Master Thesis

Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Radio Engineering

Indoor Bluetooth localisation

Bc. Jakub Horáček

Supervisor: Prof. Jan Sýkora
Field of study: Open Electronic Systems
Subfield: Communications and Signal Processing
January 2023

ČVUT
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Horáček** Jméno: **Jakub** Osobní číslo: **465990**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra radioelektroniky**

Studijní program: **Otevřené elektronické systémy**

Studijní obor: **Komunikace a zpracování signálu**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Indoor Bluetooth lokalizace

Název diplomové práce anglicky:

Indoor Bluetooth localisation

Pokyny pro vypracování:

Cílem práce je navrhnou lokalizační systém pro vnitřní prostory budov založený na Bluetooth technologii. Systém by měl být založen na Bluetooth zařízeních periodicky vysílajících své zprávy, které se zpracovávají v síťových uzlech a ve fusion centru a z jejich signálových parametrů se vyhodnocuje poloha zařízení. Práce by měla obsahovat rešerši možných strategií řešení, jejich teoretickou analýzu včetně pokročilých metod estimace parametrů signálu a jejich distribuovaného zpracování. Vybrané řešení by mělo být ověřeno počítačovou simulací a podle dostupnosti vhodné HW platformy i implementováno a otestováno v reálném provozu.

Seznam doporučené literatury:

[1] S. M. Kay: Statistical signal processing, vol. I + vol. II, Prentice Hall, 1998
[2] U. Spagnoli: Statistical signal processing, Wiley 2018

Jméno a pracoviště vedoucí(ho) diplomové práce:

prof. Ing. Jan Sýkora, CSc. katedra radioelektroniky FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **07.02.2022** Termín odevzdání diplomové práce: ______________

Platnost zadání diplomové práce: **30.09.2023**

prof. Ing. Jan Sýkora, CSc.	doc. Ing. Stanislav Vítek, Ph.D.	prof. Mgr. Petr Páta, Ph.D.
podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

.
Datum převzetí zadání Podpis studenta

Acknowledgements

I would like to thank prof. Jan Sýkora for his patience and consultations during making of this thesis. Furthermore I would like to thank my family and friends for their unprecedented support during my studies at the university.

Declaration

I declare that I completed the presented thesis independently and that all used sources are quoted in accordance with the Methodological instructions that cover the ethical principles for writing an academic thesis.

Prague, 10. January 2023

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze, 10. January 2023

Abstract

Indoor localisation has attracted increasing interest in recent years with rising need for accurate asset tracking. As the devices enabling localisation has to be able to operate for a long period of time in real-life conditions, Bluetooth low energy (BLE) was considered as an especially suitable technology for use due to its low energy consumption. In this work a BLE-based localisation system was designed with focus on modularity, extendibility and future usage with various methods of localisation. A multiple RSSI based methods were implemented and evaluated in an experimental deployment of the system.

Keywords: Bluetooth, BLE, RTLS, indoor localisation, RSSI

Supervisor: Prof. Jan Sýkora

Abstrakt

Systémům umožňující lokalizaci ve vnitřních prostorách se v posledních letech dostává více pozornosti vzhledem k nárůstu počtu aplikací pro které jsou nezbytné jako je sledování objektů. Vzhledem k tomu, že tato zařízení musí být schopná fungovat po dlouhou dobu v reálných podmínkách, Bluetooth low energy (BLE) byl zvolen jako vhodná technologie kvůli svým nízkým nárokům na spotřebu. V této práci byl navržen lokalizační systém pro vnitřní prostory budov založený na BLE se zaměřením na jeho modularitu, rozšiřitelnost a možné budoucí použití s různými metodami lokalizace. Několik method lokalizace na bázi RSSI bylo implementováno a otestováno v experimentálním nasazení systému.

Klíčová slova: Bluetooth, BLE, RTLS, lokalizační systém, RSSI

Překlad názvu: Indoor Bluetooth lokalizace

Contents

1 Introduction **1**

1.1 Aim of the work 1

2 Theory **3**

2.1 Indoor localisation techniques . . . 3

2.1.1 Received Signal Strength Indicator 4

2.1.2 Angle of Arrival 4

2.1.3 Time of Flight 4

2.2 Bluetooth 5

2.2.1 Bluetooth Low Energy (BLE) 5

2.3 Signal propagation in indoor environment 7

2.4 Research of existing work 9

2.4.1 Kalman-Based Fusion 9

2.4.2 Particle Filtering Based Indoor Positioning System 10

2.4.3 Separate Channel Information 10

2.4.4 Angle of Arrival-Based BLE Localization 10

2.4.5 Summary of research 11

2.5 Solution strategy 12

2.5.1 Requirements 12

2.5.2 Approach to localisation 13

2.5.3 System architecture 14

2.6 Hardware 15

2.6.1 Locator 15

2.6.2 Tag 16

2.6.3 Server 17

2.7 Experimental deployment 18

3 Initial experiments **21**

3.1 Single ranging measurement 21

3.2 Multiple ranging measurement . 27

3.3 Internode channels analysis 30

4 Implementation **35**

4.1 Locators firmware 36

4.1.1 MicroPython 36

4.1.2 Operation 37

4.2 Server application 41

4.2.1 Docker 41

4.2.2 Backend 42

4.2.3 Engine 43

4.2.4 Frontend 44

4.3 Performance 45

4.3.1 Proximity variant 2D 46

4.3.2 Proximity variant 3D 47

4.3.3 Nearest neighbours variant . . 47

4.3.4 Comparison 47

5 Conclusion **51**

5.1 Next steps 52

A Bibliography **53**

Figures

2.1 Overview of BLE channels [1] . . . 6

2.2 Comparison of technologies for real-time localisation systems [2] . . . 9

2.3 Time flow of the localisation system operation . . . 14

2.4 Architecture of designed localisation system . . . 15

2.5 Photo of a locator . . . 16

2.6 Photo of a tag . . . 17

2.7 Floorplan . . . 18

2.8 Picture of mounted locator . . . 19

3.1 RSSI measurement . . . 22

3.2 Window length effect with median filter . . . 23

3.3 Window length effect with mean filter . . . 23

3.4 View on median filtered RSSI measurement . . . 24

3.5 Path loss model fitted with measured data . . . 25

3.6 Visualization of path loss model with parameters averaged across all locators . . . 26

3.7 MSE of α, β parameters estimated at locator BLO-04 based on advertisement packets received from individual locators . . . 31

3.8 Polar plot centred at locator BLO-04 depicting directionality of α parameter . . . 32

3.9 Comparison of path loss models fits based on locator-locator and tag-locator communication . . . 33

3.10 Course of α, β parameters estimated during runtime . . . 33

4.1 Detailed architecture of designed localisation system . . . 35

4.2 MicroPython filesystem overview 37

4.3 MicroPython*boot.py* workflow . . 38

4.4 MicroPython *update.py* workflow 38

4.5 MicroPython *main.py* workflow . 40

4.6 Engine architecture diagram . . . 43

4.7 Engine workflow . . . 44

4.8 Frontend visualization . . . 45

4.9 Plan of area with highlighted rooms and test datasets waypoints 46

4.10 MSE of 2D and 3D proximity variants on 10s and 30s test datasets 48

4.11 Floorplan with position results from 2D proximity variant on 10s dataset . . . 49

4.12 Floorplan with position results from 2D proximity variant on 30s dataset . . . 49

4.13 Floorplan with position results from 3D proximity variant on 10s dataset . . . 50

4.14 Floorplan with position results from 3D proximity variant on 30s dataset 50

Tables

2.1 Basic transmission loss coefficients [3] 8

3.1 Parameters of fitted path loss models 25

3.2 Results of distances estimate and its MSE to real distances for different methods of estimating model parameters 27

3.3 Computational difficulty measurement 30

4.1 Accuracy of room estimate 48

Chapter 1

Introduction

An indoor positioning systems aims to bring similar advantages as the outdoor positioning systems, such as GPS, presented and proved essential to many industries and personal lives over past decades. With already being indispensable in a modern warehouses, advancements are being made to smaller scale fields as healthcare centres, offices, retail stores, and exhibition centers among others. Even for same scales of indoor positioning systems a different variants are needed as each use-cases and applications has different requirements on positioning accuracy, latency, battery life and also the total cost of ownership with regards to expected Return of Investment. This leads to importance of researching various methods for localisation and their extensive testing.

1.1 Aim of the work

Aim of this work was to design a low cost easily deployable indoor localisation system with ability to estimate location on a sub-room level. Bluetooth technology, specifically Bluetooth low energy (BLE), was chosen as a basis for the system, because of its broad usage in a various applications, thus wide range and supply of suitable off the shelf hardware is available for a low cost. Furthermore, with recent direction finding support introduced to Bluetooth standard, it is expected for it to became more used in real time localisation systems (RTLS). Based on conducted research a suitable technique for the localisation task was selected and implemented along with a whole system around it. With regards to limited resources for this work, emphasis was on development of all software parts in a modular, reusable manner to ensure possibility of future usage even for different techniques of localisation.

As most of research focuses solely on an proposes and evaluation of methods in a small scale limited environment, it was deemed beneficial to consider conditions of a real use-case with its specifics during design of the system. Evaluation was aimed to be performed in an real life environment with a scope allowing test both sub-room and room level precision. For comparison with selected technique and other works aim was to also implement a commonly used technique for BLE-based localisation as a reference.

Chapter 2

Theory

In this chapter we introduce basic knowledge preceding design of the indoor localisation system, principle of Bluetooth wireless technology with emphasis on Bluetooth low energy (BLE). Specifics of signal propagation in an indoor environment are discussed along with its impact on signal attenuation. Results of existing work on topic of indoor Bluetooth based localisation are described with notes options deemed to be viable of in depth exploring. Following that proposed solution strategy is introduced from a high-level perspective along with selected hardware.

2.1 Indoor localisation techniques

This section describes briefly main techniques in an indoor localisation systems. More detail description is for selected method in later chapter.

All mentioned techniques can be used for both common architectures of indoor localisation systems. First commonly used architecture is referred to as a device based localization, where moving device gathers information form static devices and performs localization process itself. Estimated location is thus known only to the moving device. This is for example case of a smartphone estimating its location with usage of static beacons. Second used architecture is a network based localization, where on the contrary moving device is a simple beacon and processing is performed at a static receiver. The later is a case of this work although the similarities in techniques and data processing are to extend where surveys on both are of use for this work.

2.1.1 Received Signal Strength Indicator

The oldest and simplest approach used in a localisation task is a received signal strength (RSS) or received signal strength indicator (RSSI). Difference between RSS and RSSI is slight, RSS is power measurement with unit of decibels while RSSI is unitless relative measure of the RSS although it's commonly complemented also with unit of decibels due to its close resemblance. This technique is impacted by high spatial variance of signal attenuation in indoor environment. Because of that advanced filtration methods are required to improve its stability and allow its usage for positioning systems. The quality of attenuation model is an important factor for distance estimation or in low precision application proximity estimation. Another approach to usage of RSSI is through machine learning methods of data processing. Over extensive datasets sufficient stability and accuracy can be achieved. This approach is commonly referred to as a fingerprinting method. [4]

2.1.2 Angle of Arrival

Angle of arrival (AoA) technique uses antenna array on receiver side to estimate direction of incoming signal based on time difference of arrival on individual antenna elements. It requires hardware specifically designed towards this application. AoA technique inherently losses accuracy with distance due to basic geometry principles. It's also susceptible to multipath effect and carrier frequency offset. In line of sight scenarios its performs well and require as low as two anchor nodes for estimation of 2D position and three for 3D position. Commonly used methods of processing data into angle estimates are MUSIC algorithm and various machine learning approaches. [4]

2.1.3 Time of Flight

Physically simplest technique of estimating distance is to measure propagation time of signal. This is called time of flight (ToF). The key issue with ToF is synchronization between transmitter and receiver. Without it the precision is low as the signal propagates with speed of light. Another problems may arise from narrow signal bandwidth, which limits signal propagation. [4]

2.2 Bluetooth

Bluetooth, or since introduction of Bluetooth Low Energy (BLE) referred to as Bluetooth Classic, is a short-range wireless technology standard. Typical applications are data exchange with portable devices and an alternative to wire connections.

Pivotal technique providing Bluetooth and BLE with ability to function in densely used frequencies is frequency hopping spread spectrum (FHSS), also called adaptive frequency hopping (AFH). This method reduce impact of a narrowband interference and enables Bluetooth to operate in the same band with other technologies.

Bluetooth is set to use frequencies in range 2400 MHz to 2480 MHz separated into 79 channels with bandwidth of 1 MHz. Channel usage is controlled by FHSS method. The hopping kernel of FHSS generates pseudo-random sequence of channels for device to use. During operation channels are divided into two categories: unused and used, where unused channels are not suitable for communication at the time. Used channels are then part of hopping sequence and unused channels in sequence are substituted by used in a pseudo-random way. [5]

2.2.1 Bluetooth Low Energy (BLE)

Bluetooth Low Energy is a wireless network technology designed to operate with minimal power consumption. It's a standard independent on a Bluetooth Classic, however it was derived from it and builds on many already proved methods.

The main features of Bluetooth LE are:

- wide support (all main operating systems, mobile phones, laptops etc.),
- very low power consumption (able to operate on battery power for years),
- good performance in noisy environments (short advertising, TDD, frequency hopping).

As per Bluetooth Core Specification, all devices must support basic 1 Msym/s modulation scheme in order to comply with BLE certification. This consists of Gaussian Frequency Shift Keying (GFSK) modulation, physical layer (PHY) with uncoded data at rate 1 Mbit/s and two multiple access schemes: frequency division multiple access (FDMA) and time division multiple access (TDMA).

Optionally 1Msym/s modulation scheme may support coded PHY, which lowers data rate to 125 kbit/s or 500 kbit/s due to added redundancy by forward error coding (FEC). This scheme performs better in noisy conditions and for greater distances. Another optional variant is 2 Msym/s modulation schemes which can only make use of uncoded PHY and is designed for high data rate transmissions.

Selection of GFSK was primarily motivated by it's low complexity on transceiver side and thus low cost. FDMA is employed together with frequency hopping to allow multiple devices communicate while suppressing narrowband interfere at the same time. TDMA is used in a polling scheme where precise predeterminate timing of packets allows device to communicate without occupying unnecessary resources.

BLE uses frequency range 2400 MHz to 2480 MHz separated into 40 channels with 2 MHz bandwidth and no overlap. 3 channels have specific functionality as primary advertising channels, the rest of 37 channels is refer to as general purpose channels. Each channel has its specific channel number. This arrangement is shown in Figure 2.1.

