UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

FIB

MASTER THESIS

TITLE: Study of Bluetooth Low Energy as a Contact Tracing Technology

DEGREE: Master in Innovation and Research in Informatics

AUTHOR: Aina Maín Nadal

SUPERVISOR: Jorge García Vidal

CO-SUPERVISOR: Jose Maria Barceló Ordinas

DATE: June 23, 2022

Abstract

In recent months we have seen how Bluetooth Low Energy has become, due to the epidemiological situation, the most used technology for contact tracing. With this in mind, the objective of this project is to test and contrast the robustness and functionalities offered by Bluetooth Low Energy for contact tracing. And overall to see the accuracy and capabilities it can offer.

In the current scenario we can see that Bluetooth Low Energy is already being used as a contact traceability technology, its wide distribution in COVID applications is an example of this. But it is not the only scenario where it is used for this purpose, as introduction to the project and evaluation to the state of art we are going to view some scenarios were Bluetooth is used in contact tracing or similar scenarios.

A challenge with current implementations of digital contact tracing is that it is not clear that whenever two devices are nearby, they can produce a contact. In general, these devices should be able to exchange beacons with privacy mechanisms such as Diffie-Hellman among others, in distances that are some meters. The objective that we will seek is to check if, with the capabilities offered by Bluetooth Low Energy (BLE), we can study the accuracy of these contact tracing techniques in sensing devices.

To study digital contact tracing techniques using sensing devices we will integrate Internet of Things elements into the Digital Contact Tracing architecture. We will not only evaluate the proximity between two people for their contagion, but we will also add other factors, such as air quality through Internet of Things network nodes. Secondly, we are trying to determine as reliably and accurately as possible, within the limitations of Bluetooth, the accuracy between the transmitter and the receiver.

For this we will first present a new architecture for digital contact tracing, and some of its adaptations. Along with the adaptation of an Internet of Things node to add the ability to communicate via Bluetooth Low Energy and the advantages to be gained in this case. Along with these modifications we will try to apply a solution that does not represent a large increase in the cost of implementation.

We will then evaluate the accuracy and reliability of Bluetooth Low Energy for determining distances between a transmitter and receiver. In addition to determining the contact risk between two people for the possibility of contact, we also evaluate the quality of the signal between contacts in several scenarios. With this we can evaluate how accurate is the communication through our Internet of Things node and other Bluetooth Low Energy elements.

As a result of the nature of Bluetooth it will be necessary to add a number of pre-processing elements before we can begin to evaluate its behavior or attempt to predict contact risk between Bluetooth Low Energy devices. For that we will use a particle filter using Monte Carlo methods to smooth the signal and clean and remove outliers and reading errors.

The factor that we want to evaluate to study its potential for contact tracing is the behavior and the relationship that we can deduce from the received signal strength and the distance between the two devices. For that, we will first generate our own beacon exchange data set between two Bluetooth Low Energy devices in different scenarios based on distance, positions and other characteristics.

We will then evaluate the mechanisms and the accuracy we can obtain for determining the contact risk in two ways. First, we will try to calculate the risk based on the relationship between the signal strength and the distance between the transmitter and the receiver. Secondly, we will try to classify contact risk based on distance ranges and apply a Machine Learning classifier to determine the approximate distance between the transmitter and the receiver.

With all this added information we will be able to evaluate and determine conclusively if BLE is a potential technology for digital contact tracing protocols. And also if adding Internet of Things elements to the Digital Contact tracing architecture is an option to improve the determination of the contagion risk.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1. DESCRIPTION OF TECHNOLOGIES

As an introduction to the study of Bluetooth Low Energy as contact tracing, we will first make a brief introduction to the characteristics of Bluetooth Low Energy. We will also look at scenarios or proposals where Bluetooth Low Energy has been used as contact traceability or similar situations.

1.1. State of the Art

Currently Bluetooth Low Energy has seen an increase in the number of fields in which it is used as a result of the COVID-19 pandemic. The high transmission rate of COVID-19 has resulted in the impossibility of manual contact tracing. Digital contract tracing as an autonomous mechanism for contact tracing and notification of exposure to the infection has been presented as a mechanism that could contain contagion in a rapid manner[6].

In this case it has emerged as the best technology to quickly and effectively implement digital contact tracing[13][9]. Bluetooth Low Energy has been established as the best technology for digital contact tracing in terms of accuracy, functionality, security and privacy.

Although it presents some solvable problems in the field of security and ethics[11], when evaluating accuracy, consumption and need for previous infrastructure, it considerably outperforms alternatives such as Wifi, GPS, classic Bluetooth, etc. And it has proven its functionality in the multiple implementations of several nations as epidemiological control mechanisms during the COVID-19 pandemic.

In other related scenarios Bluetooth Low Energy has also been used as a mechanism to evaluate distances in an approximate manner, similar to the main objective in contact tracing. For example in the mechanisms of cars to detect the distance between the keys and the vehicle to unlock it automatically[8]. Or to detect if all the children in a school bus are inside the vehicle[15].

1.2. Bluetooth Low Energy

Bluetooth Low Energy[1] is a wireless personal area network technology designed by the Bluetooth Special Interest Group. It is mainly oriented to new applications in the fields of health, fitness, security and home entertainment. It is independent of the original Bluetooth and has no compatibility, yet both technologies can coexist and be supported by the same device. Its first implementation is developed by Nokia in 2006 under the name of Wibree and is integrated into Bluetooth 4.0 in December 2009 as Bluetooth Low Energy.

The technical characteristics of Bluetooth Low Energy are specified in the Bluetooth Core specification 4.0[3]. Bluetooth Low Energy operates in the 2.4Ghz spectrum, the same band as traditional Bluetooth, but using a different set of channels, 79 1 Mhz channels of the original Bluetooth versus 40 2 Mhz channels of Bluetooth Low Energy. As in Bluetooth

[1] Bluetooth LE or BLE and marketed as Bluetooth Smart

Low Energy the transmission power and bandwidth are lower in contrast to the original Bluetooth. In table 1.1 we can see a comparison of some of the technical characteristics between original Bluetooth and Bluetooth Low Energy.

