NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

ANDRÉ LOURENÇO NUNES E SILVA

BSc in Electrical and Computer Engineering Sciences

LOCATION ENGINE BASED ON RECEIVED SIGNAL STRENGTH INDICATOR APPLIED TO SCHOOL AND OFFICE BUILDINGS

MASTER IN ELECTRICAL AND COMPUTER ENGINEERING

NOVA University Lisbon
September, 2024

NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

DEPARTMENT OF ELECTRICAL
AND COMPUTER ENGINEERING

LOCATION ENGINE BASED ON RECEIVED SIGNAL STRENGTH INDICATOR APPLIED TO SCHOOL AND OFFICE BUILDINGS

ANDRÉ LOURENÇO NUNES E SILVA

BSc in Electrical and Computer Engineering Sciences

Adviser: Filipe de Carvalho Moutinho
Assistant Professor, NOVA University Lisbon

Co-adviser: David Abreu
IoT Engineer, Crowdkeep Portugal

Examination Committee

Chair: João Pedro Abreu de Oliveira
Associate Professor with Habilitation, NOVA University Lisbon

Rapporteur: Luís Filipe dos Santos Gomes
Associate Professor with Habilitation, NOVA University Lisbon

Member: Filipe de Carvalho Moutinho
Assistant Professor, NOVA University Lisbon

MASTER IN ELECTRICAL AND COMPUTER ENGINEERING

NOVA University Lisbon
September, 2024

Location Engine Based on Received Signal Strength Indicator Applied to School and Office Buildings

This document was created with the (pdf/Xe/Lua)LaTeX processor and the NOVAthesis template (v7.1.7) [1].

Acknowledgements

I would like to express my gratitude to my advisor, Professor Filipe Moutinho, for accepting this topic and for providing invaluable guidance and support throughout the project, and the availability and responsiveness in addressing my continuous questions as quickly as it could. I would also like to express my gratitude to all members of Crowdkeep for welcoming me so warmly during my time with them. I am particularly indebted to my co-advisor, David Abreu, and also Rui Fernandes for their invaluable guidance and support from the moment I joined the project.

Furthermore, I would like to express my gratitude to my family, particularly my parents and sister, for their patience and support during the most challenging periods of the project. They have consistently encouraged me to pursue my objectives since my early years, which have ultimately led to my current position.

Finally, I would like to express my gratitude to all my colleagues at AEFCT, in particular Joana, Catarina, João, Diogo, Gonçalo, Marta, Frederica, Pedro, and Guilherme. Their emotional support and encouragement were invaluable during this challenging period, and I am indebted to them for their support.

Abstract

The primary purpose of this work is to develop a location engine capable of providing real-time location data with an accuracy of approximately 3 to 5 meters. This thesis explores the development of a location engine based on Received Signal Strength Indicator (RSSI), specifically applied to multi-storey school and office buildings. The research addresses the challenge of achieving accurate indoor location in different environments where traditional outdoor solutions, such as GPS, are ineffective. While various methods such as *Time Difference of Arrival* (TDoA) and *Angle of Arrival* (AoA) have been explored in previous research studies, RSSI-based solutions are a promising alternative due to their compatibility with existing infrastructure and low energy consumption.

The methodology involves a phased development approach, starting with local testing of various algorithms, that apply different location methods in controlled environments, followed by their integration into a cloud-based system. This process ensures that the chosen algorithm is not only accurate but also adaptable to different environmental conditions. The final system's effectiveness was validated through a series of tests, demonstrating its ability to provide reliable location data within the specified environments.

The results indicate that the engine developed meets the desired levels of accuracy, with an accuracy of around 4 meters, making it a valuable contribution to the real-time location systems. This work led to a result that proved to be adaptable and flexible in the tests carried out (in 6 different locations), and scalable when adding new locations.

Keywords: Indoor Location, Real Time, Location Engine, Received Signal Strength Indicator, Internet of Things

Resumo

O principal objetivo deste trabalho é desenvolver um mecanismo de localização capaz de fornecer dados de localização em tempo real com uma precisão de aproximadamente 3 a 5 metros. Esta tese explora o desenvolvimento de um mecanismo de localização baseado no Indicador da Intensidade do Sinal Recebido (Received Signal Strength Indicator (RSSI), na sigla inglesa), especificamente aplicado a edificios escolares e de escritórios, e de vários andares. A investigação aborda o desafio de conseguir uma localização interior precisa em diferentes ambientes, onde as soluções exteriores tradicionais, como o GPS, são ineficazes. Embora vários métodos como o *Time Difference of Arrival* (TDoA) e o *Angle of Arrival* (AoA) tenham sido explorados em investigações anteriores, as soluções baseadas no RSSI constituem uma alternativa promissora devido à sua compatibilidade com a infraestrutura existente e ao baixo consumo de energia.

A metodologia utilizada envolve uma abordagem de desenvolvimento faseada, começando com testes locais de vários algoritmos, que aplicam diferentes métodos de localização em ambientes controlados, seguido da sua integração num sistema integrado na cloud. Este processo garante que o algoritmo escolhido não só é adequado, mas também adaptável a diferentes ambientes. A eficácia do sistema final foi validada através de uma série de testes, demonstrando a sua capacidade de fornecer dados de localização fiáveis nos ambientes especificados.

Os resultados indicam que o mecanismo desenvolvido cumpre os níveis de precisão desejados, com uma precisão de cerca de 4 metros, tornando-se uma contribuição valiosa para os sistemas de localização em tempo real. Este trabalho originou um resultado que se mostrou adaptável e fléxivel nos testes realizados (em 6 locais diferentes), e escalável comprovado pela adição de novos locais.

Palavras-chave: Localização Interior, Tempo Real, Motor de Localização, Indicador da Intensidade do Sinal Recebido, Internet das Coisas

Contents

List of Figures

List of Tables

Acronyms

A-GPS Assisted-Global Positioning System
AoA Angle of Arrival

BLE Bluetooth Low Energy

CSS Chirp Spread Spectrum

GNSS Global Navigation Satellite System
GPS Global Positioning System
GRoA Gain Ratios of Arrival

IoT Internet of Things

NLOS Non-Line-of-Sight

RFID Radio-Frequency Identification
RSSI Received Signal Strength Indicator
RTLS Real-Time Location System

TDoA Time Difference of Arrival

UWB Ultra-WideBand

1

Introduction

In this chapter the introduction for the development of a Location Engine is going to be stated. Layering its foundation by reporting the motivation for this work, as well as its context and objectives that it aims to solve, followed by the approach and contributions to be achieved with its conclusion, finishing with the structure of the document.

1.1 Motivation

With advances made regarding the context of Internet of Things (IoT), the struggle to keep up the pace with what could be the best option for the specific problem that someone aims to solve, grows exponentially. For example, by taking a look at the number of existing communication protocols, "it is quite hard to conclude which one is perfect" [2]. Diving into a more specific topic inside the IoT such as location systems, the selection of a location method adds up on the number of variables needed to get on the right track for the development of the system. Accuracy, energy efficiency and scalability are only some examples that can occur but the question arises, are there any others that are just as important?

When it comes to locating a person, a vehicle or just a device, whether inside or outside a building, it can seem easy to solve for an output of "only" two or three coordinates (depending on the dimensions of the referential). Techniques such as Time Difference of Arrival (TDoA) and Angle of Arrival (AoA) or the application of the Received Signal Strength Indicator (RSSI) can be difficult to implement and complex systems like Global Positioning System (GPS) are a good example of accuracy positioning for outdoor environments that relies on a lot of theory.

For indoor scenarios, the access to that mentioned accuracy is very limited, therefore, the need to develop a location engine that can handle multiple scenarios to achieve it. The engine's robustness can be achieved in a number of ways, depending on the end goal, through algorithms that are adaptable to the environment and the data it receives or through the use of different techniques, communication protocols and other concepts.

These days, there's more and more discussion when it comes to tracking objects (and people), and each example is set in a completely different environment [3–5]. Examples of potential applications include hospitals, where it can be used to locate patients or essential equipment; schools, to track children who might leave the premises; warehouses, to improve the efficiency of finding parcels; or construction sites, to quickly locate workers in case of serious accidents. With this wide variety of environments comes the need to test the ability to obtain a location solution that can meet all these objectives, without compromising the efficiency of the solution with the increase of the footprint where the system is going to act.

1.2 Context and Objectives

As we have already discussed the motivation for this work, it is necessary to understand the objectives and the context in which it is taking place, as well as the problems that may be encountered and will need to be addressed during the next stages. These topics are important to take into account before making any decisions regarding the techniques, methods, and technologies to be used during the development and testing or before pointing out what are the requirements necessary to meet.

As it might be expect, in order to obtain a result for the location, it is necessary to use an infrastructure capable of supporting not only the hardware that will be located on each of the sites that collect the information, but also the cloud infrastructure, so that the information can be stored, processed, and visualized. The infrastructure, services and all the data that will be used to support this work belongs to the company Crowdkeep.

With this in mind, this work will focus on a Real-Time Location System (RTLS) that has several sites already structured and with this infrastructure in place, requiring the development of the algorithm that will process the information and obtain a location. Within these sites there are a number of devices that the people in them use by "wearing" them and walking within the site itself, these devices are responsible for the communication with the rest of the network, and are the ones that the work aims to solve the location.

The pre-existing infrastructure for this work is based on Wirepas technology (through Bluetooth Low Energy (BLE)), which offers a decentralized wireless connectivity solution, enabling scalable and energy-efficient mesh networks. The devices deployed at six different sites are operated by long-life batteries, designed to ensure that the sensors and actuators can operate for months without the need to replace the batteries, which is crucial for operational continuity.

For this work, the final goal will be to obtain the 2 coordinates needed to describe the location of people and or assets that are located in school and office environments, inside

multi-storey buildings. This result will need to be accurate to the point of understanding whether they are within zone A or B. For this use case, there is no need to obtain a highly accurate location, where it will make a difference to the final result whether the tag is 1 or 2 meters from a specific point, but rather, for large spaces (of around $10,000m^2$ and more), in which area they are located, in which room, what floor they are on or if they are inside the building.

With this in mind, we can foresee some problems and obstacles arising during the preparation and development of this work, such as, with the information collected by the devices on each site, is it possible to apply any existing techniques? Is it necessary to develop a new one? Is there sufficient coverage so that there are no devices for which it is not possible to obtain a solution for their location when they are in a given area of the site?

1.3 Approach and Contribution

Bearing in mind that the algorithm for the location engine that will be developed is embedded in a cloud infrastructure, it is necessary to develop and use tools that can be correctly integrated into it without there being a difference between what is tested locally until it reaches the desired accuracy and what is integrated into the system. Also, given that we're talking about developing something for an RTLS, there's a need for efficient data processing so that the time interval between data collection and the output of its location doesn't stretch out too long and the system no longer has the "Real-Time" feature, which is important in this context.

The devices that will be used collect data from the site, more specifically the data that will be used to estimate their position, the RSSI values of each communication between anchors and tags. This feature will make implementation easier, as it is already implemented in the system, so there is no need to change the behavior of the devices (their firmware) or the network. The data collected is then transmitted to the cloud infrastructure, where the information is processed to obtain the location.

However, before fully integrating into the cloud, a phased development strategy will be adopted. Firstly, different location methods will be designed and tested in different environments. This local testing phase is essential as it allows rapid interaction on the methods, enabling errors to be identified and corrected more effectively and easier to make adjustments to the parameters, before eventually migrating to the cloud infrastructure.

Each method will be submitted to a set of tests to validate its accuracy and efficiency. The aim is to identify the algorithm that best suits the specific requirements of the work, such as the capability of operating in indoor environments, where absolute accuracy is less critical, and the real-time feature. That said, the cloud implementation phase will

only have to focus on integration and compatibility between the tools used locally and in the cloud, since the behavior of the algorithm itself and the expected result have already been tested. This raises the need to carry out a preliminary analysis of the structure of this cloud system.

Finally, the contribution of this work goes beyond simply developing a location algorithm. A structured approach is proposed for selecting and optimizing the most effective location method, ensuring that the final system is not only functional, but also adaptable and scalable with the addition of more sites. Therefore, the outcome of this work will provide a scalable and efficient location solution that can be adapted and applied in a variety of real-time usage scenarios.

1.4 Document Structure

After the introduction to the problem, the rest of the thesis is divided into four more chapters.

Chapter two is called “State of the Art” and serves the purpose of establishing the theoretical foundation necessary for the development of the work in question and for a better understanding of the decisions and problems encountered. The third chapter, “Location Engine”, presents and describes the structures and technologies used, as well as the steps taken during the development phase. The fourth chapter, “Tests and Results”, focuses on the analysis of the system developed, detailing the testing procedures in both local and remote environments. This chapter evaluates the accuracy and effectiveness of the location system, as well as the different conditions that can influence the results, such as distance, number of readings and environmental variables. Finally, the fifth chapter, “Conclusion”, summarizes the main findings of the thesis, discusses the limitations of the work carried out, and suggests possible directions for future improvements and further research.

2

State of the Art

The main focus of this chapter is to explore the theory that supports the topics that are going to be essential for the thesis work. Starting by looking at Internet of Things in general and exploring commonly used communication protocols in general. Location concepts and mechanisms are also going to be described, as well as studies and papers that support them. Bearing in mind that someone's location can be a considered a private type of data, depending on the context, this topic is going to be looked at. At last, presenting already deployed or studied products will finish this state of the art with a strong sense of what can and already is done.

2.1 Internet of Things

The use of the words Internet of Things (IoT) has increased exponentially in recent years, since it was first used. In 1999, Kevin Ashton used these exact words for a title of a presentation. This presentation used the meaning of these words, relating them to the use of Radio-Frequency Identification (RFID) technology and the internet [6].

The word "internet" is a generalization, it is not mandatory to have a system connected to the internet for it to become a part of IoT. For any system or project to be considered IoT, according to [7], there needs to be connected objects, trough infrastructures, collecting data and enabling their access. The objects that collect data can range from simple sensors to our smartphones. The authors of [7], also suggest dividing an IoT architecture into 4 different layers:

- Local environment: Small devices (usually sensors) that are continuously collecting data;
- Transport: Transportation of the data collected to the storage;
- Storage: Storage of the data initially collected to enable its access through multiple platforms;
- Availability: Access to every data collected, from the different sources and types of data of the system.

[8] also suggests a similar architecture, with the same purpose in mind, collecting data from the real world through a network, to be accessed for multiple purposes.

From the first mention of IoT to our current world, 23 years have passed, the access to simple electronics such as sensors, microcontrollers, and other products has become easier. Also, with the evolution of access to online classes and tutorials, learning how to collect information from the real world has also become easier, making it so that there are more and more ideas being developed each year. All of these reasons, add to the growth of the IoT in the past years. From easier to use protocols to smarter and more precise sensors, the variety of choice is almost overwhelming.

The company Statica is a well trusted platform (by more than 230 thousand companies like Google, Samsung, and other) that presents insights and facts about 170 industries. According to [9], the IoT market as grown around 700 billion US dollars in the last 6 years and is expected to reach an impressive value of over 2000 billion US dollars by 2028.

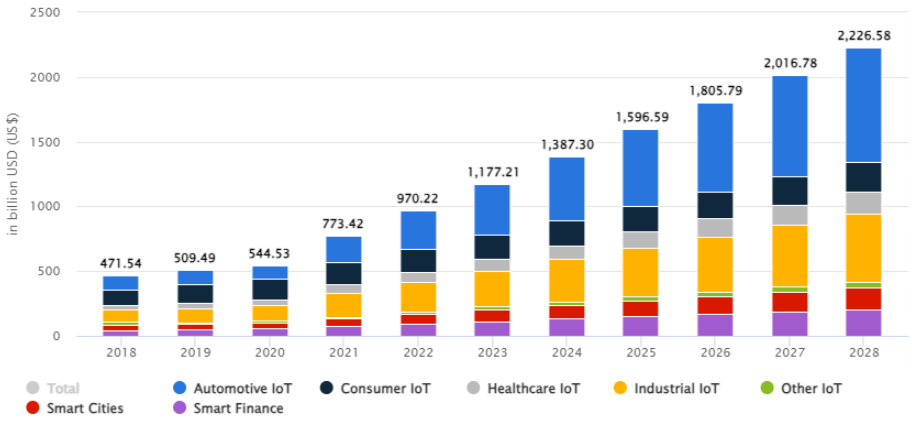


Figure 2.1: Graph of the market revenue by segment (2018 to 2028), from [9]. Last updated in September 2023.

After all that was mentioned before, it is indisputable to say that this project is part of this context. Making possible to access a referenced position of a person (or object) in a floor (or room) based on their real world location, checks all the layers on the previously mentioned architectures as we can confirm with the Figure 2.2.

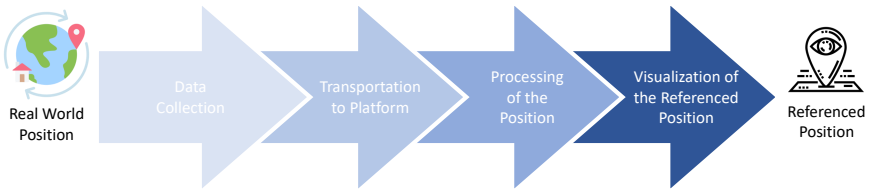


Figure 2.2: Similar Architecture of the project based on the suggestions referred

2.2 Low-Power Wireless Communication Protocols

There are currently a considerable number of Low-Power wireless communication protocols being used to deploy IoT projects and other solutions. I'm going to present a simple comparison between some of them in order to sum up, to better understand their strengths and weaknesses and for which cases they can be considered a good fit.

For that, a list of topics is going to be presented for comparison, keeping in mind that some features may not be as relevant as others for the theme of this thesis and, therefore will not be listed.

The topics that are going to be compared are, frequency at which the protocols operate, energy consumption, maximum data throughput, and the reliable range.

Table 2.1: Communication Protocols Comparison [2, 10].

Protocol	Frequency	Consumption	Throughput	Range
WiFi	2.4/5 GHz	Moderate	Up to 1Gbps	50-200 m
ZigBee	2.4 GHz	Low Power	250 Kbps	10-100 m
Bluetooth	2.4 GHz	Low Power	1Mbps	15-30 m
BLE	2.4 GHz	Very Low Power	Up to 2 Mbps	25-100 m
Z-Wave	868/908 MHz	Low Power	40 Kbps	30-100 m
UWB	3.1-10.6 GHz	Low Power	Up to 2 Mbps	10-500 m
Chirp	2.4 GHz ISM	Very Low Power	Up to 2 Mbps	10-500 m

2.3 Real-Time Location Systems

The goal of this section is to clarify what is a Real-Time Location System (RTLS), what are the requirements for a system to be considered a Real-Time Location system and what are the main features that it possesses.

As the name implies, a RTLS should be able to provide live data about the measurements the system makes [11]. However, the latency between the deployable devices, the anchors that sense their signals, the process done by the location engine and the platform, makes it almost impossible to achieve a "live" feed. Most RTLS are able to provide refreshes with less than a minute period, giving users a feeling that is very close to "Real-Time".

The location engine built in to the system is responsible for the application of concepts and mechanisms that use the signals between the devices and anchors to generate a location to be shown to the user. Some of these concepts and mechanisms will be studied in section 2.5.

The structure of a RTLS is dependent of the specifications that the developers want to meet. With that in mind, the general structure can be divided into 3 main parts, site infrastructure, backend infrastructure and visualization platform.

- Site Infrastructure: Multiple devices, usually previously installed, that communicate with each other to collect the necessary data to calculate the pretended location. This Layer is also responsible to send the data to the Backend;
- Backend Infrastructure: Consists of the Location Engine, which processes the information and calculates the desired location;
- Visualization platform: This layer will be responsible for displaying all the information and data collected and generated by the system, to the end user.

By taking a close look at all the system parts mentioned in 2.1 and the ones referred in this section, we are able to compare the two and relate them with each other. The use of small devices to collect data from the real world and bring it into the digital world via different protocols, message formats and platforms, turns RTLS part of the IoT world.

Lastly, there is one very important aspect of this type of system that needs to be mentioned. In 2015, research was made in order to discuss the performance and results of RTLS developed for construction purposes, which can be taken into the general context. The study concluded that there are many system characteristics that can be implemented, but there is not one that will fit every environment [11].

2.4 Location and Position

Throughout this document, the use of the words "location" and "position" can be confusing, so it is important to clearly define the difference between them.

According to [12] the word "location" refers to a physical or geographical point, whitin the real world or, where it is situated. While on the other hand, "position" refers to a placement relative to a reference point or system. The Oxford english dictionary uses the following meanings to specify these two words:

- Location: "The action of situating something; (also) the fact or condition of being placed; settlement in a place.";
- Position: "The place in which a person, thing, etc., is located or has been put; situation, site, station. In (also into) position: in (also into) its, his, or. . . ".

With these definitions, we can understand the two words are related but, it can still be confusing if we think that everything in the "real world" uses earth or space as a reference point.

