `https://www.favendo.com/de/`

livestock data.The design should encourage user feedback and facilitate user training sessions, optimizing the user experience and system performance.

The second document is a whitepaper provided by Favendo GmbH called *Whitepaper Asset & People Tracking*. The asset tracking system provides both asset and person tracking using signal strength-based positioning methods (trilateration, triangulation, multilateration), and must integrate with pre-existing IT systems. It should display continuous real-time data, be cost-effective, scalable, and adaptable across diverse environments while respecting the privacy constraints of the General Data Protection Regulation(GDPR). The system relies on at least three signal transmitters and needs consultation for perfect coverage. Users require training, and the application must support alerts and search functions. Primary stakeholders are businesses optimizing processes and worker safety, while secondary stakeholders encompass healthcare facilities prioritizing patient safety and infection control.

3.1.3 Interviews

In order to gain a thorough comprehension of the system requirements and address unresolved queries, we conducted interviews with Prof. Dr. Nicklas during kickoff and midterm talks. These interviews were instrumental in obtaining her expert insights, perspectives, and expectations. The supervisor consistently provided essential information pertaining to the study through online interviews and email correspondence and delivered pre-established responses and high-level requirements as conveyed by the stakeholders. Furthermore, one-on-one or group interviews were conducted with industry experts, employing open-ended questions to delve into their testimony and capture valuable insights.

3.1.4 Observation

An extensive examination of the existing system was conducted and analyzed data obtained from the WeideInsight project. In particular, we closely examined the high-level overview of the Safectory solution, considering its functionality and its relevance to the project. During this process, we noted integration points, data exchanges, or dependencies that influence requirements. The provided material includes specific site descriptions, installation photos, floor plans, and ground truth data. These data points enabled us to create a comprehensive representation of the project and translate it into a simulated environment, as well as establish attributes for the testbed. Additionally, as Favendo GmbH was involved in this study, we obtained access to and utilized their indoor location system and made significate observations of an operational system firsthand. The observation consists of users in their natural environment to understand their needs, behaviors, and challenges. In later chapters, we discuss more regarding the utilization of their system. Finally, during all the observation, we took note of exceptional or edge cases that may not be apparent through interviews or discussions alone.

[0]`https://en.wikipedia.org/wiki/General_Data_Protection_Regulation`
[0]`https://safectory.com/`

3.2 Requirement Analysis Progression

Following each cycle of the elicitation process, we employed reasoning techniques to scrutinize the requirements for conflicts or inconsistencies, combine related requirements, and identify system functionalities. This systematic approach enabled us to clearly establish the goals, scope, and constraints of the system, thus aligning with the core objective of this study. Subsequently, we analyzed and categorized the gathered requirements into functional and non-functional categories. Lastly, we assessed the significance and priority of each requirement by employing techniques such as the MoSCoW (framework Clegg and Barker (1994).

3.2.1 Defining Objectives

This research investigates the efficacy of advanced filtering techniques, namely the particle filter and Kalman filter, in improving the accuracy and reliability of commonly used lateration approaches for indoor location determination based on BLE beacon data. The availability of both beacon data and labeled ground truth data provides a unique opportunity to assess the effectiveness of these filtering methods in reducing noise and mitigating errors. By applying the particle filter and Kalman filter to the existing lateration approaches, the objective is to enhance the precision and robustness of indoor positioning systems. The research seeks to contribute to the field by evaluating the impact of these filtering techniques on the reduction of errors and the improvement of location estimation quality. The findings will provide insights into the effectiveness of advanced filtering techniques and their potential to enhance the performance of indoor positioning systems based on beacon data.

3.2.2 Categorize Requirements

The requirements were categorized into functional and non-functional categories based on their relevance to our objective. Functional requirements and non-functional requirements are key components of the requirement analysis process in software development Sommerville (2016); Kotonya and Sommerville (1998). Functional requirements specify the system's specific functions and behaviors, while non-functional requirements address the system's quality attributes and constraints. Both types of requirements are crucial for developing software systems that meet stakeholders' needs and expectations Dardenne et al. (1993); van Lamsweerde (2009). The interplay between functional and non-functional requirements ensures the system's desired functionalities are accompanied by the necessary qualities and constraints for optimal performance. Understanding and incorporating both types of requirements are essential for successful software development projects.

Functional Requirements

The functional requirements for the proposed advanced filter-based indoor positioning systems are

- **Data Collection:** The system must facilitate the collection of sensor data from diverse sources, including WiFi access points, Bluetooth beacons, and

inertial sensors, to capture environmental information and target object or user characteristics.

- **Preprocessing:** The system should preprocess the collected sensor data to enhance its quality and reliability. This step involves filtering and cleaning raw measurements, eliminating noise and outliers, and preparing the data for subsequent processing stages.
- **Position Estimation:**The system must implement both the Kalman filter and particle filter algorithms to process the preprocessed sensor data. These algorithms will estimate the indoor position of the target object or user based on the processed sensor measurements.
- **Sensor Fusion:**The system should employ sensor fusion techniques to integrate the data from multiple sensors. By combining and analyzing the measurements obtained from various sources, the system can enhance the accuracy and reliability of the position estimation process.
- **Real-time Tracking:**The system must continuously update the estimated position in real-time as new sensor data becomes available. This capability ensures that the tracking information remains current and aligns with the target object's or user's movements.
- **Calibration and Initialization:** The system should incorporate a calibration and initialization step to calibrate the sensors, establish initial values for the filtering algorithms, and ensure accurate position estimation. This step is crucial for aligning the system's measurements with the physical environment.

Non Functional Requirements

In order to implement the system, the following non-functional requirements must be met:

- **Accuracy:** The system should strive for a high level of accuracy in position estimation by minimizing errors and providing precise indoor location information. This requirement is vital for supporting reliable tracking and navigation within the indoor environment.
- **Responsiveness:** The system should exhibit low latency in updating the estimated position, ensuring real-time tracking and responsiveness to changes in the target object's or user's movement. Prompt position updates are essential for providing an up-to-date and seamless user experience.
- **Scalability:** The system must be capable of scaling to accommodate multiple tracked objects or users simultaneously without experiencing significant degradation in performance or accuracy. This requirement ensures that the system remains effective and efficient in different deployment scenarios.
- **Integration:** The system should offer integration capabilities to seamlessly integrate with existing indoor positioning infrastructure, such as

WiFi networks or beacon networks. This feature enhances positioning accuracy by leveraging complementary technologies and data sources.

- **Security and Privacy:** The system should incorporate appropriate security measures to protect the privacy and confidentiality of sensitive user data collected during the positioning process. Adhering to privacy regulations and ensuring data security are essential considerations for user trust and system adoption.

3.2.3 Prioritize Requirements

The process of prioritization involves the assessment and allocation of priorities to different requirements based on their relative significance and impact on the overall success of the project Cohn (2004). Given the constraints of limited time for our work, we employed techniques such as the MoSCoW method as illustrated in figure 3.2. The MoSCoW approach allows for the categorization of requirements into four priority levels: Must-Have, Should-Have, Could-Have, and Won't-Have. Must-Have requirements are indispensable and non-negotiable for the project's success. Should-Have requirements are important but can be deferred if necessary. Could-Have requirements are desirable but not critical. Won't-Have requirements are intentionally excluded or postponed for future consideration.

Figure 3.2: MoSCoW Prioritization and Scoping

[0]https://en.wikipedia.org/wiki/MoSCoW_method

3.3 Requirement Validation and Results

As part of this validation process, the completeness of the requirements is reviewed in order to ensure that all essential functionalities and characteristics have been captured. Additionally, the requirements are assessed to ensure that they do not conflict with one another or create contradictory conditions. Lastly, after requirements have been analyzed, validated, and approved by supervisors, a baseline of requirements is established.

3.3.1 Refinement

A comprehensive refinement process was undertaken to validate the accuracy of the requirements in capturing the underlying needs and goals of the systems. This rigorous process was instrumental in enhancing the clarity, relevance, and effectiveness of the requirements. The details of this refinement process will be discussed in Chapter 5, as it forms an integral part of the agile methodology adopted for the implementation of this system.

3.3.2 Requirement Baseline

Following are this system's baseline requirements that serve as a reference for development activities and change control

- **Beacon Signal Receiver:** A beacon scanning application should be designed to actively search for BLE beacons within its surrounding environment. Depending on the specific configuration, the application can operate either as a standalone BLE beacon tracker or as a group of trackers. Its primary function is to scan and retrieve beacon data, which is then parsed during the pre-processing stage. However, the application can also be extended to collect and store data for experimental purposes if required.

- **Beacon Signal Pre-processing:** One of the primary requirements of the study is to reduce noise from BLE beacon signal. Consequently, upon reception of the beacon broadcast by the scanning application, it should be processed to eliminate noise; different techniques should be employed to achieve that. The pre-processor should be able to handle the stream of multiple beacon data and manipulate and forward it to other components if needed.

- **Position Estimation Algorithms:** Position estimation algorithms are crucial requirements that must be included in the system since they are responsible for providing key features of the system such as indoor localization, autonomous navigation, and asset tracking. These algorithms need to be implemented on the pre-processed data received from the BLE beacon. To enhance the accuracy of BLE-based indoor positioning, the position estimation needs to make advancements over the current state. Design an appropriate system architecture to integrate the estimation of position based on the processed data from BLE beacons, with a focus on achieving high accuracy and efficiency.

- **Real-time Tracking:**The final requirement of the system is the real-time tracking functionality. Real-time tracking is a crucial component of the systems, as it enables the continuous monitoring and updating of an object's location or status. A visualization tool needs to be implemented to allow the user to see the real-time data relevant to the target object in an indoor environment.

Chapter 4

Implementation

This chapter will delve into the implementation procedures employed for the proposed solution. The chapter is structured into four sections, starting with a detailed explanation of the system design and architecture, focusing on establishing a feasible standalone BLE-based Indoor Positioning System. Moving forward, the second section outlines the data collection process essential for the functioning of our position algorithm, along with a discussion of the dataset's attributes. The final section encompasses pertinent information regarding the technologies employed in the project and the system's visualization.