Specification	Traditional Bluetooth	Bluetooth LE
Application throughput	0.7–2.1 Mbit/s	0.27–1.37 Mbit/s
Wake latency	100 ms	6 ms
Minimum total time to send data	0.625 ms	3 ms
Power consumption	1 W	0.01–0.50 W
Peak current consumption	30 mA	15 mA

Table 1.1: Feature comparison between Bluetooth Oringial and Bluetooth LE

Communication between Bluetooth LE devices is based on two main elements GAP, which handles communication and advertise, and GATT, which sets up the exchange of information.

1.2.1. GAP

Generic Access Profile(GAP) is the protocol that manages the advertise and controls the connection between BLE devices. It determines the visibility of the device and determines how two devices can communicate with each other. In GAP we have defined several roles to encompass the behavior of the devices but they can be grouped into two main roles.

- **Peripheral:** Small, low-resource devices with limited features. In these cases they are devices that also have a specific and unique function in Bluetooth Low Energy. These devices are some such as a pollution sensor, heart rate monitor, etc.
- **Central:** Devices with more capabilities and features that connect to peripherals. They are usually mobile phones, tablets, computers, etc.

This definition of roles can also be classified differently. As the device that will actively connect to request information (Central) and the passive device that will provide the information (Peripheral). In addition, these roles are not fixed to a device. A device can be peripheral for some specific communications and devices and act as central for other.

In GAP we have two main messages to use in the process of advertising our device Advertising Data and Scan Response. In both cases the two messages have the same payload and can contain up to 31 bits of data.

- **Advertising Data:** It is the main and mandatory message in GAP, it announces the existence of our device to others in the range and the identifiers of the services that have our device.
- **Scan Response:** This is a secondary GAP message, when a central device detects us it may request more information about the device, in that case this message is sent with extra information such as name, manufacturer, etc.

1.2.2. GATT

The Generic Attribute Profile(GATT) it defines the way that two Bluetooth Low Energy devices transfer data back and forth between them. For this Bluetooth Low Energy define three concepts, Profiles, Services and Characteristics. GATT is used after the connection between two devices has been correctly established and one of the important characteristics of GATT is that the connections are exclusive, a peripheral device can only be connected to only one unique central device.

GATT used a client-server relationship for the data exchange where the Peripheral is the Server and the Central is the client. The Client will always establish communication with the Server and the Server will respond to the Client's requests. For this purpose, the client will communicate at an interval determined by the client and the server when establishing communication to obtain new available information. In figure 1.1 we can see diagram illustration the communication between Client and Server.

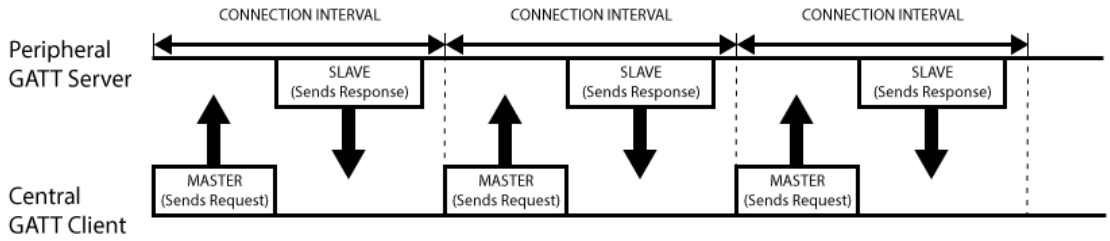


Figure 1.1: Communication in GATT

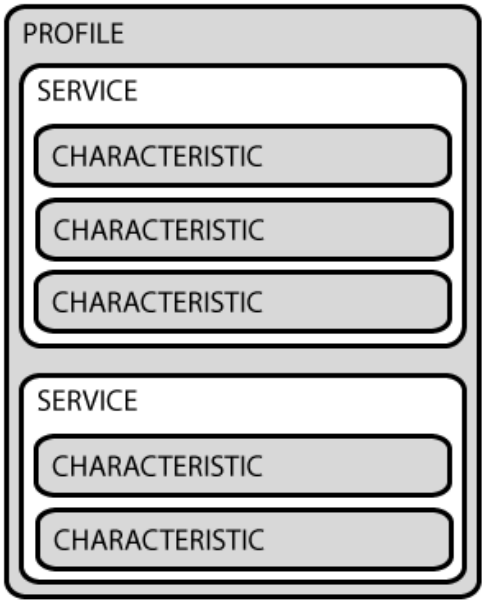


Figure 1.2: A combination of profiles, Services and Characteristics in Bluetooth Low Energy

In GATT transactions in Bluetooth Low Energy are based on high-level, objects called Profiles, Services and Characteristics. This objects follow a specified hierarchy where one includes the other. In this case Service include one or more Characteristics, and Profile includes one or more Services, a schematic of the hierarchy is available in figure 1.2.

Profile, the superior of the three, does not exist in the peripheral, it is an abstraction to identify a predefined collection of services determined by Bluetooth SIG or the device manufacturer. An example of this is the HRT, which combines the Hate Rate Service and the Device Information Service[2].

Services is the division of information into logical entities, where each specific element of that information is a characteristic. Following the previous example of the Heart Rate Service, this is made up of 3 characteristics,

[2]All officialy adopted profiles by Bluetooth SIG[16]

Heart Rate Measurement, Body Sensor Location and Heart Rate Control Point. We can see how all of them are different specific information related to the Heart Rate information in general. A service can contain several characteristics and they are differentiated by means of a numerical ID, this can be of 16-bits if it is a standard service or 128 bits for other cases.