With this in mind, the meaning of the two words will be clarified, throughout this document, so that there is no misunderstanding when using them. The word "Position"

will be used to refer the real world position, using planet earth as a reference, for example: "The device's position is at the building number 10 of the university campus". Therefore, the word "Location" will be used to identify the location solved by the location engine, method or technique, using the specific scenario as a reference, for example: "Node number one's location is inside the room Alpha of the floor plan".

2.5 Location Related Concepts and Mechanisms

In this section, a quick look will be taken at the multiple concepts and mechanisms that have already been used in location projects and studies. By looking at them from theoretical perspective and analyzing their strengths and weaknesses, a baseline will be drawn up for comparison to related work that has already been implemented. This section will also allow me to take well-formed conclusions before starting the development of the project. It is also important to highlight that this concepts can be used for a variety of purposes but, focusing on the location characteristics of each one, as well as combinations between them.

2.5.1 2D trilateration theory

Before explaining how to locate something on the Earth's surface or floor plan, a quick look at this concept in 2 dimensions will be taken. For that, we are going to consider a point, P, that we want to locate, and a 2-axis referential.

Initially, we need to determine what is required to look for. Since we are in 2D, we have to solve for the two coordinates, $P(x, y)$, that represent a given point, inside the considered referential. If we know the location of 3 different reference points beforehand (from now on referred as "anchors", A_1, A_2, A_3), and also know the distance between P and the anchors then, considering the equation:

$$(x - A_{nx})^2 + (y - A_{ny})^2 = D_n^2 \tag{2.1}$$

D_n being the distance from the anchor to P, and the equation, the Euclidean distance between 2 points in 2D [13]. By solving the system of the 3 equations (for all 3 anchors), we can find the location of P, because the only unknown variables are x and y.

In figure 2.3, we can visualize the application of this technique. The point, P location as it was expected, is situated where the 3 circumferences (representing the Euclidean distance between the 3 anchors and P) meet [13, 14].

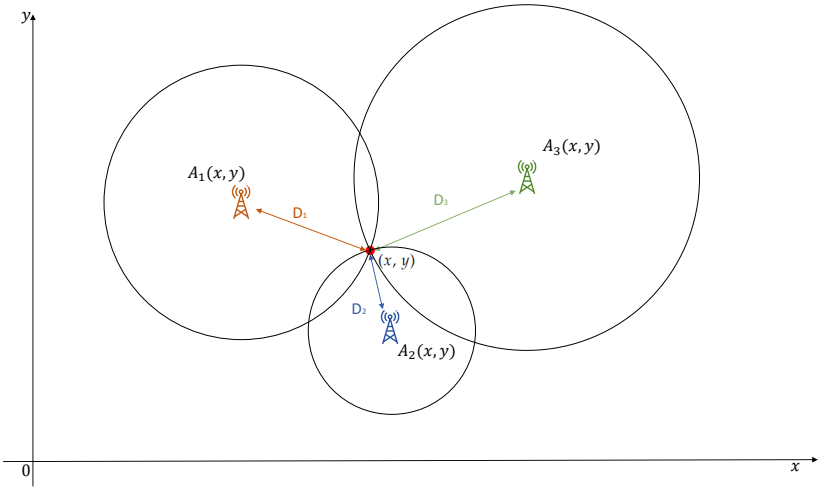


Figure 2.3: Visualization of trilateration in a 2-axis referential

2.5.2 Time Difference of Arrival

A Time Difference of Arrival (TDoA) based location method focuses on utilizing the differencing on arrival of the same signal to multiple anchors, sent from the source which position the method aims to solve [15]. However, there are more than one way to implement TDoA with additional help, that have already been tested. Because of that variety, a closer look will be taken at the ones deemed more interesting to mention and better suited for the later work.

Taking in consideration that the method utilizes precise time measurements, we can start to name some difficulties that may occur. There are multiple variables that can be unknown and therefore, increase the complexity of this mechanism, such as: real position of the anchors, non-synchronization between anchors clocks and, unpredictable noise scenarios.

For the specific scenario that it is going to be studied in this thesis, the exact position of the anchors, is known. Because of that, the changing factors it can involve and mechanisms that do not assume the same will not be considered.

The first important variable to take into account in TDoA, is the synchronization between the source of the signal and the reference stations. As it was mentioned by [16], in TDoA the measurements define a hyperbola instead of the already mentioned earlier, circle. Usually there is one of the anchors that is used as a reference, and for that, the precise time of the signal sent is not needed, clearing the problem of the clock synchronization between anchors.

In [17], the authors discuss the optimal sensor placement for TDoA localization for uncertain source location. They recommend placing the anchors as far away from the source as possible and distributing the remaining sensors evenly at equal angles.

To improve the localization accuracy, [18] tested the combination of Gain Ratios of Arrival (GRoA) (ratio of the received signal amplitudes at the referenced sensor to the

other sensors). The author came to a conclusion that it can significantly improve accuracy when the factor between the signal propagation speed and bandwidth is "large". It is also mention that the typical performance of the location algorithm can be characterized by the Cramer-Rao bound (CRB) and the Fisher information matrix (IFM) [17–19].

It is also important to present the equations for this topic using TDoA, as they were described in [19]:

$$s\Delta t_{ij} = \sqrt{(x-x_i)^2+(y-y_i)^2+(z-z_i)^2} - \sqrt{(x-x_j)^2+(y-y_j)^2+(z-z_j)^2}, i, j = 1, \ldots, N \tag{2.2}$$

Where s is the signal propagation velocity, Δt_{ij} is the time difference between travel times t_i and t_j, x_i, y_i, $\ldots$, z_j represent the position of anchors i and j. Keeping in mind that for 2D and 3D representation we need at least 3 and 4 anchors(N = 3 or 4), respectively. Finally, the terms x, y, z, are the coordinates of the source that we are solving for.

This equation is not linear, which makes the estimation of the source location, "potentially complex and expensive" [19]. The z variable can be omitted if we just want to locate something on a 2D plane, making the equation simpler and requiring 1 less variable to find the now 2 coordinates.

2.5.3 Received Signal Strength Indicator

Another possibility for source location can be achieved with the use of received signal strength and, in a wireless connection, there is a feature called Received Signal Strength Indicator (RSSI) that can be used for that purpose. The RSSI (in dB) indicates the received power level by the antenna, where higher is the RSSI level, stronger is the signal and therefore closest from the destination [20, 21].

Unfortunately, the level of the RSSI is not linear in relation to the distance between the source and the anchors. Because of the non-linearity, there are different approaches to analyze this indicator and correctly estimate the closest location possible of the source.

In order to better understand this relation between the measurement and the distance, [22], has done an evaluation the reliability around this subject.

As it is referred in [21], this indicator measurements can be related with the distance to the source by the Friis transmission equation, for an ideal case:

$$P_r(D) = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 D^2} \tag{2.3}$$

Where P_t is the transmitted power, G_t is the transmitters' antenna gain, G_r is the receivers' antenna gain and λ is the wavelength of the transmitter signal in meters. Taking in consideration that it is hard to determine the antennas gains, [20] also suggests a simplified version for the equation:

$$P_r(D) = P_{r1} - K.\log_{10}(D) \tag{2.4}$$

As we can see, this equation is considerably simpler to solve, with P_{r1} being the received power (in dBm) at one meter, K the loss parameter and D the distance that we aim to solve for. In [21], the authors also suggest a similar variation of the equation.

If the relation between the signal strength and the distance to the source of the signal can be achieved, similarly as it was explained in 2.5.1, the true location can be solved.

In addition to everything said, [23] has proposed and tested an algorithm that can achieve a considerably good location accuracy by combining the measurements of the RSSI with the use of Filters. One of the filters, The Kalman Filter, will be addressed in the next sections.

2.5.4 Angle of Arrival

According to [24], the Angle of Arrival (AoA) can be defined as the angle between the propagation direction of an incident wave and some reference direction (orientation), measured on the receiving (arrival) end of the considered device. For a device to be able to measure the angle of the signal, it needs to be equipped with an array of antennas [25], where multiple antennas sense the same signal but with different phases or a directive antenna [26], that uses its gain variation and phase difference with respect to other antenna elements in a uniform circular array.

In [21] the author specifies that the AoA measurements can be divided into two subclasses, by utilizing the receiver antenna's amplitude or phase response. Problems for both types of measurements, and ways of dealing with them are also explained in detail.

For an angle to be measured, it is also needed to consider or define a reference orientation. By considering a fixed orientation to all the devices it serves as a reference to measure all the received signals and, measure the angle in a non-clockwise direction. By doing this, all the AoA of the same signal to different reference points can be related in a later stage.

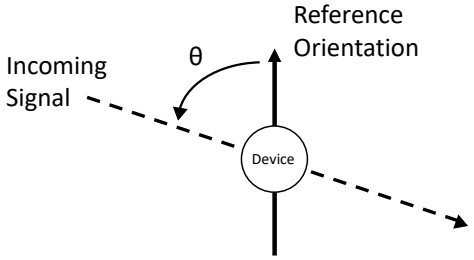


Figure 2.4: Angle of arrival, θ, measurement with reference orientation.

The AoA method can be used for location purposes, when a device needs to be located by measuring the angles at which the same signal arrives at several reference points. In contrast to the last referred methods, this method does not need to solve for the distance to the source of the signals, instead, crossing all the imaginary lines created by the solved angles until there is a reliable estimate. The authors of [16] mentioned that this method needs a minimum of 2 anchors for 2D (one more for 3D), but keeping in mind that the more reliable measurements available, more accurate can the method be.

Even with the ability to calculate the angle of arrival of the signal, it is important to bear in mind that it is not enough to accurately locate anything. After the anchors solved the angle of the arrival of the signal, the only step that it is left, is to apply simple geometry equations and locate the source of the signal. This can be achieved by the difference of the multiple AoA of each anchor [24].

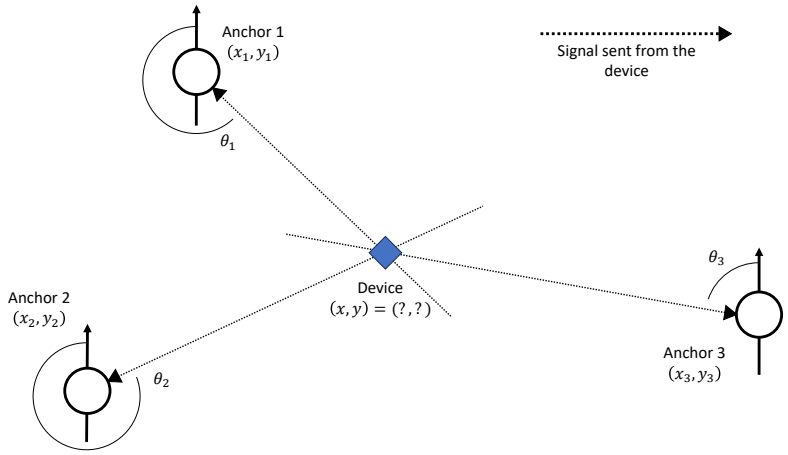


Figure 2.5: Example of 3 different anchors receiving the same signal, finding each AoA and intersecting the 3 imaginary lines to find the device coordinates.

2.5.5 Kalman Filters

In 1960 R. E. Kalman published his research introducing a new look at problems such as: separation of random signals from random noise and, detection of known form signals in the presence of random noise [27]. One of its most known application was in the Apollo project (in the 1960s) where it helped estimate the trajectories of the mission [28].

A Kalman filter, in simple terms, is an algorithm that takes noisy or inaccurate inputs and produces less noisy or more accurate estimates [29]. Given that it is highly unlikely that the data collected in the context of positioning systems, will always be noise-free and accurate, Kalman filters can play an extremely important role in obtaining better results.

The paper [30], produced an illustrative description of the concepts behind Kalman Filters, used for positioning applications. The author addressed topics such as kalman gains and the state evolution estimation and predictor-corrector estimator.

The Global Positioning System (GPS) is an example of the usage of Kalman filters for localization purposes, for both satellites and receivers [28]. A scenario where the majority of measurements are Non-Line-of-Sight (NLOS), was tested in [31] and proven to reduce error. [32] achieved a reduction of error for mobile tracking using a Squared Root Unscented Kalman Filter(SR-UKF) algorithm for localization in IoT.

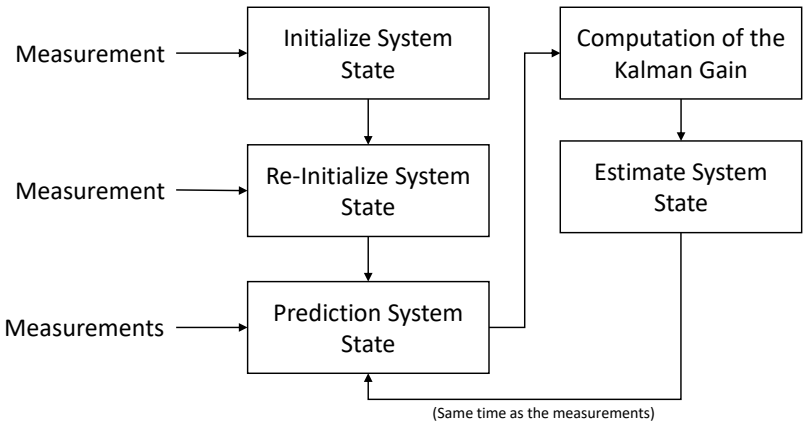


Figure 2.6: Kalman Filter block diagram, based on [29, 33].

2.5.6 Fingerprinting

The Oxford English Dictionary describes the word "fingerprint" as the mark made on a surface by the tip of a person's finger, a finger mark. In location scenarios, the "fingerprint" has a relatable meaning to its original. By analyzing the characteristics of the network signals throughout all the designated site, storing it to later use, it is possible to come to a more accurate result. This concept is possible to apply, taking in account that, like human fingerprints, the signal patterns can be used to draw a map of those readings, as long as the coverage of the site is significant, if not total [34, 35].

The first question this concept may arise is how the data used for this analysis, is collected. Different scenarios can generate different approaches and the data collection methods depend on the specific context, for example [34] mentions that it is possible to cover the area with sensors, but it is a time-consuming process. Because of that, "crowdsourcing" may be a promising approach, this method, with the help of users smartphones or other wearable devices, gathers the information throughout the floor plans, eliminating the need to install new infrastructures.

With all the information collected, a map of the readings can be drawn where it is visible the signal characteristics. Figure 2.7 is an example of the result generated after the reading process and consequent interpolation to fill in the map.

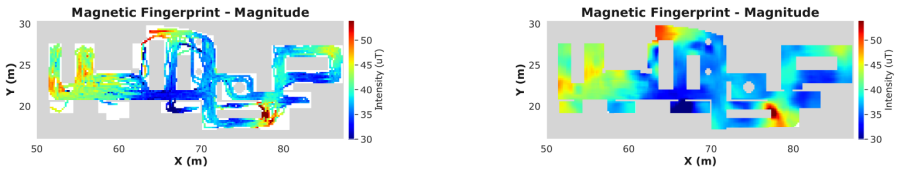


Figure 2.7: Example of a "heatmap" generated with fingerprinting, before and after an interpolation. Taken from [34].

Finally, with the map of the site built, it is now possible to use the known signal characteristics to relate with the signals inside the network and locate more accurately the object or person.

2.6 Privacy and Data Sharing of a RTLS

Since scenario under study involves calculating real people's location, even if it is in a work environment, we still have a privacy problem in hands. Therefore, it is important to be careful when dealing with these data, so that there are no related issues during the development and deployment of a project related to this topic.

2.6.1 Privacy Risks

Tracking the real-time location of a package that is traveling from a warehouse to a costumer, may not have any major privacy risk. If, for example, we consider a person inside a mall where a RTLS is being used to track customer or employees that are connected to the malls network, the problem becomes much bigger.

An unintended tracking may be in action, while the person may not have granted permission to do so, if this is the case, it is necessary to distinguish between who gives this permission and who does not. Using the context of a worker who is tracked while working, raises other concerns, that depending on the context (for security reasons inside a warehouse, for example) will also need the permission of the person.

A Data breach related to a real person's location can be a huge problem, because of the highly sensitive information that the data may carry. The power of this kind of information in the hands of someone with the wrong intentions, can lead to harmful consequences.

2.6.2 General Data Protection Regulation

All organizations and entities that handle personal data of individuals in the European Union must comply with the regulations described in the General Data Protection Regulation (GDPR) document [36]. Apart from setting regulations, this document imposes strict requirements regarding privacy and data collection of all sorts and aims to give control of personal data to the respective individuals.

2.6.3 Ethical principles

Much like every solution that involves personal data, a RTLS must incorporate ethical principles that should be a common practice for everyone. Some of this principles may be: transparency of what information is collected and where is it going to be used and the individual's ability to refuse to give that information.

2.7 Commercial Related Solutions

The following section introduces the key studies, research, and products that match with the focus of this thesis. These different cases have been designed to accommodate distinct scenarios, leading to multiple approaches for similarly the same end result, and that is why it is important to take their details into account.

2.7.1 Satellite Based Systems

Global Navigation Satellite System (GNSS) is a location system that relies on satellites that orbit the earth (referred to as constellation). The most known, GPS, is a system developed by the U.S. Department of Defense, with military purposes (late 60s, early 70s) and uses a constellation of 24+ satellites. Nowadays, it can be freely used for conventional geolocation, as long as the device has a compatible GPS module [37]. It is noteworthy that there are more examples similar to GPS such as Galileo, GLONASS and IRNSS, developed by Europe, Russia, and India respectively, which have some differences [38].

Throughout the years, this technology suffered upgrades. For example, modern receivers use simultaneously GPS and previously mentioned satellite systems, in order to combine the satellite usage and improve the performance [38]. Also, there is Assisted-Global Positioning System (A-GPS) and, as it was described in [39, 40], offers a better accuracy, availability, and coverage. A-GPS works by having multiple ground monitoring stations and servers that provide data and, assist GPS receivers to quickly process and compute the location. The fixed monitoring stations have a high chance of having the same satellites in their reach, has the receiver.

Every satellite is at an altitude of around 26559 km, with a 12-hour period and are strategically placed so that regardless of the receivers position, any device can always "see" at least 4 satellites [41].

2.7.1.1 Basic overview of the system

Previously, when we were taking a look at trilateration in a 2D referential, we came to a conclusion that we needed 3 reference points to locate an object in their reach. Moving to a 3D plane, we will now need 4.

By taking a look at Figure 2.8, we can see that, with 2 reference points (in this case, 2 satellites), in 3D, the possible solutions are now within a circle.

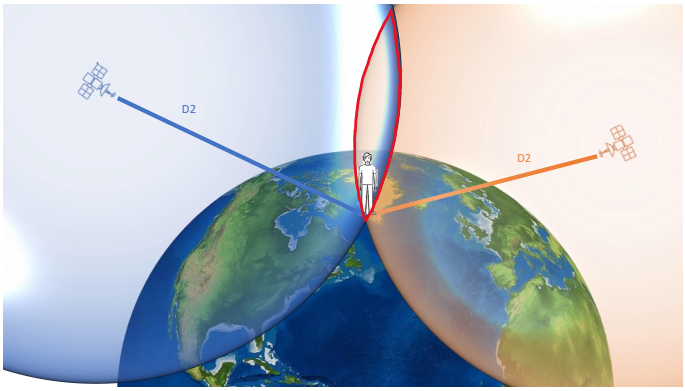


Figure 2.8: Visualization of trilateration in 3D using 2 satellites.

If we add another satellite to this scenario, there will remain only 2 possible points for the correct position. By finally adding the satellite number 4, will give us the solution of the true location of the device (marked with the red circle).

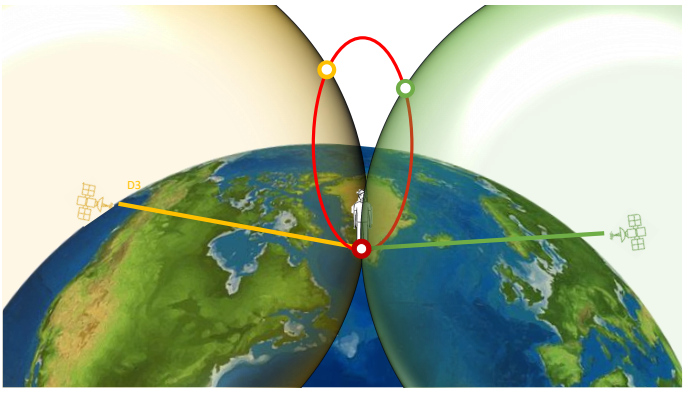


Figure 2.9: Visualization of trilateration in 3D adding the third satellite.

The final step of adding the fourth satellite not only solves the problem of finding the true location, but also helps to solve a problem that we are going to take a look in the next section. The “bias in the receivers clock”.

2.7.1.2 How is the distance measured?

Satellite based systems similar to GPS use a TDoA method. Every satellite has an atomic clock, which are capable of maintaining a continuously precise clock signal (stable to a few parts in 10^{-13} [37]), in contrast with the crystal clocks that the receivers’ module have [40]. "Bias in the receivers clock" is the difference in precision of the two clocks and needs to be taken into account in the process of the TDoA method.