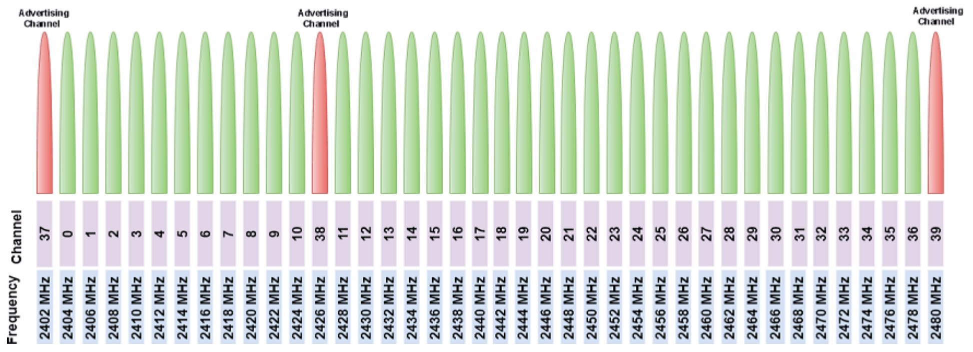

Advertising Channel
Advertising Channel
Advertising Channel
Channel
37 0 1 2 3 4 5 6 7 8 9 10 38 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 39
Frequency
2402 MHz 2404 MHz 2406 MHz 2408 MHz 2410 MHz 2412 MHz 2414 MHz 2416 MHz 2418 MHz 2420 MHz 2422 MHz 2424 MHz 2426 MHz 2428 MHz 2430 MHz 2432 MHz 2434 MHz 2436 MHz 2438 MHz 2440 MHz 2442 MHz 2444 MHz 2446 MHz 2448 MHz 2450 MHz 2452 MHz 2454 MHz 2456 MHz 2458 MHz 2460 MHz 2462 MHz 2464 MHz 2466 MHz 2468 MHz 2470 MHz 2472 MHz 2474 MHz 2476 MHz 2478 MHz 2480 MHz

Figure 2.1: Overview of BLE channels [1]

BLE link layer can be described as a state machine with 7 states: Standby State, Advertising State, Scanning State, Initiating State, Connection State, Synchronization State and Isochronous Broadcasting State. In this work the Advertising State and the Scanning State were used. Advertising State has several modes of operation from which periodic advertising mode was selected as the most suitable for desired functionality. A device in this mode is sending advertisement messages at periodic and deterministic intervals on primary advertising channels. FHSS is responsible for channel selection if not statically configured to use specific one. A device in the Scanning State is listening for advertisement messages on primary advertisement channels. This state is used on receiver side.

The advertisement message, by full name Advertising Physical PDU (protocol data unit), consist of a 16 bit header and a payload of variable size $1 - 255$ octets. The header 4 bit for PDU Type, which identifies role of the message in communication, 1 bit for RFU (reserved for future usage), 1 bit for ChSel, 1 bit for TxAdd, 1 bit for RxAdd and 8 bit for Length of the payload in octets. Values of ChSel, TxAdd and RxAdd are dependent on PDU Type in their meaning. PDU Type used in this work is ADV_IND, which is specified for usage in connectable and scannable undirected events. For this PDU Type the TxAdd represents whether the advertiser's address is public (TxAdd$= 0$) or random. The ChSel bit represents advertisers support of LE Channel Selection Algorithm #2, which differs from #1 with capability of subevent channel selection. In case of this work ChSel$= 0$ as the LE Channel Selection Algorithm #2 is not needed. RxAdd is not used in this PDU Type. As to payload, first 6 octets stores AdvA (advertiser address), following $0 - 31$ octets stores AdvData (advertisement data) which is programable data field, which is leaved empty for case of this work. [5]

2.3 Signal propagation in indoor environment

Signal propagation in indoor environment differs from outdoor settings in multiple respects. While it's simpler in a definition of the area covered, complexity arise with complicated inner geometry, various materials used, movement of objects/people and time variance of it.

Main causes of propagation impairments are:

- reflection/diffraction from and around objects including walls and floors,
- transmission loss from passing through objects,
- channelling in corridors,
- motion of people and objects,
- motion of transceivers and receivers. [3]

These causes affects signal propagation in a several ways. Signal passing through objects, walls, and free space deteriorates in power. Contrarily, narrow spaces like corridors mitigates transmission loss to some extend as they channel the propagated signal. As a consequence of mentioned effects, there might be a significant spatial variation of transmission loss. Another consequence of a complicated inner geometry are multi-path propagation effects. Multiple versions of the same signal arriving in a same time window interfere with each other causing power and phase distortion. Further, motion of communicating devices cause polarization mismatch of propagating wave and receiver antenna.

In scope of this work focus was on frequency 2.4 GHz and indication of received signal strength (RSSI). Frequency range of interest is prone to all above mentioned effects. This as expected leads to high variance in propagation loss and subsequently RSSI. Relation of RSSI and distance between transmitter and receiver is described by path loss model. In a general case, this model is formalized in Equation 2.1.

$$L_b\left(d, f\right) = 10\alpha \log_{10}\left(d\right) + \beta + 10\gamma \log_{10}\left(f\right), \tag{2.1}$$

where L_b represents median basic transmission loss, d is 3D direct distance between transmitter and receiver in meters and f is frequency in GHz. α is parameter representing transmission loss with distance, β represents offset of this transmission loss and γ is associated with frequency dependency of the transmission loss. [3]

With regards to scope of this work, where frequency deviation is negligible, γ element in 2.1 is constant and thus it can be merged with β offset parameter. Next, indicative measurement of received signal power through $RSSI$ is used to represent L_b. Subsequent model is as follows.

$$RSSI\left(d\right) = 10\alpha \log_{10}\left(d\right) + \beta, \tag{2.2}$$

where RSSI is received signal power in decibels. Values of α, β are site specific and should be adjusted through measurement. Recommended parameter ranges by ITU-R are provided in Table 2.1.

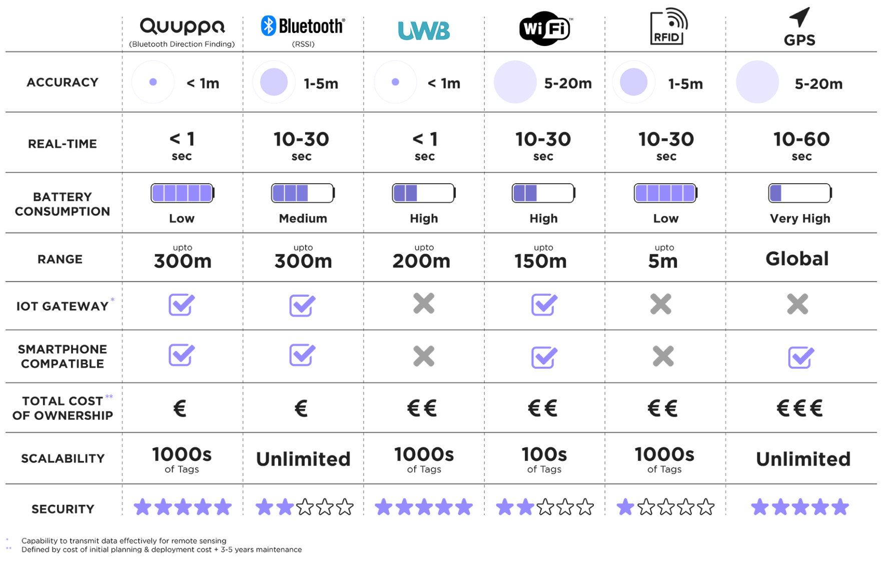
Environment	LoS/NLoS	Frequency range [GHz]	Distance range [m]	α	β
Office	LoS	0.3 – 83.5	2 – 27	1.46	34.62
	NLoS	0.3 – 82.0	4 – 30	2.46	29.53
Corridor	LoS	0.3 – 83.5	2 – 160	1.63	28.12
	NLoS	0.625 – 83.5	4 – 94	2.77	29.27
Industrial	LoS	0.625 – 70.28	2 – 101	2.31	24.52
	NLoS	0.625 – 70.28	5 – 108	3.79	21.01

Table 2.1: Basic transmission loss coefficients [3]

In a later chapter assessment of combine influence of propagation impairments on RSSI and distance estimation based on it is made. Methods of obtaining site specific values for α and β are also discussed.

2.4 Research of existing work

In order to get familiar with current knowledge in a field, a thorough research was conducted. As many articles proved to be similar in their objectives and approaches, only subset deemed representational is presented. It is important to note that scope of this work was limited to Bluetooth based approaches to task of indoor localisation. Other technologies in use for localisation as Wi-Fi, UWB (Ultra-wideband), visible light communication, ultrasound etc. were not considered to keep extend of the work feasible. Their comparison is for illustration shown in Figure 2.2.



	Quuppa (Bluetooth Direction Finding)	Bluetooth (RSSI)	UWB	WiFi	RFID	GPS
ACCURACY	< 1m	1-5m	< 1m	5-20m	1-5m	5-20m
REAL-TIME	< 1 sec	10-30 sec	< 1 sec	10-30 sec	10-30 sec	10-60 sec
BATTERY CONSUMPTION	Low	Medium	High	High	Low	Very High
RANGE	upto 300m	upto 300m	upto 200m	upto 150m	upto 5m	Global
IOT GATEWAY*	✓	✓	✗	✓	✗	✗
SMARTPHONE COMPATIBLE	✓	✓	✗	✓	✗	✓
TOTAL COST** OF OWNERSHIP	€	€	€€	€€	€€	€€€
SCALABILITY	1000s of Tags	Unlimited	1000s of Tags	100s of Tags	1000s of Tags	Unlimited
SECURITY	★★★★★	★★☆☆☆	★★★★★	★★☆☆☆	★☆☆☆☆	★★★★★

* Capability to transmit data effectively for remote sensing
** Defined by cost of initial planning & deployment cost + 3-5 years maintenance

Figure 2.2: Comparison of technologies for real-time localisation systems [2]

2.4.1 Kalman-Based Fusion

As studies on RSSI based approaches are being performed for almost two decades, attempts to improve its accuracy comes to limitations of this approach. One way of achieving higher accuracy is to combine RSSI approach with other techniques. This is for example descripted in a study "An Improved BLE Indoor Localization with Kalman-Based Fusion: An Experimental Study", where combination of RSSI based trilateration and dead reckoning technique is proposed. Dead reckoning is a technique extracting movement of device from accelerometer measurements. With filtration to suppress considerable noise in this measurement, information about movement can be extracted. This in fusion with position estimated by trilateration can compensate for imprecision to some extend. In experimental measurement the article reports achieved accuracy under 1 m in area of 30 m × 5 m covered by 8 receivers. [6]

2.4.2 Particle Filtering Based Indoor Positioning System

The paper "Particle Filtering-Based Indoor Positioning System for Beacon Tag Tracking" proposed particle filter-based indoor positioning system to localize BLE tags from RSSI data. Approach chosen uses Kalman Filter for compensation of fluctuation in the RSSI data. Subsequently particle filter with Gaussian kernel conducts positioning task. This system was tested with low cost off-the-shelf devices having 4 receivers at corners of a 5.4 m long and 4.95 m wide room. Proposed particle based approach performed with error of 1.38 m for 80% of time. In comparison trilateration method with site adjusted path loss model resulted in 3.1 m error for 80% of time. [7]

Statistical approach as described in this paper is a reasonable approach to localisation task. However, conducted experiments doesn't consider non-line-of-sight scenarios and their effect on fluctuations of RSSI and particle filter behaviour.

2.4.3 Separate Channel Information

In the article "A Robust Indoor Positioning Method based on Bluetooth Low Energy with Separate Channel Information" authors proposed approach based on RSSI and common path loss model with site specific coefficients. Novelty in their approach was evaluating separately each of BLE primary advertisement channels. They reported finding of 33% improved accuracy in comparison to usage of RSSI averaged over all channels with regard to their experimental setup. [8]

This approach is simple as it uses basic regime of BLE beacons operation. In theory, with possibility to adjust BLE beacons firmware, this approach could be extended to usage of all 40 BLE channels instead of only 3 primary advertising channels using chained advertisement method available in BLE protocol. This would add granularity for cost of higher overall complexity.

2.4.4 Angle of Arrival-Based BLE Localization

Previously described articles focused on RSSI based localisation systems. With advances in available hardware and support from Bluetooth specification side, Angle of Arrival based localisation is being explored as potentially technique to outperform state of the art RSSI based systems. Amount of quality sources describing more than a single angle measurement is relatively low. Because of this the comparison of achievable localisation accuracy under representable circumstances is imprecise.

Based on article "AOA-Based BLE Localization with Carrier Frequency Offset Mitigation", it is possible with AoA approach to achieve level of accuracy around 0.15 m. However their testing was performed in an area of size $4\,\text{m} \times 4\,\text{m}$ with direct line of sight. [9]

2.4.5 Summary of research

From reviewed works following finding were marked. As to refer to used approaches, for RSSI based methods two approaches are commonly used, fingerprinting and proximity. Fingerprinting uses extensive dataset gathering for evaluation either with traditional classification methods such as k-nearest neighbours or decision tree, or with more modern methods as deep neural networks. In general fingerprinting approach is stated to perform well, however, with need for extensive datasets and thus low adaptability for changes in environment, it can't scale well and this issue is not solved in a satisfying manner. Proximity approaches from median or Kalman filtered RSSI values extrapolates distances through path loss model or site specific machine learned model instead.[10] Articles with this approach adds interesting tweaks for improvement in accuracy of localisation, however, many of them are either site specific or specific for used hardware.

Very few works performs tests on a area larger than $10\,\text{m} \times 10\,\text{m}$ or with covering more than two rooms. This is concerning, because most of signal propagation impairments occurs due to complicated structures in environment across larger area.

To sum up, all of mentioned and for scope of this work reviewed indoor localisation system can be consider as well performing. Subset of methods indicates potential in combination of several approaches. However, none of currently existing indoor localisation systems is close to the same level of accuracy, reliability and low cost like those of GPS system in outdoor environment. This field of study is thus still more than relevant. With regards to new features as Angle of Arrival (AoA) and Angle of Departure (AoD) introduced in the most recent updates to Bluetooth standard, the more accurate approaches will drop in price and have thus great potential in broader usage.

2.5 Solution strategy

In previous sections several existing strategies for indoor localisation systems was reviewed. The aim of this work is to design the localisation system based on Bluetooth technology. First, requirements list is put together for this system and constraints given conditions out of reach. Then proposed solution strategy is introduced from high level perspective with comment on meeting the requirements.

2.5.1 Requirements

Requirements it was decided to put on the system were following:

- low cost (off-the-shelf equipment),
- Bluetooth Low Energy compatible,
- low energy consumption of ,
- real-time tracking (<30s delay),
- room level precision (sub-room at best),
- easy deployment,
- extendibility,
- modularity.

Requirement on low cost is always present when designing a system. In this case it was one of the most important requirements, because of limited budget and expected high number of locators necessary. Availability and quality of development toolchain were considered during selection of suitable hardware components. Linked with this is requirement on Bluetooth Low Energy (BLE) usage. BLE devices with low power consumption capable of operating for months or several years are commonly used for many applications. They are produced in quantities which makes them cheap, reliable and easily available. Detailed description of all hardware components is in following chapter.

The localisation system is intended to be used for live tracking of people and objects. This puts limitations on timings, data transfers and methods of processing data. Easy deployment of whole system is not the most important aspect as it's not market ready product. However, keeping that in mind should help with controlling the work to be easily tested, moved to other locations and modified. With extendibility is meant ability to adjust installation of the localisation system, add new devices or remove them. This requirement is directly connected with the requirement on easy deployment.