The lowest level in GATT communication, the Characteristic, represents a single piece of information. In the case of the heart rate, it will be the value of the beats per minute, but it is not tied to a single value, in the scenario of a gyroscope the Characteristic is an array of three values, the three axes. As in the services, each Characteristic is differentiated by a standard value of 16 bits or a custom value of 128. The Characteristic are the primary element of communication in Bluetooth Low Energy, because when we communicate with a device what is finally done is to read or write the information of a particular Characteristic.

1.3. Contact Tracing

Contact tracing, in relation to public health, is the process of identifying persons who have had contact with an infected person. By identifying contacts and acting accordingly, the aim is to reduce the number of infected persons in the general population. Although this seems to be a recent subject with the COVID-19 pandemic, it is being done for other infections, such as tuberculosis, Ebola, HIV, measles, etc.

The main objectives of contact tracing are the following:

- Interrupt transmission and reduce spread.
- Alerting contacts to the possibility of infection
- Provide diagnosis, counseling and treatment
- If the infection is treatable, help prevent reinfection
- To understand the epidemiology in the population

Contact tracing has proven to be one of the most important pillars of infectious disease control in recent decades. One of the main scenarios where contact tracing has been used and has proven its effectiveness was the WHO smallpox eradication campaign in the 1970s as this was one of the vital elements for its eradication.

Contact tracing has evolved in recent years with digital contact tracing, as we have seen from his proposals and implementations in the COVID-19 epidemic. In the traditional model, contact tracing and infectivity is done by interviewing the infected person to obtain a list of contacts and identify possible infected persons. In digital contact tracing, the method for identifying contacts is based on a tracking system, mainly based on mobile devices. Considering the mobile limitations there are two main mechanisms for this, GPS and Bluetooth. These mechanisms can be reinforced with other contact tracing methods such as those based on QRs.

In figure 1.3 we can see a schematic of one of the original contact tracing proposals for COVID-19, in this case based on GPS and QRs. The example in this case works as follows

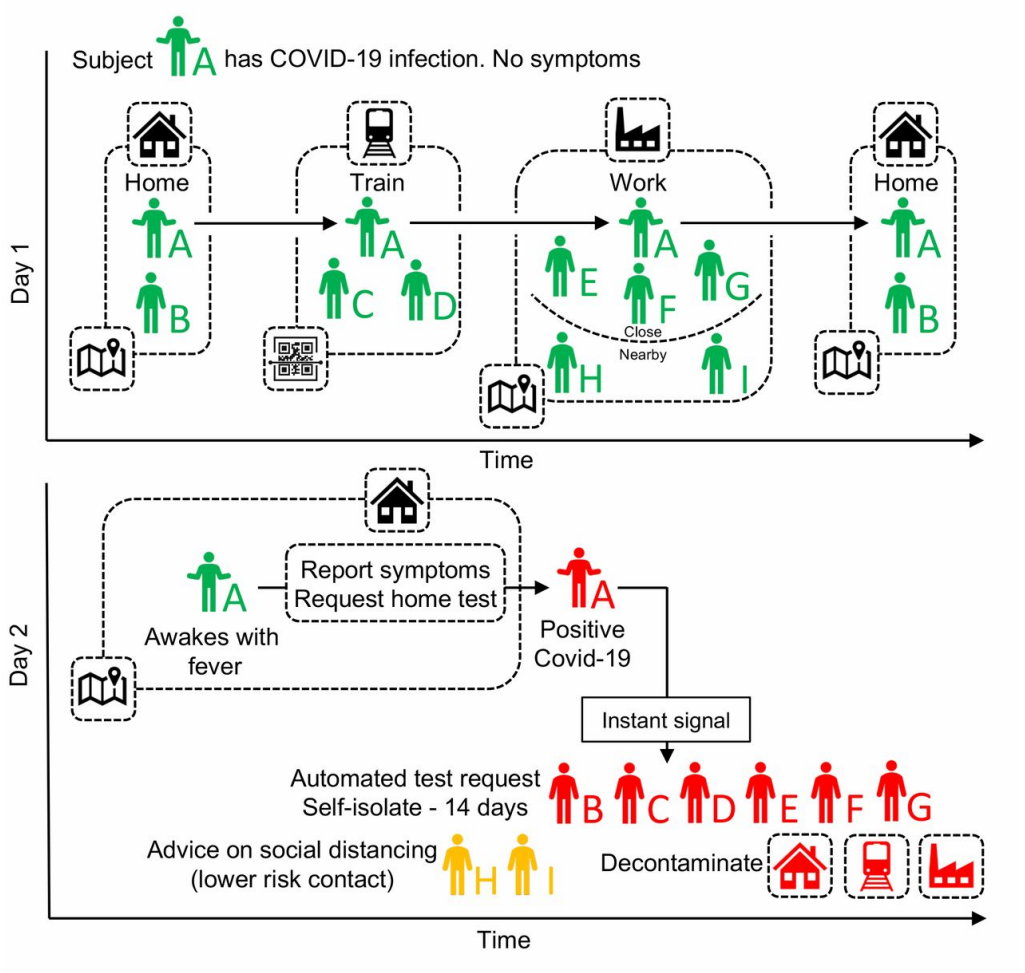


Figure 1.3: A schematic of app-based contact tracing[7]

Individual A, and all other app users, are tracked by GPS and located in relation to other app users. Supplementing the location with QR codes in scenarios where GPS is not accurate enough, in this case it would be the train scenario.

Individual A requests a COVID-19 test and obtains a positive result. Based on the location information collected from all devices, all users in the vicinity of A are immediately notified that they are close contacts of a positive test. With the same App, A's isolation is communicated and A's contacts are notified of the need for testing and quarantine.

In the case of the current pandemic, Bluetooth will end up being used for contact traceability as opposed to GPS. In this case the exact location data is not used, as in GPS, but the devices exchange ID's with each other to have information about nearby contacts.