4.1 System Design

System design plays a crucial role in software development as it encompasses the process of defining, planning, and organizing the structure, components, and interactions of a software system Bass et al. (2012). It lays the foundation for the entire development process, ensuring that the final product meets the desired objectives, performs efficiently, and is scalable and maintainable. Effective system design leads to a robust and reliable software system that meets the needs of users and stakeholders.

Designing a BLE based IPS requires consideration of several factors. Firstly, accurate location determination using BLE technology involves careful consideration of signal strength measurements, positioning techniques, and calibration processes. Secondly, the system design should include methods for collecting, processing and analyzing data received from BLE devices, including handling data synchronization, noise filtering, and implementing algorithms for converting raw data into meaningful position information. The system design should also consider scalability and efficient real-time data processing by selecting appropriate hardware components and software frameworks to support desired system requirements and performance benchmarks Gortázar and Kurpas (2016). The following figure4.1 from Favendo illustrates a simple system design for IPS.

4.1.1 Architecture

In order to transform the system design into a software architecture, we utilized a combination of the pipe-filter and broker architecture patterns. By leveraging the strengths of both patterns, the system can achieve a modular and scalable design while facilitating data flow and coordination among components. Figure 4.2 presents a high-level architectural overview of our proposed indoor positioning system. It is important to note that the implementation of each component within this software architecture falls beyond the scope of this study, therefore only the essential parts of the architecture were implemented to carry out the study. Yet we present a positioning system architecture that incorporates a backend in order to demonstrate how to utilize and provide LBSs.

This combined architecture ensures efficient data processing, modular design,

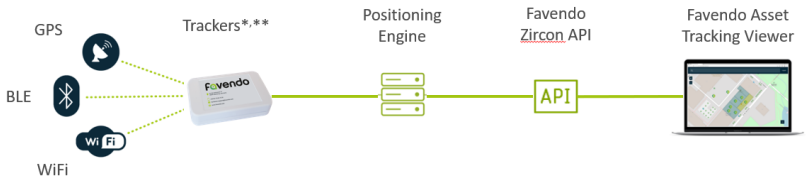


Figure 4.1: A Simplistic IPS Design

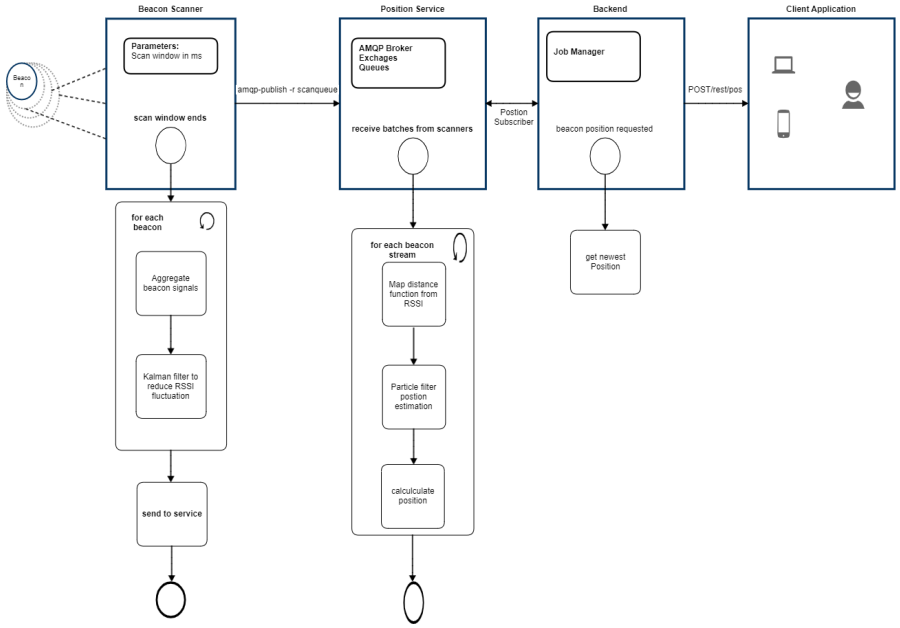


Figure 4.2: High-level Architecture

and seamless communication between components. In designing the architecture for the proposed indoor positioning system, we considered the following characteristics.

- **Data Processing Pipeline:** The pipe-filter pattern can be used to create a data processing pipeline within the system. BLE beacon data flows through a series of filters, each responsible for a specific processing step. In this case these filters perform tasks like beacon whitelisting, scanning window and aggregation, signal pre-processing, distance calculations, coordinate conversion, etc. Each filter in the pipeline encapsulates a specific data transformation operation.
- **Broker as Communication Hub:** The broker architecture pattern is employed as a central communication hub within the system. The broker receives data from the filters in the pipeline and distributes it to the appropriate components based on specific criteria or topics. It acts as a message broker, facilitating communication and coordination between different modules.
- **Publish-Subscribe Model:** The broker implements a publish-subscribe model where the filters act as publishers, sending processed data to specific topics or channels. Other components within the system, such as location visualization modules or data storage modules, act as subscribers, expressing interest in specific topics to receive relevant data. This decoupling allows for dynamic scaling, extensibility, and flexibility in handling different types of data and integrating new components.
- **Event-Driven Architecture:** The combination of both patterns enables an event-driven architecture within the system. As filters process data and generate events or updates, the broker triggers notifications or actions in other components subscribed to relevant topics. For example, when a device enters a specific area, the filter can generate an event that triggers a notification or updates the location visualization module.
- **Modular and Scalable Design:** By combining these patterns, the system achieves a modular design where each filter represents a separate component, ensuring encapsulation and reusability. The broker serves as the central communication component, allowing for scalability by facilitating the connection and coordination of multiple filters and other components.

4.1.2 Backend

The backend platform could be designed to be a versatile and scalable solution that can possibly cater to multiple tenants and applications. The platform offers extensive capabilities for data management across various areas, including floorplans, beacons, navigation, points of interest, offers, push messages, analytics, and asset tracking.

A variety of services are available in the system illustrated in figure 4.3, for

[0]`https://en.wikipedia.org/wiki/Pipeline_(software)`
[0]`https://en.wikipedia.org/wiki/Broker_pattern`

example, the geo server service facilitates users to upload and oversee floorplans of their buildings. The hardware manager service facilitates the management of beacons and their associated properties, such as Universally Unique Identifier (UUID), major, and minor values. Moreover, the navigation feature provides indoor navigation services to users, assisting them in navigating within indoor spaces. The points of interest feature allows users to create and manage specific points of interest, such as shops, restaurants, and restrooms. The platform also offers a zone alarm management service enabling users to define their assets' zones and alarms.

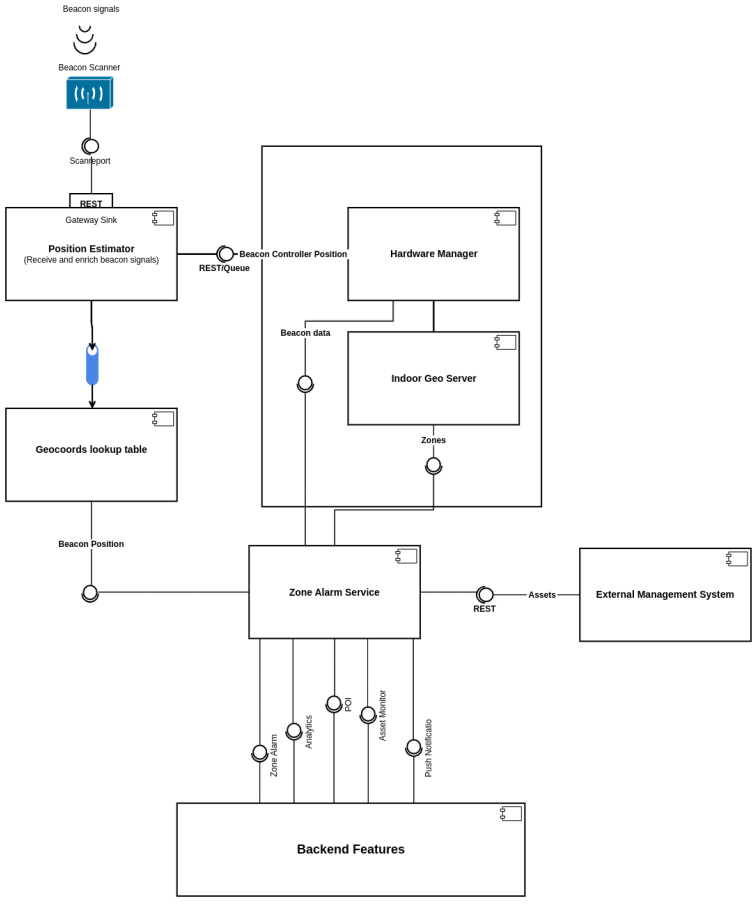


Figure 4.3: Backend Architecture

Users would be able to leverage the offers feature to create and manage promotional offers for their customers. Push messages enable sending push notifica-

tions to their customers, enhancing engagement and communication. The analytic service delivers insights into customer behavior and preferences, enabling informed decision-making. The asset tracking feature also allows users to track assets, such as shopping carts or wheelchairs. Furthermore, the Application Programming Interface (API) service provides users with access to backend data and services through a RESTful interface, facilitating integration with logging and visualization clients. Overall, the backend system can offer a wide range of services that can be based on many different IPS applications. Our proposed solution can be adopted in such a backend system and provide position estimation while taking advantage of its modularity.