Each satellite does a continuous broadcast on which passive receivers will perform the precise ranging measurements that were previously referred. The message sent from the satellites contains their location and clock information [41]. The satellite knows its own location because of its specific altitude and period, but the operation control system uploads corrections daily to improve precision.

Knowing that the received signal travels at the speed of light ($c = 300000000m/s$) and the time at which the satellites sent the signal (t, in seconds), the device is able to measure a "pseudorange". It is called pseudorange because of the difference in the precision of the clocks, that makes the possible solutions to be around a larger set of points instead of a single intersection.

The fourth satellite helps to correct this "pseudorange", where instead of having circles crossing and forming points, there will be areas of possibilities. Adding another reference to the equation, causes the number of possible points to converge to become smaller and smaller.

2.7.1.3 What makes GPS not suitable for indoor

After all of this explanation, the studies and improvements already developed to the satellite Based Systems, what makes these systems unsuitable for indoor location? The answer to this question may help understand future problems and what to take into consideration in the development stage of this project.

The signals sent from the satellites already have to cross several kilometers, for indoor location, they would also need to go through the buildings walls, decreasing the signal strength. The presence of metallic structures in the foundations and other parts of the buildings will also add up to this loss. The Faraday cage effect also needs to be taken into account [34].

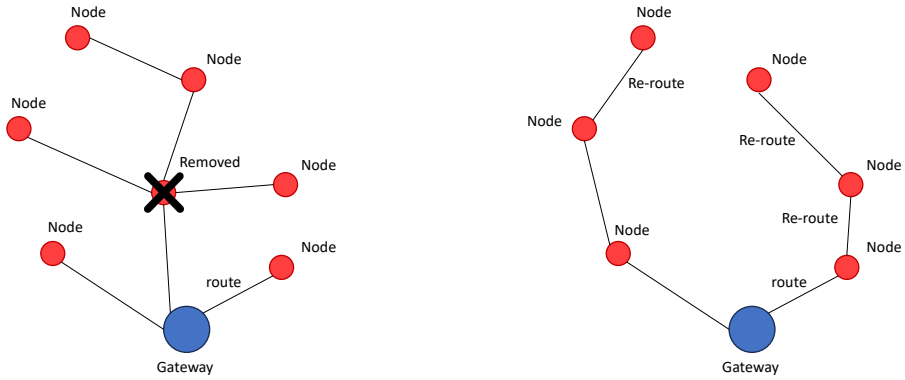
2.7.2 Wirepas Smart Tracking

2.7.2.1 Wirepas network overview

Wirepas is a company that aims to solve real world problems with the help IoT technologies and without the use of wires or infrastructures. The company offers a variety of solutions to make it possible, such as, mesh networks with built-in end-to-end services and smart tracking for multiple purposes in different industries [42].

What makes this company stand out is their mesh massive reliability and scalability, regardless of the environment it is deployed in to. This is made possible by decentralizing all network management operations, leaving the nodes to do that job.

In a Wirepas network there are 2 main components, the gateways and the nodes. When a node finds a gateway, it creates a connection and informs every node in its reach that it knows a "route" to the gateway. To achieve optimal connectivity, every node manages their next hop (to another node or gateway) to spread their intended message. When adding or removing a device to the network, the mesh automatically rearranges to achieve the optimal connection between gateways and nodes.



(a) Wirepas network before removing one node

(b) Wirepas network after removing one node

Figure 2.10: Wirepas adaptability visualization.

Also, considering the fact that every device can manage the power used during transmissions, it is possible to have long-lasting devices while battery operated.

2.7.2.2 Tracking overview

The first step to understand the tracking system behind Wirepas, is to get to know what is the main goal of every component of this system:

- Gateways: Responsible to send all the data to the Wirepas backend server;
- Anchors: Previously mentioned as nodes, these are also used as references for the positioning system;
- Asset Tags: These are Nodes that are also responsible for collecting the RSSI values of signals sent by multiple anchors, and sending all the data collected, along with timestamps, to the gateways.

Figure 2.11: Examples of a gateway, anchor, and asset tag respectively, from [43].

The system has some requirements for both the gateways and the anchors. Some of these requirements are: Central position for the gateway, line-of sight to as many anchors as possible and at least one gateway per floor. Anchors should be placed around the area (forming a square, for example) that the asset tag should be located inside.

Two cases are provided by [43] to illustrate how the devices can be arranged for two different scenarios in order to comply with the listed recommendations. The illustrated scenarios are a "per-room" and area localization.

2.7.3 Cisco Asset Tracker

Cisco, is a well known company in the overall engineering world because of the number of different solutions they offer not only to single costumers but also companies of big or small sizes. One of their solutions, Cisco's Asset Tracking offers a simple tool to manage, monitor and visualize the customers assets, regardless of whether they are simple sensors or vehicles inside a warehouse. The tool's document, [44], states that the location engine uses the RSSI measurements to calculate the tags' location.

This tool is a cloud-based solution that can be deployed with Bluetooth Low Energy (BLE), Wi-Fi or RFID tags and beacons as long as they are compatible with all of their environment. After acquiring, either by their starter kit or by purchasing independently, the devices can be registered in Cisco Spaces web-site or mobile app. After the registration process (helped with video tutorials), all the tags' information, either their location, route or measurement, can be used by multiple applications available on the platform.

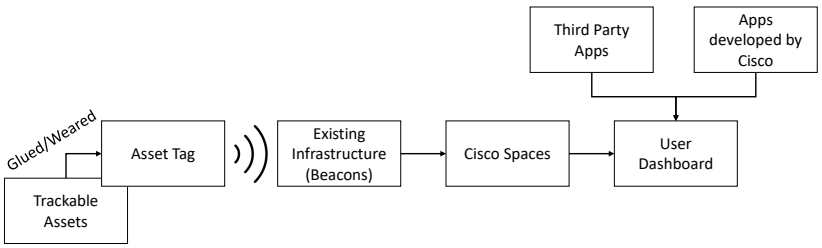


Figure 2.12: Cisco's Asset Tracker information flow, based on [44].

One key feature, of this tool, is the ability to use applications developed by Cisco or third party applications that can enhance the experience of the user, enabling the faster development of customized solutions that customers see fit to their goals. For example, it is possible to draw areas where if at any point, a tag leaves, the user will get a notification.

2.7.4 Inpixon RTLS Platform

2.7.4.1 Inpixon platform overview

Inpixon (merged with XTI Aircraft) has developed a RTLS that, much like the one referred in section 2.7.3, is "Technology-Agnostic" which mean that it is compatible with multiple hardware technologies. Making use of multiple protocols like BLE, Ultra-WideBand (UWB) and Chirp Spread Spectrum (CSS) as the transport layer, having the possibility to also use their platform with third party applications and integrate tags developed by other companies.

This solution, according to their web-site documentation, seems to have been developed for the industrial sector or larger companies, enabling the deployment of smart warehouses, factories, and other facilities without the need of a complete redesign of the existing infrastructures [45].

2.7.4.2 Inpixon tracking overview

The platform, has all the previously mentioned systems and methods, uses tags that continuously send signals that will be detected by known fixed position anchors. TDoA is the method applied by the location engine to calculate the tags position [46]. Inpixon claims the ability to reach an accuracy from 5 meters to 50 centimeters, with optimal conditions (depending on the protocol used).

With the objective of integrating position and identification data, some examples that solution features, aside from Real-Time Location Tracking, are alerts based on predefined conditions, definition of zones to trigger actions, integrating users rights to achieve a better facility management.

Figure 2.13: Inpixon RTLS Platform design and workflow, from [45].

2.8 Related Academic Projects

The final section of this chapter aims to sum up all the projects that have been discussed throughout this chapter. Table 2.2 provides a summary of the academic papers referenced in this chapter, indicating the source of the reference, the location technique under consideration, and the conclusion or information that can be drawn from it.

Table 2.2: Overview of the Academic Projects.

Reference	Location Technique	Conclusions and Information taken
[17]	TDoA	Distance between source and sensing node is influential, splay configuration placement of the sensors can achieve optimal results.
[18]	TDoA	Combination of GRoA and TDoA can improve accuracy, the lower the bandwidth of the signal.
[19]	TDoA	Factors like NLOS, multipath fading and synchronization can influence location algorithms. TDoA can be used to provide higher accuracy.
[20]	RSSI	Valuable experimental research on RSSI measurements. RSSI-based location using standard sensors can provide accuracies between 2 and 5 meters, depending on the site configuration.
[23]	RSSI	Combining the use of RSSI measurements with Kalman Filter can improve a significantly the accuracy of the algorithm.
[21]	TDoA, RSSI and AoA	Review of measurement techniques and investigation of connectivity and distance-based algorithms. Approaches to problems of distance-based techniques.
[24]	AoA	Proposal of a AoA location technique and orientation scheme, achieving good accuracy and precision with inaccurate angle measurements and low number of beacons.
[25]	AoA	The tests made with AoA proved to be well-suited for outdoor wireless sensor networks, by shifting the complexity of the sensing hardware to the anchors.

3

Location Engine

With the theoretical foundation laid, the next step will be to move on to the development and implementation phase of the location engine that has already been discussed. The following sections will introduce the topic through various points. It is essential to take these points into account in order to better understand this work as a whole. As this project has been implemented with the help of Crowdkeep, providing several networks of battery-operated devices spread across different spaces in the USA and Portugal, as well as the data they gather, it is important to emphasize these contextual introductions. From now on, this system will be addressed as Real-Time Location System (RTLS).

Thus, this chapter will begin with an introduction to the frameworks and services used during this phase, a detailed description of all the elements that integrate the RTLS pre-existing infrastructure. This is followed by a description of the development and implementation phase of this work, and concludes with the description of the location techniques used in.

3.1 Frameworks and Services

This section aims to introduce the frameworks used throughout the work's development to make the description of the work easier to understand. For each framework, a brief introduction will be provided, followed by the reasons for its use and its benefits. Additionally, if relevant, the different packages used in conjunction with each framework will be discussed.

3.1.1 Firestore Database

Firestore Database, part of Google Firebase, is a cloud-hosted NoSQL database designed for real-time data synchronization and scalability. Firestore grants the option to store, sync, and query data for multiple application (web and mobile for example), without the need to manage its underlying infrastructure. Inside the database, data is organized into collections and documents, making it easy to handle structured data

and queries. It supports real-time updates, enabling the applications to reflect changes instantly, providing a responsive user experience [47].

As there is already a pre-existing infrastructure (described in section 3.2), my use of the firestore was essentially to verify values. Given that much of the information coming from the local sites is written directly into the firestore, it was necessary to check values, formats and interactions as the work was developed. Given the use of 6 Sites, it was also necessary to check how the data was structured both when it was received and when it was passed on to another layer of the system.

Using Firestore for a Location Engine in a Real-Time Location System (RTLS) provides real-time data synchronization, ensuring accurate and up-to-date location data. Adding the ability to scale and support varying loads, making it easier to handle small or larger sites. These features collectively strengthen the reliability, responsiveness, and scalability of the system.

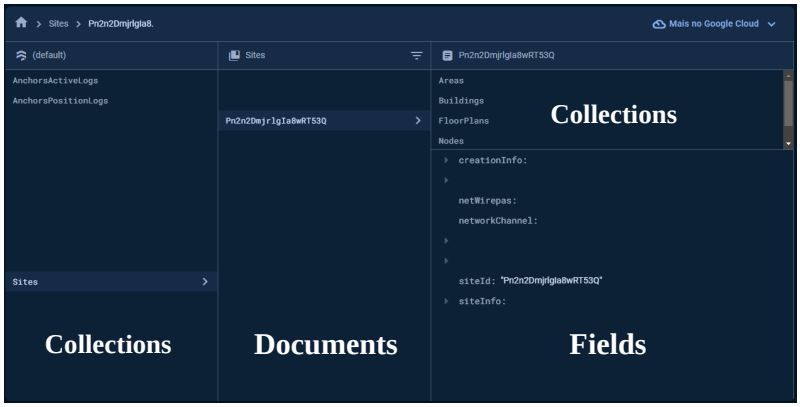


Figure 3.1: Example of how data is visualised and divided in the Firestore Database, with multiple collections, documents and fields.

3.1.2 Google Big Query

Google BigQuery is a data warehouse solution provided by Google Cloud, designed for large-scale data analysis. This solution allows SQL-like queries to be executed to massive datasets, taking advantage of the power of Google's infrastructure. It can also be integrated with other Google Cloud Services, allowing a number of different applications to generate several results. This service, like Firestore, also supports real-time data handling, but leaning more towards complex tasks and no need to manage its underlying infrastructure [48].

Similar to the use of the Firestore database, the SQL query feature of Google Big Query was used to read data from it. These queries generated files with a large amount of data (between 600,000 and 1.5 million lines) in order to carry out local tests with the developed

algorithms, before integrating it into the system. This service was also used to visualise and validate the results of the algorithm once it had been implemented in the pipeline and integrated into the system.

Google Big Query also includes data coming directly from the local sites but also some already processed through the pipeline implemented by the company, such as the data processed by Wirepas regarding to its location solution (mentioned in section 3.2.4). This Database differs from Firestore Database in that it is not designed to process data in real time, but has many other advantages. Some of these are: the ability to support the processing of large volumes of data, to store it to keep a history of events over a long period of time, for visualisation and other types of analysis.

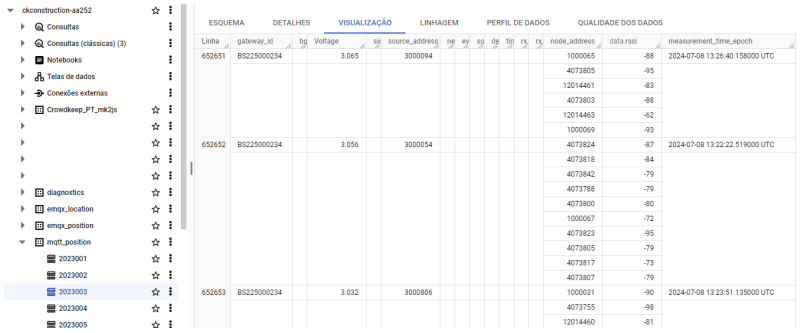


Figure 3.2: Example of how data is visualized in the Google Big Query, with multiple tables, variables and values.

3.1.3 Python

Python is a high-level, interpreted programming language that was designed with an emphasis on code readability and simplicity, making it easy for beginners to learn but also powerful for experienced developers to use for a wide range of applications. The language's focus on readability and ease of use is intended to reduce the cost of maintaining programs and allow programmers to express concepts in fewer lines of code [49].

Initially, the use of Python as the programming language to develop this work was due to my previous experience using it in various curricular units and personal projects over the past few years. The ease of implementing complex algorithms in a simple way and its ability to process and analyze data were some of the important aspects considered for this choice in the early stages of the development. Later, during the integration phase of the algorithm (that appied the chosen method) with the RTLS, this choice became even more relevant, since the same language is used for the data pipeline.

Python supports multiple programming approaches, including procedural, object-oriented, and functional programming, and its extensive standard library and community

contributed modules make it versatile for tasks such as web development, data analysis, artificial intelligence, and more [50].

3.1.4 Interactive Python Notebook

An Interactive Python Notebook, as the name suggests, is a notebook that combines several python snippets, which are written in "cells" and can be executed separately, keeping the memory of what has been processed between them. The simplicity offered by this type of file makes it easy to develop small sections of code, separating them into blocks and adding the possibility of using "markdown" cells to write text (generally used to describe the content/utility of the upcoming code cell).

As with the use of the python language, the reason for using this type of file was also the use and experience I had previously had with this type of notebook and the tools that help improve the user experience.

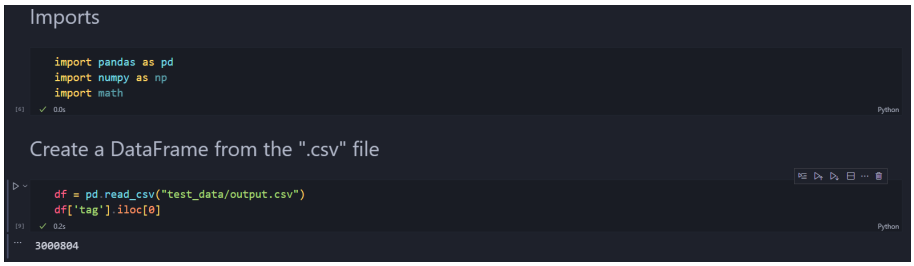


Figure 3.3: Example of the markdown and Python cells of an Interactive Python Notebook file, showing some imports and a simple printout of the corresponding tag Id of the first line of the file.

3.1.5 Anaconda

Anaconda is a Python and R distribution widely used for data science and artificial intelligence. The use of this distribution has focused on the "conda" package and environment manager [51].

The benefit that "conda" brought to this work was the ability to create Python environments in which specific packages were installed for the use of several libraries. Once these environments had been created, they could then be exported and imported, making it easier to use the code in different systems. By having curricular classes and personal projects that use Python environments during the thesis development, this feature enabled me to go back and forth with the different environments without having the necessity of loading to many packages at once, when working in different projects at the same time.

Some of the libraries used during the development were: pandas, numpy, matplotlib, math, folium, seaborn, and more.

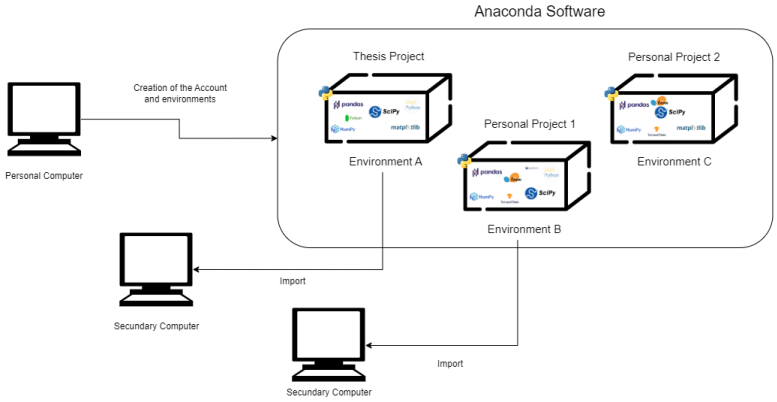


Figure 3.4: Anaconda environments setup.

3.1.6 Apache Beam

Apache Beam is an open-source unified programming model designed to define and execute data processing pipelines, both batch (processing of large volumes of data that have been collected over a period of time) and stream (continuous ingestion and processing of data in real-time) [52], in a platform-agnostic manner [53].

Developed originally by Google and later (early 2016) moved to the Apache, Beam provides a set of abstractions for expressing data workflows and offers flexibility through its portability framework. This allows developers to write their pipeline code once and execute it on multiple environments such as Google Cloud Dataflow, without modifying the pipeline logic. Beam supports several programming languages, including Java, Go, and Python, the language that it is going to be used for the integration of the pipeline [54].

A data pipeline is responsible for performing a series of tasks, receiving data in a particular form, transforming it with an end in mind (which can vary from pipeline to pipeline), and usually storing it in different places for different purposes. There is not much to say as the reason to why the use of Apache Beam since it is also something that was already implemented in the RTLS, and will be needed to understand in order to integrate the developed code in it.

3.2 Pre-Existing Infrastructure

The last aspect that needs to be understood before moving on to the actual development of the work, is the presentation of the entire existing system in which the location engine will be integrated. By understanding how the entire local and cloud structure has been set up, it will be easier to understand how the work has been done and why certain decisions have been made. Therefore, the pre-existing infrastructure consists of two parts, a local part which has a network of fixed and mobile devices that communicate using Wirepas

technology (section 2.7.2). And a cloud part, which uses technologies that mostly belong to the Google Cloud Platform (GCP).

3.2.1 Local Infrastructure

The local infrastructure provides the main element of the system, combining a network of fixed and mobile devices that rely on Wirepas technology (section 2.7.2) for continuous communication. This network has been built to provide connectivity between numerous devices and floors, ensuring efficient data transmission. Mobile devices (or nodes), carried by users (or attached to assets) as wearable bracelets or key-like cards, interact with fixed nodes (anchors) to provide dynamic and flexible coverage. These are integral to the system's adaptability, allowing for real-time location tracking as they constantly move within the site. The use of Wirepas technology ensures that both fixed and mobile devices can form a mesh network, enhancing connectivity and redundancy by allowing devices to communicate directly or indirectly through intermediate nodes.

This local setup is critical for achieving high accuracy in location tracking, as the dense network of devices provides multiple reference points for triangulation and an efficient data transmission. Additionally, the scalability of Wirepas technology allows for an easy expansion of the network, accommodating more devices as the need arises without significant changes to the existing infrastructure. Examples for these changes can be the expansion of the indoor facility or a physical change in the layout of the current area. Figure 3.5 is a small-scale representation of how the infrastructure operates, serving as an example for a better understanding of the description given so far.