1.3.1. COVID Apps

One of the most new scenarios we have seen in recent years in relation to contact tracing via Bluetooth Low Energy has been the traceability of COVID-19 infections. Added to this is the fact that it is one of the first real digital contact tracing scenarios. The main mecha-

nism on which COVID applications are based is Exposure Notification[3][4], a protocol and framework designed by Google and Apple to facilitate contact tracing during the COVID-19 pandemic.

The Apple/Google protocol is similar to the Decentralized Privacy-Preserving Proximity Tracing protocol created by the European DP-3T consortium, but in this case the protocol is implemented at the operating system level which allows for better optimization as a background process.

The operation for the contact tracing of GAEN Apps and similar Apps is as follows:

1. The system will generate a random ID for your device encrypted with a secret daily key. These IDs and the Bluetooth MAC address change every 10-20 minutes to help ensure privacy and that a user's location and contacts cannot be identified.
2. The phones will exchange in the background the generated keys or ID's using only the Bluetooth Low Energy GAP protocol. The devices will never fully connect to each other.
3. The device record the received IDs retaining them locally for 14 days.
4. If a user tests positive, it will send the secret daily key of the last 14 days to a central server. These are then made available to all devices.
5. The devices periodically check if the server keys match any of the received messages. If a match is found, it indicates that contact has occurred and the user is notified of the risk of infection.

Although GAEN is the most widespread method for COVID tracing, there are other implementations, some centralized, such as the French application, others decentralized, such as the Covid Watch proposal. But even with multiple implementations they all work the same at the Bluetooth Low Energy level, they all exchange an ID, key or beacon using GAP to identify contacts. The main changes are at the level of the network architecture, or the cryptography to generate the keys.

1.3.2. Similar Scenarios to Contact Tracing

Among the possible scenarios in the Bluetooth Low Energy world, there are more possibilities than contact tracing. There are also those that also determine positions or distances,h related to what is being pursued in contact tracing.

One of the main scenarios where Bluetooth Low Energy is used for distance calculation is for indoor positioning systems. In these scenarios the use would be the same as traditional GPS, but focused indoors where GPS accuracy and coverage is limited. For example in the case of a hospital, to know our location in it, area and floor, and to be able to navigate through it knowing our location inside.

In this case the operation would be similar to GPS, knowing the approximate distance between the receiver and multiple BLE transmitters with a fixed location we can calculate the position of the receiver.

[3] Also called GAEN(Google/Apple Exposure Notification) and originally known as Privacy-Preserving Contact Tracing Project

CHAPTER 2. NEW ARCHITECTURE

We have seen the contact tracing model based mainly on the devices of each user exchanging a token between them to identify people nearby. But in this case we are limiting the possibilities that digital contact tracing can offer. In this case we combine the functionalities of Digital Contact tracing with those of an Internet of Things Node that detects the air quality and has been adapted to communicate its information through Bluetooth Low Energy. In this way we extend the functions of the Internet of Things node and offer its information to a digital contact tracing network.

2.1. Contact tracing and Internet of Things

The proposal in this case would be to expand the functionalities of contact tracing by adding the features and functionalities offered by Internet of Things devices. In this case the tracing and risk calculation, present in these applications based only on contact time and distance, would be extended to the possibility of using environmental factors. Figure 2.1 shows a schematic of the new digital contact tracing model.

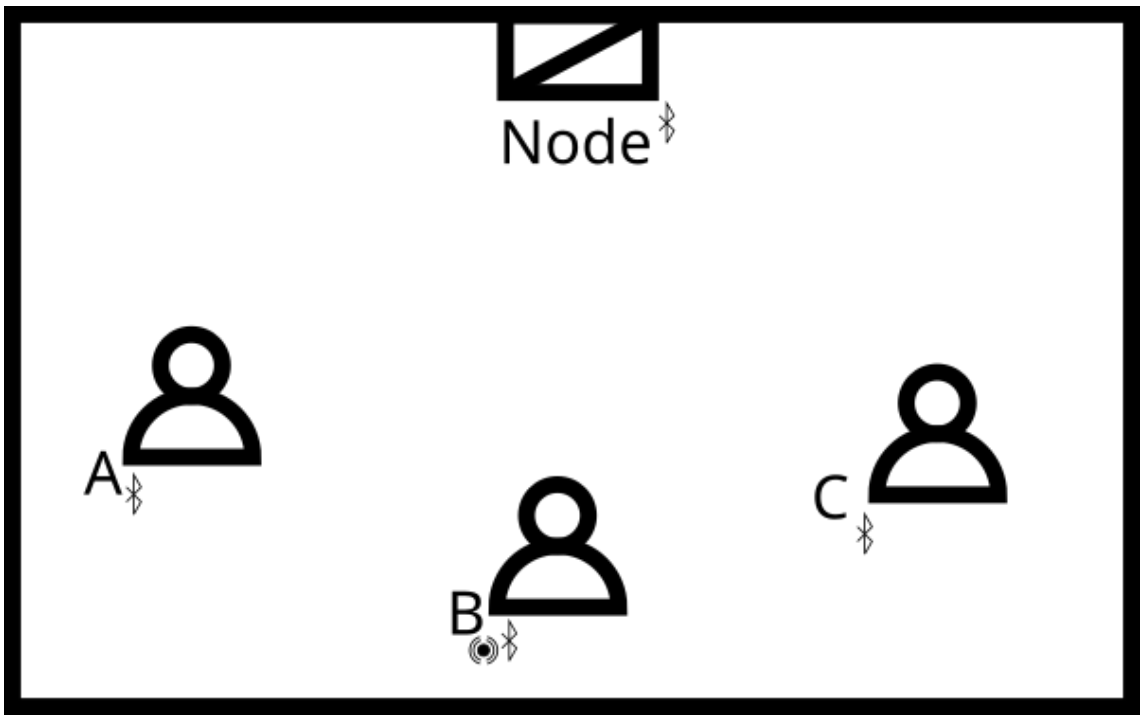


Figure 2.1: Schematic of the new digital contact tracing model

In the new model we can find the elements already present in digital contact tracing in users A and C. In these cases both have a telephone with its respective application that has the task of generating and exchanging the keys to perform the tracking between them.