4.2 Data Collection Framework

The collection of BLE beacon data is essential for the implementation of advanced filtering algorithms in indoor positioning systems. These datasets typically comprise information on received signal strength, which can exhibit high levels of fluctuation and noise. In the literature reviewed in chapter 2, a widely used approach involves the utilization of BLE beacon advertisements that are transmitted to BLE receivers, as these advertisements contain valuable information on the RSSI, a critical parameter in indoor positioning. The collected data is often used for various positioning techniques and correlate algorithms such as Kalman Filter or Particle Filter. Unfortunately, while numerous RSSI datasets exist for this purpose, they may lack key attributes required for effective data processing, and some datasets may not be suitable for integration into proposed filtering algorithms due to non-Gaussian noise characteristics and other issues. To address these limitations, we propose collecting a new dataset using BLE beacons and BLE receivers, which will ensure the suitability of the collected data for advanced filtering algorithms, thereby facilitating the development of more accurate indoor positioning systems.

Why Collect Own Data?

Publicly accessible datasets for BLE RSSI-based IPS present both advantages and disadvantages. While employing them, several shortcomings arise, impairing the suitability of these datasets for filtering algorithms. These limitations primarily stem from the datasets' nature and the associated challenges encountered during data collection and processing.

Firstly, publicly available datasets often consist of fingerprinting positioning data Torres-Sospedra et al. (2016); Hisham et al. (2022). While this data can be valuable for training phase purposes, it may not accurately represent real-time or apply to our system. A significant drawback of publicly available datasets lies in the insufficient or lack of essential ground truth data, environmental data, and reference distance data. Ground truth data serves as a benchmark for evaluating the accuracy of positioning algorithms. Its absence within the dataset jeopardizes the reliability and credibility of the results obtained through filtering algorithms. Furthermore, these datasets often fail to incorporate environmental factors such as obstructions, interference, or signal attenuation, significantly impacting RSSI readings. The absence of reference distance data also hinders

the calibration and fine-tuning of filtering algorithms, making them less effective in distance mapping for relative positioning scenarios.

Moreover, publicly available datasets, such as the specific type of BLE devices employed, may vary in their data collection properties. Factors such as the antenna design or BLE version of a given device might have an impact on the data collection process. In addition, the sampling frequency and the availability of positional information within the indoor environment are critical considerations for developing effective indoor positioning systems. Finally, these datasets can sometimes be too large, covering massive tracking areas along with a great amount of hybrid data Shu et al. (2021); Torres-Sospedra et al. (2016). While a large dataset may seem advantageous, it can pose challenges for computational complexity. Cherry-picking only BLE RSSI data from such extensive datasets can be overwhelming, increasing the processing time and eventually hindering real-time positioning applications.

4.2.1 System Setup

To collect data, we followed a three-stage approach: setting up all the equipment in a controlled indoor environment, preparing data collection tools, and recording data along with measurement variables. The collected dataset later also were used for obtaining the reference distance of anchor points to calibrate the distance mapping function. This new dataset allows us to ensure that the collected data is suitable for applying filtering algorithms and can effectively contribute to developing proposed indoor positioning systems.

Controlled Environment

For collecting BLE beacon, we have chosen Favendo office premises. since the environment there simulates a highly noisy area, however not in terms of structural attributes but with great signal traffic. The area consists of a large amount of signal-transmitting devices, not only BLE but many other wireless devices with different radio frequencies. The multitude of devices transmitting on various frequencies can create a noisy environment that makes it difficult for positioning systems to identify signals from the target device accurately Liu et al. (2013). Some devices may generate electromagnetic interference that can impact the quality of the received signal, particularly for devices operating in the same frequency range Piyare et al. (2018). The following figure 4.4 demonstrates the BLE beacon position, indoor area, and the path taken for collecting data. The environment is considered controlled as it was a key requirement for our study to conduct in a noisy environment. Furthermore, we have known reference points and detailed floor maps that can be used to calibrate and validate the dataset. Finally, our controlled movement helped to capture data in different orientations and physical positions.

[0]https://geojson.io/
[0]https://kontakt.io/
[0]https://www.raspberrypi.org/
[0]https://www.quuppa.com/

Hardware and Software Tools

Hardware equipment such as BLE beacons and beacon trackers are crucial for this procedure since these are the primary source and collector of the data. We carefully choose BLE 5.0 beacon from the manufacturer Kotakt.io, called Anchor Beacon 2. These beacons offer great features such as long battery life, robust wireless connectivity, easy deployment and management, etc. An actual ble beacon provides more realistic performance and behavior than a simulated smartphone beacon. For both scanning and tracking the BLE beacons, Bluetooth 5.0 capable wireless BLE transceivers were used, there are two different scanning devices, firstly a Dell Precision 3571 laptop, a Raspberry Pi 4 Single Board Computer. One critical hardware component we used is called Quuppa, A high-precision indoor positioning system that can reach centimeter-level accuracy and provides highly accurate location information that serves as ground truth for our data collection. This allowed us to benchmark the accuracy and reliability of regular BLE beacon systems.

4.5 presents some of the software tools we utilized for data collection and storage. Some tools not in the figure, such as geogson.io, were used to map the indoor environment and reference points with actual geo-coordinates. A message broker, redis, was used as a data communication hub and cache memory. Once the beacon data were collected, a data stream processor was used to ingest real-time data from BLE beacons and affixes data information. Afterward, Elasticsearch indexed and stored the BLE beacon data collected in real-time. This allows for fast and efficient searching and retrieval of the data. Finally, an open-source data visualization and exploration platform Kibana, were used to create real-time dashboards and visualizations of the BLE beacon data. This allows for easy monitoring of the indoor environment, such as identifying areas of high traffic or congestion, several statistics were created using the tool. Qu-

Figure 4.4: Indoor Environment

uppa also offers several applications to record, stream and store highly precise data.

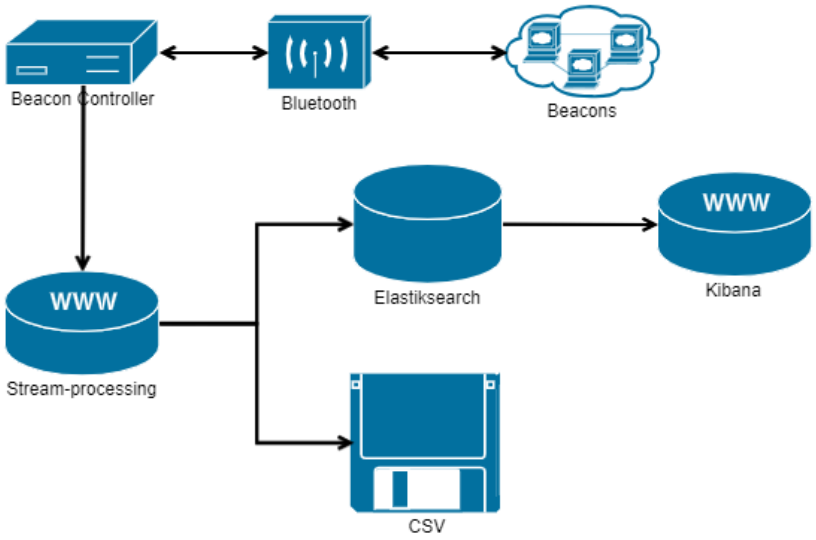


Figure 4.5: Data Collection Setup

Parameters

Important parameters needed to be taken into consideration for BLE beacon data collection process as follows

- **Beacon Configuration:** The configuration of the beacon itself is an important consideration for data collection. This includes parameters such as Transmission Power (TX), advertising interval, and beacon type (iBeacon, Eddystone, etc.). The best practices are a lower TX setting for smaller deployment areas or when the beacon range needs to be limited. This takes into account the impact of the TX setting on battery life and interference and helps ensure optimal signal quality for the specific use case. It is also important to balance the advertising interval with the beacons' desired range and signal quality and consider the impact on system performance. Usually, it comes down to application requirements to use a longer interval for applications that require longer battery life and a shorter interval for

[0]https://redis.io/
[0]https://www.elastic.co/
[0]https://www.elastic.co/kibana/
[0]https://en.wikipedia.org/wiki/Eddystone_(Google)

```
scanner_aggregation_timeWindow=1000
scanner_aggregation_reportDuration=1
scanner_heartbeat_interval=5
scanner_redis_host=localhost
scanner_redis_port=6379
```

Figure 4.6: Beacon Scanner Configuration

applications that require more frequent updates. For our particular use case, we employed iBeacon advertisement, with 500 ms of interval and -12 dbm TX power.

- **Beacon Scanner Configuration:** The configuration of the scanner used to collect beacon data is also important. This includes parameters such as scanning frequency, scan window, and message broker config. A sample configuration for the beacon scanner
- **Data Collection Settings:** This includes parameters such as the sampling rate, data storage format, and collecting duration. These parameters can affect the granularity and accuracy of the data collected and the amount of storage required. As configured in the beacon tracker, the sampling rate was 1000 mili-second, and data storage was made into JavaScript Object Notation (JSON) format due to its easy parsing and portability. We parsed this data within our system to obtain reference distance and stored that in CSV format, which is more suitable for Python scripting.

4.2.2 Dataset Properties

An example of a collected dataset contains the following properties:

- scannerID: a string representing the ID of the BLE scanner used to collect the data.
- windows: an array of objects representing time windows during which measurements were taken.
- measurements: an array of objects representing individual BLE beacon measurements.
- uuid: a string representing the UUID of the beacon.
- major: a string representing the major value of the beacon.
- minor: a string representing the minor value of the beacon.
- minRssi: an integer representing the minimum RSSI RSSI value of the beacon during the window.
- maxRssi: an integer representing the beacon's maximum RSSI value during the window.
- avgRssi: an integer representing the beacon's average RSSI value during the window.

```
"scannerID": "b827eb0f6dfe",
"windows": [
      {
        "measurements": [
              {
                "uuid": "626c756b-6969-2e63-6f6d-626561636f6e",
                "major": "3",
                "minor": "21772",
                "minRssi": -84,
                "maxRssi": -84,
                "avgRssi": -84,
                "txPower": -57,
                "lastEventTime": 1689204471832,
                "sampleCount": 1
              }
          ]
      }
 ]
```


Figure 4.7: BLE Beacon Dataset Properties

- txPower: an integer representing the beacon's TX.
- lastEventTime: a timestamp representing the time of the last event detected from the beacon during the window.
- sampleCount: an integer representing the number of samples taken for the beacon during the window.