Modularity is a requirement arising from deployment and extendibility. The aim is to have each part of system encapsulated and with documented API, so either different hardware can be used with it, approach to evaluation of data can be changed or multiple instance for live testing can be run simultaneously. Especially possibility of changing hardware platform was considered important if the selected hardware based on cost prove to be unfit for the task.

Constraints limiting the options of meeting requirements were following:

- limited processing power on nodes
- fixed antenna
- office building environment

With hardware selected to meet the cost limits, decision was made to use only RSSI as information source for the localisation system. It's the most simple source of information with significant downsides for usage, however, any other sources of information were not usable without inherent error of magnitudes over the target usage. Even with limited RF processing capabilities, suitable antenna design could help mitigate certain issue arising from specifics of signal propagation in indoor environment. Decision was made to not explore this option as it would broaden scope of the work excessively. Thus it was considered as a constraint on the system. As mentioned with requirement on modularity, future change and testing should be possible if the requirements are met.

Another key constraint for this work is indoor environment. Specifically modern office building, with substantial number of devices using wireless communication and materials with varying interactivity in regards to part of spectrum used by Bluetooth. This is discussed in detail implications of indoor environment on signal propagation in chapter 2.3.

2.5.2 Approach to localisation

Based on related work two main approaches to localisation task based on RSSI were explored. First, from RSSI values estimate distances to nodes and through their combination obtain location estimate. Subsequently either directly from estimated location if possible or with additional processing determine a room. Second, use RSSI fingerprinting method with k-nearest neighbours classifier for determining a room directly. This approach was selected for comparison with the first one as it's one the most popular among related work.

In order for both approaches to operate in real-time a time window for data points and longer time window over which localisation is performed were define. Accumulation of RSSI values to data points both helps mitigate errors in RSSI values and arise as necessity from hardware capabilities. Extended time window for localisation process adds additional time dependency and thus reduce variance in location estimates. On the other hand it introduces inertia to location estimate results and slows the localisation process. To meet requirements set for this work, the localisation time window needs to be under 30s.

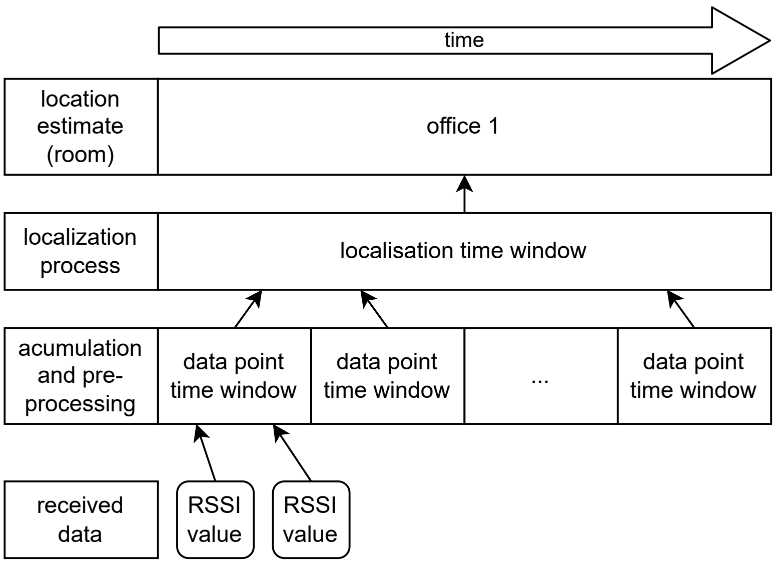


Figure 2.3: Time flow of the localisation system operation

2.5.3 System architecture

Starting point in functionality of the system is the "tag" i.e. device which periodically transmits BLE advertisement messages. The messages are received in nodes, referred to as "locators", mounted in rooms and halls. From each message locator identifies ID of the tag, adds RSSI measurement and timestamp from its time base. This represents one data point in the system. Locators periodically sends all gathered data points onto local server.

Server is divided into 4 parts, Backend, Database, Engine and Frontend. Each part on server is run as asynchronous instance for better performance in manner of responsivity. Backend provides API for locators and other instances. Data received from locators on Backend are added to Database with correct UTC time. Database is time series based type as it suits are application the best. Engine periodically requests new data from Backend, estimates location of tags and results returns to Backend, where they're again stored into database. Frontend operates in similar manner, periodically requests data and provides them to user on local website.

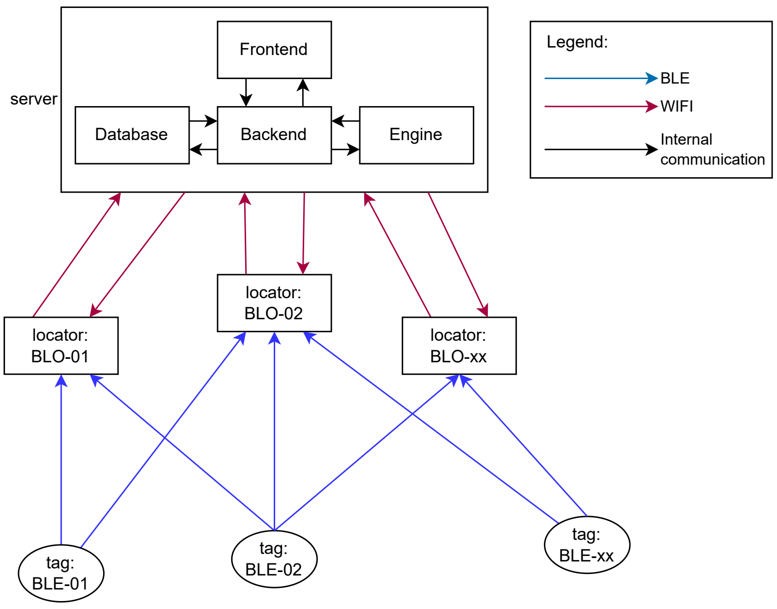


Figure 2.4: Architecture of designed localisation system

2.6 Hardware

There are two key hardware components in the localisation system. The transceiver, referred to as a "tag" and receiver or "locator". Selection of hardware components for they're realization along with unchangeable configuration options limits were put onto possibilities of the localisation system. The aim was to utilize them in the best way possible while keeping possibility of them being exchanged in future.

2.6.1 Locator

The receiver is custom build board hosting ESP32-C3-MINI-1 module depicted in Figure 2.5. The board contains AC-DC power source, voltage regulator, status LED, JTAG connector and USB connector. The intention was to power locator from electrical installation as it substantially simplifies process of installation and overall cost. Appropriate steps for isolation and protection were made during board design process. Additional option is to power the device from 5V source or via USB connector from a power bank. Last option was used during development as it allowed flexibility for initial measurements.

ESP32-C3-MINI-1 is a general purpose Wi-Fi and Bluetooth module with planar antenna of meandered PIFA design. It is built around ESP32-C3 32-bit RISC-V single-core processor with frequency up to 160 MHz, 384 kB ROM, 400 kB SRAM and 4 MB flash memory. It has single RF peripheral which is shared by Wi-Fi and Bluetooth stack thus allowing to use only one of them at the time. Wi-Fi stack is IEEE 802.11 b/g/n-compliant. Bluetooth stack supports Bluetooth 5.0 standard and Bluetooth Low Energy. From specified parameters only limiting factor for scope of this work is shared RF peripheral, which makes simultaneous receiving and transferring data impossible. [11]

Locators were running custom made firmware which implementation is described in chapter 4.1.

For identification each locator was labelled as "BLO-xx" where "xx" is substituted with double digit number. In total 21 locators were available for usage, from which 13 were installed in experimental deployment and the rest was used for firmware testing.

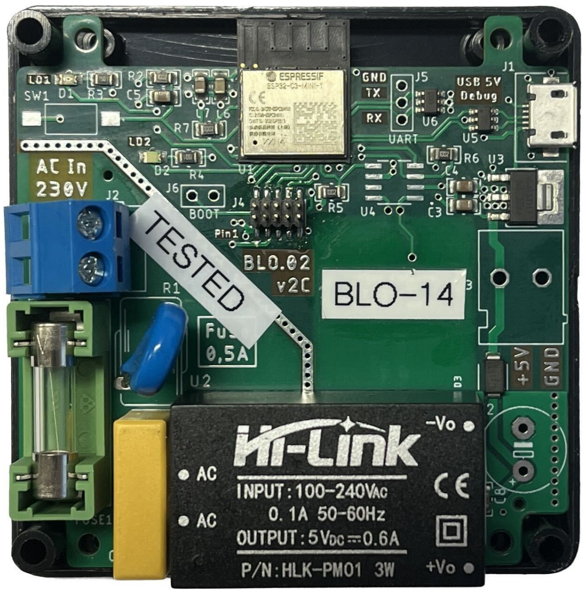


Figure 2.5: Photo of a locator

2.6.2 Tag

From available off-the-shelf devices Quuppa QT1-1 Tags were selected. It's a compact device with Nordic nRF52832 Bluetooth chip, accelerometer, button, non-rechargeable battery capable powering device for up to 2 years and a IP67 cover. Dimensions are 39 mm × 39 mm × 9 mm and it weights 10 g. It was selected for use as it's commercially used model in a various RTLS systems and thus enables to test the system in a more realistic respect.

Another upside of this device is its robustness and simplicity in usage. Unfortunately, there's also significant downside in closed source firmware and inability to change configuration of most of BLE properties. Inability to adjust transceiver inherently worsens the situation at receiver and limits an options. The trade-off was deemed acceptable for scope of this work.

Basic functionality of a tag is to periodically transmit BLE advertisement packets. The period and payload message of advertisement packets can be configured. In the configuration the period of transmission is 1 s. The message is empty as it's not needed for development, however, for potentially future usage both locators and server application were prepared for its receiving and storing.

Similarly as with locators, each tag was labelled as "BLE-xx".

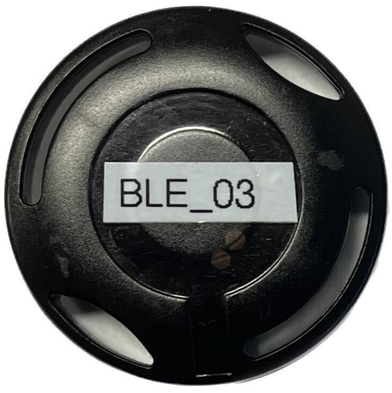


Figure 2.6: Photo of a tag

2.6.3 Server

As a server laptop HP EliteBook 830 G5 was used, with Intel Core i5-8250U CPU at 1.60GHz, 32GB RAM and with Windows 10 installed. To have a controlled environment a separated local network without internet access was created to which server was connected via Ethernet. This was both for development reasons and also it mirrors deployment mode.

The server is running custom made software in a virtualized containers, which makes it independent on operating system to extend of underlaying virtualization service support. Early in development usage of Orange Pi 3, a small computer with ARM architecture CPU and UNIX based operating system, was successfully tested and deemed as option for deployment. Details of server software are discussed in chapter 4.2.

2.7 Experimental deployment

This section describes setting and layout of experimental deployment, tools developed to facilitate this process and specifics of this deployment.

The experimental deployment was situated in part of a one floor in an office building. Area size was 17 m × 8 m covering 5 offices and adjacent hallway. The locators were mounted on ceilings of rooms and a hallway. The intention was to place locators in a square grid with at least 1 m separation from any wall or similar obstacle and having between one and two locators per room. Existing technical installations compromised some locations. Final placement of locators depicted on Figure 2.7 is a trade-off. Drawn connections between locators represent electrical cabling originating in a distribution board providing circuit break protection.

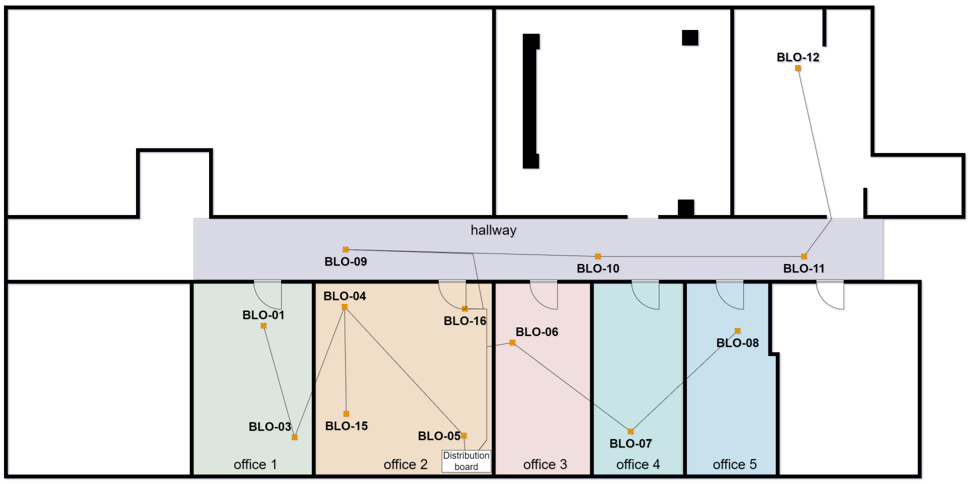


Figure 2.7: Floorplan

For simplifying process of deployment a software tool was development. It makes use of floorplan drawing made in "diagrams.net", open-source graphing software for diagrams, flowcharts, etc. From the floorplan drawing it can export positions and labels of rooms, locators, walls, etc. and whole floorplan as an image background for live visualization. All exported information are in formats which are imported to corresponding parts of the localisation system application. Positions of locators are processed into their distances and are used for path loss model estimation in Chapter 3.3 and for Engine operation. Similarly with room coordinates which are used for room level localisation.

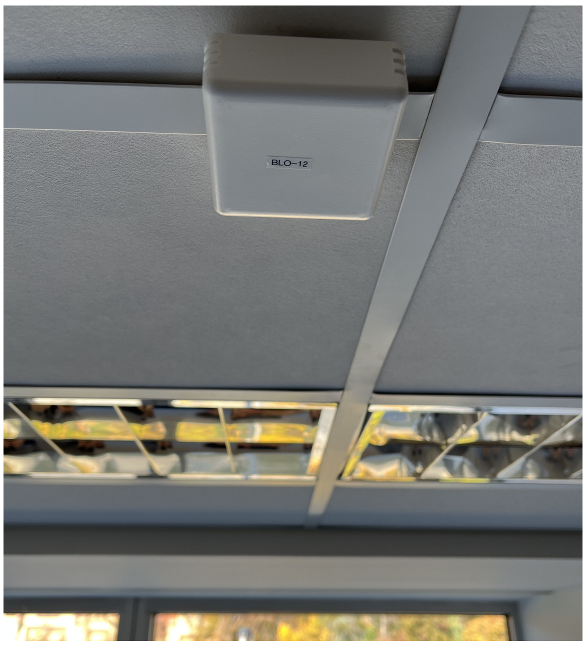


Figure 2.8: Picture of mounted locator

Chapter 3

Initial experiments

Several experiments was performed to verify feasibility of using selected hardware for localization task. Firstly single ranging measurement was considered where using one locator and one tag in known distances path loss model parameters were adjusted. Accuracy of such measurement was evaluated with MSE (mean square error) as a metric. Based on results of it process of estimating location of a tag using multiple locators is discussed.