A new element in this case would be B, who is not performing the exchange of Bluetooth Low Energy beacons for contact tracing through his phone. In this case he is using a token. In this case the Bluetooth token would be a small device with the sole purpose of sending and receiving packets for contact tracing, with this alternative get the advantage of having a device adapted only for contact tracking. This also allows to deploy these elements in

specific cases of a large accumulation of people, such as concerts, sporting events, etc..

In these cases an external mechanism would be required to send the tokens to the central server, either an application on the phone that communicates with the token, or a wired connection to a computer to send information. In our model the communication is done with an application that uses standard Bluetooth periodically to connect, collect the messages received and send the new tokens to be issued.

In our scenario, the most differentiating element with respect to the usual digital contact tracing would be the Internet of Things node, a sensor node that collects environmental and atmospheric values.

As seen in the pandemic, one of the most important determinants of risk has been indoor air quality. A poorly ventilated environment leads to a higher risk of infection. In our scenario, the node is dedicated to measuring air quality, through airborne particles, CO2, NO and others. The node would have the objective to offer through Bluetooth Low Energy the data it has collected to make them available to the users' digital contact tracing applications, both in the case of A and C, as well as in the case of B.

This new element offers an extension of the functionalities of Digital Contact tracing. First of all it allows to calculate the risk based on more factors than just time. Knowing the distance also to the node we can deduce how dangerous the current situation is, from the point of view of a contagion.

This also allows to extend the information offered to the users to actively avoid contagion. Let's assume that this design is applied in public interiors, from restaurants to hospitals. The application can indicate to users when entering a new room the risk of being infected based on the quality of air renewal. This value can also be used as an indication for the authorities assigned to the management of the interior. Notifying when it is necessary to initiate a new process to renew the air of the instance.

2.2. The Node

For our scenario we will rely on the contamination nodes that have been previously used by the research group, and in which I participated in the design[5]. The node is originally designed to operate in two ways, either connected to the mains with continuous operation, or using a set of batteries, where the node is switched on and off for fixed periods where it reads pollution data and stores it.

The architecture of the node can be seen in Figure 2.2. It can be seen that the node can be divided into two blocks or characteristic elements. First, a Raspberry Pi as a central board. This central board has several objectives, the two main ones are the management of the communication with the sensors and the sending of the collected data through a wireless network, in this case it would be through the mobile network. The central board also performs some additional functions, such as synchronizing an Real Time Clock or storing the readings in case errors occur when sending data over the mobile network.

The second element would be the sensor boards. In this case the sensor boards are an Arduino Nano connected to two electrochemical sensors, following the manufacturer's instructions. Then through an I2C bus transmits the information to the central board. In the generic architecture, any sensor or communication element that supports I2C communi-

cation can be added as a sensor board.

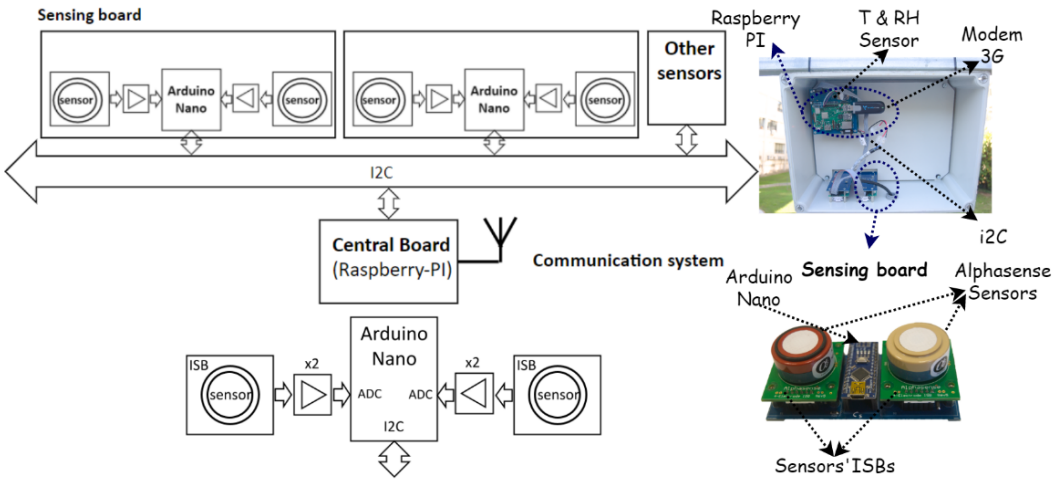


Figure 2.2: Schematic of the Node

In comparison between the model continuously connected to a power supply network or the functional ones with batteries, the differences are much smaller. The main difference would be the frequency of the readings, as it is necessary to reduce the consumption for the batteries it is not read continuously but in cycles of readings every 5 minutes. In the following section we will cover in more detail the differences and the causes of these modifications between the node continuously connected to the grid and the battery node.

2.2.1. An Autonomous Node

One of the first elements to take into account is that we have two nodes, which read the same continuum values but one of them does it by feeding continuously to the network and the other one does it by using batteries. This presents a first problem, if both of them were to be operated in the same way in the readings, the battery life would be reduced to a few hours.

In contrast, the continuously operating node takes readings from all sensors every 0.5 seconds. In addition to this it has a Raspberry pi running continuously, running a full Linux operating system and maintaining a permanent connection to the 3G network. To avoid this high consumption in the node with batteries the solution is to turn on and off the different elements of the board as they are needed. For this there are 2 duty cycles.