As for the ground truth, while Quuppa positioning data provides 3-dimensional location data, it can still be used to provide ground truth for a 2-dimensional BLE positioning system by ignoring the Z-coordinate value. Quuppa packet contains the following information, among others, that are not necessary for our use case:

- tagId: a string representing the unique identifier of the tag.
- tagName: a string representing a human-readable name assigned to the tag.
- lastPacketTS: a timestamp representing the time at which the last packet was received from the tag.
- locationType: a string representing the type of location data being reported (in this case, "position").
- locationMovementStatus: a string representing the current movement status of the tag (in this case, "moving").
- locationRadius: a floating-point number representing the estimated radius of the tag's location.

```
{
  "tagId": "a4da22e4ee8c",
  "tagName": "Kamrul's Thesis",
  "lastPacketTS": 1679929471172,
  "locationType": "position",
  "locationMovementStatus": "moving",
  "locationRadius": 0.21,
  "location": [
    -20.72,
    -57.75,
    1.2
  ],
}
```


Figure 4.8: Quuppa as Ground Truth Data

- location: an array of three floating-point numbers representing the estimated x, y, and z coordinates of the tag's location.

Finally, the CSV files contain the measurements of signal strength (in dBm) at various distances (in meters) between a transmitter and a receiver. The first row lists the distances for which the signal strength measurements were taken, ranging from 0.1m to 10m. The rest of the row lists the corresponding RSS values, approximately 100 samples per distance.

4.3 Positioning System

The proposed positioning system is implemented in several components and technologies, the following UML diagram 4.9 shows the workflows of the proposed system. The following sections explain each primary component's purpose and the technologies used. The solution was developed using Python and NodeJS programming languages, the source code can be found in the GitLab repository.

4.3.1 Beacon Scanner

The beacon scanner is responsible for continuously scanning for these beacon signals in its range and collecting essential data, such as the beacon's UUID, major and minor numbers, and signal strength RSSI. This information is crucial for the beacon tracker or the positioning algorithm to calculate the relative distance between the beacon and the tracking device. The distance estimation is essential for accurate indoor positioning, as it enables the system to determine the position of the tracking device based on its proximity to multiple beacons. As mentioned earlier, we have used a Dell Precision 3571 laptop and a Raspberry

[0]https://en.wikipedia.org/wiki/Python_(programming_language)
[0]https://en.wikipedia.org/wiki/Node.js
[0]https://gitlab.rz.uni-bamberg.de/mobi/theses/2022-ma-kamrul-hasan

Pi 4 as a beacon tracker, both were running Linux Operating System (OS). The cross-functional beacon tracker application offers great portability and scalability since the application can be adopted in other OS or can be containerized.

The NodeJS application utilizes the Noble library for scanning iBeacon devices and publishes the collected data to a Redis server. It imports the required libraries, such as Noble and Redis, and defines an array of Media Access Control (MAC) addresses corresponding to the iBeacons of interest, in this case, the array serves as a beacon whitelist. The ***stateChange***event of the library is triggered when the Bluetooth adapter state changes. Upon detecting a ***poweredOn*** state, the program starts scanning for iBeacon devices using the ***startScanning()*** method of the Noble library. Additionally, the program sets up an event listener for the discover event, which is triggered when a new beacon is discovered. If the discovered beacon device has a MAC address that matches one of the iBeacon MAC addresses of interest, the program extracts the iBeacon data from the advertisement data and sends it to the Redis server using the ***publish()*** method of the Redis client.

4.3.2 Pre-processing

Kalman Filter for RSSI Noise

The Kalman filter serves as a state estimator, estimating an unobserved variable using noisy measurements. We applied KalmanJS to the 1D RSSI data, and the underlying Kalman filter can be understood through its two primary components: the transition model and the observation model shown in figure 4.10. The transition model accounts for the state's evolution over time, which can be mathematically expressed as:

Transition Model:

$$x_t = A_t x_{t-1} + B_t u_t + \epsilon_t \tag{4.1}$$

Here, x_t represents the state at time t, A_t is the state transition matrix, x_{t-1} is the state at the previous time step, B_t is the control input matrix (if applicable), u_t is the control input vector (if applicable), and ϵ_t represents the process noise.

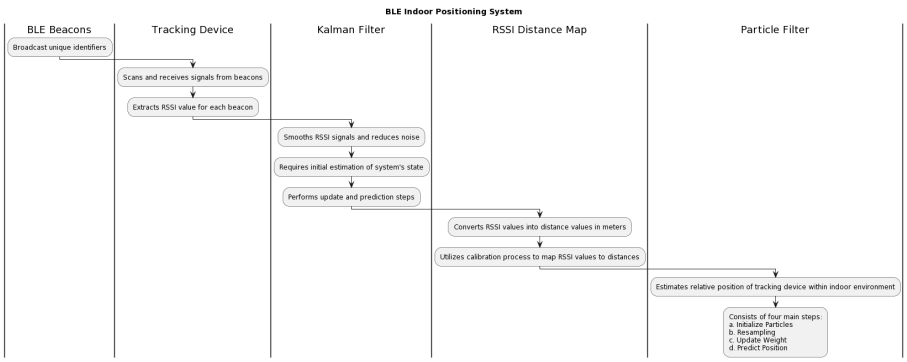


Figure 4.9: UML Diagram of the Proposed IPS

On the other hand, the observation model establishes the relationship between the state and the measurements, and it is expressed as:

Observation Model:

$$z_t = C_t x_t + \delta_t \tag{4.2}$$

Here, z_t represents the measurement at time t, C_t is the measurement matrix, x_t is the state at time t, and δ_t represents the measurement noise.

In the prediction step, the Kalman filter utilizes the previous state estimate to forecast the current state. The prediction step equations are as follows:

Prediction Step:

$$\bar{\mu}_t = \mu_{t-1}, \bar{\Sigma}_t = \Sigma_{t-1} + R_t \tag{4.3}$$

Here, $\bar{\mu}_t$ and $\bar{\Sigma}_t$ represent the predicted state estimate and the predicted state covariance at time t, respectively. μ_{t-1} and Σ_{t-1} are the state estimate and the state covariance at the previous time step, and R_t is the process noise covariance matrix.

Subsequently, the update step involves integrating the prediction with the current measurement to refine the state estimate. In this process, the Kalman gain plays a crucial role by balancing the influence of the prediction and the measurement based on their respective uncertainties. The updated step equations are as follows:

Kalman Gain:

$$K_t = \bar{\Sigma}_t(\bar{\Sigma}_t + Q_t)^{-1} \tag{4.4}$$

Update Step:

$$\mu_t = \bar{\mu}_t + K_t(z_t - \bar{\mu}_t), \Sigma_t = \bar{\Sigma}_t - (K_t\bar{\Sigma}_t) \tag{4.5}$$

Here, K_t represents the Kalman gain at time t, $\bar{\mu}_t$ and $\bar{\Sigma}_t$ are the predicted state estimate and the predicted state covariance at time t, respectively. z_t is the measurement at time t, and Q_t is the measurement noise covariance matrix. μ_t and Σ_t represent the updated state estimate and state covariance at time t, respectively, after incorporating the measurement. The complete processing steps are illustrated in the following UML diagram 4.10.

Distance Model Calibration

We employed the Log-Distance path loss model to estimate the distance from the RSSI value. The parameters for the model need to be determined through a calibration campaign. During this campaign, the RSSI value is measured at different known distances between the transmitter and receiver devices, as discussed in the Dataset section. By fitting the Log-Distance path loss model to these measurements using a regression technique, Our calibration script calculates the Log-Distance path loss model parameters using the calibration data collected from a set of beacons. Firstly, we define the list of beacon names, a list of reference distances, and the file names for the CSV files containing the RSSI measurements we collected. The `path_loss_model()` function defines the Log-Distance path loss model, which takes in the distance between the transmitter and receiver devices, the reference distance d_0, the RSSI value at the reference

[0] `https://en.wikipedia.org/wiki/Log-distance_path_loss_model`

distance rss_{d_0}, and the signal propagation exponent γ, which we explain in the position estimation section. The function returns the estimated RSSI value at the given distance.

The `fit_model()` function takes in a beacon name and fits the Log-Distance path loss model to the RSSI measurements for that beacon using the minimize function from scipy. The function returns the best-fit values for the parameters d_0, rss_{d_0}, and γ. The primary function of the script loops over each beacon, calls the `fit_model()` function for that beacon and stores the determined path loss exponent γ in a dictionary called `path_loss_exponents`. Finally, it provides the path loss exponent for each beacon.

4.3.3 Position Estimation

In order to implement the proposed positioning system, we first needed to configure and provide calibrated parameters to apply the position estimation algorithms to real-time data. The primary functionality can be divided into two parts: the Log-Distance path loss model and the Particle Filter algorithm.