3.1 Single ranging measurement

To assess capability of extracting distance from RSSI, initial set of measurements was performed. For each locator RSSI was measured at various distances with a single tag. Character of error in RSSI measurement was taken into account and compensated for with filtration. Model 2.2 derived in previous chapter was used to fit the filtered data and error of distance measurement using this method was evaluated. All data manipulation descripted in this chapter is illustrated on data from single locator to keep number of figures acceptable.

Starting with raw RSSI measurement depicted in Figure 3.1, distortion of values was significant.

For filtration mean filter and median filter were reviewed both with sliding window implementation. Comparison for various window lengths was performed. The character of error in the measurements is of occasional significantly different values. Mean filter was expected to supress these outliers only partially as the frequency of noise is rather low. Median filter should provide better results, if the length of averaging window is longer than occurrence of outline values. Comparison of Figures 3.2 and 3.3 shows better

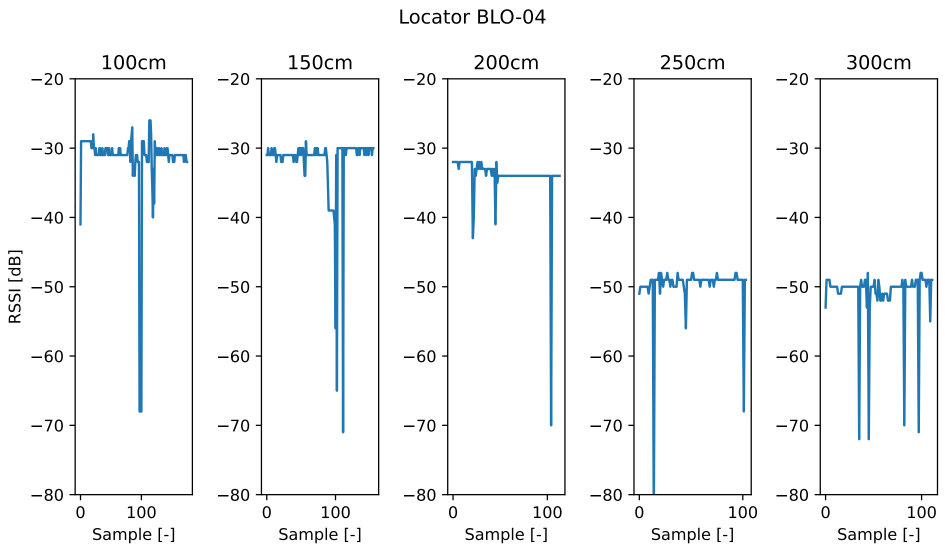


Figure 3.1: RSSI measurement

performance of median filter. If taken into account future usage in runtime of whole localisation system, shorter window length would be more beneficial in cases of lower number of samples for evaluation. Decision was made to use window length $w_{len} = 9$ as it's able to compensate for most of outline values while being as short as possible. However, for different set up of system or different installation, this value may need adjusting or be replaced by adaptive averaging window length.

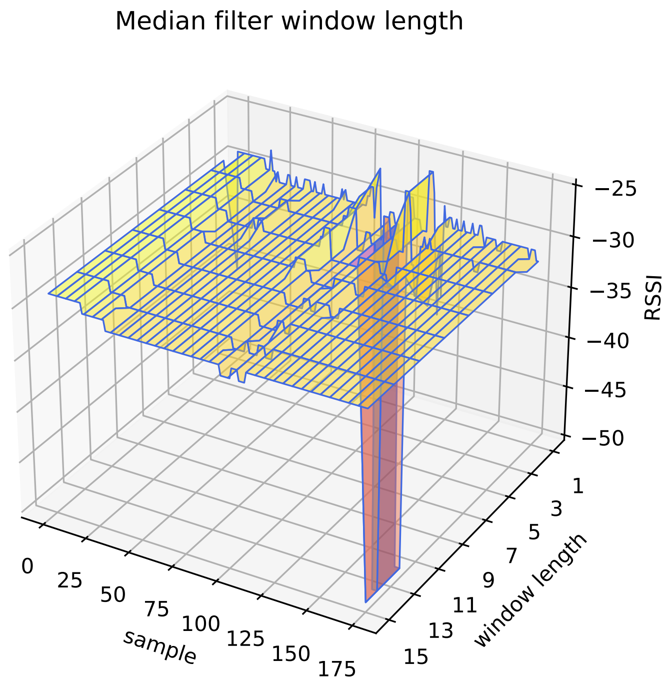


Figure 3.2: Window length effect with median filter

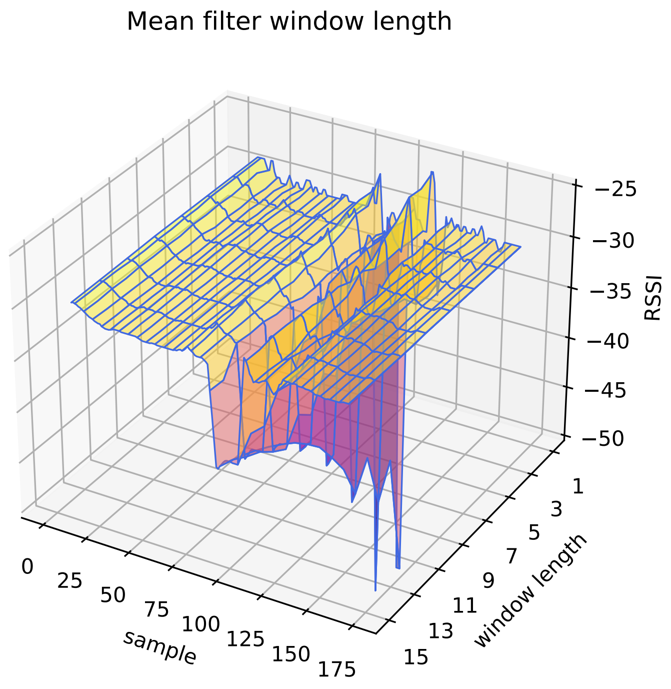


Figure 3.3: Window length effect with mean filter

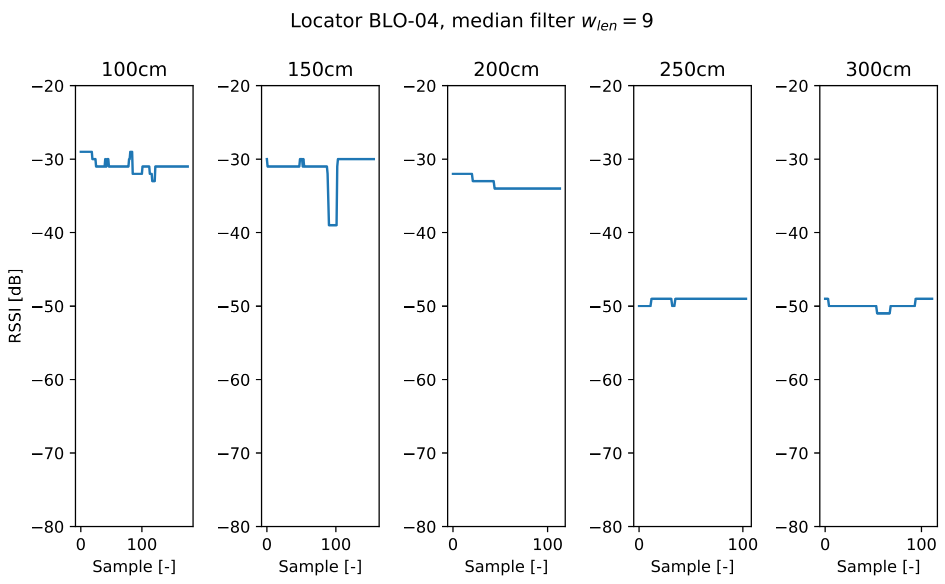


Figure 3.4: View on median filtered RSSI measurement

To obtain distance representation of RSSI values path loss model 2.2 was used. Based on it matrix equation 3.1 was formed, where d_n represents distances in which measurement was performed and $RSSI_{avg}(d_n)$ represents average of filtered RSSI values. Least squares method was used to solve the equation and obtain approximation of α and β parameters.

$$\begin{pmatrix} 10\log_{10}(d_0) & 1 \\ 10\log_{10}(d_1) & 1 \\ \vdots & \\ 10\log_{10}(d_n) & 1 \end{pmatrix} \cdot \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} RSSI_{avg}(d_0) \\ RSSI_{avg}(d_1) \\ \vdots \\ RSSI_{avg}(d_n) \end{pmatrix} \tag{3.1}$$

This step was repeated for averages of RSSI values adjusted by their standard deviation to approximate also bounds for parameters. Results are illustrated in Figure 3.5 and complete summary in Table 3.1.

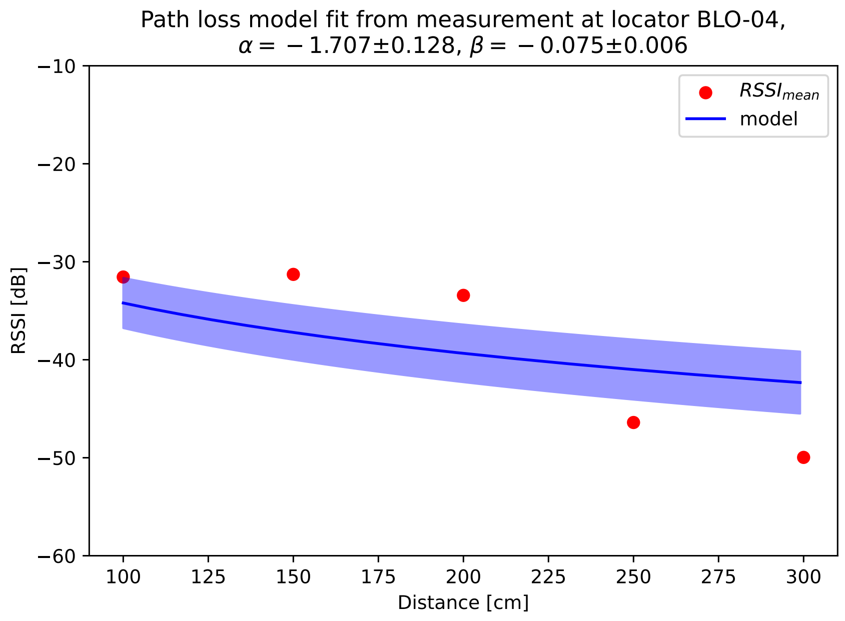


Figure 3.5: Path loss model fitted with measured data

Locator	RSSI mean					α		β	
	1m	1.5m	2m	2.5m	3m	mean	σ	mean	σ
BLO01	-31.13	-33.08	-41.00	-44.04	-44.24	-1.709	0.093	-0.075	0.004
BLO03	-31.05	-31.25	-32.13	-44.26	-44.98	-1.624	0.134	-0.071	0.006
BLO04	-31.56	-32.29	-33.43	-46.40	-49.96	-1.707	0.128	-0.075	0.006
BLO05	-29.28	-31.28	-33.41	-46.90	-51.85	-1.711	0.124	-0.075	0.005
BLO06	-33.76	-38.39	-40.55	-45.32	-52.88	-1.864	0.164	-0.082	0.007
BLO07	-28.13	-30.62	-31.00	-45.31	-46.45	-1.610	0.135	-0.071	0.006
BLO08	-31.26	-31.28	-34.95	-46.38	-45.72	-1.677	0.176	-0.073	0.009
BLO09	-29.75	-30.24	-33.46	-35.20	-39.97	-1.485	0.103	-0.065	0.004
BLO10	-30.12	-30.05	-34.18	-37.00	-43.33	-1.542	0.140	-0.068	0.006
BLO11	-29.80	-31.72	-33.98	-38.73	-46.40	-1.597	0.147	-0.070	0.006
BLO12	-18.00	-21.01	-30.27	-42.32	-42.36	-1.378	0.072	-0.060	0.003
BLO15	-32.13	-34.86	-35.09	-51.63	-52.23	-1.826	0.179	-0.080	0.008
BLO16	-30.96	-31.05	-40.10	-40.27	-43.87	-1.644	0.089	-0.072	0.004
average	-29.76	-31.32	-34.89	-43.37	-46.48	-1.644	0.130	-0.072	0.007

Table 3.1: Parameters of fitted path loss models

There's a significant variance in estimated model parameters depicted in Figure 3.6. As signal propagation environment is different for each location, may differ with specific pieces of hardware and is changing over time, these results were expected. Several options of mitigating this issue were considered.

First and the most simple option is to use at each locator the same values of α, β parameters for all subsequent distances estimation. These values would be selected as the mean of estimated values in Table 3.1 across all locators. This option would not reflect particularity of locators placements

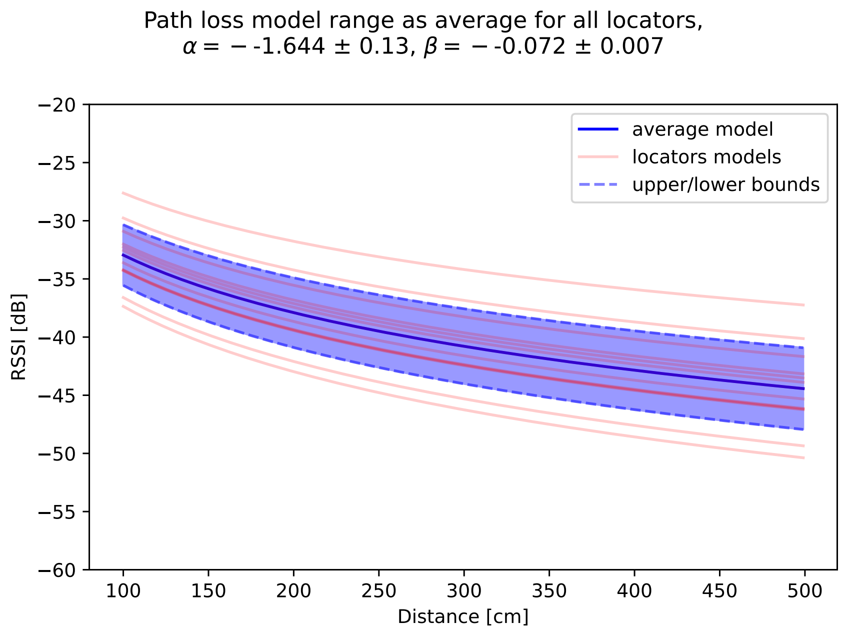


Figure 3.6: Visualization of path loss model with parameters averaged across all locators

and thus introduce significant error to calculation of distances. Even though poor performance of this option was expected, it was evaluated to serve as a baseline in comparison of other options.

Another simple option would be to choose specific values of parameters α, β for each locator. This approach would provide results with vary accuracy depending on dataset size. Using values from 3.1 should be illustrative. This option should compensate for locator placement, but won't reflect short term changes in environment due to either movement of people or communication of devices. Long term changes in environment wouldn't be possible to take into account without gathering additional dataset. This is another downside of this approach, although it doesn't affect day-to-day running of the system. For room level precision this option may be sufficient.