First of all, the sensor board has a 5-minute cycle in low power mode where it does not perform any task. In those five minutes the sensor board will activate and take a series of sensor readings and store them in a non-volatile memory. The other cycle is every 6 hours, the raspberry will turn on, connect to the network and perform the transfer of the data collected by the sensor that are in the non-volatile memory and send them to a server.

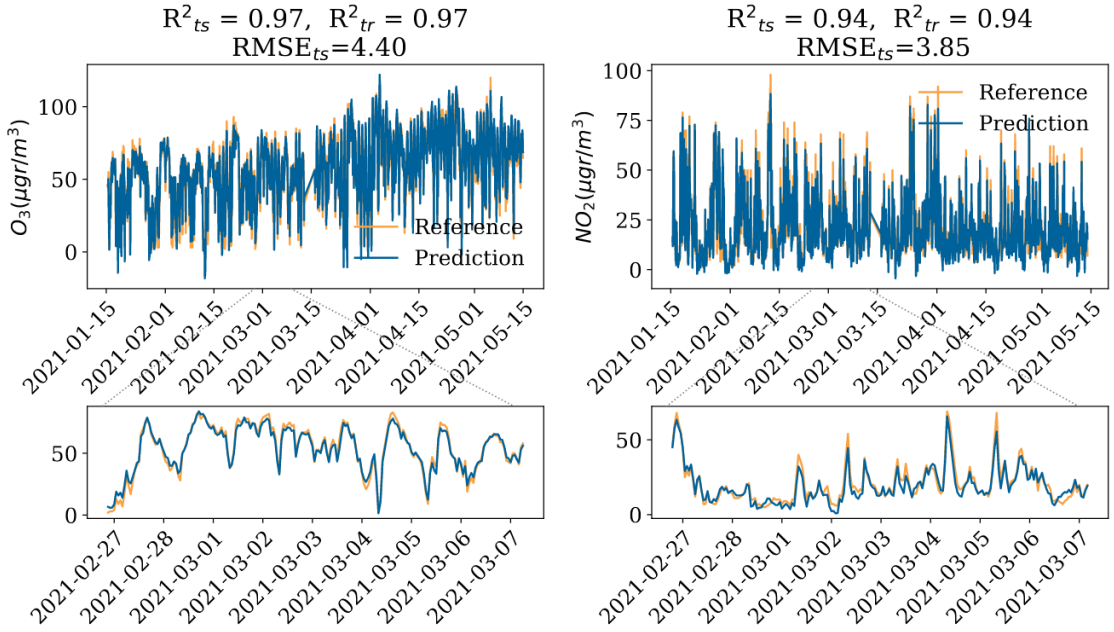
In the battery model we have an element that is not found in the other models, the non-volatile memory. Along with the duty cycles this is the second and only difference between the nodes, this memory exists to make up for a capacity problem, the arduino does not have enough space to accumulate the readings of 6 hours to offer them to the raspberry.

The solution has been to add an intermediate storage system with capacity for the readings of the last 48 hours.

2.2.2. The Node Data

Next we will observe the quality and fidelity of the pollution readings obtained by the node, in this fist based on the nodes running on grid power.

In Figure 2.3 we can see the results of the calibration of both NO2 and O3. In this case the part of the contamination data processing and calibration has been carried out by a PhD student. In the figure R^2_{ts} and R^2_{tr} denote the training and testing coefficient of determination, while $RMSE_{ts}$ denotes the root-mean-squared error for the testing using Machine Learning Regression.



(a) 20001 O_3 calibration.

(b) 20001 NO_2 calibration.

Figure 2.3: Calibration results for the O3, NO2. Bottom plots are a zoom of the plots above for a specific time interval

We can observe that the results obtained by the node are good and that the values obtained are very similar to the real ones. We can confirm that the node works and is able to reliably assess contamination.

In the case of the node with continuous power supply we have a very high number of readings, with 0.5 readings per second we have 7200 readings per hour. A value much higher than that of the battery. In this case we have a total of 8 readings every 5 minutes which leaves us with a total of only 96 readings per hour. The problem that may arise in this case is, considering that the contamination values do not vary very quickly, is to determine if this is enough readings or we will lose accuracy. In the case of the battery model the pipeline of readings works as follows.

1. Readers are sampled periodically over time. This is done to reduce battery consumption. The Arduino will be in low power mode, it will activate, take readings and

store them in non-volatile memory.

2. Once every 6 hours data will be sent through the raspberry, for this to happen there are a series of specific steps.

   (a) The Arduino will turn on the Raspberry, at the same time that this happens it will transfer the role of Master on the I2C bus to the raspberry.

   (b) The raspberry will start its operating system with the tasks required for the shipment. It will identify the provider and connect to the mobile network. It will synchronize the Real Time Clock, which is used by the Arduino to indicate the time of the readings and check the internet connection.

   (c) The Raspberry will then read the readings of the last 6 hours from the non-volatile memory, store them in its SD memory as a backup and send the readings via the mobile network to a central server. It will also evaluate the battery level to check if it is necessary to charge the batteries of the node.

   (d) The raspberry will notify the Arduino that its task has been completed and upon receiving an acknowledge from the Arduino will initiate the shutdown of the operating system. After a safety period the Arduino will cut power to the Raspberry and resume the Master role on the I2C bus.

3. The data available in the network is filtered to eliminate outliers and aggregated. Then the data is estimated using Machine Learning models.

To check if this new pipeline presents problems, the complete readings have been taken and filtered to obtain only the values read according to different time intervals. We will check the behavior in the case of different time periods and the number of readings taken.

Figure 2.4 shows the results of this test where sampling period T_{sen} and number of consecutive samples N_s taken every period T_{sen}. The solid lines denote the strategy that the N_s samples are taken consecutively after a sensor response period, and the dotted lines denote the strategy that the N_s samples are taken uniformly at T_{sen}.