From RSSI to Meters

The relationship between RSSI and distance can be quite complex and depends on various factors such as signal propagation characteristics, environment, and

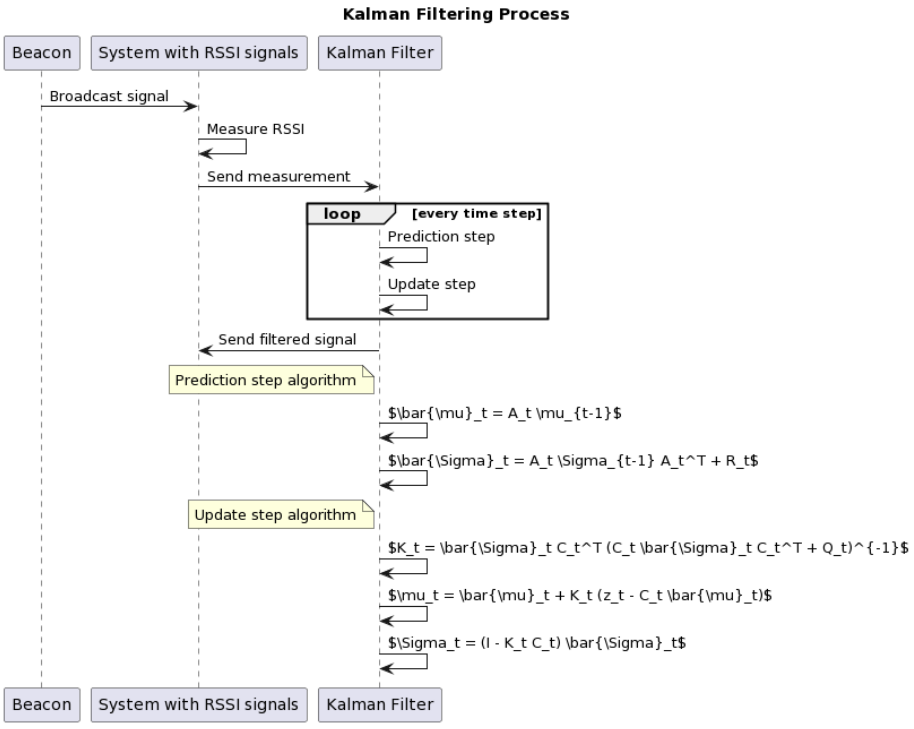


Figure 4.10: Kalman Filter RSSI Processing

hardware-specific parameters. The Log-distance path loss model is a common approach, which relates RSSI to distance through an exponential decay function. The Log-Distance path loss model is mathematically expressed as follows:

$$d = 10^{\frac{rss_d0 - rss}{10 \cdot \gamma}} \times d_0$$

(4.6)
Where:

- rss: The RSSI at the receiver (user device).
- $rssd0$: The reference RSSI value at a reference distance (d_0) from the transmitter. It represents the RSSI at a known distance and is used to calibrate the model.
- d: The distance between the transmitter (BLE beacon) and the receiver (user device).
- d_0: The reference distance, usually set to 1 meter, at which $RSSI_0$ is measured.
- $gamma$: The path loss exponent characterizes the rate at which the signal strength attenuates with distance. It depends on the environment and the radio frequency used.

To utilize the Log-Distance path loss model for indoor positioning, the model needs to be calibrated for the specific environment and BLE beacon setup, which we already explained. The main process begins with parameters and constants such as the dimensions of the indoor space, window size for smoothing RSSI measurements, and the MAC addresses and physical locations of the BLE beacons. The Calibrator class calibrates the received RSSI values to distance estimates using parameters obtained from the calibration process d0, rss_d0, and gamma. Finally, the method takes an array of received RSSI values (rss) as input and returns distance estimates (in meters) for each beacon, calculated using the calibration parameters.

Particle Filter Positioning

The Particle Filter utilizes a set of particles to represent possible user positions. Each particle has an associated weight, indicating its likelihood of being in the true position. At the start of the localization process, the ***ParticleFilter*** class generates an initial belief of the user's location within the indoor space. The room dimensions, ***roomWidth***, and ***roomLength*** are considered to define the boundaries for particle generation. The filtered distance estimates and beacon locations are fed into the Particle Filter class. The algorithm then updates the weights of particles based on their likelihood concerning the filtered distances. Particles that better align with the filtered distance estimates receive higher weights. Figure 4.11 goes through each step of the algorithm.

Initialization: A function initializes the number of particles, ***nParticles***, and the dimensions of the indoor space, ***roomWidth*** and ***roomLength***. Then another function randomly generates the initial set of particles within the specified room dimensions. It also initializes the weight of each particle to be equal and returns both the particles and their weights.

Prediction: The ***prediction*** function responsible for calculating the probability of observing the measured distance from a given particle location to a beacon assumes a normal distribution with mean and variance.

Weight Calculation: Next step is to update the weight of each particle based on the measured distances to the beacons and their locations, and the variance calculates the likelihood of each particle location given the measured distances and beacon locations. It also performs a boundary check to ensure particles are within the room dimensions. Finally, it normalizes the weights so that they sum up to 1.

Re-sampling: ***resampling*** the particles based on their weights select particles to create a new set of particles. It then adds a random offset to each particle in the x and y direction, sampled from a normal distribution with mean 0 and variance. Finally, it resets the weight of each particle to be equal.

Target State Estimation: Finally, to predict the user's location, calculates the weighted average of the particle locations, where the weight of each particle is given by its weight calculated in the ***weight_update*** function.
The system operates in real-time, continuously receiving iBeacon data and updating the particle filter with each new data point. As more data points are

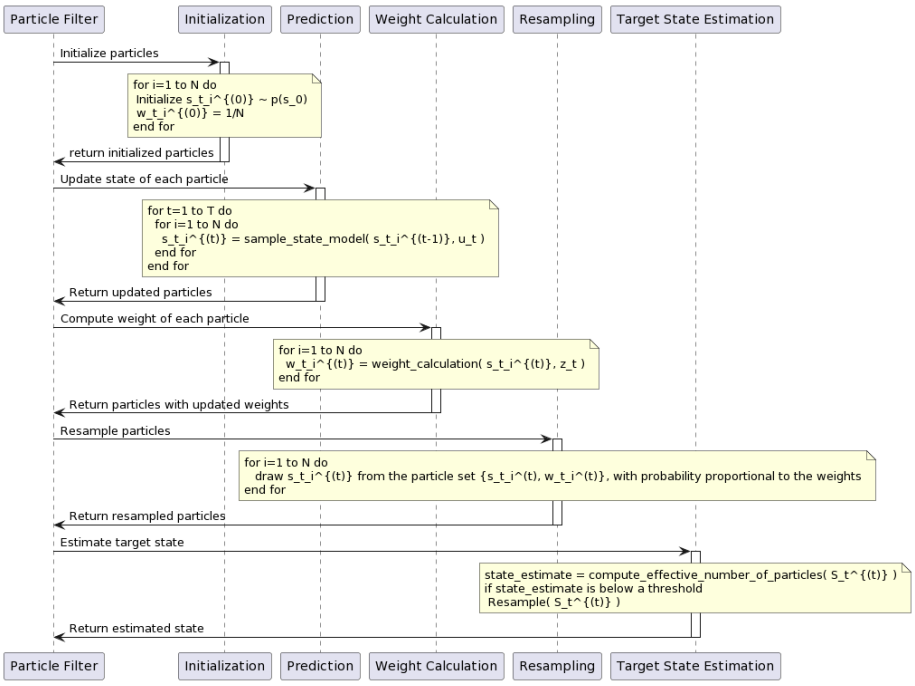


Figure 4.11: Particle Filter Position Estimation

collected, the accuracy of the predicted indoor position improves due to the iterative updates of the Particle Filter.

4.4 Visualization

To visualize the position in real-time, we developed a virtual room environment that shows the position information and provides the beacon with the real-time location. We used the Matplotlib library to create plots for visualizing the results of the Particle Filter algorithm for indoor localization using BLE beacon data. The implementation contains functions for initializing the plot, plotting the beacons, particles, predicted location, reference circles, and real location, updating the plot, and saving the plot. A depiction can be found in figure 4.12.

To initiate the visualization, the system initializes a figure and axis object using the Matplotlib library, setting the width and length of the plot to provide an appropriate canvas for displaying the positioning elements. The first step in plotting the visualization is to depict the beacon locations as blue circles. These beacons act as reference points within the indoor space and serve as essential anchors for positioning calculations. Next, the Particle Filter generates a set of particles, each representing a potential user location. These particles are displayed as red crosses on the plot, estimating the user's position based on the received beacon data.

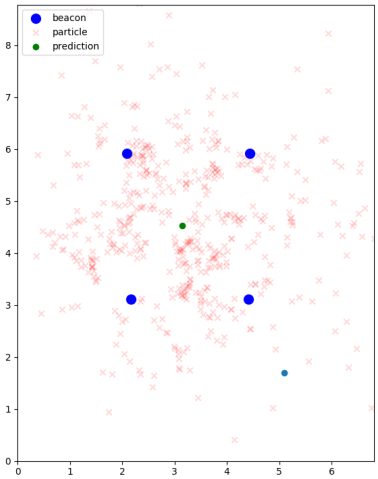


Figure 4.12: Visualization of the System

As the Particle Filter predicts the user's indoor location, the predicted position

[0] `https://matplotlib.org/`

is illustrated on the plot as a green circle. This predicted location represents the system's best estimate of the user's current position within the indoor environment.To better understand the positioning accuracy, reference circles are plotted around each beacon, visualizing the measured distances between the user's device and each beacon. These circles aid in assessing the system's performance by comparing the measured distances with the predicted position. If available, the user's real location can be optionally displayed on the plot as a black square, acting as a reference point for evaluating the accuracy of the predicted position.

The plot dynamically adjusts to reflect the changing positions as the system continuously receives new beacon data and updates the Particle Filter. The update function is responsible for clearing the previous plot and plotting the most recent beacon locations, particle positions, predicted location, reference circles, and real location if provided. Additionally, the axis limits are adjusted to accommodate the changing positions, and a legend is included to identify the different elements displayed on the plot.

Chapter 5

Realisation and Evaluation

In this chapter, we present the outcomes obtained by applying the methods explained in earlier sections and assess them using established evaluation criteria. Firstly, to mitigate the fluctuations in RSSI, we explore the efficacy of the Kalman filter in effectively reducing signal noise. Consequently, we examine the impact of this noise reduction on the accuracy of the Log-distance path loss model. Lastly, we evaluate the system's performance by comparing the estimated positions with the ground truth data. Upon reviewing the results of our evaluation, we revisit our initial research question to determine whether the study has effectively provided an answer.