More complex option is to estimate model parameters during runtime. Using locators to also transmit advertisement packets, receive them with other locators and with their distance being known estimate model parameters in the same way as with tag above. With this approach the model should represent current state of environment more accurately and provide better distance precision. However, the locators have only one RF peripheral and can't transmit and receive at the same time. Balance between time spent estimating model parameters and localization task itself needs to be addressed. This option is assessed in detail in section 3.3.

Mentioned options of obtaining path loss model parameters were used to calculate distances and compared them by its error. Results are shown in Table 3.2.

Locator	Option	Result	Distances [cm]				
			100	150	200	250	300
BLO04	measured model averaged over locators	$\hat{d}$	73	80	152	1046	1570
		$MSE\left[\hat{d}\right]$	759	5536	19129	691887	7217923
	measured model per locators	$\hat{d}$	62	68	125	808	1180
		$MSE\left[\hat{d}\right]$	1446	7157	15619	343361	3444152
	runtime estimated model	$\hat{d}$	62	67	122	676	1201
		$MSE\left[\hat{d}\right]$	1473	7405	14788	215117	3211224

Table 3.2: Results of distances estimate and its MSE to real distances for different methods of estimating model parameters

Results over all options indicates that used path loss model in combination with measured RSSI can't accurately estimate distance from a locator. At best the estimated distance can be interpreted as a proximity metric to a locator. Usage of such metric was considered as potentially viable for room level precision localisation. Differences between individual options are rather minor. In conclusion runtime estimated model performs as good as models calculated from manually performed measurements.

3.2 Multiple ranging measurement

Next step in analysis of feasibility of propose localization system is to combine several proximity measurements to estimate location of a tag. In this section chosen technique is described along with its limitations and key aspects with regards to potentially extending area for localization system.

Position of both locators and tags in the area is denoted as a vector $r = (x, y, z)$. From research of existing works on this topic, many solutions prefers simplified 2D positions where $r = (x, y)$. Argument, that reducing dimensionality decreases computational complexity is definitely valid and in theory this trade-off is reasonable.

The issue with this simplification is following. In a situation, where tag is directly below a locator, its distance from it is in $(x, y) = 0$ while calculated distance from RSSI is nonzero and this will cause significant error in localization. Even in a scenarios where it's not physically possible to place tag under a locator e.g. locator mounted on side of a wall. The closer the tag will be to this locator, the higher the error of localization will be as it depends directly on RSSI measured on this locator. Simply, the estimated distance to this locator would be proportion of error introduce to localization process. As the installation has almost all locators mounted in a way, that person with a tag can walk under them, the trade-off of lower computational complexity for lower accuracy is not viable for us in theory. To verified the reasoning decision was made to explore both 2D and 3D position options and compare them.

Trilateration (or multilateration) is one of traditional techniques for computing position of a node. It requires N ranging measurements for calculating position in a N dimensional space if zero-error case is considered. For non-zero-error case, higher number of measurements may increase accuracy. However, for case of this work with having proximity measurement instead of precise distance, this doesn't necessary apply. This issue is addressed in section 4.3 dedicated to performance of individual variants of data processing engines. For calculation of trilateration task, approximation method needs to be used. Least squares was selected. [12]

To formalize, a tag position is denoted as $r_{tag} = (x, y, z)$. For total of N locators each locator position is denoted as $l_i = (x_i, y_i, z_i)$. Then distance estimated between locators and a tag is as follows.

$$d_i = \|r_{tag} - l_i\| \tag{3.2}$$

$$d_i = \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2} \tag{3.3}$$

By subtracting last equation expressing d_n from all the others, r_{tag} coordinates can be isolated [6].

$$\begin{aligned} d_i^2 - d_n^2 =& (x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2 \\ &- (x - x_n)^2 - (y - y_n)^2 - (z - z_n)^2 \end{aligned} \tag{3.4}$$

$$\begin{aligned} d_i^2 - d_n^2 =& x^2 - 2xx_i + x_i^2 + y^2 - 2yy_i + y_i^2 + z^2 - 2zz_i + z_i^2 \\ &- x^2 + 2xx_n - x_n^2 - y^2 + 2yy_n - y_n^2 - z^2 + 2zz_n - z_n^2 \end{aligned} \tag{3.5}$$

$$\begin{aligned} d_i^2 - d_n^2 + x_i^2 - x_n^2 + y_i^2 - y_n^2 + z_i^2 - z_n^2 =& 2\left(x_n - x_i\right)x + 2\left(y_n - y_i\right)y \\ &+ 2\left(z_n - z_i\right)z \end{aligned} \tag{3.6}$$

Now a matrix equation can be formed.

$$Ar = b \tag{3.7}$$

$$\begin{pmatrix} 2\left(x_0 - x_n\right) & 2\left(y_0 - y_n\right) & 2\left(z_0 - z_n\right) \\ 2\left(x_1 - x_n\right) & 2\left(y_1 - y_n\right) & 2\left(z_1 - z_n\right) \\ \vdots & \vdots & \vdots \\ 2\left(x_{n-1} - x_n\right) & 2\left(y_{n-1} - y_n\right) & 2\left(z_{n-1} - z_n\right) \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_0^2 - x_n^2 + y_0^2 - y_n^2 + z_0^2 - z_n^2 + d_n^2 - d_0^2 \\ x_1^2 - x_n^2 + y_1^2 - y_n^2 + z_1^2 - z_n^2 + d_n^2 - d_1^2 \\ \vdots \\ x_{n-1}^2 - x_n^2 + y_{n-1}^2 - y_n^2 + z_{n-1}^2 - z_n^2 + d_n^2 - d_{n-1}^2 \end{pmatrix} \tag{3.8}$$

Similarly for simplified case with two dimensions.

$$\begin{pmatrix} 2\left(x_0 - x_n\right) & 2\left(y_0 - y_n\right) \\ 2\left(x_1 - x_n\right) & 2\left(y_1 - y_n\right) \\ \vdots & \vdots \\ 2\left(x_{n-1} - x_n\right) & 2\left(y_{n-1} - y_n\right) \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_0^2 - x_n^2 + y_0^2 - y_n^2 + d_n^2 - d_0^2 \\ x_1^2 - x_n^2 + y_1^2 - y_n^2 + d_n^2 - d_1^2 \\ \vdots \\ x_{n-1}^2 - x_n^2 + y_{n-1}^2 - y_n^2 + d_n^2 - d_{n-1}^2 \end{pmatrix} \tag{3.9}$$

For solving these matrix equations least squares method was chosen as mentioned before. There are many solvers in existence with various specific advantages, but for this case common implementation with adjustable bounds was sufficient. TRF (Trust Region Reflective) method from SciPy library was used[13]. TRF can use either matrix decomposition approach or iterative procedure for its calculation of steps. The iterative procedure uses LSMR algorithm [14], which main advantage here is monotonic decrease in residuals of states and thus it's safer to be terminated early.

Accuracy of estimating location proved to be dependent on several factors linked with implementation and thus is addressed fully in chapter 4.3 dedicated to comparison of localization engines.

To consider extend of the system from computation perspective, the installation, which is descripted in a detail in separate section, consists of 12 locators and up to 5 tags. This means the dimensions of A matrix was at most $(12, 3)$, b was $(12, 1)$. Time spent computing using method described above was measured, results are listed in Table 3.3. From results follows that computing one position of a one tag takes little under 1 ms.

Number of computations	804
Average time for one computation $[ms]$	0.818
Standard deviation $[ms]$	0.363

Table 3.3: Computational difficulty measurement

The localization system can be extended in two ways. First way is number of tags being tracked simultaneously. With tag frequency of 1 Hz and thus minimal evaluation window of 1 s, this computational approach was considered usable to up to hundred tags without any other changes. Second way of extending the localisation system is to extend the area over which localisation is performed. With extending the area the amount of locators will grow. However, with distance RSSI values decline significantly and thus accuracy of ranging measurement using it. There would be a state from which additional locator wouldn't improve accuracy of localization. From performed experiments, which are based on single installation and are not resolute for other possible installations, the opinion is that limiting localization task to use at most of 10 locators doesn't lead to decrease in accuracy.

3.3 Internode channels analysis

In this section method of estimating parameters for path loss model during runtime of the localization system is analysed. This method was intended to address two issues. First, short-term accuracy of ranging measurement. Second, diminish the need for initial measurement for deployment of system as a whole. Throughout this section data from locator BLO-04 continue to be used for illustration.

As stated in previous section, proposed method would take advantage of a beforehand known position of all locators. With modification to locators firmware, locators were set up to periodically switch modes between receiving advertisement packets from tags and transmitting their own advertisement packets. Issue with setting values for receiving and transmitting periods is addressed along with other implementation details in section dedicated to firmware of locators. The advertisement packets from locators are received by other locators in the same manner as ones from tags and they require same filtration before usage. With filtered values same matrix equation was formed as in 3.1 and using least square solver obtain approximation of path loss model parameters.

This process treats all locators as to be in one direction which is in contradiction with reality of the experimental installation. To address this potential issue, measurement was performed from which were firstly calculated model parameters between all locators. In next step MSE for both model parameters for each locator was calculated to determine directional variance. Results shown in Figure 3.7 were similar for all locators. Based on this approximation of model parameters for locator with average of estimated model was considered as viable. For spatial illustration of directional behaviour of model parameters refer to Figure 3.8.

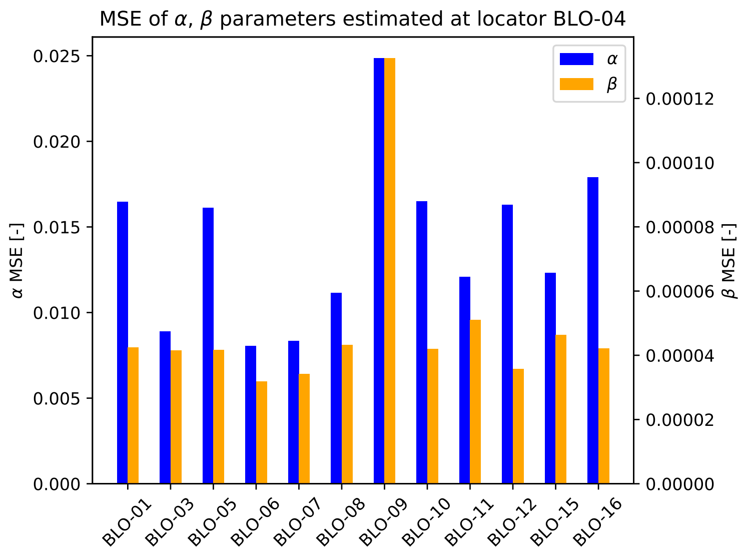


Figure 3.7: MSE of α, β parameters estimated at locator BLO-04 based on advertisement packets received from individual locators

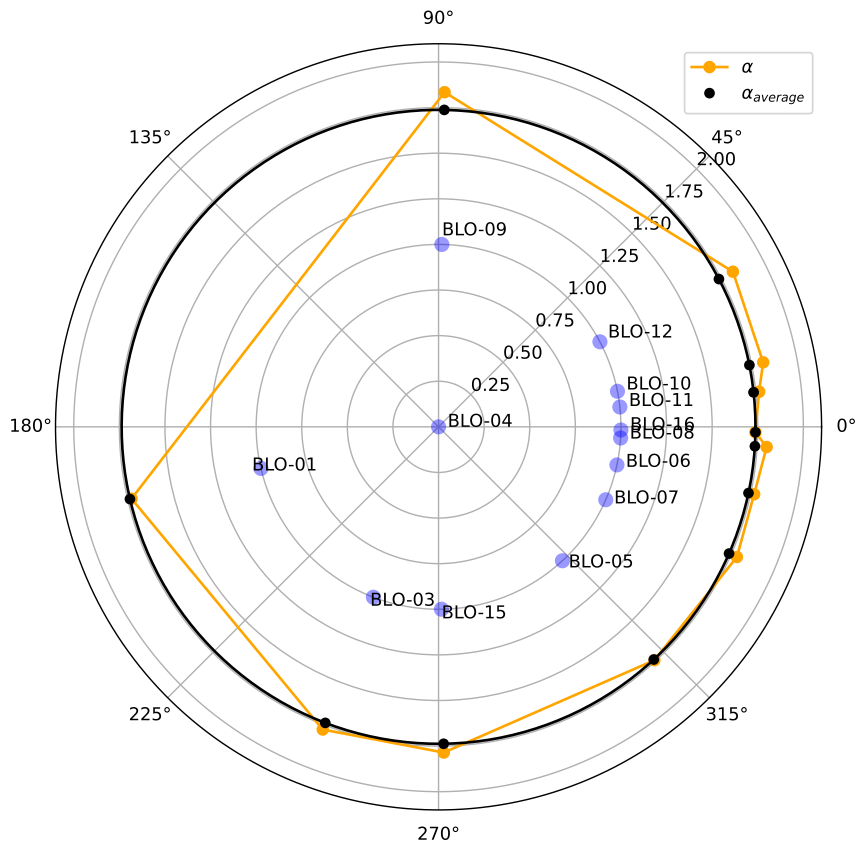


Figure 3.8: Polar plot centred at locator BLO-04 depicting directionality of α parameter

With regards to the usage of locators for estimation of signal propagation environment there was also need to address difference in vertical position of locators and tags. All locators were mounted in similar height around 2.75m above ground. For intended usage of the localization system tags were expected to be most of the time in between 1 m and 1.5 m above ground. This means communication between two locators passes through a slightly different part of the space than communication between tag and locator. For both scenarios path loss models fit with data gathered over long period of time were compared. As shown in Figure 3.9, the difference was rather minor.

With behaviour similar over long period of time, short time differences were assessed. It was expected that runtime estimated model would be able to perform better as it would compensated for temporal changes in environment. In Figure 3.10 the variance of parameters can be seen. However, when compared for distance calculation, the difference between the two models is not large enough to be considered significant.