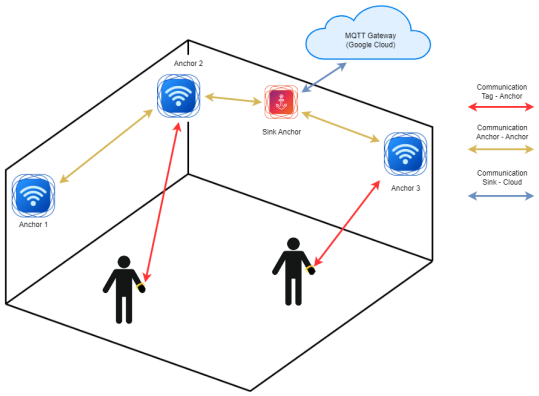


Figure 3.5: Demonstration on how data flows through the local sites.

3.2.2 Cloud Infrastructure

The second part, unlike the first, is, as the name suggests, fully cloud-based, mostly on the Google Cloud Platform. On this platform, the system consists of various distributed services and microservices such as Google Big Query (GBQ), Firebase's Firestore Database,

Data Flow and more. All the data from the local infrastructure reaches the cloud via communication through an MQTT server. This server publishes all the data in multiple pub/sub topics where the other cloud services already mentioned (Firestore, Big Query), by subscribing to these topics, have access to them whenever something is published in them.

From here, each service does what it's meant to, the most important of which, and the one my work will focus on, is Apache Beam (mentioned in section 3.1.6), the data processing pipeline that runs the algorithm that calculates the location of each tag. Whenever a tag reports information from the local infrastructure, whether it's located in Portugal or the USA, that data will pass through the algorithm developed.

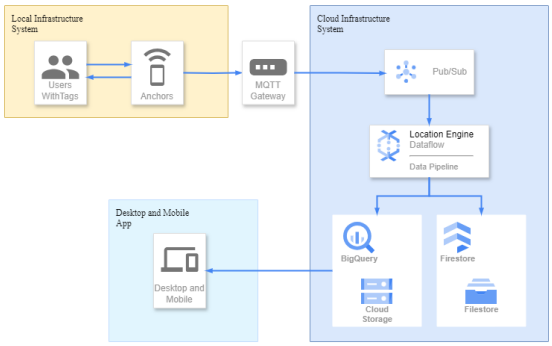


Figure 3.6: Demonstration on how data flows through the local and cloud infrastructure.

3.2.3 Sites

This section explores the specific local sites where the location engine was tested and implemented. By examining these sites through their description and characteristics, we can better analyze the performance of the engine in different environments.

The sites can be divided into two sub sections, the ones in the United States of America and the ones in Portugal. These sites provide a variety of environments that are critical for evaluating the robustness and adaptability of the location engine in different real-world scenarios. The sites used were those where the local infrastructure was already installed, and the data collection was ready to be utilized, unlike other sites that were not yet prepared

The following subsections provide detailed descriptions of each site, including its unique characteristics, area, perimeter, and number of floors. The total number of anchors and the average number of personnel inside each site will also be presented. In both cases, for reasons of privacy and of little relevance to the final result, no information will be used relating to the real locations, which could link them to their identity.

3.2.3.1 USA's Sites Description

As for the sites located in the United States of America, there are 4 different sites with a higher density of Anchors and people moving around on them (by which is meant Tags.), and of the spaces used to test the work, these are the ones with the largest dimensions. In this case, all 4 sites are located in a school environment, where the users can be teaching and non-teaching staff, as well as students.

Starting with the site, which from now on will be referred to as "Site A", it has an area of around $9,000m^2$ (doesnt have a rectangle shape, with 3 different floors), a perimeter of around $460m$, 68 anchors in total and 64 different tags, recorded over the course of 5 months. The second one, which from now on will be referred to as "Site B", it has an area of around $30,000m^2$ (around $145m$ x $65m$, over 4 different floors), a perimeter of $410m$, 201 anchors in total and 299 different tags, recorded over the course of 5 months.

The third site, which from now on will be referred to as "Site G", it has an area of around $9,500m^2$ (doesnt have a rectangle shape, with 3 different floors), a perimeter of $350m$, 114 anchors in total and 49 different tags, recorded over the course of 5 months. The forth and last site in the USA, which from now on will be referred to as "Site S", it has an area of around $17,000m^2$ ($115m$ x $50m$, over 3 different floors), a perimeter of $330m$, 169 anchors in total and 279 different tags, recorded over the course of 5 months.

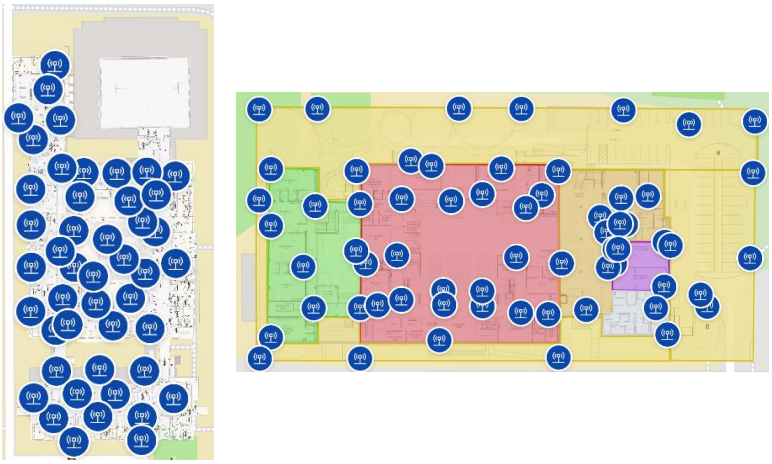


Figure 3.7: Visualisation of 2 sites (one floor only) with the corresponding anchors for that floor plan.

3.2.3.2 Portugal's Sites Description

As for the sites located in Portugal, there are 2 different sites with considerably less density of Anchors and people (by which is meant Tags.), and therefore, with smaller dimensions. The size of these sites should not be overlooked when compared to the size of

sites in the USA, as the larger sites can be considered to be several smaller rooms (similar to those tested in Portugal) next to each other.

In this case, both locations are in an office environment where the tags are used by office workers who move around approximately every half hour, as well as other stationary objects (such as spare tags that would remain in the same place for an entire day). Starting with the site, which from now on will be referred to as "Site M", it has an area of around $30m^2$ ($7m$ x $5m$, over 1 floor), a perimeter of $24m$, 4 anchors in total and 10 different tags over the course of 1 month. Moving on to the second and last site, which from now on will be referred to as "Site T", it has an area of around $165m^2$ ($14m$ x $11m$, over 2 different floors), a perimeter of $50m$, 6 anchors in total and 9 different tags over the course of 3 months.

Figure 3.8: Visualisation of one of the Portuguese sites.

All of the information listed above is summarized in Table 3.1.
* - These values are not precise and are rounded down.

Table 3.1: Summary of site information.

Site	Country	Area* (m^2)	Perimeter* (m)	Floors	Total Anchors	Total Tags
Site A	USA	9000	460	3	68	64
Site B	USA	30000	410	4	201	299
Site G	USA	9500	340	3	114	49
Site S	USA	17000	330	3	169	279
Site M	PT	30	25	1	4	10
Site T	PT	165	50	2	6	9

3.2.4 Wirepas Positioning Engine

The Wirepas positioning engine is a location engine developed by Wirepas, which can be integrated into its “Wirepas Massive” networks (such as those used for the project), which also uses Received Signal Strength Indicator (RSSI) data to obtain the result of the location of its tags. For this work, the aim will be to replace the use of this engine, using

it as a reference for the tests that will be carried out, and also to help draw conclusions about the the remote test, where it is not possible to know the exact location of each tag (explained in the 4.3 section).

By analyzing the documentation, provided by Wirepas, we can conclude that there is no exact value for the accuracy of this location engine developed by them, as this value depends on a number parameters such as the physical environment in which it is inserted "(i.e. with a lot of concrete or metal)" and the arrangement of its devices. For this, Wirepas suggests dimensioning the site settings to provide an approximate accuracy, based on the objectives of each client. This dimensioning depends on: "Number of Tags, Tags location update rate, Tags (density), Anchor (density), Anchors (quantity), Gateways", and can be done using a tool they provide: [55].

Focusing now on the sites used to carry out the tests, their dimensioning was done in order to obtain an accuracy of around 3 to 4 meters, since this is a dimensioning and it is not possible to obtain an exact accuracy value. So we can conclude that, for our case, the accuracy of the wirepas location engine is around 3 to 4 meters. This result is to be expected considering the multiple examples discussed in the chapter 2, such as [10], where the accuracy that can be achieved using RSSI is very difficult to go below accuracy values of between 2 and 5 meters.

3.3 Location Engine Development

In order to present the development of the Location Engine, it is important to note that this development is divided into two main parts, which divide the work so that it can be tested first and then integrated into the RTLS.

The first part of the work concerns the initial development of the algorithm, the "testing phase". In this phase, two different methods were developed for calculating the location. To do this, different algorithms were developed for each one, as well as different ways of testing the results obtained. The second part, as already mentioned, is based on integrating the tested and finished algorithm into the RTLS so that it is operational in all the locations from which the system receives information of the local structure.

3.3.1 Data Overview

This section provides an overview of the data used in this work. The data is essential for testing and refining the location algorithm. The data was mainly obtained from Google Big Query, where information from the local infrastructure and the Wirepas location solution was stored.

For a clear understanding, this discussion was divided into two parts. First, the data sets are described, detailing the types of files and their formats. Then explained how the data was collected, highlighting the methods and time periods involved.

3.3.1.1 Data Description

Before starting describing the data used throughout the project, it's important to note that in this context it only makes sense to talk about the use of "files" in the case of the algorithms used for testing. Keep in mind that once the tests are done and the location engine is integrated with the RTLS, it doesn't need large documents (or many variables), since it only consumes about 5 to 10 lines of data at a time.

With this initial input, the main source of data, as already mentioned, was Google Big Query, where the data relating to the information collected by the local infrastructure and the wirepas location solution is stored. Several queries were made (some examples in Figure 3.9, containing some blocked information for privacy reasons) to the different tables relative to each site, where the size of the files was roughly between 600,000 and 1.5 million lines (data collected for each month and/or site).

Once these queries had been made, the resulting tables were converted into comma-separated value (".csv") files using the platform's own tool. These files were then read locally and used in the testing algorithm. While this algorithm is running, a table is created in which lines are added as it calculates the location results, these lines compare the output of what was calculated to the wirepas location solution, or the real position (depending on the test). This table is then converted back into csv so that the results can be stored and analized later.

During the development of the algorithm integrated into the RTLS, several ".json" files were created, taking into account that once integrated, the algorithm will receive information in this format. Several different files have been created in order to simulate possible scenarios of information input and to correct possible problems in the algorithm. The lines of data received by the algorithm are used to obtain the result for the location of only one tag. Each tag performs a reading for each anchor that is in its "field of view" at each reading time creating a group of readings with all the anchors seen. Once the calculation is complete, a message of the same type is created, displayed and saved (in the same format), so that it can be analyzed locally and the data passed on to the next "layer".

The query in Figure 3.9 a) creates a file with the data of all the readings made by each tag over the course of a month and contains the following variables: *source address*: id of the tag that made the reading; *"anchor"*: id of the anchor to which the reading was made; *"rssi"*: value in dBm of the reading made between tag and anchor; *"time"*: instant

```sql
SELECT
  source_address,
  node.node_address AS anchor,
  node.rssi,
  measurement_time_epoch AS time
FROM
  `                    .mqtt_position.2023003`,
  UNNEST(data) AS node
WHERE
  measurement_time_epoch >= TIMESTAMP("2024-04-01")
  AND measurement_time_epoch < TIMESTAMP("2024-05-01")
ORDER BY
  measurement_time_epoch ASC
LIMIT 50000000;
```

(a) Query to retrieve the local infrastructure data collected through April.

```sql
SELECT
  node_address,
  latitude,
  longitude,
  measurement_time_epoch as time
FROM `                    .wnt_location.2023005`
WHERE
  measurement_time_epoch >= TIMESTAMP("2024-04-01")
  AND measurement_time_epoch < TIMESTAMP("2024-05-01")
ORDER BY
  measurement_time_epoch ASC
LIMIT 500000
```

(b) Query to retrieve the Wirepas solution data calculated through April.

```sql
SELECT
  node_address as source_address,
  anchor.anchor_address as anchor,
  anchor.rssi as rssi,
  measurement_time
FROM `                path_to_the.correct.table                `,
UNNEST(anchors) as anchor
WHERE
    measurement_time >= TIMESTAMP("2024-07-09 14:18:00")
    AND node_address = 300107
ORDER BY
    measurement_time DESC
LIMIT 10000
```

(c) Query to retrieve all entries of the tag 300107 at a specific time.

Figure 3.9: Examples of SQL Queries used.

of time when the reading was made. Keep in mind that a group of readings is a group of measurements that was made at exactly the same time, from the same tag to different anchors.

The query in Figure 3.9 b) generates a file with the data from all the results of the calculations made by wirepas over the course of a month, based on the same information from the local infrastructure. The variables used in the query are: *"node address"*: id of the tag that made the reading; *"latitude"*, *"longitude"*: values of the coordinates corresponding to the calculation made; *"time"*: instant of time when the reading was made. These timestamp variables will be used to compare the groups of readings that the tag has made, with the same wirepas solution. Table 3.2 summarises and presents a description of all the variables used and extracted by the queries.

An important detail to keep in mind is the importance of the variable *"time"*, because in the test algorithm this variable is used to compare the local reading with the result calculated by wirepas. Since this value is unique for each reading made by a tag (on each network), this comparison is made in order to check if the timestamp of the readings matches.

Table 3.2: Summary of the data variables used.

Name	Type	Description
source address	Integer	Identification of the Tag used in the communication.
node address	Integer	Identification of the Tag used in the communication (different query).
anchor	Integer	Identification of the Anchor used in the communication.
rssi	Float	Value (in dBm) of the received signal strength.
time	Timestamp	Instant of time when the communication happened.
latitude	Float	Value of the coordinate corresponding to the horizontal axis of the earth.
longitude	Float	Value of the coordinate corresponding to the vertical axis of the earth.

3.3.1.2 Data Collection

Once the data has been described, it is important to understand how the data was collected. With regard to its collection, as explained above, the users of each site use weareable tags which, when they enter the respective site, start communicating with the network that has been installed there.

Initially, to test the results generated by the algorithm, tags were placed in specific (and recognized) positions, so that the data collected could be used to calculate the location and then compared with the final result. Similarly, collections were made to optimize the location calculation, shown in sub-section 3.3.6. This data arrives at the Google Big Query platform in exactly the same way as the data from the USA, but only one tag at a time was taken into account so that the calculated and the real position could be correctly compared.

In the case of the remote sites (in the USA), through the queries already presented, data was collected from March to July (5 months) almost continuously, hence the considerable volume of data. The data collected in person, in Portugal, was set at irregular intervals, depending on how far the work had progressed.

3.3.2 Algorithm Design

The architecture of the system, in general, is defined in 3 parts: data pre-processing, calculations, and data formatting as presented in Figure 3.10 (in addition to the input and output). As already mentioned, there will be two different but similar algorithms that are responsable to apply the location methods. The local testing algorithm and the system's final algorithm, although they are different, they were built using the same principles and basically with the same result in mind (for what their purposes were).

The location engine works in this way so that it is able to receive the data from the physical part, in the specific way in which it is structured. This way the system carries out all the necessary processes to calculate the position and other parameters (if necessary), checking for possible flaws in the data received. The next step is to re-structure the data so that it can be sent to the next step.

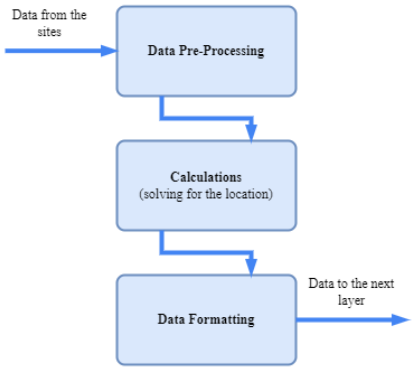


Figure 3.10: General overview of the Architecture of the Location Engine.

3.3.3 Local Testing Algorithm

The main reason for this algorithm was to be used only for local testing and corrections, before the final algorithm was developed and integrated into the system. It was taken into account that speed would not be a priority, but rather that the calculations would be done correctly and conclusions drawn from them.

This led to the need not only to save the calculations in several files, but also to display small values on the terminal that were of interest, and to produce a short report. That said, the algorithm, as mentioned above, is divided into parts where the data is collected, processed and sent, either to a local file or to the terminal itself.

The following Figure 3.11 is an UML sequence diagram for a better understanding of the explanation given regarding the general operation of the testing algorithm. "Auxiliary Functions" is a Python file that was created to contain all the functions that are responsible for the locations calculation methods and other generic functions that were needed to use continuously. In this figure, we can see that when you run the algorithm in the notebook cell, it first performs all the necessary imports, secondly the method's calculations, and finally displays some information on the terminal, and saves the results locally.

Figure 3.12 is a flow chart that represents the functioning of the algorithm that was used to test all the methods presented in this work. This algorithm begins by processing the data it receives and grouping the readings in such a way that it is possible to apply the

desired methods, after applying them, it saves the data and continues processing until it has finished with all the lines, exporting the data to a ".csv" file.

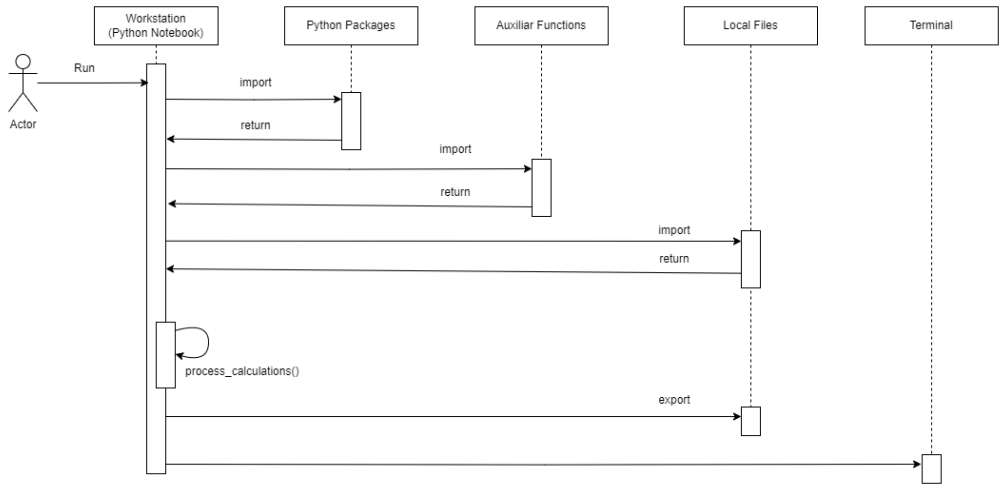


Figure 3.11: Local Testing Algorithm UML Sequence Diagram.

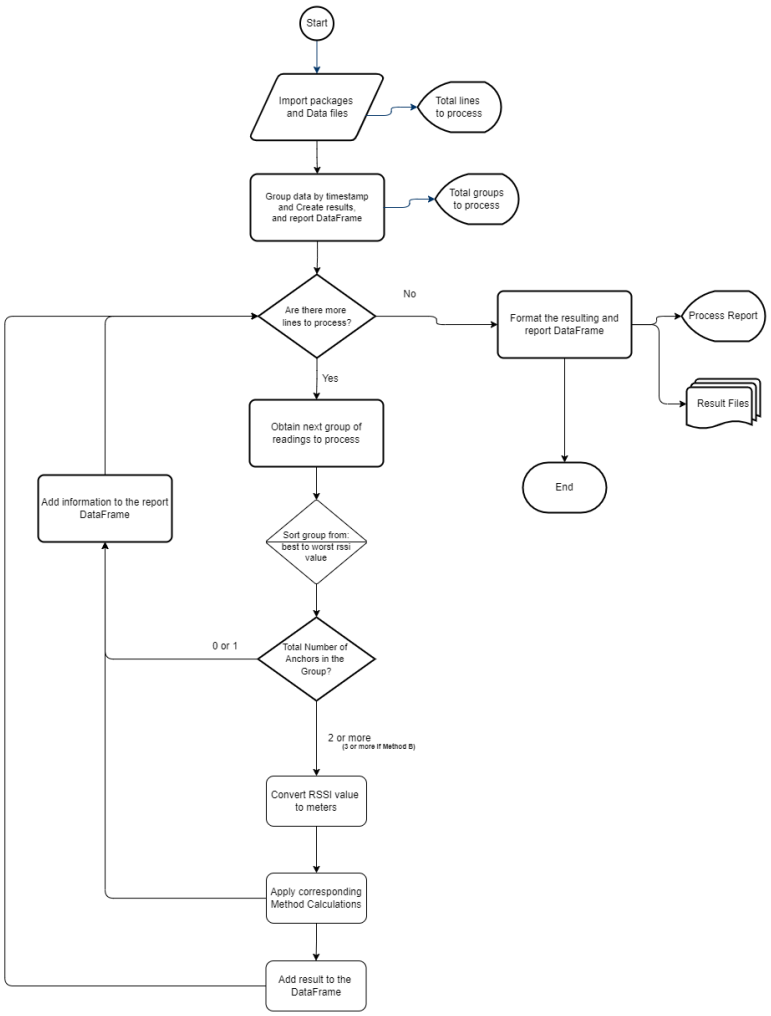


Figure 3.12: Flowchart of the Testing Algorithm functionality.