5.1 Noise Reduction

To determine the effect of the Kalman filter on noisy data, the filter was employed in a simplified model, assuming a static system with no movement of the target beacon. The application can process exponential data, considering the nature of real-time IPS. The RSSI values were sampled at a 1Hz sample rate, while other configurations remained the same as the previous configuration. The collected dataset includes both raw RSSI values, obtained directly from the BLE beacon, and corresponding filtered RSSI values achieved through the beacon scanner application that utilizes the KalmanJS library.

To visually assess the impact of the Kalman filter on noise reduction, we plotted 5.1 the raw and filtered RSSI values over time. The noise level visualization clearly showcased the substantial fluctuations present in the raw RSSI data. In contrast, the filtered RSSI values exhibited a notably smoother trajectory, effectively mitigating the noise and resulting in a more stable signal representation.The prediction step of the filter was based on the previous certainty and the process noise caused by the system itself. The update step used the Kalman gain as a weighting function between the estimate's certainty and the measurement's certainty, influenced by the measurement noise. The larger the Kalman gain, the more the measurement was integrated into the state estimate.
The filter also utilizes parameters for measurement noise covariance and the

[0]https://en.wikipedia.org/wiki/Signal-to-noise_ratio
[0]https://en.wikipedia.org/wiki/Root-mean-square_deviation

process noise covariance denoted by $\boldsymbol{R}$ and $\boldsymbol{Q}$. By tuning the noise levels R and Q, the Kalman filter's behavior can be tailored to best suit the specific characteristics of the exponential RSSI data, striking a balance between smoothing the RSSI and accurately tracking variations in the underlying exponential pattern. We also calculated the SNR for both the raw and filtered RSSI data and evaluated the improvement in SNR due to the filter. The improvement is evaluated by dividing the SNR for the filtered data by the SNR for the raw data. The SNR for for raw and filtered RSSI data are calculated respectively 1.94 and 3.48, making the improvement by a factor of 1.79 times, or 79.27%.

5.2 Distance Accuracy

the Kalman filter's noise reduction effect on RSSI data significantly enhances the Log-Distance Path Loss Model, making it a powerful framework for accurately predicting wireless signal strength in diverse environments. The Root Mean Square Error (RMSE) was used to compare the distance estimation accuracy from raw and filtered RSSI signals. To calculate RMSE, a set of ground-truth distances between the transmitter and receiver is required for a set of reference points. As we considered Quuppa as ground truth data, therefore an empirical Log-Distance Path Loss Model function was used in the Quuppa system to calculate the reference points. The RMSE measures the average distance between the estimated distances and the ground-truth distances. The scatter plot 5.3 below shows the ground truth distances are plotted against the distance estimates for both raw and filtered RSSI data. The points close to the gray dashed line representing the ground truth indicate accurate distance estimates.

Additionally, the following plot shows the RMSE as a function of the reference points, and demonstrates the difference between the ground truth and the distance estimates is visualized with error bars. The error bars represent each dataset's RMSE. Which is 0.84 meters for raw RSSI data and 0.46 meters for

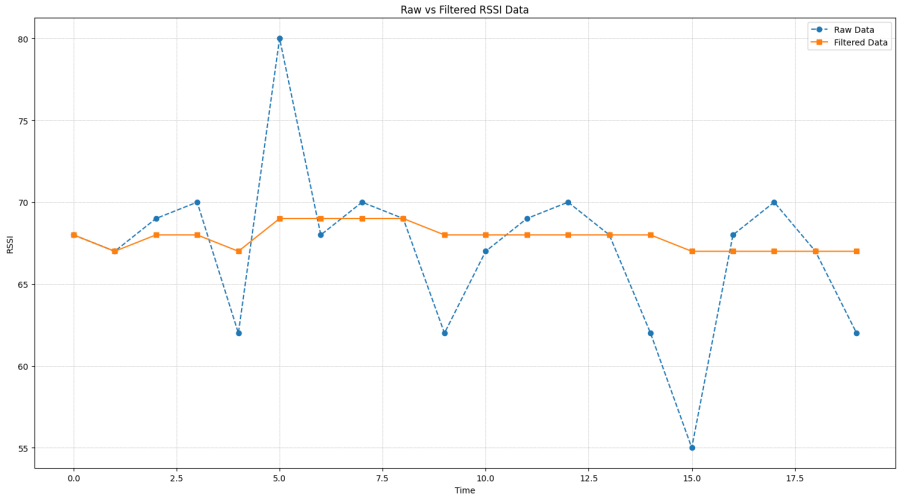


Figure 5.1: Raw vs KF Filtered RSSI Data

filtered data, here, smaller error bars suggest better distance estimation accuracy.

5.3 Performance Analysis

To evaluate the performance of the proposed particle filter-based IPS, we opted to calculate the error between the estimated location and the real location of the device in the 2D space. The error is typically defined as the Euclidean distance

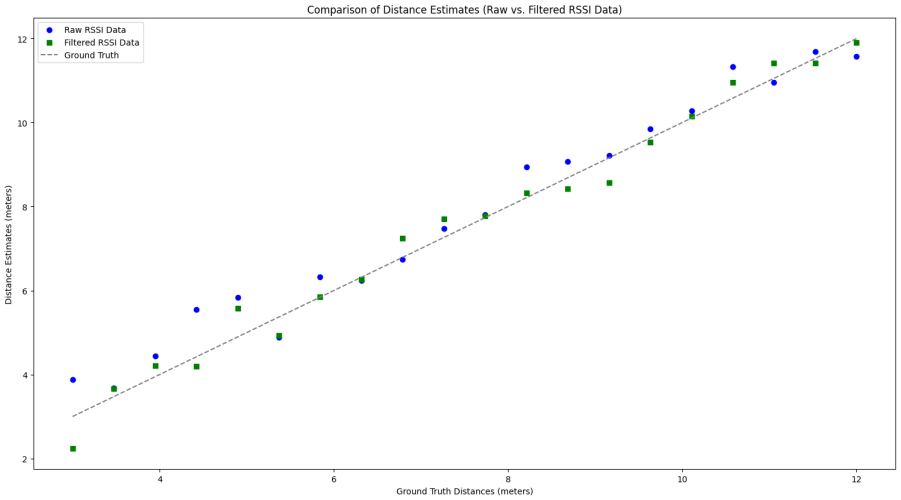


Figure 5.2: Comparison of Distance Estimates

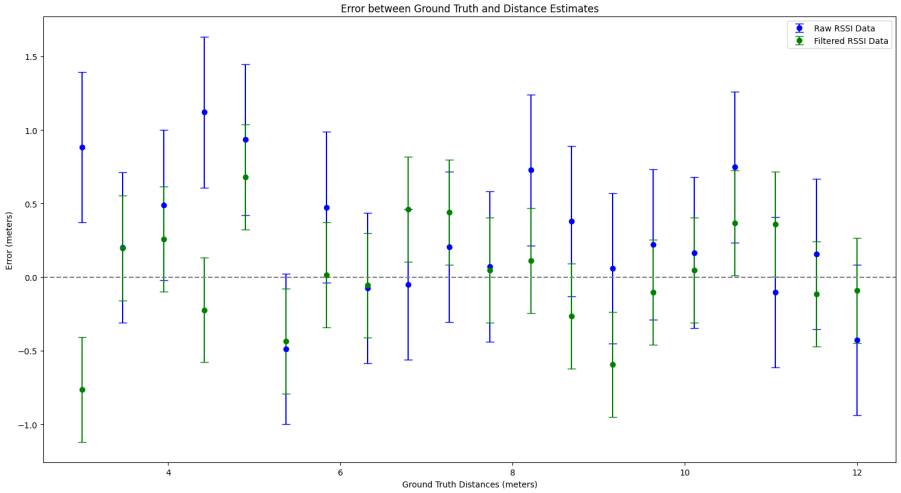


Figure 5.3: Error between Ground Truth and Distance Estimates

between the estimated position and the true position of the beacon. Again we utilized the Quuppa system to be the true position of the beacon.

$$error = \sqrt{(x_{est} - x_{true})^2 + (y_{est} - y_{true})^2} \tag{5.1}$$

The assessment is performed for different true positions for the beacon, specified in a list. For each true position, the script executes the proposed particle filter position system iteratively for multiple time slots. At each time slot, the particle filter uses the current RSS measurements from the beacons to estimate the user's location. The localization error method computes the Euclidean distance between each time slot's estimated and corresponding real positions. The error values are then appended to the error list, creating a record of the algorithm's performance over time. Once the particle filter has processed all time slots, the code generates the following plot 5.4 to visualize the localization error over time. The plot displays the error values on the y-axis and the time slots on the x-axis. The x-axis represents the progression of time during the localization process, while the y-axis indicates the magnitude of the localization error at each time slot.