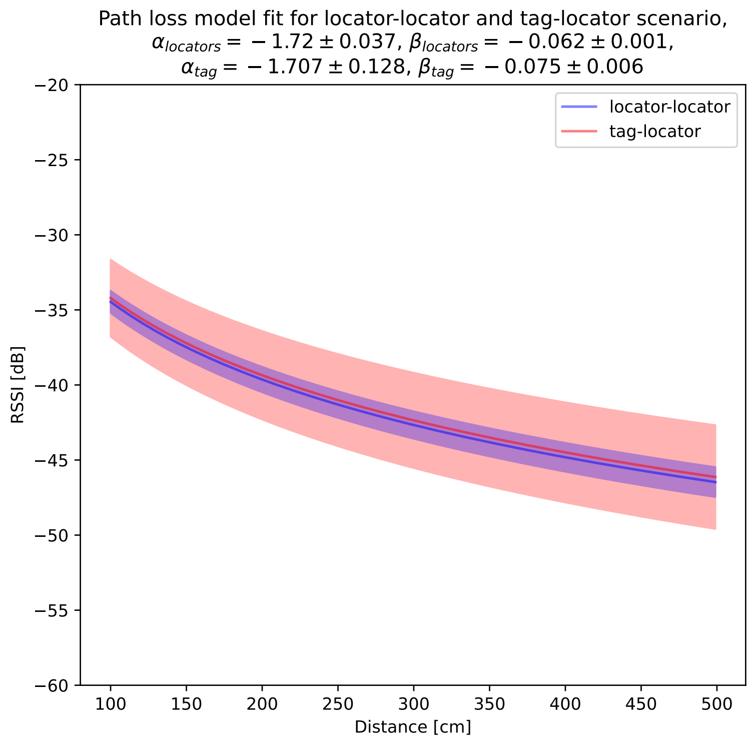


Figure 3.9: Comparison of path loss models fits based on locator-locator and tag-locator communication

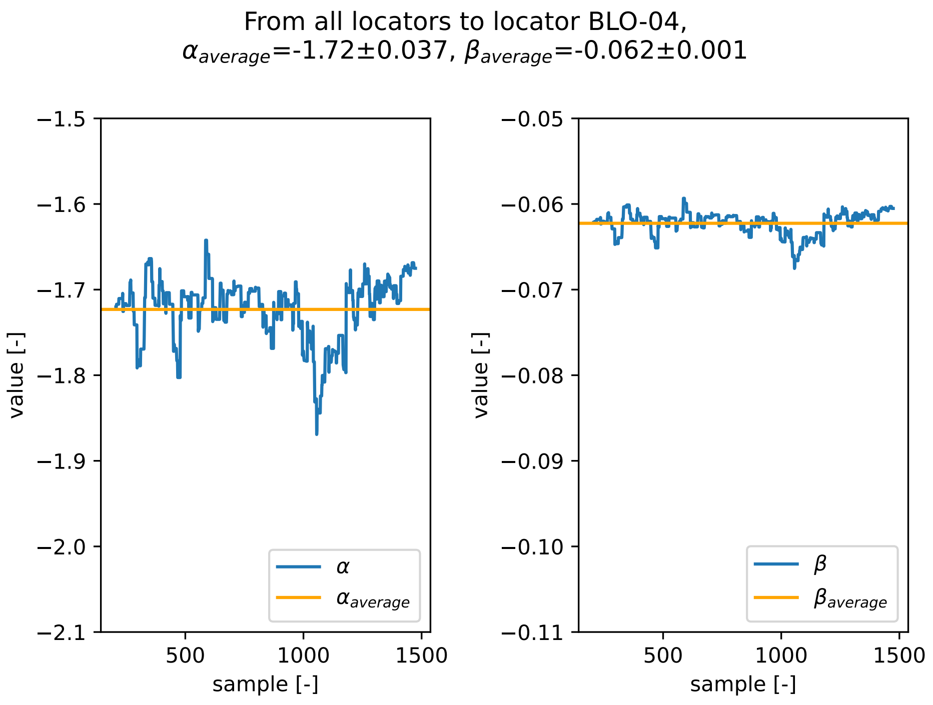


Figure 3.10: Course of α, β parameters estimated during runtime

To summarize, although this method proved to be unable to increase accuracy of localisation noticeably, it achieves similar results as with manually gathered and calculated dataset. With regard to requirements set on the localization system, the benefit of automatic path loss model parameters estimation was considered important to overall system. This method was used for all implemented variants of the localisation system based on ranging measurement principle.

Chapter 4

Implementation

This chapter focuses on design and implementation of a whole localisation system. Basic system architecture, which was expanded on to meet requirements on modularity and real-time operation, is depicted on Figure 4.1. In following section inner working of each system part is described in detail along with used third-party tools.

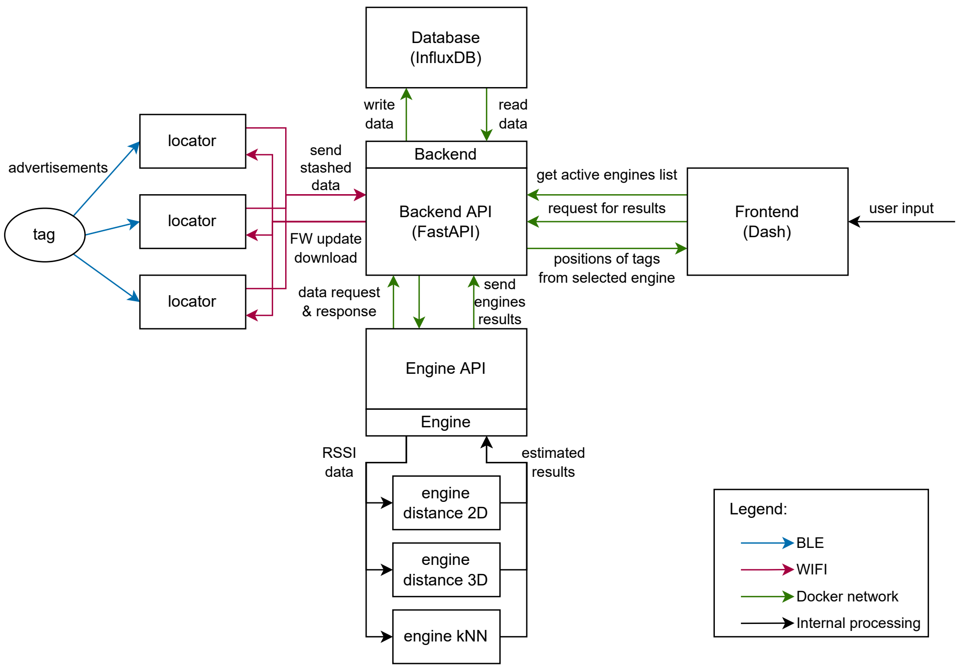


Figure 4.1: Detailed architecture of designed localisation system

4.1 Locators firmware

Based on the proposed localisation system architecture locators needed to be able to continually receive and filter BLE advertising messages. Pre-process, stash and subsequently sent them to server with given period. Possibility of remote update of configuration was deemed as necessary to allow adjustments after deployment. Optional capabilities were remote update of firmware itself and remote access for testing purposes.

To implement desired functionality consideration was made about suitable Software Developing Kit (SDK). There are several SDKs available for the ESP32C3 around which the locators are built. Two major ones are C SDK and MicroPython. Decision was made to use MicroPython. It's limited in capabilities and slower in runtime than C. This cons may potentially lead to rewriting the firmware in C in the future. However, for purposes of this work, MicroPython was deemed more suitable. It takes significantly less time to set up its toolchain and implement desired functionality. Moreover, useful functions as remote update and remote access are simple to implement. Especially possibility of remote update was consider beneficial for size of presented installation and requirement on easy deployment.

4.1.1 MicroPython

MicroPython is an implementation of the Python 3 programming language optimised to run on a microcontrollers. Firstly created for STM32 platform, but over time spread to most of popular microcontroller platforms. It aims to be compatible with standard Python 3. However it is necessary to keep in mind that it is still simplified solution with its limitations. From the range of a standard Python libraries only small subset of functions is currently implemented. With its relatively short history and usage not aimed for industrial applications, the documentation and support resources are not extensive. Especially with usage on bare metal, good knowledge of principles in firmware programming is necessary to produce robust application.

After powering on embedded device with MicroPython on it, it firstly initialize its interpreter and simple shell for user input. Afterwards filesystem is checked for "boot.py" file, which if present is then executed. If not, system is waiting for user input in shell. The interpreter behaves in a same way as one of standard Python 3 on PC. Each inserted command is executed and result returned to user. Interaction with filesystem as upload or remove file can be performed with shell commands. This can be made easier with using IDE, for example Thonny.

4.1.2 Operation

Firmware application was separated into 3 executable scripts, *boot.py*, *update.py* and *main.py* with additional libraries, config files and log file. The filesystem is depicted in Figure 4.2, where *app/* folder contains main application with its configuration and *lib/* folder contains implemented libraries.

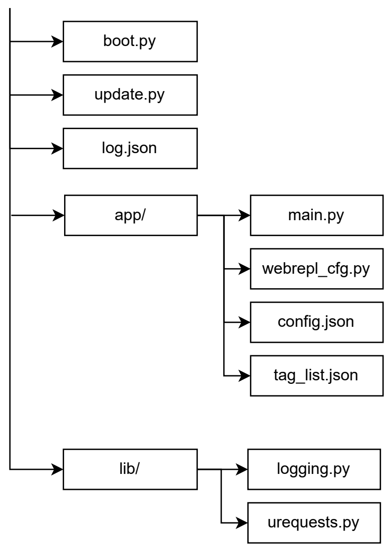


Figure 4.2: MicroPython filesystem overview

Workflow of the *boot.py* is depicted in diagram Figure 4.3. It's checks if update flag is set and either runs the *update.py* script, or *main.py* script with main functionality. If an error is encountered in any of the scripts, the error counter is incremented and device restart is triggered. When error counter reaches its limit, device enters failsafe mode in which it limits its functionality only to remote access through WebREPL shell.

update.py workflow is in Figure 4.4 diagram. It performs update of the firmware by exchanging files in folders *app/* and *lib/*. This means application code, its configuration and libraries can all be remotely updated.

Key functionality is implemented in *main.py* script with workflow shown in Figure 4.5. The application first loads its configuration from file, which besides storing information about firmware version and local network address allows to set following parameters:

- logging verbosity,
- number of records kept in log file,
- BLE functionality timings,

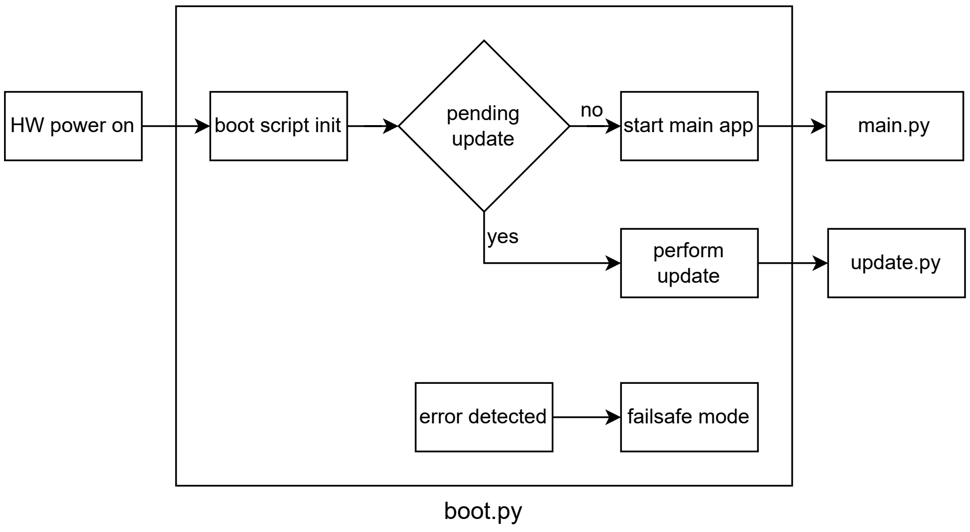


Figure 4.3: MicroPython*boot.py* workflow

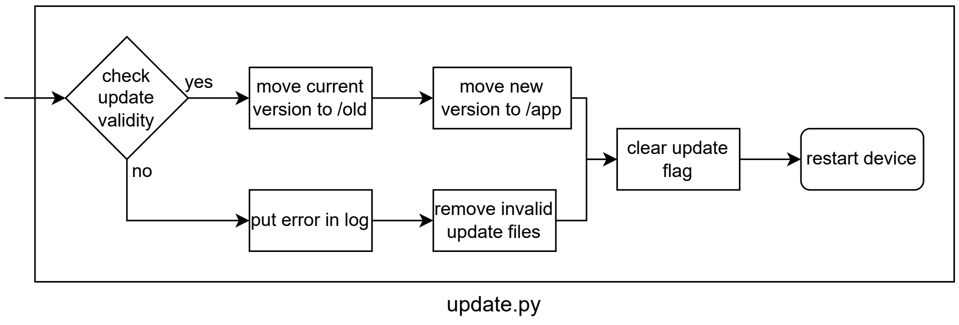


Figure 4.4: MicroPython *update.py* workflow

- period of sending data to server,
- period of sending logs to server,
- error counter limit.

With configuration loaded connection to local network via Wi-Fi is established. WebREPL, REPL shell working through Wi-Fi instead of serial port, for remote access is started. Following that Bluetooth peripheral is configured for receiving advertisement messages matching whitelist of devices specified in *tag_list.json* file. The receiving is an interrupt routine which puts received messages into queue. Both data sending and logs sending are also implemented as an interrupt routines to minimize load on the CPU. They're both triggered either with their period or when their queue is full.

Sending of received data and logs is both done over HTTP POST requests. This method is not optimal as the HTTP header is not negligible part of transmission. Steps were made to allow future change for other protocol.

However, for scope of this work HTTP requests were deemed as an acceptable solution.

As stated before, the locator can either be receiving BLE advertising messages or be sending stashed data to server. Time spent on both needed to balanced. With tag advertising period set to 1 s and desired maximum delay in localisation of 30 s, after several tests period of data sending was set to 5 s.

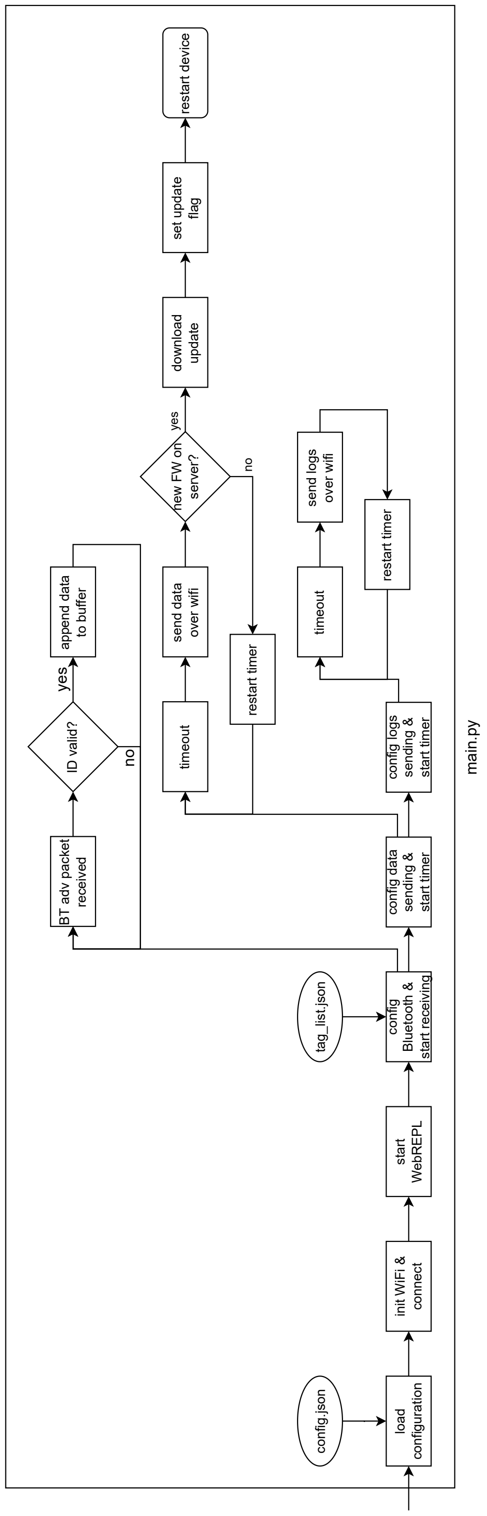


Figure 4.5: MicroPython *main.py* workflow

4.2 Server application

Server application consists of four parts, each responsible for one task in overall operation of the localisation system. Task of Backend is receiving and distributing data, Database stores data as a time series, Engine performs calculations over the data and Frontend runs website with real-time visualization of results.