3.3.4 System's Final Algorithm

Regarding the algorithm used to integrate with the RTLS, after what has already been said about the similarities with the one described above, the idea during its development was to imagine it as a black box that has a data input and output. This I/O must comply with the format that is expected in both input and output (JSON) of the RTLS.

This algorithm does about the same as the previous one, the main difference being that it processes only one line of data, which makes it much simpler. "One line of data" means that the input is a reading from a tag that captured multiple anchor messages at the same time (figure 3.13).

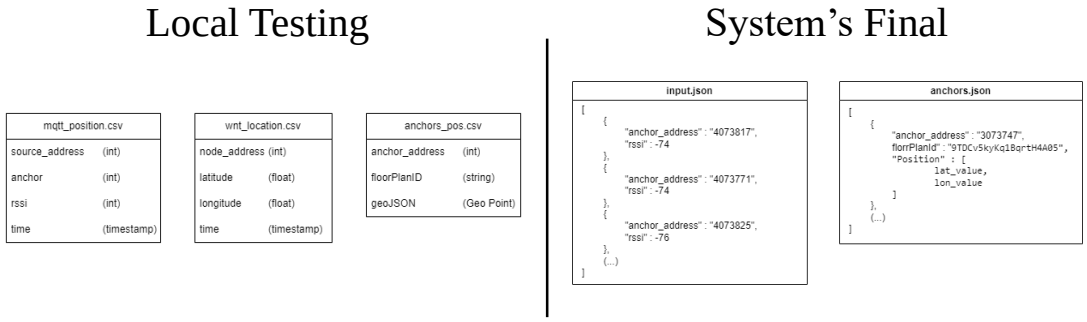


Figure 3.13: Difference between the test and final algorithm data input format.

Within the previously mentioned "black box", the location technique is applied to the input, where the result is then used to format the output and add the appropriate fields to complete the message. When it comes to the location technique used (within the 2 developed), there are some details to consider in order to understand the behavior of the algorithm. Considering that the triangulation is done with 3 anchors, there is a problem if the tag only "sees" 1, 2 or no anchors, which can happen at any time, so it is necessary to be careful with this detail.

If the tag cannot read any signal coming from any of the anchors, it is impossible to make any connection between its location and the position of the anchors. Relating the position to be calculated to the positions previously calculated is also not an option because there is too little information to make any conclusion. If the tag reads the signal from only one anchor, the algorithm uses the RSSI value to convert this value into meters. After this, the fixed position of the anchor that sent the message is used so that the location is generated at the calculated distance to that anchors position.

If the tag reads the signal from only 2 anchors, the algorithm will use method A to calculate the location result, as described in 3.4.1. These two abilities to obtain the location, even without having a reading group with 3 anchors, reduces the number of times a location is not “calculated”, thus improving the efficiency of the algorithm.

For readings that manage to collect the signal from 3 or more anchors, Method A is then used to make the normal calculation of the tag’s location, relative to the position of the anchors. The reason for using the methods described is explained in the next chapter (Tests and Results).

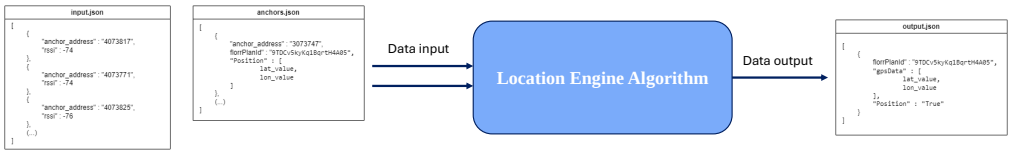


Figure 3.14: Diagram of the input and output data format for the final algorithm.

3.3.5 Integration with the Existing System

As mentioned above, the integration that was done to make the location engine algorithm work within the RTLS's pipeline consisted of using the Data Flow framework (also known as Apache Beam) in Python.

It was therefore necessary to learn how this framework worked and what functions it already performed within the system. To do this, access was provided to the various pipelines used by the RTLS to gain a basic understanding of how they worked, taking into account the scope of what was already integrated there. As a result, the process of writing data to the RTLS databases was put on hold until there was some confidence in what was being written. For this reason, the algorithm was first developed to take readings from the firestore, do the processing, and then print the results locally.

Once this was done, the part responsible for writing in Google Big Query was then added and tested until it worked correctly. To make sure that all the processing in the pipeline was working correctly, it was necessary to simulate movements within a local site in Portugal, check that the data arrived correctly in Google Big Query, and consequently that it was displayed on the RTLS web and mobile platforms. This last part regarding the correct presence on the platforms is extremely important because this is where the results are presented to the company's customers, so if there are errors and they don't appear, something has not been processed correctly.

3.3.6 Performance Optimization

This sub-section will only describe the action taken to optimize the results obtained, and the results of these changes will be presented and discussed in the next chapter 4 (Tests and Results).

3.3.6.1 RSSI to meters Conversion

The first step in optimizing the results was to convert the RSSI value (in dBm) to meters, obtained through the communication between tag and anchors.

As an initial state, the expression mentioned in [20] was taken into account. This expression was reorganized so that it could be solved in order to the distance (in meters), using 2 constants that could be adjusted depending on the environment used. The values

used initially were also those presented in the same article: $K = 25, P_{r1} = -55dBm$, taking into account that K is the loss parameter and P_{r1} the power felt at one meter.

Attempting to optimize, several readings were taken at specific distances (1 to 10 meters), so that it would be possible to deduce a logarithmic curve with similar characteristics. After these measurements and the deduction of these variables, the following values were obtained: $K = 16, P_{r1} = -56dBm$.

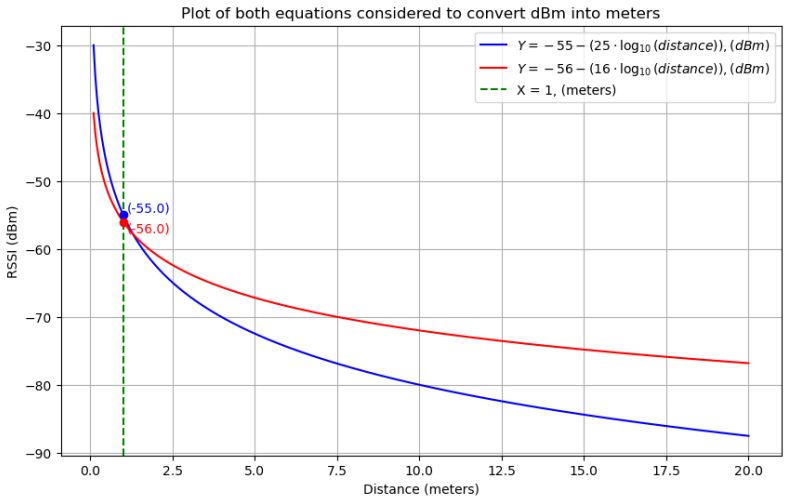


Figure 3.15: Plot of the 2 considered equations where $X =]0, 20.0[$ meters.

3.4 Location Technique

In this section it is discussed the two location techniques developed, based on the use of the Received Signal Strength Indicator (RSSI). To do this, these topics will be described briefly so that it is clear how they both work, including their differences. Some diagrams will be presented to help understand the description given.

3.4.1 Method A

Unable to implement the theory presented in Chapter 2, another approach was attempted. This method can be called the weighted intermediate point method. The initial point to be made regarding the functionality of this method is that it is capable of calculating the position with just two anchors. This feature, which takes into account the algorithm's accuracy requirements, makes this possible without compromising the algorithm's desired accuracy.

The second point that is important to look at is the ability to convert the RSSI value (dBm) into distance (meters), making it possible to convert the signal strength value into

something that can be used in a coordinate system to calculate the desired results. For this conversion, the equation 2.4 was used (presented by [20]), and also described in section 2.5.3.

To carry out this method, let's assume that the algorithm is going to use 3 anchors to calculate the location. The Figures 3.16, 3.17, 3.18 help to better understand the theory behind this explanation. Once it has the position of each anchor, it takes into account the RSSI value that each has obtained up to the tag and weights the distance from the best to the 2nd best and from the best to the 3rd best (A to B, and A to C, when ordered by highest to lowest RSSI value). It calculates a weighing based on the strength of the signal and assigns weights to the position calculations so that the point is placed in the middle of the two locations. This then makes it possible to calculate with 2, 3, 4 or more (section, no advantage was found in using more than 3 or 4).

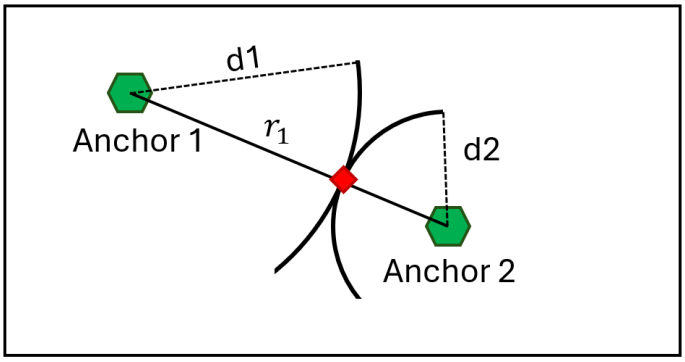


Figure 3.16: Visual application of the Method A using 2 anchors.

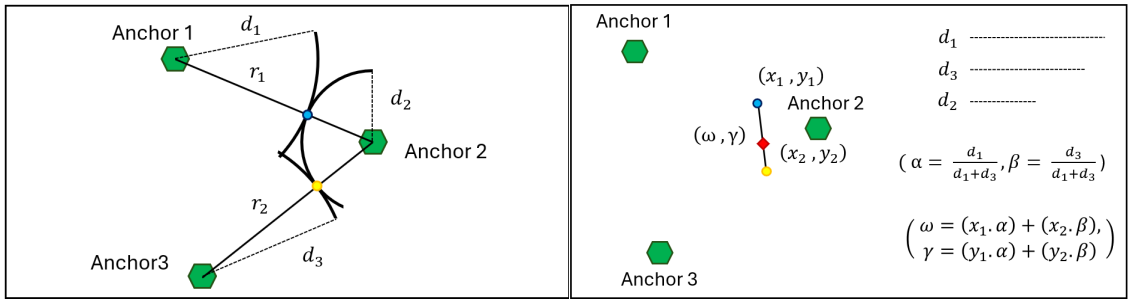


Figure 3.17: Visual application of the Method A using 3 anchors.

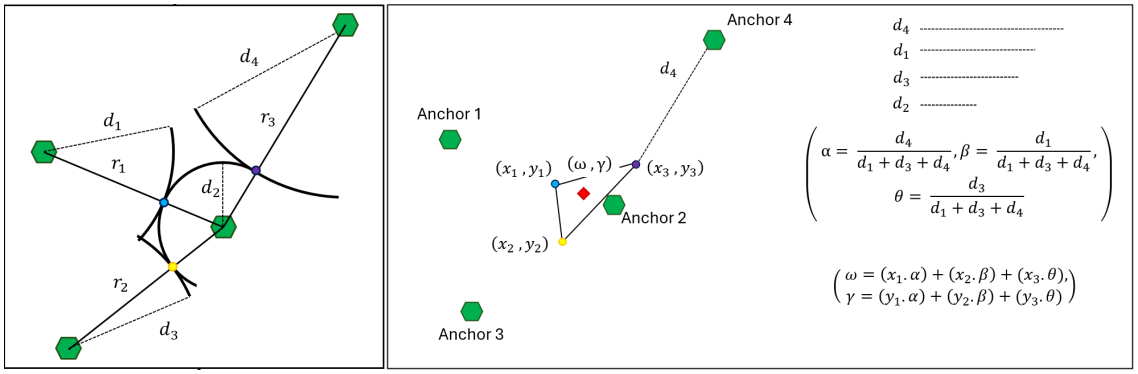


Figure 3.18: Visual application of the Method A using 4 anchors.

3.4.2 Method B

The second method that was developed works very similarly to what was described in section 2.5.1 (2D trilateration theory), also using the concept of transforming the RSSI value from dBm to meters that was used in the previous method. With this, given that there are 2 unknown variables (latitude and longitude), 3 reference points must be used to calculate the location output. This feature makes it impossible to get a result if there are readings with fewer than 3 anchors in the group. Something that, as we have already seen, can be important to take into account. As expected, there are errors associated with the measurements made, which cause the RSSI value to be mathematically unsolvable, as shown in Figure 3.19.

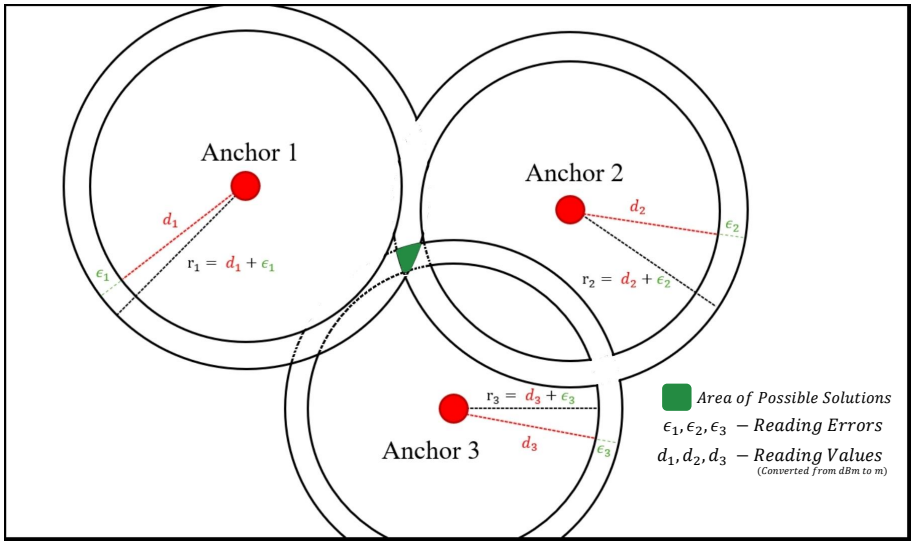


Figure 3.19: Visual application of the Method B.

As it can be seen from the Figure 3.19, there are several points where the location could be, due to the various errors and the way the circles shown, cross each other. If these circles do not intersect, it is very difficult to obtain a result for the location, so it was

necessary to correct these errors associated with the RSSI values and optimize the final result obtained. This was taken into account in the algorithm by using a Python library called "scipy.optimaze", which ensures that, given the initial values, even if they contain some errors, they are approximated and converge to an "optimized" value. By applying this, as we can see from figure 3.19, it is possible to obtain a result for the location of the tag, which is within the green area of the Figure.

4

Tests and Results

This chapter, as the name suggests, is destined to present all the tests that have been carried out throughout the course of this work, as well as their results. For this purpose, the chapter is divided into several parts, in which tests and the results that can be drawn from their analysis are presented, described, and discussed. In addition to the tests, some graphs will be presented with the aim of adding information and knowledge about the different environments in which the tests take place, and drawing conclusions not only from the graphs that are presented but also from the tests that have been carried out.

It's important to remember two aspects, the description given in the previous chapter of the different sites on which the tests and all the information presented here are linked. The second aspect to highlight is the fact that there are not only 2 methods, but also that method A is divided into 3 different parts. By using 2, 3 and 4 anchors for the solution, it is always necessary to present the results for all 4 occasions if the values vary significantly.

Section 4.1 presents the initial tests conducted at two Portuguese sites, where multiple tags were placed and readings compared with calculated locations. Section 4.2 describes distance-based Received Signal Strength Indicator (RSSI) measurements at various distances to understand the correlation between RSSI and distance. Section 4.3 explains the remote tests and accuracy evaluation using previously stored data and a summary of the results in order to draw conclusions. Section 4.4 details integration tests verifying the algorithm's operation within the Real-Time Location System (RTLS), while Section 4.5 discusses site environment data relevant to understand some results.

4.1 Initial Tests

The tests that will be presented in this section were carried out in presence, on the two Portuguese sites already described, the "M" and "T" sites. To do this, multiple tags were placed in various locations whose positions were known and multiple readings were stored for each tag. The location was calculated using the various methods mentioned in section 3.4 and finally compared with their actual position.

4.1.1 Site M Tests

For the first group of tests, which was carried out on Site M, 2 tags were used, where one was placed exactly underneath one of the anchors and the other exactly 1 meter away from another anchor, as shown in the following figure:

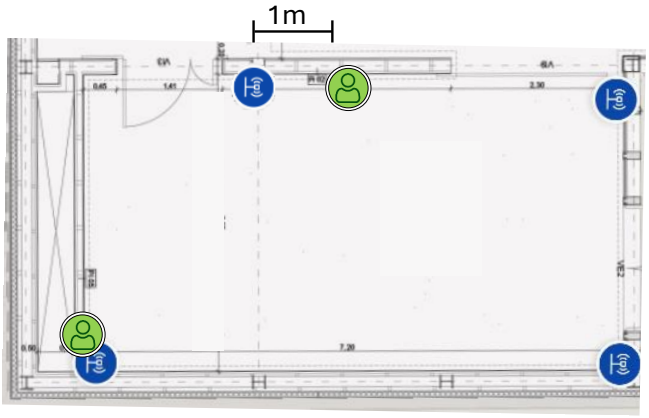


Figure 4.1: Visualization of the setup used for the local accuracy tests. In blue, the position of the anchors, and in green, the position of the tags, in Site M.

A total of 822 readings were made over 2 hours and 10 minutes, for the two tags, where around 740 were for the tag that was located 1 meter away from the anchor and 80 for the one that was located exactly below the other anchor. These readings produced 185 location results in total, 165 for the first case and 20 for the second.

4.1.1.1 Tag at 1 meter

Looking at the following figure, we can see how the results varied over time, and by averaging the distance between the tag's actual position and the one calculated, we can see that on average there was a distance between the two of approximately 2.32 meters. These results were obtained by applying method A, using 3 anchors.

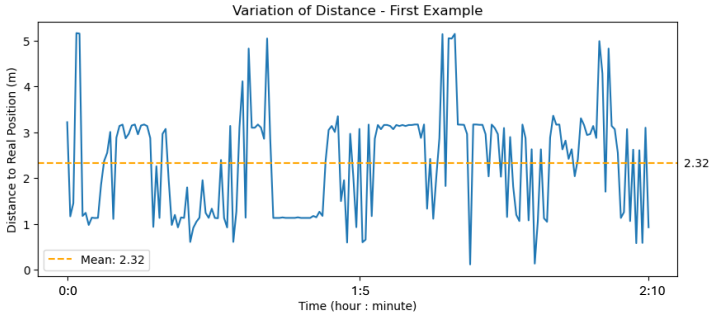


Figure 4.2: Variation of the distance between the calculated location and the real position of the tag, in 2 hours of collection, using Method A.

The following figure shows the difference in this calculations, when using method A (with 2, 3 and 4 anchors) and B (that uses 3 anchors).

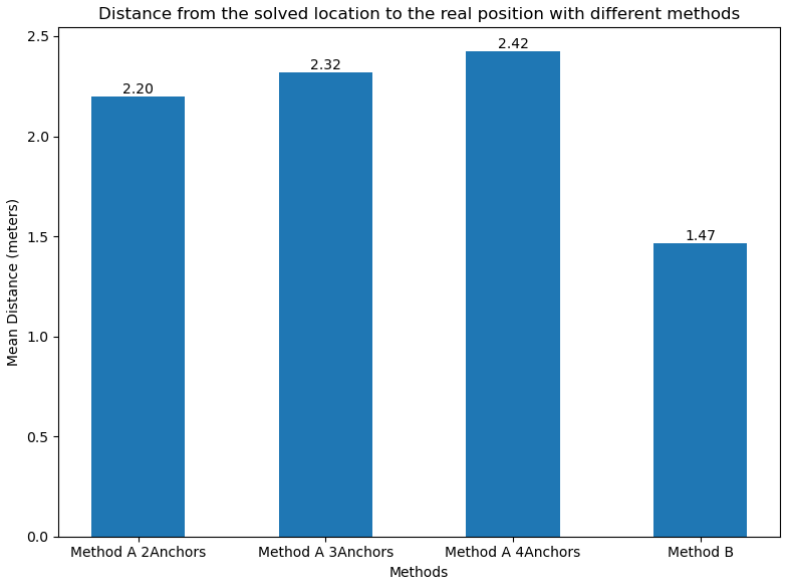


Figure 4.3: Variation of the distance between the calculated location and the real position of the first tag, using different methods.