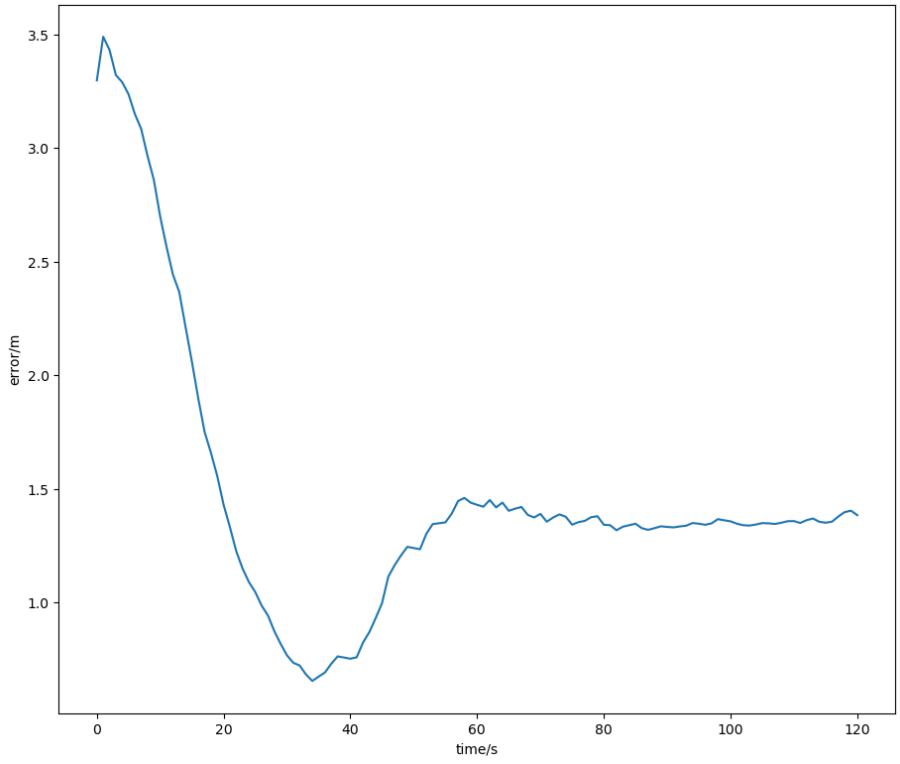
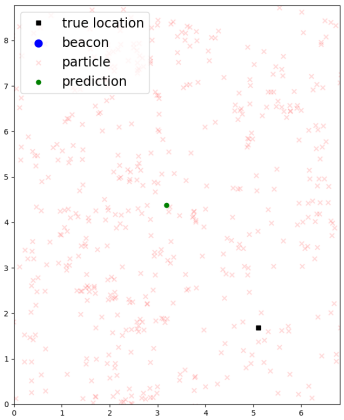


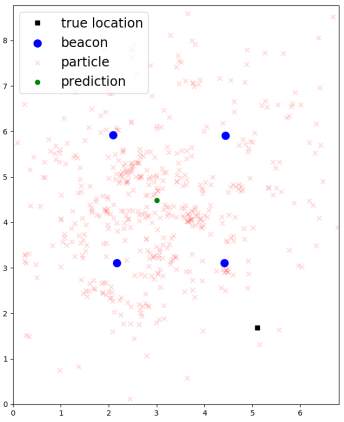
Figure 5.4: Particle Filter Prediction Error

[0]https://en.wikipedia.org/wiki/Euclidean_distance

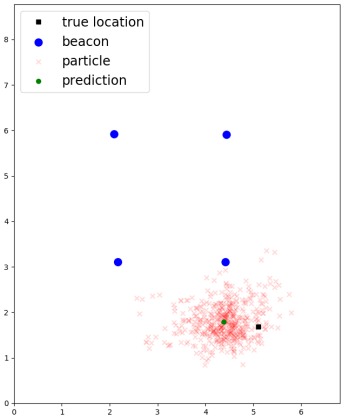
The resulting plot shows a decreasing trend, indicating that the particle filter's estimates converge toward actual positions over time. A decreasing error trend signifies improved accuracy and successful localization.



(a) Initial Belief Step



(b) Updated Weight and Resampled



(c) Converging towards True Position

Figure 5.5: Location Prediction by Particle Filter

Finally, we show the real-time location prediction utilizing the visualization plot5.5 described in the previous chapter. The first plot generated by the visualization class shows the initial generation of particles, with each particle represented as a red x marker. The second plot shows the improved resampled particles after the updated weight. The particles are represented as red x markers with reduced opacity, and a green circle marker represents the predicted location. The third and final plot shows the predicted position converging near the device's true location. In the plots, the particles are represented as a red x marker. The beacon location is represented with blue circles, a green circle

marker represents the predicted location, and a black square marker represents the device's true location. By calculating the Euclidean distance between the estimated location derived from the beacon and the true location, our system's location estimation has achieved sub-1.5-meter accuracy. In comparison to traditional IPS claims, this result shows a significant improvement.

The study was designed to address the following research question

Research Question: What filtration method can reduce BLE beacon signal noise, consequently improving the accuracy in a highly reflecting and noisy area?

To answer the research question posed above

Answer: The Kalman filter and the Particle filter, both recursive Bayesian filters, can be employed independently or synergistically to mitigate the noise in BLE beacon signals. As a result, this aids in enhancing the precision of location estimations in areas characterized by high signal reflection and noise.

Chapter 6

Conclusions and Future Work

In conclusion, we investigated the use of advanced filtering techniques, namely the particle filter and the Kalman filter, to enhance the accuracy and reliability of BLE RSSI-based IPS. A dataset of RSSI measurements and ground truth data collected from relatively noisy environments was used in the study. To collect the data, a three-stage approach was utilized. First, all the necessary equipment was set up in a controlled indoor environment. Second, data collection tools were prepared. Finally, data along with measurement variables were recorded.
The study includes a comprehensive examination of system requirements. A systematic approach was implemented, utilizing various techniques to ensure a clear comprehension of the study's scope and objectives.
The proposed system design includes three key components: a beacon scanner, pre-processing tools, and a position estimation module. The implementation uses a Linux device for beacon scanning and Python scripts for pre-processing and position estimation.
The study offers not only theoretical insights but also practical implementation that could be reproduced, proving its practical applicability. The system's performance was evaluated based on noise reduction, distance accuracy, and overall performance. The results indicate a substantial improvement in the evaluation metrics, thereby validating the proposed solution and answering the research question. The findings and methods presented in this study are expected to be of high relevance to future researchers and the broader scientific community involved in indoor positioning systems.

Limitations

While our proposed system addresses some limitations, in BLE based IPS, it is crucial to acknowledge that the system itself is not without its limitations. One potential drawback is that the datasets we collected may not fully encompass all scenarios where indoor positioning systems are utilized. Different scenarios can introduce factors. Increase the complexity of implementing the system.
In some situations, the proposed indoor positioning systems may face limitations when the device is in motion or frequently changing direction. This limitation

arises from the fact that a motion model integration is not incorporated into the particle filter-based IPS. Furthermore, although the selected filtering techniques are powerful, they also have limitations and assumptions that may not always be applicable in every scenario.

Future Works

One area for future work is the integration of a motion model into the particle filter-based IPS Rad and Mosavi (2016). A motion model could be used to predict the likely movement of the mobile device based on its previous location estimates and other sensor data, such as accelerometer or gyroscope readings. By incorporating a motion model, the particle filter-based IPS could improve its performance in situations where the device is moving rapidly or changing direction frequently.

Another area for future work is the use of an exponential moving average to smooth the particle weights over time. This could help to reduce the effects of noise and improve the accuracy of the location estimates. The exponential moving average could be updated at each time step using the most recent particle weights and a smoothing factor, allowing for dynamic tuning of the filter Torrieri (1984). In addition, the particle filter-based IPS could be integrated with other IPS techniques such as trilateration or multilateration Lane et al. (2010). We could potentially improve the accuracy and robustness of the location estimates. Nonetheless, incorporating these methods into pre-existing systems could potentially lead to new complications and discrepancies. Consequently, in such situations, it becomes necessary to embrace a newly designed system architecture.

[0]`http://stefanosnikolaidis.net/course-files/CS545/Lecture8.pdf`

Abbreviations

IPS Indoor Positioning Systems

BLE Bluetooth Low Energy

RSSI Received Signal Strength Indicator

COTS Commercial off-the-shelf

UWB Ultra-Wideband

APs Acess Points

LBS Location-based Services

WLAN Wireless Local Area Networ

RSS Received Signal Strength

RPs Reference Points

IMU Inertial Measurement Unit

RNN Recurrent Neural Networks

JSON JavaScript Object Notation

CSV Comma-separated Values

API Application Programming Interface

TX Transmission Power

UML Unified Modeling Language

UUID Universally Unique Identifier

MAC Media Access Control

OS Operating System

RMSE Root Mean Square Error

Bibliography

(2016). Ipin 2016 : Indoor positioning and indoor navigation conference. `https://ipin2016.web.uah.es/index.php`. (Accessed on 06/26/2023).

Abade, B., A. D. P. C. M. (2018). A non-intrusive approach for indoor occupancy detection in smart environments. *Sensors*, 18:3953.

Agrawal, D. (2020). Mind the gap: Real-time decentralized distance estimation using ultrasound and bluetooth across multiple smartphones.

Alshami, I. H., A. N. H. S. S. F.-F. (2017). Adaptive indoor positioning model based on wlan-fingerprinting for dynamic and multi-floor environments. *Sensors*, 17:1789.

Barsocchi, P., C. S. M. F. P. F. (2018). Indoor bluetooth low energy dataset for localization, tracking, occupancy, and social interaction. *Sensors*, 18:4462.

Bass, L., Clements, P., and Kazman, R. (2012). *Software Architecture in Practice*. Addison-Wesley Professional.

Bi, J., Wang, Y., Yu, B., Cao, H., Shi, T., and Huang, L. Q. (2022). Supplementary open dataset for WiFi indoor localization based on received signal strength. *Satellite Navigation*, 3(1).

Cetin, M., Ozdemir, S., and Yigit, T. (2016). Indoor positioning using kalman filter and multi-trilateration techniques. In *2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, pages 2177–2180. IEEE.

Chatzimichail, A., Tsanousa, A., Meditskos, G., Vrochidis, S., and Kompatsiaris, I. (2020). *RSSI Fingerprinting Techniques for Indoor Localization Datasets*, pages 468–479.

Chen, L., Ahriz, I., and Le Ruyet, D. (2020). Aoa-aware probabilistic indoor location fingerprinting using channel state information. *IEEE Internet of Things Journal*, 7(11):10868–10883.

Chen, W. and Wu, X. (2016). Indoor positioning and navigation system based on kalman filter algorithm. *Journal of Physics: Conference Series*, 741(1):012127.

Clegg, D. and Barker, R. (1994). *CASE Method Fast-track*. Addison Wesley Longman.