In order to fulfil requirement on modularity and easy deployment decision was made to built the server application as a group of virtualized containers. This for cost of higher memory usage allowed to keep each part of the application separated, encapsulated with specifically defined interface and individually controlled. Especially individual control, possibility to modify source code and rebuilt the part of application without stopping others was deemed as beneficial during development process. Separation of this type also brings up possibility to move containers, parts of the application, to self-contained hardware (or cloud) without any other changes necessary than changing network addresses.

Most of development was done in Python programming language because of existence of high-performing third party tools and substantial previous knowledge. Docker was chosen as a containerization platform due to it wide usage and broad supporting resources. Each of the parts is described in detail in following section including description of used third-party tools.

4.2.1 Docker

Docker is a set of tools for OS-level virtualization of applications. It can encapsulate application with all of its dependencies and libraries in a virtual container. Optionally persistent data space, called volume, can be mapped to a container. These containers are run in Docker Engine and can communicate between each other through defined channels. The channels can be of several types from which networking is the most common. Its functionality is the same as with networking in a physical network. [15]

Main difference between virtual machines and virtual containers is that each virtual machine runs complete operating system, while virtual containers shares single operating system kernel. This allows containers to take up fewer resources for their operation in comparison with the virtual machines. Docker runs the operating system kernel in Docker Engine instance. [15]

4.2.2 Backend

Backend is built on FastAPI framework as it provides high-performing API for HTTP communication. Having multitude of endpoints to serve asynchronous operation was deemed necessary for Backend to operate in a sufficiently responsive manner. From FastAPI framework principles each endpoint for each type of HTTP request has its callback function with defined model of data to receive. The framework performs check on the data validity with regards to model and using standard HTTP codes communicates eventual issues with the client side.

Key endpoints for serving were */advertisements/locators* for receiving data from locators, */advertisements/tags* for providing data to Engine and */location/tags* for receiving estimated locations of tags from Engine and providing them to Frontend. Besides them many supporting endpoints for providing information were hosted. As to reduce communication with locators and complexity in its firmware, decision was made to limit it as a one-way communication only. Backend can't by itself communicate with a locator, it can only answer on locators requests and uses HTTP responses to transmit information to a locator. The responses contains current firmware version available for download. If locator receives in a response firmware version newer than its own, it automatically downloads the new version form hosted static files on the Backend and updates itself as described in a chapter 4.1.

FastAPI

FastAPI is a web framework for building RESTful APIs in Python. It's developed around Pydantic data models, which makes validation, serialization and deserialization of data simple. The data models specify structure, variable names and data types. With this construction any data transfer is straightforward. Another feature of FastAPI is automatic data model documentation generation. FastAPI fully supports asynchronous programming and is one of the fastest Python frameworks, on par with NodeJS and Go. It can also work by itself with Static Files, mechanism of providing whole files for download, GZip compression, streaming, security and authentication methods as OAuth2 and many others. [16]

REST API

REST API is a type of API (Application Programming Interface) which follows REST (REpresentational State Transfer) architectural style. The API following REST style is called RESTful. It's one of the most common styles used in todays websites. It's constraints ensures high performance, scalability, simplicity, modifiability, visibility, portability and reliability. In a typical application client sends HTTP request (GET, POST, . . .) with header containing metadata, authorization information, etc. to a server. The server responds with state representation of requested resource in one of supported formats. The most common is JSON (JavaScript Object Notation), both human and machine readable file. [17]

4.2.3 Engine

Part of the server application referred to as Engine is responsible for processing of data and estimating locations of the tags from them. Architecture of the Engine is depicted in Figure 4.6. There's separate *Engine_API* class for communication and distribution of data. Processing is performed by implementations of *Engine* class. This separation allows simple addition of various algorithms for location estimation and their simultaneous operation.

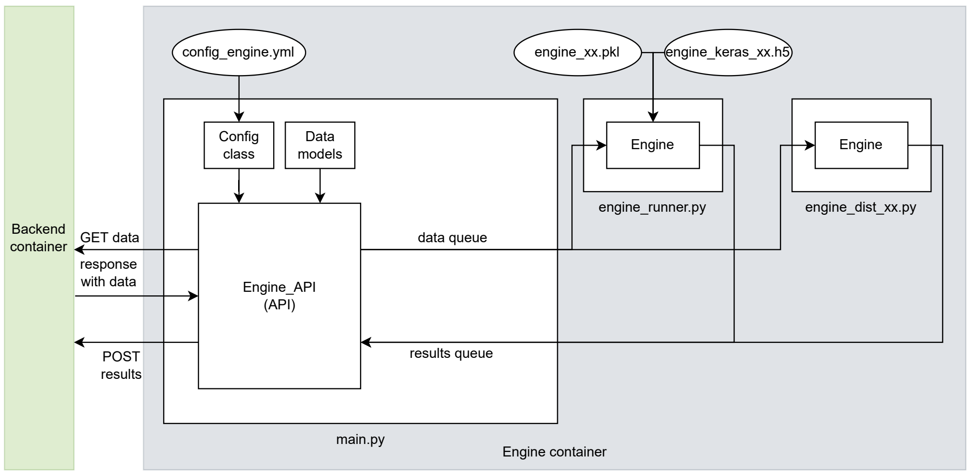


Figure 4.6: Engine architecture diagram

Basic operation flow as depicted in Figure 4.7 is following, *Engine_API* is initialized with loaded configuration from file which sets time periods, endpoint address and list of engines to run. Each listed engine has stated the corresponding implementation of *Engine* class. This means pretrained algorithms which differs only in their values are run through same *Engine* class. If different pre-processing is required, *Engine* class may be reimplemented to matched it. Algorithms which doesn't need training and storing its model are implemented directly.

Each engine is run in a separate thread for better performance and to force decoupling from rest of the Engine container application. The decoupling is important for code quality and for better possibilities in future usage with regards to extending the localisation system. With engines initialized, main loop of operation is stared. In this state the API class periodically requests data for configured timespan from the Backend. When data are received, they're put into the data queue, from which the engines can only read to ensure immutability. The individual engines pre-process data for their specific algorithm, calculate result and append it to the results queue. The API takes results from the queue, packs them and sends them to Backend for storing in the Database.

In scope of this work only approaches using multilateration and traditional classifiers from Scikit-Learn library were explored. However, interface for Keras neural network model was implemented for future usage.

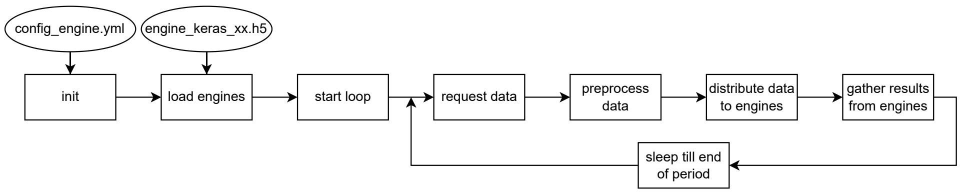


Figure 4.7: Engine workflow

Scikit-Learn

Scikit-Learn, also know as a sklearn, is a free open-source Python module. It features various classification, regression and clustering algorithms in both unsupervised and supervised variants for medium-scale problems. Design of it puts emphasis on ease of use, performance, API consistency and interoperability with the Python numerical and scientific libraries NumPy and SciPy. Over the past decade with rising popularity of machine learning, it has become the starting point capable producing quality results. [18]

4.2.4 Frontend

For better understanding of results in the real time, simple visualization website was created using Dash Open Source framework. Key advantage with using website instead of an application is accessibility from all devices connected to the local network. With current standards in modern web frameworks, it's out of the box compatible with displaying on mobile phones allowing simple real-time testing of the localisation system.

Floorplan generated from diagrams.net application using developed tool server as a map for displaying location results of tags. The map window has zoom and pan functions. Filtration of tags for visualization is implemented with multiple selection box. Results shown are from selected engine and are periodically updated through requests on Backend API.

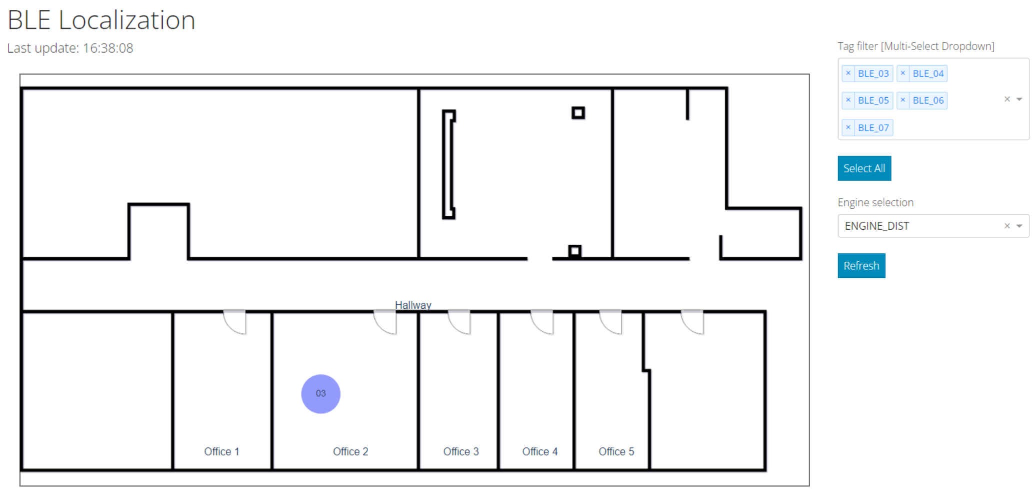


Figure 4.8: Frontend visualization

Dash Open Source

Dash Open is an open-source library hosting low-code web framework for building dashboard applications in Python, R and Julia. It is written on top of JavaScript graphical engine Plotly.js bringing responsivity, performance and ease of use. It abstracts away various technologies and protocols that are required to build a full-stack web app with interactive data visualization. [19]

4.3 Performance

In this chapter performance of all implemented localization engines is evaluated based on recorded datasets. Comparison is made with regards to accuracy, speed and usability in real world deployment.

Although key aspect of this work was to design real-time localisation system, thorough evaluation of performance can be performed only on a recorded dataset. For this purpose two datasets were created. Both with a one tag carried by person through selected waypoints with stopping on each for either 10 s for first dataset or 30 s for second one. This was done to compare besides localisation accuracy the time needed to estimate location.

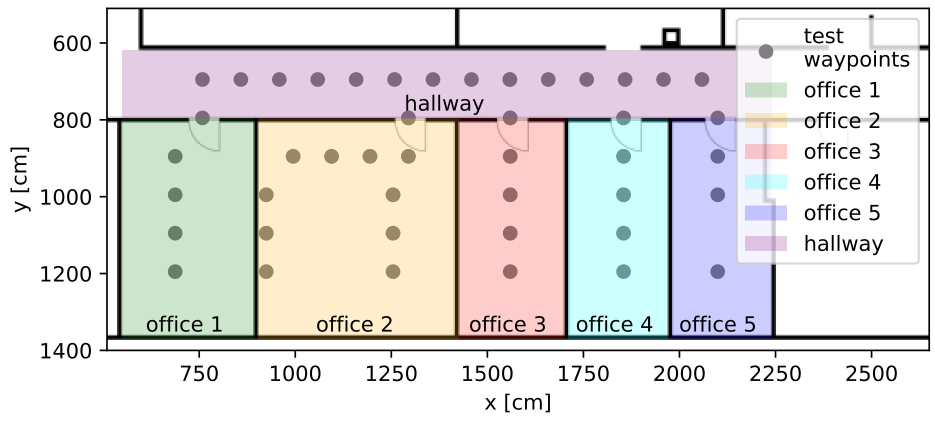


Figure 4.9: Plan of area with highlighted rooms and test datasets waypoints

4.3.1 Proximity variant 2D

To summarize workflow of this variant, RSSI from advertisement messages is gathered over 5s period and sent to the server. There engine every 2s recalls data received in past 30s over which pre-processing is performed. In this variant pre-processing consist of median filtration with window of length $w_{len} = 9$ and calculation of mean from filtered values. This is done for data from each locator. Obtained mean values are then interpreted using Equation 4.1 as a proximity to locators. Due to inaccuracy of this proximity metric, higher number of values doesn't lead to higher, on the contrary it leads to rise in error. From the proximity values is thus taken only subset of 5 lowest over which multilateration task is solved with unbounded least square method as described in chapter 3.2. Values for parameters α and β are taken from periodic estimation of path loss model parameters described in chapter 3.3.

$$d = 10^{\frac{RSSI_{mean-\beta}}{10\alpha}} \tag{4.1}$$

4.3.2 Proximity variant 3D

Variant with 3D position differs from previously described 2D variant only in process of estimating location from proximity values. This variants follows equation 3.8 and uses bounded least squares method with bounds set to borders of covered area.

4.3.3 Nearest neighbours variant

To complement main approach to localisation task nearest neighbours method was also tested. It is a widely used classification method, especially in a variant k-nearest neighbours (k-NN). As stated in research section, it's common approach to localisation task. From this reason it was chosen as a benchmark to main proximity approach.

Classification with k-NN is performed by calculating metric between classified vector and all vectors in a labelled dataset. Then k closest vectors from dataset is taken and label with highest quantity is a result of classification. Training phase for k-NN consist only of storing dataset of vectors with its labels. [20]

On a gathered dataset kNN was trained for various values of k parameter. The highest accuracy was for $k = 5$.

4.3.4 Comparison

Firstly to made comparison between proximity variants with regards to precise position MSE was calculated. From results shown in Figure 4.10 follows that 3D variant performs better although it's precision is still quite low. If represented as distance error, it varies from 3 to 5m. Another observation is made about difference with regards to 10s and 30s dataset, it appears that both methods performs better with longer time. This probably follows from slow frequency of the tag operation and subsequent long processing windows in order to compensate for errors in RSSI values. Thus with longer time the methods used can compensate for error and variance in RSSI better.

With established inability to estimate precise location, attempt was made to estimate correct room in which a tag is located based on proximity measure. Conversion of location estimate to room level was made simply through occurrence in area of the room. Results for this method are in Table 4.1.