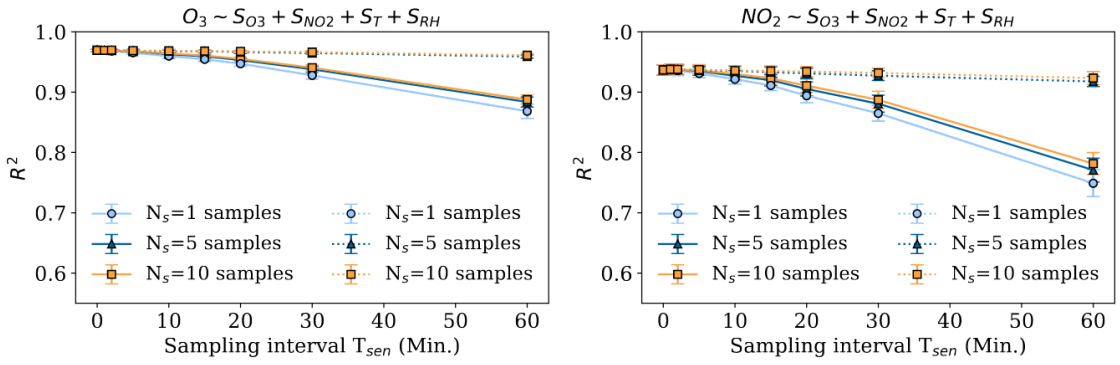


Figure 2.4: Average R^2 and 95% confidence intervals for different sampling settings using Machine Learning Regression

In this case we can observe that the readings are reliable in a considerable number of scenarios. First, if we spread the total readings over the sampling period, the result is

the same as if we had all the readings. Secondly, if we take all the readings at the same instant, we can go up to about 20 minutes without any difficulties in calibration.

As in the case of the battery powered nodes the readings are every 5 minutes, this does not present any difference in the results with the nodes connected to the mains. Therefore the pollution readings are valid in both cases.

2.2.3. Adaptation to Bluetooth Low Energy

The next objective is to add Bluetooth Low Energy functionality to the node, this is for multiple purposes, the first, to provide the necessary functionality for the Contact tracing architecture that we have seen previously, the others to be able to adapt and modify the node architecture that we have so far to make it more flexible. In this second case the main objective is to separate the sensor boards from the raspberry.

The main problem with the nodes is the communication to the central server via the mobile network. From our experience of use we have found that it presents a large number of problems, mainly in the stability of the connection as well as in the costs of using the network. In this case there would be several adaptations to the current model, firstly the advertise node, which would be used for contact tracing and is the base for the other two implementations, secondly a wireless structure and finally a mobile node model.

The Advertise Node

This is the node with the least differences with respect to the previous implementation. In this case it would be a node, either by battery or with connection to the mains that would perform the same processes, but at the same time that Arduino receives the readings from the sensors, it offers them in open via Bluetooth Low Energy.

In this case the only thing is that the node offers the possibility of consulting in real time the readings it is collecting. But in the following models we will see how with this possibility the functionalities of the node are extended. And all of them would offer the possibility described in the preliminary architecture where we used the node as an added enhancement to contact tracing.

A Wireless Structure

This implementation aims to mitigate or eliminate the problems caused by, firstly the need to use the mobile network to send data, and secondly the consumption problems caused by having a raspberry pi in the node powered by batteries. A schematic of the new implementation is available in figure 2.5.

In this model the I2c bus used for the nodes to communicate with the Raspberry is replaced by Bluetooth Low energy communication. This offers several advantages, firstly in this implementation the Raspberry, which acts similarly to a gateway is continuously powered from the mains and has access to an Ethernet network for access to the internet and the central server. In the case of not having an Ethernet network, it would be possible to maintain the communication of the mobile network.

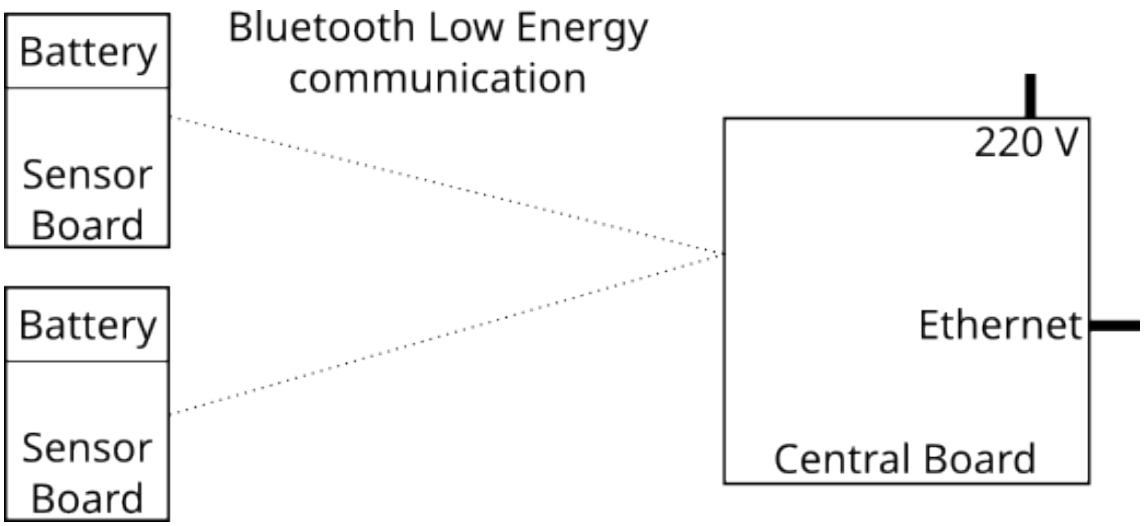


Figure 2.5: Schema for the Wireless structure

With this model, the battery life of the nodes is considerably extended. The biggest consumption that a node suffers is the consumption of the raspberry pi at the moment of starting the system and connecting to the 3G network. It also facilitates the deployment of sensors in larger areas, in this case with a single central board and several Bluetooth Low Energy nodes, could cover the surface of a park or a city block without difficulty where each of the sensors could be distributed over the surface and not all accumulated in a single point.