Cohn, M. (2004). *User Stories Applied: For Agile Software Development*. Pearson Education.

Dardenne, A., van Lamsweerde, A., and Fickas, S. (1993). Goal-directed requirements acquisition. *Science of Computer Programming*, 20(1-2):3–50.

Dong, H. (2020). Modeling and simulation of english speech rationality optimization recognition based on improved particle filter algorithm. *Complexity*, 2020:1–10.

Doucet, A., De Freitas, N., and Gordon, N. (2001). *Sequential Monte Carlo methods in practice*. Springer, New York.

Eyng, A. C., Rayel, O. K., Oroski, E., and Rebelatto, J. L. (2020). Kalman filtering-aided hybrid indoor positioning system with fingerprinting and multilateration. In *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, pages 1–5.

Faragher, R. and Harle, R. (2015). Location fingerprinting with bluetooth low energy beacons. *IEEE Journal on Selected Areas in Communications*, 33(11):2418–2428.

Fronckova, K. and Prazak, P. (2020). Possibilities of using kalman filters in indoor localization. *Mathematics*, 8(9).

Gomez, C., Oller, J., and Paradells, J. (2012). Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology. *Sensors*, 12(9):11734–11753.

Gortázar, F. and Kurpas, P. (2016). Bluetooth low energy (ble) indoor positioning system for smartphones. *IEEE Access*, 4:2547–2558.

Hidayat, Z., B. R. S. B. D. (2011). Decentralized kalman filter comparison for distributed-parameter systems: a case study for a 1d heat conduction process. *Etfa2011*.

Hisham, A. N. N., Ng, Y. H., Tan, C. K., and Chieng, D. (2022). Hybrid Wi-Fi and BLE Fingerprinting Dataset for Multi-Floor Indoor Environments with Different Layouts. *Data*, 7(11):156.

Hoang, M. H., Y. B. D. X. L. T.-W. R. R. K. K. (2019). Recurrent neural networks for accurate rssi indoor localization. *IEEE Internet Things J.*, 6:10639–10651.

Huang, S.-Y., Huang, H.-Y., Chong, H. Y., Jiang, J.-B., Leu, J.-S., and Vítek, S. (2022). A directional particle filter-based multi-floor indoor positioning system. *IEEE Access*, 10:116317–116325.

Jose, A., Rishikesh, P., and Shaju, S. (2023). *Mitigation of RSSI Variations Using Frequency Analysis and Kalman Filtering*, pages 227–239.

Julier, S. J. and Uhlmann, J. K. (2004). Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422.

Kang, W., H. Y. (2015). Smartpdr: Smartphone-based pedestrian dead reckoning for indoor localization. *IEEE Sensors J.*, 15:2906–2916.

Karlsson, J. and Ryan, K. (1997). A cost-value approach for prioritizing requirements. *IEEE Software*, 14(5):67–74.

Kim, M., Han, D., and Rhee, J. K. K. (2020). Machine learning for practical localization system using multiview csi. *IEEE Access*, 8:184575–184584.

Kotonya, G. and Sommerville, I. (1998). *Requirements engineering: Processes and techniques*. John Wiley & Sons.

Kumar, A. and Kumar, P. (2012). Requirements elicitation techniques–a systematic literature review. *International Journal of Computer Science Issues (IJCSI)*, 9(1):169–177.

Lane, N. D., Miluzzo, E., Lu, H., Peebles, D., Choudhury, T., and Campbell, A. T. (2010). A survey of mobile phone sensing. *IEEE Communications Magazine*, 48(9):140–150.

Lauesen, S. (2002). *Software requirements: Styles and techniques*. Addison-Wesley Professional.

Lee, S., Cho, B., Koo, B., Ryu, S., Choi, J., and Kim, S. (2015). Kalman filter-based indoor position tracking with self-calibration for rss variation mitigation. *International Journal of Distributed Sensor Networks*, 11(8):674635.

Lee, S.-H., Lim, I.-K., and Lee, J. S. (2016). Method for Improving Indoor Positioning Accuracy Using Extended Kalman Filter. *Mobile Information Systems*, 2016:1–15.

Li, B., Hao, Z., and Dang, X. (2019). An indoor location algorithm based on Kalman filter fusion of ultra-wide band and inertial measurement unit. *AIP Advances*, 9(8).

Liu, H., Darabi, H., Banerjee, P., and Liu, J. (2007). Survey of wireless indoor positioning techniques and systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(6):1067–1080.

Liu, R., Chen, W., and Li, X. (2013). A survey of wireless indoor positioning techniques and systems. *IEEE Communications Surveys & Tutorials*, 15(3):1347–1363.

Lohan, E. S., Torres-Sospedra, J., Leppäkoski, H., Richter, P., Peng, Z., and Huerta, J. (2017). Wi-fi crowdsourced fingerprinting dataset for indoor positioning. *Data*, 2(4).

Mackey, A., Spachos, P., and Plataniotis, K. N. (2020). Smart parking system based on bluetooth low energy beacons with particle filtering. *IEEE Systems Journal*, pages 1–12.

Malekzadeh, P., Mohammadi, A., Barbulescu, M., and Plataniotis, K. (2019). Stupefy: Set-valued box particle filtering for bluetooth low energy-based indoor localization. *IEEE Signal Processing Letters*, PP:1–1.

Mendoza-Silva, G. M., Matey-Sanz, M., Torres-Sospedra, J., and Huerta, J. (2019). BLE RSS Measurements Dataset for Research on Accurate Indoor Positioning. *Data*, 4(1):12.

Mendoza-Silva, G. M., Richter, P., Torres-Sospedra, J., Lohan, E. S., and Huerta, J. (2018). Long-term wifi fingerprinting dataset for research on robust indoor positioning. *Data*, 3(1).

Mohammadi, M. and Al-Fuqaha, A. (2018). BLE RSSI Dataset for Indoor localization and Navigation. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C54G80.

Mohammadi, M., Al-Fuqaha, A., Guizani, M., and Oh, J.-S. (2018). Semisupervised deep reinforcement learning in support of iot and smart city services. *IEEE Internet of Things Journal*, 5(2):624–635.

Nikitin, A., L. C. C. G. K. P. Z.-Y. D. (2017). Indoor localization accuracy estimation from fingerprint data.

Piyare, R., Bhatia, S., and Sarin, V. (2018). Impact of electromagnetic interference on wireless communication systems. *International Journal of Advanced Research in Computer Science and Software Engineering*, 8(3):46–50.

Qureshi, U. M., U. Z. H. G. P. (2019). Evaluating the implications of varying bluetooth low energy (ble) transmission power levels on wireless indoor localization accuracy and precision. *Sensors*, 19:3282.

Rad, A. B. and Mosavi, M. R. (2016). A motion model-based particle filter for indoor positioning. In *2016 24th Iranian Conference on Electrical Engineering (ICEE)*, pages 1031–1036.

Shen, Y., Hwang, B., and Jeong, J. P. (2020). Particle filtering-based indoor positioning system for beacon tag tracking. *IEEE Access*, 8:226445–226460.

Shu, Y., Xu, Q., Liu, J., Choudhury, R. R., Trigoni, N., and Bahl, V. (2021). Indoor location competition 2.0 dataset.

Sikeridis, D., Papapanagiotou, I., and Devetsikiotis, M. (2018). Blebeacon: A real-subject trial dataset from mobile bluetooth low energy beacons. *CoRR*, abs/1802.08782.

Sikeridis, D., Papapanagiotou, I., and Devetsikiotis, M. (2022). Crawdad unm/blebeacon.

Sommerville, I. (2016). *Software Engineering*. Pearson.

Torres-Sospedra, J., Montoliu, R., Martínez-Usó, A., Avariento, J., Arnau, T., Benedito-Bordonau, M., and Huerta, J. (2014). Ujiindoorloc: A new multi-building and multi-floor database for wlan fingerprint-based indoor localization problems.

Torres-Sospedra, J., Montoliu, R., Mendoza-Silva, G. M., Belmonte, O., Rambla, D., and Huerta, J. (2016). Providing Databases for Different Indoor Positioning Technologies: Pros and Cons of Magnetic Field and Wi-Fi Based Positioning. *Mobile Information Systems*, 2016:1–22.

Torrieri, D. J. (1984). Statistical theory of passive location systems. *IEEE Transactions on Aerospace and Electronic Systems*, 20(2):183–198.

van Lamsweerde, A. (2009). *Requirements engineering: From system goals to UML models to software specifications*. John Wiley & Sons.

Wan, J. and Balakrishnan, H. (2002). Using dynamic time warping to find patterns in time series. In *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, pages 359–370.

Welch, G. and Bishop, G. (2006). An introduction to the kalman filter. Technical report, University of North Carolina at Chapel Hill.

Wu, C., Zhang, F., Wang, B., and Liu, K. J. R. (2020). Easitrack: Decimeter-level indoor tracking with graph-based particle filtering. *IEEE Internet of Things Journal*, 7:2397–2411.

Xu, Y., J. A. (2019). Particle filters for inference of high-dimensional multivariate stochastic volatility models with cross-leverage effects. *Foundations of Data Science*, 0:0–0.

Zhang, C., T. A. M. P. G. (2016). Attitude estimation with feedback particle filter. *2016 IEEE 55th Conference on Decision and Control (CDC)*.

Zhou, B. and Chen, Q. (2017). Device-free localization: A physical model and algorithm. *IEEE Internet of Things Journal*, 4(6):2093–2103.

Declaration of Authorship

In accordance with § 9 Para. 12 APO, I hereby declare that I wrote the preceding master's thesis independently and did not use any sources or aids other than those indicated. Furthermore, I declare that the digital version corresponds without exception to content and wording to the printed copy of the master's thesis and that I am aware that this digital version may be subjected to a software-aided, anonymized plagiarism inspection.

Bamberg, Date _____ _____

Kamrul Hasan