4.1.1.2 Tag bellow the Anchor

For the reads made where the tag was directly below the anchor, we have a slightly different scenario. The tag was reporting at a substantially longer interval, which meant that the number of reads dropped considerably compared to the other tag. This site was subsequently deactivated, which made it impossible to perform new readings under the same conditions for this same site.

So, as we can see from the graph in the Figure 4.4, using method A (using 3 anchors) the distance between the calculated location and the actual position is approximately 0.5 meters (in green). For the remaining cases of method A, with 2 and 4 anchors, the result is approximately the same. When applying Method B to the same dataset, the average was approximately 5 meters.

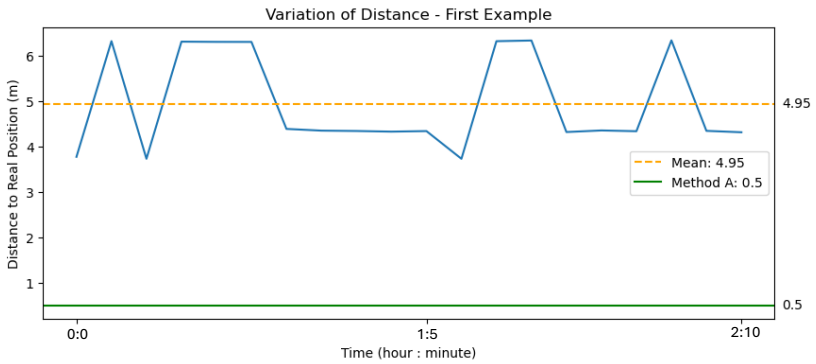


Figure 4.4: Variation of the distance between the calculated location and the real position of the second tag, in 2 hours of collection.

With this site configuration, we can conclude that, for cases where the tag is too close to an anchor, method B presents a considerable error compared to method A, but this result is not exactly "bad" taking into account the requirements presented initially. Since calculating the location using RSSI and obtaining accuracies of less than 3 meters is a difficult task. For the results of Method A, obtaining values with a difference of around 0.5 meters,and an average of around 2.50 meters is a good indicator and something to keep in mind for the next steps.

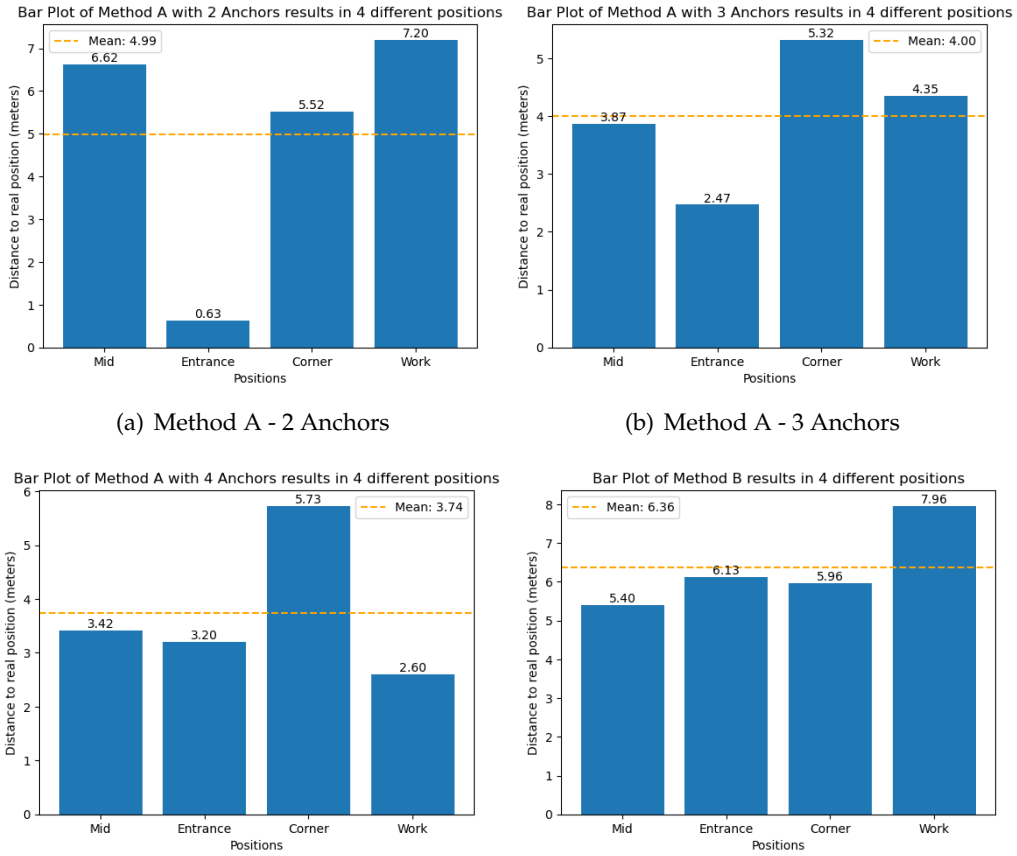
4.1.2 Site T Tests

As already mentioned, the tests on this site were carried out in person, in order to obtain exact measurements of the accuracy that can be achieved with the methods developed and, unlike the previous on-site test, the number of readings taken is considerably higher in order to understand the veracity of the conclusions drawn previously. To do this, a tag will be placed at various locations on the site and around 1000 readings will be taken at each, in order to obtain a considerable number of samples for the location and measuring the distance between the result obtained and the actual location of the tag.

In the Figure 4.5 we can see the 4 locations where the measurements were taken. These locations were chosen to cover the various possibilities of where a tag could be located over time, such as in the middle of the site, on the sides and in the corners.

Figure 4.5: Positions of the tags during the measurements, inside site T.

The figure below is divided into four sub-figures, representing the two developed methods (with Method A divided into three), and contains four bars per sub-figure, corresponding to each position described in Figure 4.5. Based on the results of the tests described, we can conclude that the method with the highest accuracy is Method A. While it shows some variation across all tested positions, it maintains relatively low values, often dropping below 1 meter and frequently ranging around 3 to 4 meters in the remaining cases. It is important to note that in some cases, it is understandable that achieving a "good" accuracy (of around 3 to 5 meters) may not always be possible.



(a) Method A - 2 Anchors

(b) Method A - 3 Anchors

(c) Method A - 4 Anchors

(d) Method B

Figure 4.6: Distance between the location calculated with every method to the real position, in Site T.

4.2 Distance Measurements

The purpose of this section is to present the readings taken locally at several distances of 1, 2 and 4 meters in line of sight, in order to get a general idea of the RSSI variation, depending on the distance. Collections were also made at other distances (3 and 5 to 10 meters), with the aim of trying to understand the RSSI to distance (in meters) curve.

The next figure shows the setup used for all measurements, varying only the distance between the anchor and the tag for the targeted distance. For the tests where it was not possible to use a single table (distances above 4 meters), 2 tables were used to get the correct placement to obtain the desired distances. All these measurements were made inside the T site.

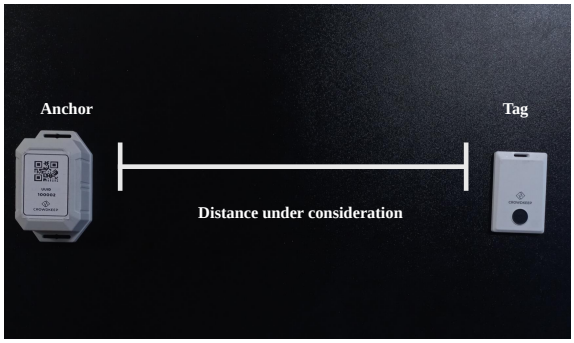


Figure 4.7: Example of the setup used to make the measurements explained.

For the first distances measured, 1000 readings were taken at each distance, but for a better visualization of the results, the values were plotted every 10 points, as was a curve with a rolling mean with a window of 50 samples, as were the maximum and minimum values obtained for each group of measurements. For the remaining measurements, the sample size was also 1000 readings.

4.2.1 Measurements at 1, 2 and 4 meters

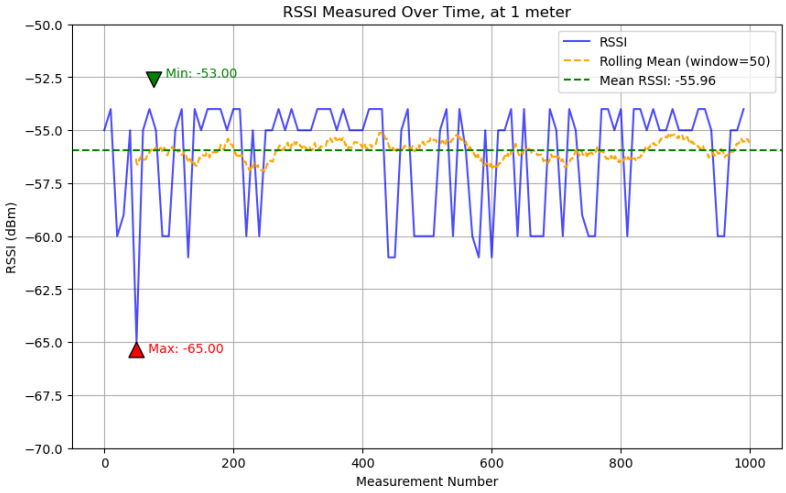


Figure 4.8: Graph of the measurements made at a distance of 1 meter from the anchor to the tag.

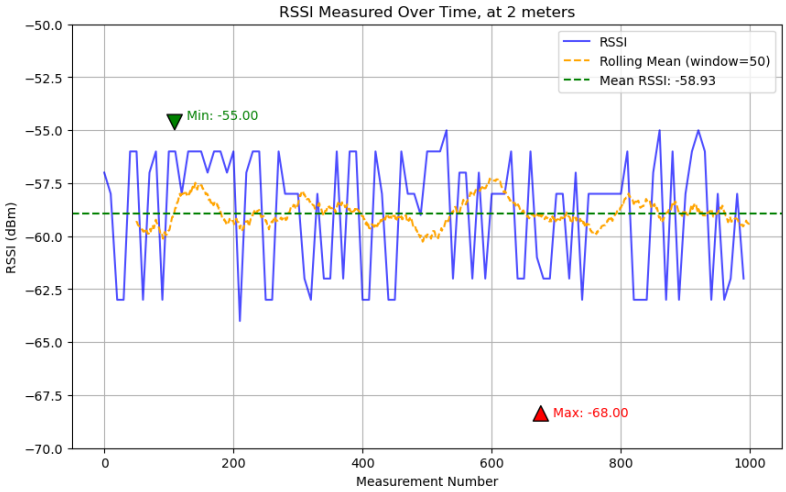


Figure 4.9: Graph of the measurements made at a distance of 2 meters from the anchor to the tag.

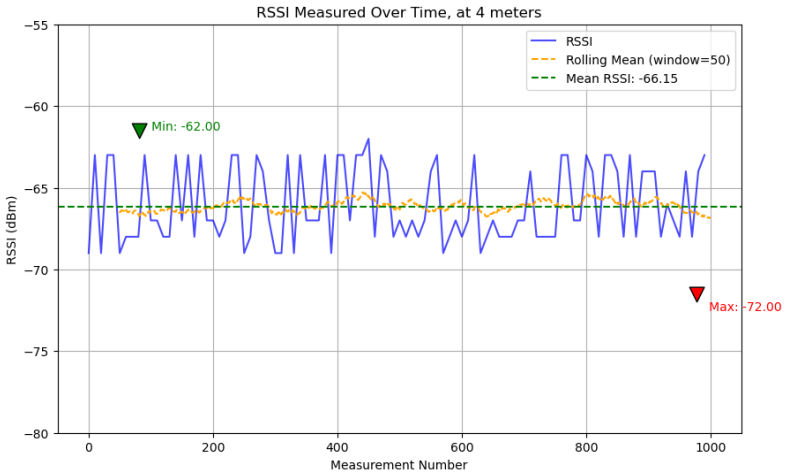


Figure 4.10: Graph of the measurements made at a distance of 4 meters from the anchor to the tag.

As we can see from the graphs presented, the RSSI values that were collected vary constantly, with a variation of up to about 10 to 13 dBm. Firstly, this means that there is a need to carry out some future cleaning of the results obtained, taking into account this oscillatory characteristic of the readings and safeguarding the data from possible noise and interference that could affect the real accuracy of the location engine. The second point to take away from this collection is the reference value for the RSSI at 1 meter, which in this case is approximately -56 dBm.

4.2.2 Remaining Measurements

For the remaining readings, the following graph shows the results obtained at distances of 3 and 5 to 10 meters (as well as those already collected at 1, 2, and 4 meters). These measurements have been taken into account and analyzed using a logarithmic model. The

logarithmic curve fitting method was applied to the data, assuming that the relationship between the distance and the measured values follows a logarithmic decay, based on the previously mentioned equation. In this case, as it was expected, as the distance increases, the measured values decrease at a logarithmic rate. This is consistent with many physical processes, such as sound or signal attenuation, where intensity or, in this case, signal strength decreases logarithmically with distance. The curve fitting allowed for the derivation of the mathematical function.

$$y = -56 - a.log_{10}(x) \tag{4.1}$$

, where y represents the measured value, x is the distance, and a is the constant determined by the regression analysis. This deduction will provide the possibility to more accurately determine the distance between two devices with any possible value of the RSSI, in dBm.

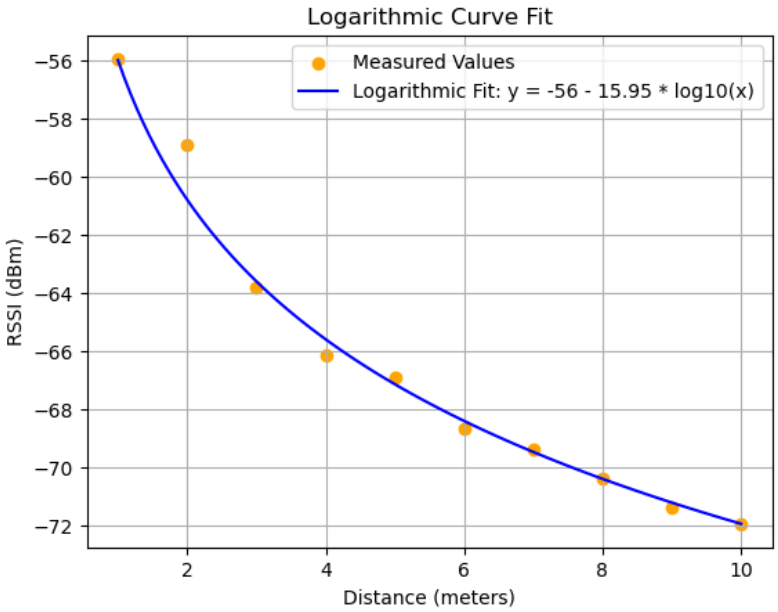


Figure 4.11: Graph of the measured points at the refered distances and the corresponding logarithmic curve fit.

It is therefore concluded from this collection that the optimized expression for converting the value of RSSI to the distance between the two devices, in meters, is as shown below

$$P_r(D) = -56 - 16.log_{10}(D) \tag{4.2}$$

It is important to keep in mind that this optimized curve was deduced for the measurements made by the devices used in the networks that are deployed in the local sites (in the USA and in Portugal), a different curve should be deduced in the case of using different devices with different antennas as the results of the measurements will most probably be different.

4.3 Remote Tests

This section will describe how the remote tests were carried out, comparing the output results of every method to the Wirepas location result. It is important to once again recall the requirements mentioned at the beginning, as they serve as a guideline for what is a good or bad result in this context. All the data used was stored in Google Big Query, and it was extracted as a ".csv" file, has it was explained previously, in section 3.3.1.

4.3.1 Introduction to the remote tests

Regarding the tests carried out remotely, since it was not possible to verify the exact location of the hundreds of tags per site, in order to compare my result with the exact value. For this reason, the solution used is based on a comparison between wirepas' location solution (described in section 3.2.4) and the solution proposed in this dissertation.

As indicated in the section 3.2.4, this algorithm created by the company "wirepas" has an accuracy of around 3 to 4meters, which means that for each solution calculated by this algorithm, there is a radius of around 3 meters (assuming a fixed number from the interval) where the true location of the tag could be. In turn, if my solution is, for example, also 3 meters (to simplify) from the previous solution, it is possible to create a second radius from which the location can be between 0 and 6 (3 + 3) meters away. The following figure helps you to understand the logic described.

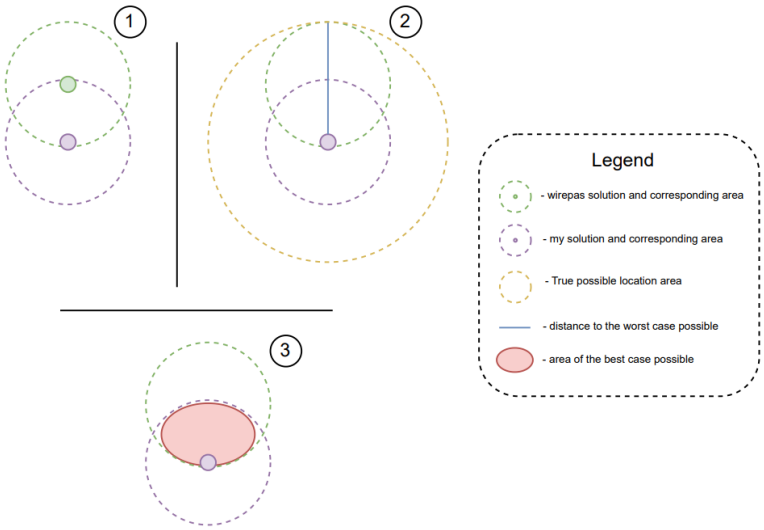


Figure 4.12: Visualization of the comparison done in the tests.

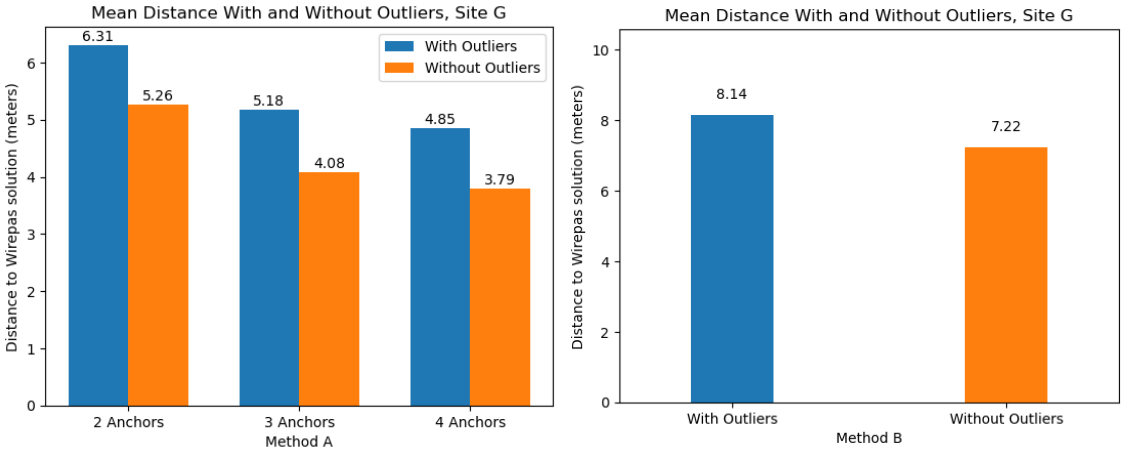
Due to the results of the tests that were carried out locally, we can assume that the actual location of the tags will most often be within the area shown in red, in the previous figure, which gives a maximum accuracy equal to the distance between my solution and the wirepas solution (in the previous example, 3 meters).

As mentioned before, RSSI measurements can vary significantly over time and depend on the environment in which they are taken, therefore, the outlier values were removed. The values considered to be "outliers" were the ones that deviated significantly from the norm, specifically those with Z-scores greater than 3 or less than -3. These Z-scores were used as a benchmark because, for normal distributions, such extreme values fall outside the range where 99.7 of data points typically lie. Hence, these values were flagged as outliers due to their substantial deviation from the mean, and then removed.

For all the tests presented in this section, the optimized equation (presented in section 3.3.6.1) was used, bearing in mind that in all the methods it is necessary to convert RSSI to distance (dBm to meters).

4.3.2 Site G Tests

Now that the introduction to remote testing is complete, the results of the assessments carried out for Site G will be presented. In this case, due to the smaller size of the dataset, there is only one group of graphs that go from February to June. The dataset has a total of approximately 240,000 lines of data, equivalent to approximately 40,000 calculations made, for this site.



(a) Method A

(b) Method B

Figure 4.13: Mean Distance With and Without Outliers using different methods, in site G.