Achieved accuracy is relatively low across all variants and test datasets. As

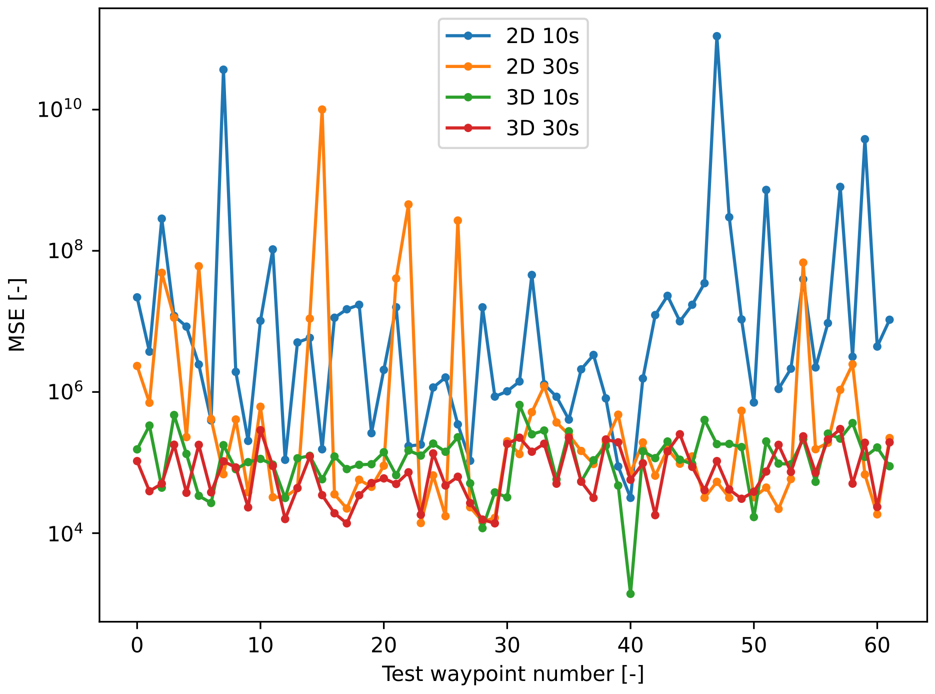


Figure 4.10: MSE of 2D and 3D proximity variants on 10s and 30s test datasets

Variant	Dataset	Accuracy [%]						
		Office 1	Office 2	Office 3	Office 4	Office 5	Hallway	Total
2D	10s	3	24	33	5	0	5	11
2D	30s	39	88	12	60	27	16	45
3D	10s	44	46	27	21	27	12	28
3D	30s	71	91	16	60	5	32	53
kNN	10s	21	94	13	15	0	16	35
kNN	30s	10	100	14	0	5	0	40

Table 4.1: Accuracy of room estimate

for 2D variant performance on 10s dataset, with regards to its MSE, it fails to estimate location with error lower than size of rooms and thus fails even as a room indicator. For 30s dataset it performs significantly better even achieving 88% accuracy on *Office 2*. Its lowest result is for *Office 3*, which looks to be problematic for all variants. This is probably due to imperfect placement of a locator in the room and proximity metric not resulting to close enough values. Moving to 3D variants, their performance was better with same difference between 10s and 30s datasets.

If considered achieved accuracy with regards to positions of test waypoints and locators, the scenario probably wasn't favourable for location engine variants based on multilateration. The multilateration approach with regards to proximity metric used in this work, would performed better for cases where test point is in middle of several locators. This pattern can be seen throughout

Figures 4.11, 4.12, 4.13 and 4.14. In the same way good performance can be expected from locations inside *Office 2*, because of 4 locators in it, and inside *Office 4* as it also lays in between 4 locators.

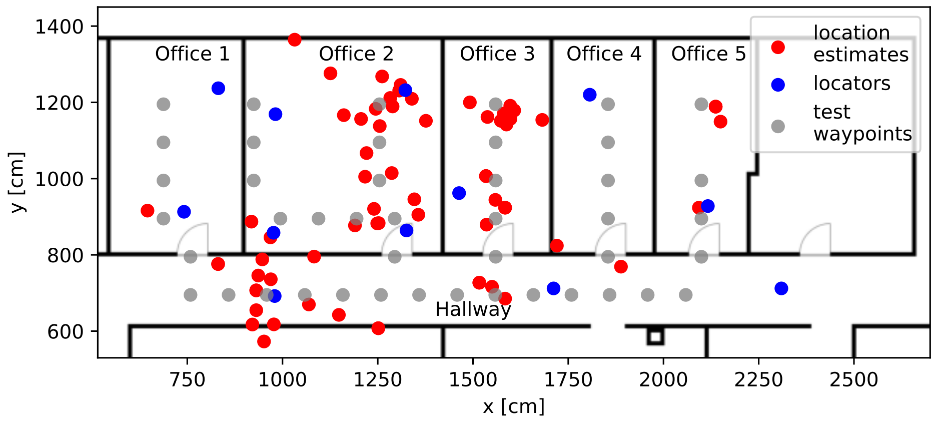


Figure 4.11: Floorplan with position results from 2D proximity variant on 10s dataset

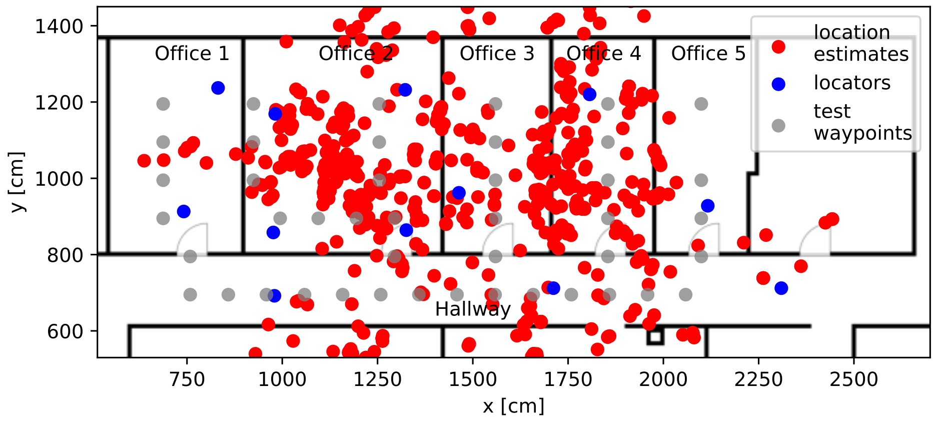


Figure 4.12: Floorplan with position results from 2D proximity variant on 30s dataset

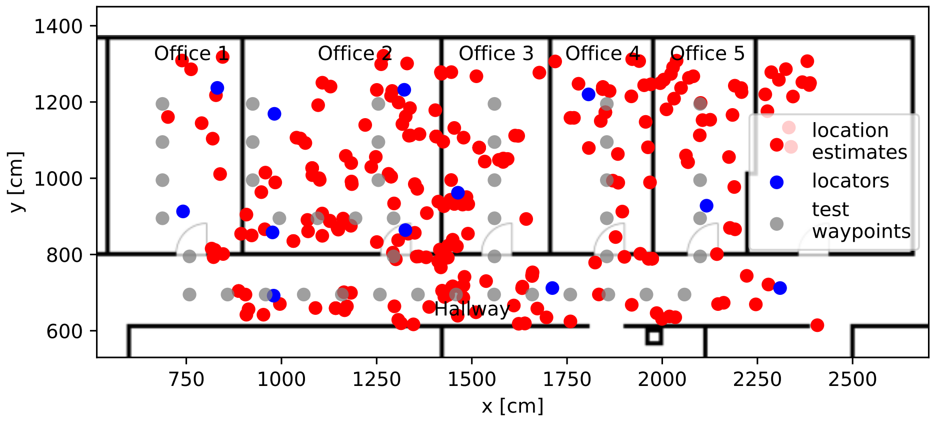


Figure 4.13: Floorplan with position results from 3D proximity variant on 10s dataset

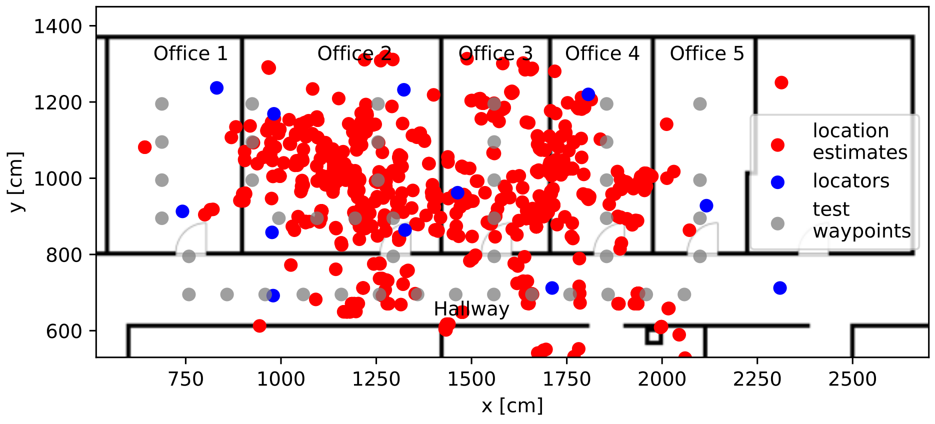


Figure 4.14: Floorplan with position results from 3D proximity variant on 30s dataset

Chapter 5

Conclusion

Aim of this work was to design and implement a Bluetooth based real-time indoor localisation system. To meet this objective a research of existing work and principles of localisation systems was conducted. Based on it the requirements for desired localisation system were set along with constraints given limiting conditions. Decision was made to use basic approach of signal strength measurement through RSSI indicator with adjusted path loss model for task of estimating location. Automatic method for continuous path loss model adjustment was designed and proved to be better performing than manual calibration of the model. From initial experiments emerged several problems caused by high variance of RSSI in indoor environment. Considerate that the resulting localisation system was designed and implemented in a modular and highly adjustable manner for it to be able to accommodate different approaches in the future. The emphasis on ease of deployment has been supplemented with implementation of utilities for translation of information from drew floorplan to the localisation system application. Following that the requirements on real-time operation, modularity, extendibility and ease of the localisation system deployment were met.

Experimental installation was built in an office space covering over 120 m^2 and 6 rooms. Proposed trilateration method of estimating location was implemented in a 2D and 3D variant. A machine learning approach was also implemented as the reference for the proposed method in task of room classification. Detailed testing datasets were created for evaluation of precise location accuracy and room classification accuracy. Achieved results were mediocre for both variants of trilateration method and also for the machine learning method. The best performing variant, 3D trilateration, achieved precision of location estimate between 3 m and 5 m. This is in line with abilities of commercially available localisation systems based on RSSI. Subsequent extension of it for room classification achieved overall accuracy of 53%.

Resulting relatively low accuracy was considered as a consequence of inadequate placement of the locators with regard to principle of trilateration method. Similarly, chosen waypoints for testing datasets proved to be borderline for combination of chosen method and the locators placement. Contemplating that the results of experimental evaluation represents performance in the worst case scenario and was deemed sufficient. However, further testing with different placement of locators is advised.

5.1 Next steps

For implemented approaches to localisation task, it is advisable to focus on the locator placement optimisation. Software tools created in this work facilitates process of it. With the localisation system implementation favourable to other approaches to localisation task, this direction should be also explored. Preferable machine learning methods with ability to simultaneously map and localize, to preserve ease of deployment benefit, should be of priority. As for software tools, with implemented utilities for feasibility of the system deployment, complementary tool for extensive dataset gathering and annotation should be implemented. This would allow broader usage of machine learning methods for data processing and thorough testing of any approach deemed applicable.

Appendix A

Bibliography

[1] MathWorks, “BLE channels overview.” `https://www.mathworks.com/help/bluetooth/ug/bluetooth-le-channel-selection-algorithms.html`. Accessed: 2023-01-06.

[2] Quappa, “RTLS Systems Overview.” `https://www.quuppa.com/wp-content/uploads/2021/08/Technology-comparison-Quuppa.png`. Accessed: 2023-01-06.

[3] Series P, “Recomendation P.1238: Propagation data and prediction methods for the planning of indoor radiocommunication systems and radio local area networks in the frequency range 300 mhz to 450 ghz,” *Recommendation ITU-R*, 2011.

[4] F. Zafari, A. Gkelias, and K. K. Leung, “A survey of indoor localization systems and technologies,” *IEEE Communications Surveys and Tutorials*, vol. 21, no. 3, pp. 2568–2599, 2019.

[5] Bluetooth Special Interest Group (SIG), “Bluetooth core specification version 5.2 feature overview.” `https://www.bluetooth.com/bluetooth-resources/bluetooth-core-specification-version-5-2-feature-overview/`, 2019. Accessed: 2023-01-06.

[6] J. Röbesaat, P. Zhang, M. Abdelaal, and O. Theel, “An improved ble indoor localization with kalman-based fusion: An experimental study,” *Sensors*, vol. 17, no. 5, p. 1, 2017.

[7] Y. Shen, B. Hwang, and J. P. Jeong, “Particle filtering-based indoor positioning system for beacon tag tracking,” *IEEE Access*, vol. 8, pp. 226445–226460, 2020.

[8] B. Huang, J. Liu, W. Sun, and F. Yang, "A robust indoor positioning method based on bluetooth low energy with separate channel information," *Sensors*, vol. 19, no. 16, 2019.

[9] X. Qiu, B. Wang, J. Wang, and Y. Shen, "Aoa-based ble localization with carrier frequency offset mitigation," *IEEE Access*, pp. 1–5, 2020.

[10] C. Xiao, D. Yang, Z. Chen, and G. Tan, "3-d ble indoor localization based on denoising autoencoder," *IEEE Access*, vol. 5, pp. 12751–12760, 2017.

[11] Espressif Systems, "ESP32-C3-MINI-1 Datasheet v1.3." `https://www.espressif.com/sites/default/files/documentation/esp32-c3-mini-1_datasheet_en.pdf`, 2022. Accessed: 2023-01-06.

[12] V. Cantón Paterna, A. Calveras Augé, J. Paradells Aspas, and M. A. Pérez Bullones, "A bluetooth low energy indoor positioning system with channel diversity, weighted trilateration and kalman filtering," *Sensors*, vol. 17, no. 12, 2017.

[13] M. A. Branch, T. F. Coleman, and Y. Li, "A subspace, interior, and conjugate gradient method for large-scale bound-constrained minimization problems," *SIAM Journal on Scientific Computing*, vol. 21, no. 1, pp. 1–23, 1999.

[14] Q. xing Huang, F. Wang, J. hua Yan, and Y. Chi, "A two-step discrete method for reconstruction of temperature distribution in a three-dimensional participating medium," *International Journal of Heat and Mass Transfer*, vol. 55, pp. 2636–2646, apr 2012.

[15] Docker, Inc., "Docker." `https://www.docker.com`. Accessed: 2023-01-06.

[16] S. Ramírez, "FastAPI." `https://fastapi.tiangolo.com/`. Accessed: 2023-01-06.

[17] R. T. Fielding, *Architectural Styles and the Design of Network-based Software Architectures.* PhD thesis, UNIVERSITY OF CALIFORNIA, IRVINE, 2000.

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[19] Plotly, Inc., "Dash Open Source." `https://dash.plotly.com`. Accessed: 2023-01-06.

[20] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.