As we can see from the results of the graph for method B, even with a considerable difference between the results before and after removing the outliers, this value for the accuracy is quite high and is not a good indicator for the tests that follow. Taking advantage of the size of the dataset and the time it took to be collected, the purpose of the next graph is to try to conclude on the ideal number of anchors to use in method A. We can see from the graph that multiple variants were used, with 2, 3, 4, 5 and 6 anchors, to calculate the position of each tag.

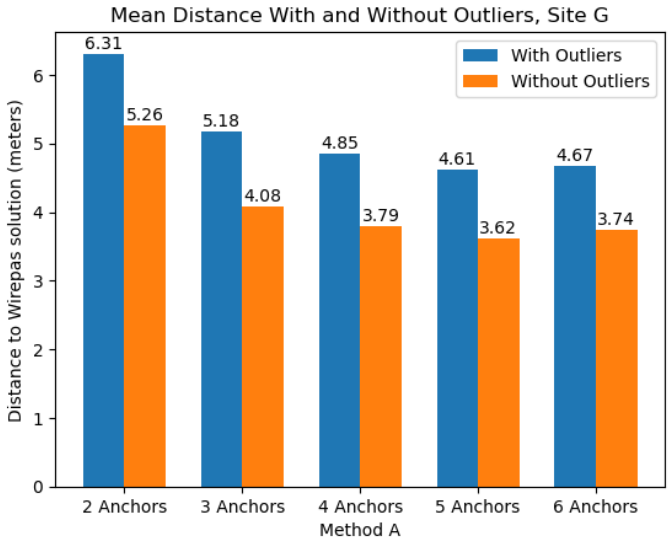
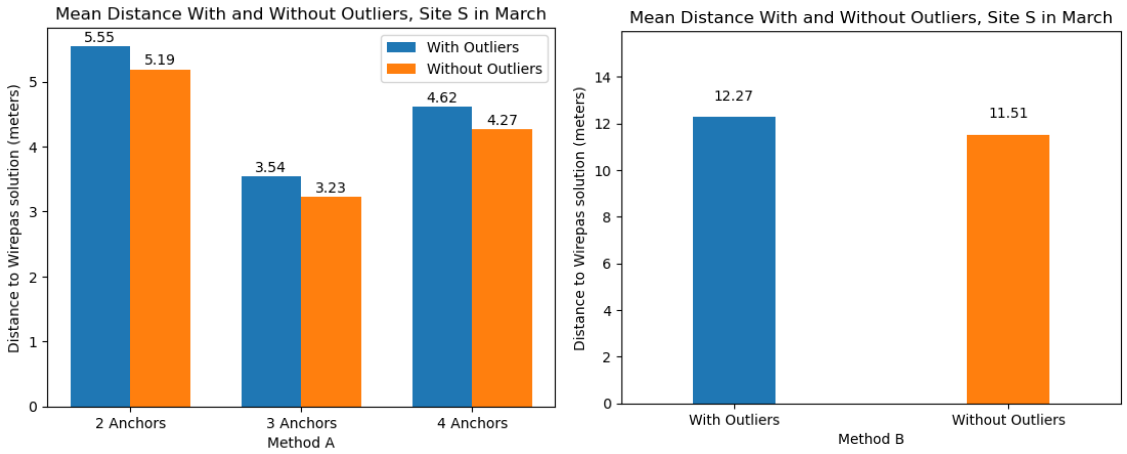


Figure 4.14: Difference of the accuracy using Method A with 2, 3, 4, 5 and 6 anchors.

As the number of anchors increases, the distance between the calculated location and the wirepas location decreases, but not linearly. Bearing in mind that, with the use of a greater number of anchors, there are more readings that will not have a location (example in section 4.5.2), the calculation time increases significantly and the requirements are met with the use of only 3 to 4 anchors in most of the examples that will be presented, so there is no need to use more than 4 anchors to calculate the location, and for that reason, they will not be included in the following tests.

4.3.3 Site S Tests

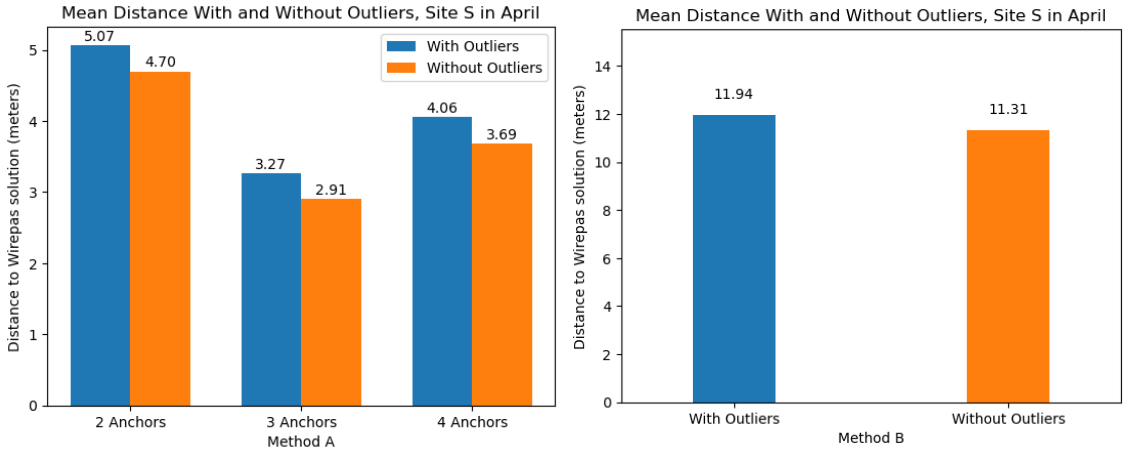
Moving into the Site S, because of the size difference of the dataset, compared to the others, and the difference of the results that they produce, the graphs were separated and they will be presented from March, April, and May, for a total of approximately 2.75 million lines of data (1.05 million, 690,000, and 1.01 million respectively), equivalent to approximately 283,000 calculations made (105,000, 68,000, and 110,000 respectively), just for this site.



(a) Method A in March

(b) Method B in March

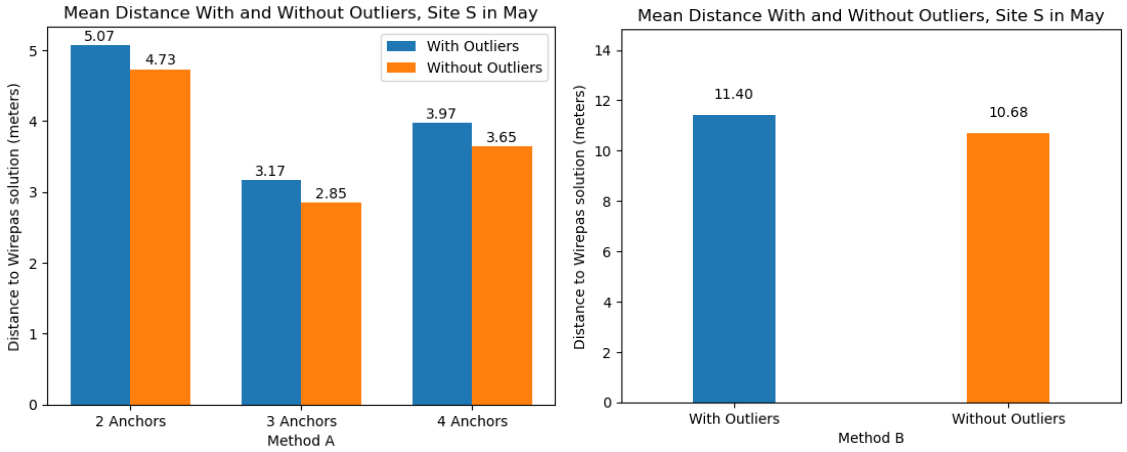
Figure 4.15: Mean Distance With and Without Outliers using different methods in March



(a) Method A in April

(b) Method B in April

Figure 4.16: Mean Distance With and Without Outliers using different methods in April



(a) Method A in May

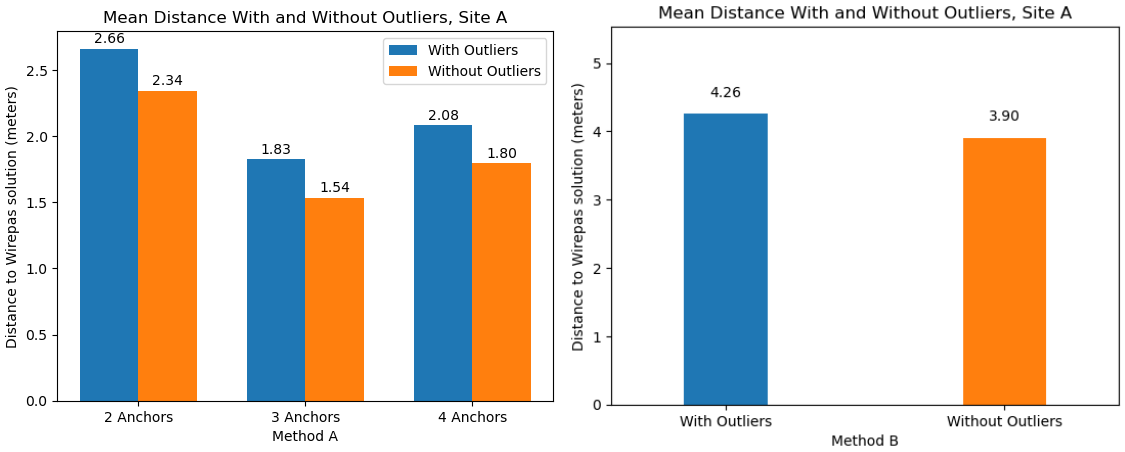
(b) Method B in May

Figure 4.17: Mean Distance With and Without Outliers using different methods in May

By analyzing the graphs above, we can see that over the months the accuracy of all the methods used has varied slightly but has always remained within the same range of values. Looking at the best values for each method, we conclude that the results obtained using method A with 3 anchors were the best ones and very suitable for the objective and requirements initially set. Once again, the results of method B were far from what was expected, with values between 11 and 12 meters that are impossible to use in this context.

4.3.4 Site A Tests

The next site to be tested is the Site A, in this case, there was a smaller ammount of data that was used to run the tests because it was a later deployed site, when compared to the others. This Sites data set has a total of almost 400,000 lines, equivalent to approximately 55,000 calculations to be done. Similar to the first site, due to the size of the dataset in question, all the data has been condensed into a single graph instead of being broken down by month.



(a) Method A

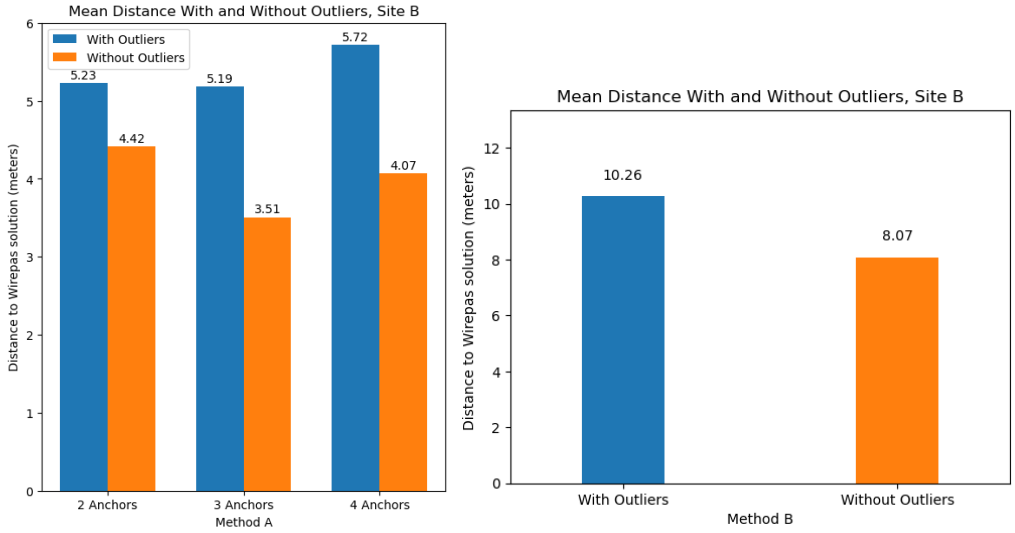
(b) Method B

Figure 4.18: Mean Distance With and Without Outliers using different methods

As we can see from the results above, this site has considerably better accuracy than the previous ones. This was expected considering that this site, for its size, has a large number of anchors in the most common areas where people pass, thus increasing and improving the coverage by the anchors. Like all the other cases, this one has slightly better accuracy when using method A with only 3 anchors, compared to the other A methods, but much better accuracy when compared to method B.

4.3.5 Site B Tests

The next and final site to be tested is the Site B, in this case even though the large size of the dataset, after running the methods through all the months separately, it was determined that the accuracy did not change significantly enough to justify separating the months into different graphs., for that reason, only one will be shown. The dataset for this site is made up of around 3,4 million lines of data, originating a total of 320,000 calculations to be done.



(a) Method A (b) Method B

Figure 4.19: Mean Distance With and Without Outliers using different methods

As we can conclude from the analysis of these last two graphs, the trend remains the same: method A continues to be accurate to the level of the initial requirements, with better accuracy when using 3 anchors. When it comes to using method B, the results are less than what was expected and are still too high for the intended purpose.

4.3.6 Remote Tests Discussion

Below is a table that provides a summary of all the information presented above in order to condense everything so that the necessary conclusions can be drawn regarding the methods developed and which will be the best to use.

Table 4.1: Summary of the tests done to all the sites.

Site	Dataset	Calculations	Method A			Method B
			2 Anchors	3 Anchors	4 Anchors	
Site G	240,000	40,000	5.26	4.08	3.79	7.22
Site S	2,750,000	283,000	5.06	3.14	4.12	10.68
Site A	400,000	55,000	2.34	1.54	1.80	3.90
Site B	3,400,000	320,000	4.42	3.51	4.00	8.07

The first feature that stands out from a quick look at the table is the considerable difference between the results obtained for method A and method B, where it was possible to achieve a minimum of 1.54 meters and a maximum of 5.26 meters of distance to the Wirepas solution correspondingly, for method A and a minimum of 3.90 meters and a

maximum of 10.68 for method B. The average for all these results was: 4.56 meters for Method A with 2 anchors, 3.27 meters with 3 anchors, 3.86 meters with 4 anchors, and 8.75 meters for Method B (with 3 anchors).

Considering the variety of environments in which all these tests were carried out, it is possible to conclude that the algorithms were tested in multiple environments, and that this was done in order to be able to draw accurate conclusions from these results regarding the accuracy of the methods. Having said all this, the best choice for implementation in RTLS is Method A using 3 anchors, due to the consistency of the results obtained by all the tests, and the efficiency in calculating the solutions, in a total of 525,000 results, spread over 4 remote sites and 65,500 m^2, based on the results of the local and remote tests.

4.4 Integration tests

The aim of this section is to present the tests that were carried out to verify the correct operation of the algorithm finally integrated into RTLS and describe how it works. Since the location result generated by the algorithm has already been tested previously and the method to be used has been chosen, there is no need to return to this point. This section will also present some figures to demonstrate the correct integration in the RTLS, these figures show the correct integration in both the data collection part, the storage part and the visualisation part of the system.

4.4.1 Pipeline and Database

The Figure 4.20 shows the output from the pipeline (when run locally) to check the steps and values being processed while the algorithm is running. The Figure 4.21 shows the local output of the pipeline running, the data that it processes (First Red rectangle) and the data that it is going to write in the Database (json message in the bottom). After the data has been processed, similar data like the one on the Figure below is grouped together (for 15 seconds) and then written in bulk to the database, to avoid constant writes and to optimize processing time since this pipeline processes data from all the sites in use.

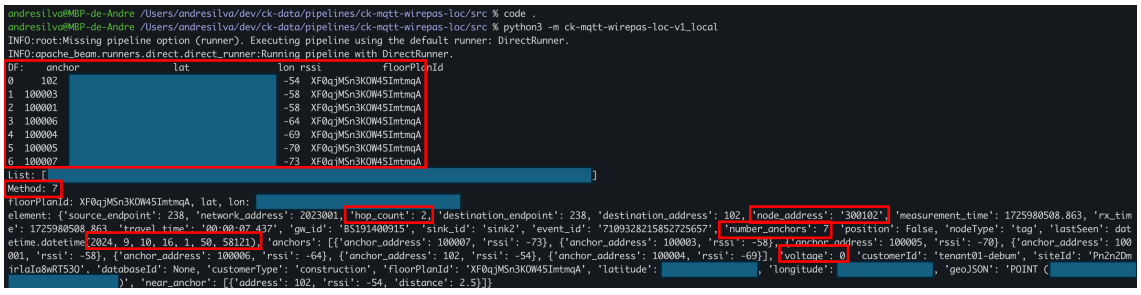


Figure 4.20: Local output from the dataflow pipeline.

If we compare the data in the red rectangles in both top and bottom Figures, we can see that the data that was processed locally was written in Google Big Query correctly, and by analyzing the result of the latitude and longitude variables (that are covered for privacy reasons) it is possible to confirm that the results match and are calculated is correct. This correct comparison makes it possible to conclude that the algorithm has been correctly integrated into the company's RTLS.

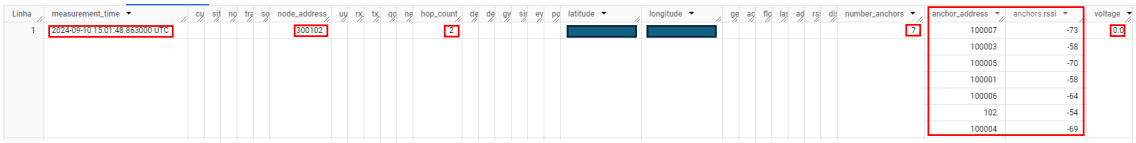


Figure 4.21: Google Big Query Entry after the pipeline process is done writing to the Database.

An important aspect to point out, which was taken into account during development, is the time it takes to red-deploy the pipeline, after making a change to the algorithm, which is around 7 minutes, hence the importance of the two development phases mentioned before. The time between a tag reading the anchors around itself and that data to reach the database is around 10 to 15 seconds.

4.4.2 Web and Mobile App

This application, both mobile and web was developed by Crowdkeep, and is responsible for presenting all the data processed and stored in the database so it can be visualized, follow the movement of each tag and receive specific alerts when the context and applicability make sense. In the figure we can see several blue circles that represent the anchors scattered around the site, in green the location of the tags, and the circles that contain a light purple color and a number inside represent a set of tags that are very close together. This application is similar for both versions, requiring a login to use. The location of these sites has been kept secret for privacy protection.

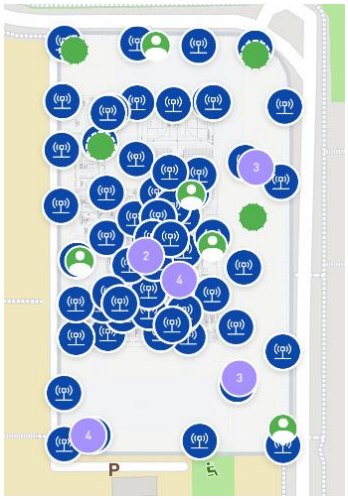


Figure 4.22: Tags real position inside site B.

Figure 4.23 illustrates the dashboard used to monitor and validate the implementation and integration results. This tool allowed the results to be observed live, ensuring that no data was lost during the process and confirming that the implementation had been carried out correctly. It also provided an easy way to verify the functionality and integrity of the system during the integration phase.

Figure 4.23: Dashboard visualization of the Site T.

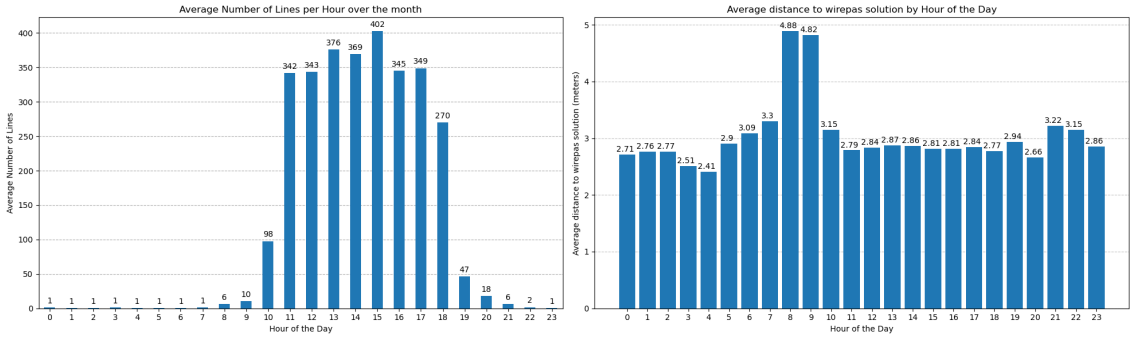
4.5 Site Environment Variations

As mentioned in the first paragraphs of this chapter, the following will provide a variety of descriptive information about the sites, as well as additional data that will help to better understand the results that have been shown so far.

4.5.1 Average tags and influence on the location

The aim of this sub-section is to try to conclude whether or not the number of people on a site and the number of tags communicating with the anchors influences the final result of the tags' location. To this end, the average number of sets of readings that the tags grouped together and sent to be processed for each hour of the entire month of May will be presented, as well as the average distance between the calculated location and the wirepas solution.

To do this, only the May data from Site S will be used, with the solution achieved using Method A with 3 anchors. The decision to use this dataset was due to the large amount of data and the low accuracy attained with this method. So, from the graphs below, we can conclude that the number of people on the site and the number of tags communicating with the anchors does not influence the location calculation made by method A with 3 anchors. However, we can see that there is a peak in the first hours of the day, between 8:00 and 9:59 am.



(a) Average line by hour

(b) Average distance by hour

Figure 4.24: Average made for the month of May

The most likely reason for this spike is the low number of readings taken during these hours. This period corresponds to times when the tags are often located at the "edges" of the site, where there is less coverage and it is more difficult to calculate a more accurate position if it is too close to the site's boundaries.

4.5.2 Usual Number of Readings per group

As has already been mentioned several times, with different methods there is a need to check certain conditions so that they can be applied, such as the number of anchors that a tag sees at any given time. If the method we want to apply needs to use at least 3 anchors but the most common, for a given site, is to see only 2, most of the readings will have no final results.

So let's evaluate the average number of anchors that tags see in each reading they make, using the following graphs in Figure 4.25:

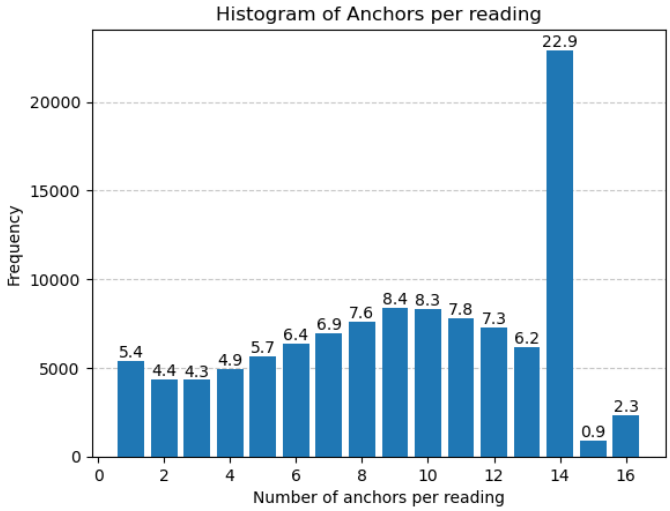


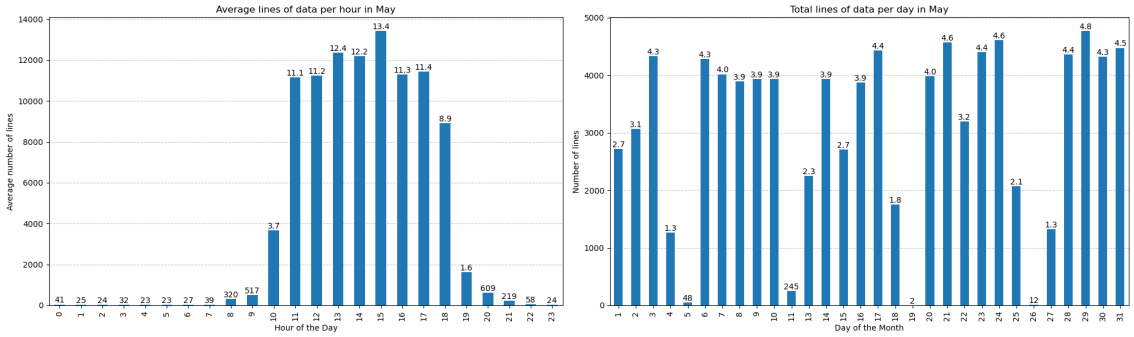
Figure 4.25: Number time that a reading of a tag has several amounts of lines for Site S.

From this we can see that for this given dataset, no algorithm can be applied to around 5,000 lines of data, in cases where only one anchor is read for each group. This number grows as we move along the graph, for example, around 10,000 lines of data will have to be ignored if we want to apply Method A with 3 anchors or Method B and 14,000 if we want to apply Method A with 4 anchors and so on.

These conclusions may not seem relevant, but these cases can be repeated over and over again and even increase the number of occurrences, where the coverage may not be very strong, causing the groups to always have a reduced number of anchors and, consequently, there is never any localization result in those areas.

4.5.3 Average Number of People on Site

Another interesting example to better understand the environment in which the algorithm will be operating is the variation of data throughout the day and month. The following graphs show the fluctuation in the number of people who have been on the site over the hours of the day (similar to the one already shown, now as a total) and over the days of the month. This data refers to the month of May, at site S, due to the size of the dataset, so it is better able to express the variations, bearing in mind that all the remote sites are in very close and similar environments when it comes to these topics.



(a) Total Lines for each hour of the day

(b) Total Lines for each day of the month

Figure 4.26: Differente views of the number of total lines on the May dataset of the Site S.

As we can see from the graphs, there is an expected variation from a school workplace, where there is a higher attendance between 10 a.m. and 6 p.m. where students and teachers work, and a lower attendance at the remaining hours, where janitors and other types of workers carry out their work. As far as the days of the month are concerned, it's easy to infer from the graph which days of the month were weekends and/or public holidays, on which there were almost no readings taken.

5

Conclusion

The aim of this thesis was to develop a location engine capable of using the RSSI values collected to obtain the location of devices operated by small batteries, integrated into school and office environments that are "worn" by their the staff. Due to the environments in which these devices are situated and the purpose they need to fulfil, the accuracy of the engine only needs to be able to indicate the approximate area in which they are located.

During the development of this work, two methods were developed that were capable of applying the objectives of the same, method A: "weighted intermediate point" and method B: "triangulation in 2 dimensions". Of these two methods, the one with the best results was method A using 3 anchors, which was able to find the location of the devices with an approximate accuracy of between 2 and 5 metres. These results show that the objectives set out in the first chapter, as well as the requirements outlined, were achieved.

However, it is important to recognize that there may be other ways of implementing location methods using the RSSI measurements, capable of achieving even better accuracies, with the help of other technologies and methods, which were discussed in the State of the Art chapter. Future research should explore the use of other location methods, using Time Difference of Arrival (TDoA) and Angle of Arrival (AoA) technologies, for cases where the accuracy of the calculated location needs to be less than 2 to 3 metres.

Ultimately, this work highlights not only the importance, but also the growing trend of adopting location technologies to track products, people and other assets in various environments such as schools, hospitals, warehouses and construction sites, standing out as an essential tool for the efficient management and optimization of resources in these spaces.

Bibliography

[1] J. M. Lourenço. *The NOVAthesis LaTeX Template User's Manual*. NOVA University Lisbon. 2021. URL: https://github.com/joaomlourenco/novathesis/raw/main/template.pdf (cit. on p. i).

[2] S. Al-Sarawi, M. Anbar, K. Alieyan, and M. Alzubaidi. "Internet of Things (IoT) communication protocols: Review". In: *2017 8th International Conference on Information Technology (ICIT)*. 2017, pp. 685–690. DOI: 10.1109/ICITECH.2017.8079928 (cit. on pp. 1, 7).

[3] E.-E.-L. Lau and W.-Y. Chung. "Enhanced RSSI-Based Real-Time User Location Tracking System for Indoor and Outdoor Environments". In: *2007 International Conference on Convergence Information Technology (ICCIT 2007)*. 2007, pp. 1213–1218. DOI: 10.1109/ICCIT.2007.253 (cit. on p. 2).

[4] P. Bahl and V. Padmanabhan. "RADAR: an in-building RF-based user location and tracking system". In: *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064)*. Vol. 2. 2000, 775–784 vol.2. DOI: 10.1109/INFCOM.2000.832252 (cit. on p. 2).

[5] N. E. Tabbakha, W.-H. Tan, and C.-P. Ooi. "Indoor location and motion tracking system for elderly assisted living home". In: *2017 International Conference on Robotics, Automation and Sciences (ICORAS)*. 2017, pp. 1–4. DOI: 10.1109/ICORAS.2017.8308073 (cit. on p. 2).

[6] K. Ashton et al. "That 'Internet of Things' thing". In: *RFID journal* 22.7 (2009), pp. 97–114 (cit. on p. 5).

[7] B. Dorsemaine, J.-P. Gaulier, J.-P. Wary, N. Kheir, and P. Urien. "Internet of Things: A Definition and Taxonomy". In: *2015 9th International Conference on Next Generation Mobile Applications, Services and Technologies*. 2015, pp. 72–77. DOI: 10.1109/NGMAST.2015.71 (cit. on p. 5).

[8] S. Li, L. D. Xu, and S. Zhao. "The Internet of Things: a survey". In: *Information Systems Frontiers* 17 (2015) (cit. on p. 6).

[9] Statica. *Internet of Things — Worldwide Revenue*. 2023. URL: https://www.statista.com/outlook/tmo/internet-of-things/worldwide (visited on 2024-01-05) (cit. on p. 6).

[10] Inpixon. *Bluetooth RTLS, Location Tracking, and Positioning*. 2023. URL: https://www.inpixon.com/technology/standards/bluetooth-low-energy (visited on 2024-01-21) (cit. on pp. 7, 32).

[11] H. Li, G. Chan, J. K. W. Wong, and M. Skitmore. "Real-time locating systems applications in construction". In: *Automation in Construction* 63 (2016), pp. 37–47. ISSN: 0926-5805. DOI: https://doi.org/10.1016/j.autcon.2015.12.001. URL: https://www.sciencedirect.com/science/article/pii/S0926580515002411 (cit. on pp. 7, 8).

[12] T. C. Authority. *Location vs Position: When And How Can You Use Each One?* 2020. URL: https://thecontentauthority.com/blog/location-vs-position (visited on 2023-12-18) (cit. on p. 8).

[13] L. Liberti, C. Lavor, N. Maculan, and A. Mucherino. "Euclidean Distance Geometry and Applications". In: *SIAM Review* 56.1 (2014), pp. 3–69. DOI: 10.1137/120875909 (cit. on p. 9).

[14] F. Bonnin-Pascual and A. Ortiz. "UWB-Based Self-Localization Strategies: A Novel ICP-Based Method and a Comparative Assessment for Noisy-Ranges-Prone Environments". In: *Sensors (Basel, Switzerland)* 20 (2020-10). DOI: 10.3390/s20195613 (cit. on p. 9).

[15] J. Bard and F. Ham. "Time difference of arrival dilution of precision and applications". In: *IEEE Transactions on Signal Processing* 47.2 (1999), pp. 521–523. DOI: 10.1109/78.740135 (cit. on p. 10).

[16] C. Laoudias, A. Moreira, S. Kim, S. Lee, L. Wirola, and C. Fischione. "A Survey of Enabling Technologies for Network Localization, Tracking, and Navigation". In: 20.4 (2018), pp. 3607–3644. DOI: 10.1109/COMST.2018.2855063 (cit. on pp. 10, 13).

[17] J. T. Isaacs, D. J. Klein, and J. P. Hespanha. "Optimal sensor placement for time difference of arrival localization". In: *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*. 2009, pp. 7878–7884. DOI: 10.1109/CDC.2009.5399478 (cit. on pp. 10, 11, 22).

[18] K. C. Ho and M. Sun. "Passive Source Localization Using Time Differences of Arrival and Gain Ratios of Arrival". In: *IEEE Transactions on Signal Processing* 56.2 (2008), pp. 464–477. DOI: 10.1109/TSP.2007.906728 (cit. on pp. 10, 11, 22).

[19] X. Li, Z. D. Deng, L. T. Rauchenstein, and T. J. Carlson. "Contributed Review: Source-localization algorithms and applications using time of arrival and time difference of arrival measurements". In: *Review of Scientific Instruments* 87.4 (2016-04), p. 041502. ISSN: 0034-6748. DOI: 10.1063/1.4947001. eprint: https://pubs.aip.org/aip/rsi/article-pdf/doi/10.1063/1.4947001/14720750/041502_1_online.pdf. URL: https://doi.org/10.1063/1.4947001 (cit. on pp. 11, 22).

[20] K. Heurtefeux and F. Valois. "Is RSSI a Good Choice for Localization in Wireless Sensor Network?" In: *2012 IEEE 26th International Conference on Advanced Information Networking and Applications*. 2012, pp. 732–739. DOI: 10.1109/AINA.2012.19 (cit. on pp. 11, 22, 40, 42).

[21] G. Mao, B. Fidan, and B. D. Anderson. "Wireless sensor network localization techniques". In: *Computer Networks* 51.10 (2007), pp. 2529–2553. ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2006.11.018. URL: https://www.sciencedirect.com/science/article/pii/S1389128606003227 (cit. on pp. 11, 12, 22).

[22] Q. Dong and W. Dargie. "Evaluation of the reliability of RSSI for indoor localization". In: *2012 International Conference on Wireless Communications in Underground and Confined Areas*. 2012, pp. 1–6. DOI: 10.1109/ICWCUCA.2012.6402492 (cit. on p. 11).

[23] W. Xue, W. Qiu, X. Hua, and K. Yu. "Improved Wi-Fi RSSI Measurement for Indoor Localization". In: *IEEE Sensors Journal* 17.7 (2017), pp. 2224–2230. DOI: 10.1109/JSEN.2017.2660522 (cit. on pp. 12, 22).

[24] R. Peng and M. L. Sichitiu. "Angle of Arrival Localization for Wireless Sensor Networks". In: *2006 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks*. Vol. 1. 2006, pp. 374–382. DOI: 10.1109/SAHCN.2006.288442 (cit. on pp. 12, 13, 22).

[25] P. Kułakowski, J. Vales-Alonso, E. Egea-López, W. Ludwin, and J. García-Haro. "Angle-of-arrival localization based on antenna arrays for wireless sensor networks". In: *Computers and Electrical Engineering* 36.6 (2010), pp. 1181–1186. ISSN: 0045-7906. DOI: https://doi.org/10.1016/j.compeleceng.2010.03.007 (cit. on pp. 12, 22).

[26] B. R. Jackson, S. Rajan, B. J. Liao, and S. Wang. "Direction of Arrival Estimation Using Directive Antennas in Uniform Circular Arrays". In: *IEEE Transactions on Antennas and Propagation* 63.2 (2015), pp. 736–747. DOI: 10.1109/TAP.2014.2384044 (cit. on p. 12).

[27] R. E. Kalman. "A New Approach to Linear Filtering and Prediction Problems". In: *Journal of Basic Engineering* 82.1 (1960), pp. 35–45. DOI: 10.1115/1.3662552 (cit. on p. 13).

[28] M. S. Grewal and A. P. Andrews. "Applications of Kalman Filtering in Aerospace 1960 to the Present [Historical Perspectives]". In: *IEEE Control Systems Magazine* 30.3 (2010), pp. 69–78. DOI: 10.1109/MCS.2010.936465 (cit. on pp. 13, 14).

[29] KalmanSimp. *Kalman Filter Explained Simply*. Website. 2020. URL: https://thekalmanfilter.com/kalman-filter-explained-simply/ (visited on 2023-11-22) (cit. on pp. 13, 14).

[30] O. Cadet. "Introduction to kalman filter and its use in dynamic positioning systems". In: *Proceedings of Dynamic Positioning Conference*. Houston, USA. 2003, pp. 16–17 (cit. on p. 13).

[31] J. M. Huerta, J. Vidal, A. Giremus, and J.-Y. Tourneret. "Joint Particle Filter and UKF Position Tracking in Severe Non-Line-of-Sight Situations". In: *IEEE Journal of Selected Topics in Signal Processing* 3.5 (2009), pp. 874–888. DOI: 10.1109/JSTSP.2009.2027804 (cit. on p. 14).

[32] J. Guo, H. Zhang, Y. Sun, and R. Bie. "Square-Root Unscented Kalman Filtering Based Localization and Tracking in the Internet of Things". In: *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*. 2012, pp. 1824–1829. DOI: 10.1109/TrustCom.2012.265 (cit. on p. 14).

[33] A. Bistri. *Kalman Filter for Intelligent Tracking*. Website. 2021. URL: https://www.skyradar.com/blog/kalman-filter-for-intelligent-radar-data-enhancement (visited on 2023-12-20) (cit. on p. 14).

[34] R. Santos, R. Leonardo, M. Barandas, D. Moreira, T. Rocha, P. Alves, J. P. Oliveira, and H. Gamboa. "Crowdsourcing-Based Fingerprinting for Indoor Location in Multi-Storey Buildings". In: *IEEE Access* 9 (2021), pp. 31143–31160. DOI: 10.1109/ACCESS.2021.3060123 (cit. on pp. 14, 15, 18).

[35] R. Faragher and R. Harle. "Location Fingerprinting With Bluetooth Low Energy Beacons". In: *IEEE Journal on Selected Areas in Communications* 33.11 (2015), pp. 2418–2428. DOI: 10.1109/JSAC.2015.2430281 (cit. on p. 14).

[36] E. Parliament and C. of the European Union. *REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL*. 2016-05-04. URL: https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679 (visited on 2024-01-04) (cit. on p. 15).

[37] J. McNeff. "The Global Positioning System". In: *IEEE Transactions on Microwave Theory and Techniques* 50.3 (2002), pp. 645–652. DOI: 10.1109/22.989949 (cit. on pp. 16, 17).

[38] D. Odijk, N. Nadarajah, S. Zaminpardaz, and P. J. G. Teunissen. "GPS, Galileo, QZSS and IRNSS differential ISBs: estimation and application". In: *GPS Solutions* 21 (2017). DOI: 10.1007/s10291-016-0536-y (cit. on p. 16).

[39] G. Djuknic and R. Richton. "Geolocation and assisted GPS". In: *Computer* 34.2 (2001), pp. 123–125. DOI: 10.1109/2.901174 (cit. on p. 16).

[40] F. S. T. Van Diggelen. *A-gps: Assisted gps, gnss, and sbas*. Artech house, 2009 (cit. on pp. 16, 17).

[41] P. K. Enge. "The Global Positioning System: Signals, measurements". In: *International Journal of Wireless Information Networks* 1 (1994), pp. 83–105. DOI: 10.1007/BF02106512 (cit. on pp. 16, 17).

[42] Wirepas. *Wirepas: Infinitely scalable IoT connectivity*. 2023. URL: https://www.wirepas.com/ (visited on 2023-12-22) (cit. on p. 18).

[43] Wirepas. *Wirepas Massive Tracking Installation guidelines*. 2022. URL: https://developer.wirepas.com/support/solutions/articles/77000513810-wirepas-massive-tracking-installation-guidelines (visited on 2023-12-22) (cit. on pp. 19, 20).

[44] C. Systems. *Cisco Spaces: Asset Locator Configuration Guide*. 2017. URL: https://www.cisco.com/c/en/us/td/docs/wireless/spaces/asset-locator/b-asset-locator-cg.pdf (visited on 2024-01-02) (cit. on p. 20).

[45] Inpixon. *Inpixon RTLS Platform*. 2024. URL: https://www.inpixon.com/rtls-platform (visited on 2024-01-03) (cit. on p. 21).

[46] Inpixon. *Industrial-Grade Asset Tracking System*. 2024. URL: https://www.inpixon.com/use-cases/asset-tracking (visited on 2024-01-03) (cit. on p. 21).

[47] Google. *Firestore | Firebase*. 2024. URL: https://firebase.google.com/docs/firestore (visited on 2024-07-18) (cit. on p. 24).

[48] Google. *BigQuery documentation | Google Cloud*. 2024. URL: https://cloud.google.com/bigquery/docs (visited on 2024-07-18) (cit. on p. 24).

[49] S. F. Python. *General Python FAQ — Python documentation*. 2024. URL: https://docs.python.org/3/faq/general.html (visited on 2024-07-18) (cit. on p. 25).

[50] S. F. Python. *What is Python? Executive Summary*. 2024. URL: https://www.python.org/doc/essays/blurb/ (visited on 2024-07-22) (cit. on p. 26).

[51] A. Inc. *Anaconda Distribution*. 2024. URL: https://docs.anaconda.com/anaconda/ (visited on 2024-07-22) (cit. on p. 26).

[52] A. P. Ltd. *Batch Processing vs Stream Processing: 7 Key Differences*. 2023. URL: https://atlan.com/batch-processing-vs-stream-processing/ (visited on 2024-07-29) (cit. on p. 27).

[53] S. F. Apache. *About Apache Beam*. 2024. URL: https://beam.apache.org/about/ (visited on 2024-07-29) (cit. on p. 27).

[54] S. F. Apache. *Apache Beam Documentation*. 2024. URL: https://beam.apache.org/documentation/ (visited on 2024-07-29) (cit. on p. 27).

[55] W. Ltd. *Wirepas Positioning Dimensioning guide*. 2022. URL: https://developer.wirepas.com/support/solutions/articles/77000499107-wirepas-positioning-dimensioning-guide (visited on 2024-09-10) (cit. on p. 32).