A Mobile Node

The latest implementation is a different approach and in this case the raspberry is replaced by a mobile device as shown in Figure 2.6. In this case, the aim is not to cover the contamination of a specific area, but of an entity that changes its position over time, for example a person throughout the day.

In this model the readout node is reduced to the smallest possible size while also reducing the size of the batteries. In this model the battery life is not required to last more than 48 hours. As they are designed to be worn by one person, they can be charged every 24 hours just like a mobile phone. With this reduction in size we obtain a node of dimensions that allow management and transport without many difficulties.

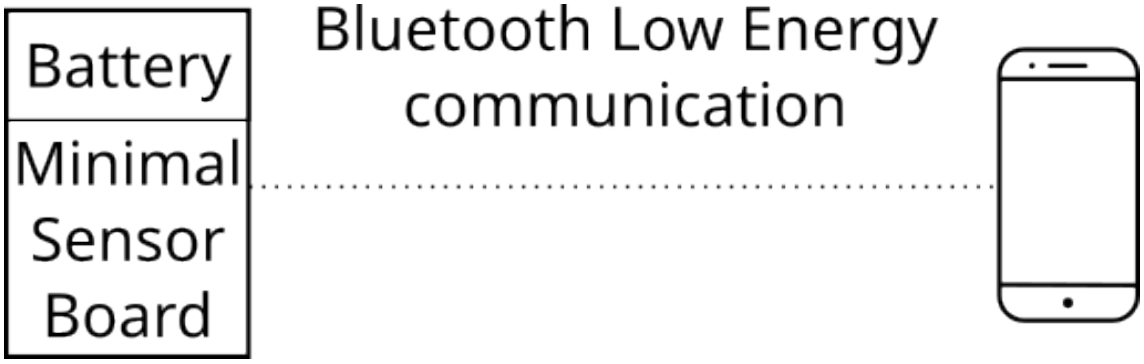


Figure 2.6: Schema for the Mobile Node

In this case the readings would be consulted in a lost way by the phone, which would add to the readings the moment in which they are consulted as well as the GPS coordinates of the signal. This would give the time and place of each reading. With this model we have the option of obtaining information on pollution levels, not of a specific area, but of the pollution to which a person is subjected throughout his day.

2.3. Node Adaptation

Next we will see the modifications to which the node undergoes to be able to offer the Bluetooth Low Energy functionality that would offer the changes that we have seen previously. In this case, based on the I2C bus that is present in the architecture, the solution would be to add Bluetooth Low Energy devices that offer I2C communication. For our implementation we will rely on two Arduino models already on the market, the Arduino Nano IoT[1] and the Arduino Uno Wifi[17].

In this case we use the Arduino Nano IoT because at the connection level it is the same as the Arduino Nano, which we currently use in sensor boards. In this case the migration process would be very simple, since the only necessary would be to add the Bluetooth Low Energy code and compile again for the new microcontroller. Without requiring any hardware modifications.

On the other hand we have the Arduino Uno Wifi, which is the one we will see in more detail. This is because it is the device that we have used both for the experimental implementation of Bluetooth as well as the device used to evaluate the performance of the Bluetooth Low Energy signals.

At the Bluetooth Low Energy level there is no difference between the two boards since they both use the same chipset and the same library for Bluetooth communication, so the implementation would be the same on both boards and the results would be very similar.

2.3.1. Bluetooth Low Energy Board

For the experimental implementation and for the performance test of Bluetooth Low Energy we will use the Arduino Uno Wifi. We have available the NINA-W102[12] chipset that allows communication via Wifi or Bluetooth Low Energy. In this structure we will use the Arduino to communicate with the chipset and be able to implement small programs that use the capabilities of Bluetooth Low Energy. For this reason we have chosen to completely disable the WiFi functionality of the chip, to avoid interference between the two features.

In our case the characteristics of the Arduino are not very important to us since the use of the Arduino is to facilitate communication with the Bluetooth chipset and the main load of the program in sensor boards is to read ADC values. Equally we will check the characteristics of the microcontroller to know its limitations and capabilities. In this case the microcontroller at our disposal is an ATmega4809[2] designed by Microchip, formerly Atmel. It is a 8-bit controller, with a set of digital and analog inputs and outputs and offers I2C, USART, Two-Wire and SPI. If we check the memory capacity of the microcontroller, available in table 2.1, we can see that although it may be limited for some features that you want to implement using Bluetooth for our case we have enough space. An important

detail that we can see by looking at the schematics of the board is that the I2C bus is used to communicate with the Bluetooth chipset.

This is an element to take into account, our previous architecture was very dependent on I2C. But if we review in more detail the schematics of both the Arduino Uno and the Arduino Nano, we can see that not all of them use the I2C protocol to communicate with the Nina chip. In the case of the Nano, the SPI protocol is used to communicate directly with the chipset, so it would not present problems in the case of sensor boards. On the other hand in the case of implementation on raspberry pi mainboards, there are several options. Use the Arduino Nano board in contrast to the Arduino Uno, or even any other Bluetooth Low Energy module. But the less complex solution to implement is the one we have used in our implementation, take this into consideration and share the I2C bus with the bluetooth chipset. As in the Raspberry implementation, the Arduino will be the I2C master, it will receive the information and send it via I2C to the Raspberry. So it is not a problem since the Arduino Uno will always be the master in the I2C protocol. Or even in a more restricted cases, we can even use another protocol for communicate the Arduino with the Raspberry like UART or SPI.

Flash Memory	48 KB
SRAM	6,144 Bytes
EEPROM	256 Bytes

Table 2.1: Atmega4809 available memory

For the implementation and the tests we will first update the firmware of the Nina chipset following the instructions given by Arduino. This is necessary first, because the BLE functionalities are not available in version 1.0 of the firmware and then to correct bugs or defects that were found in previous versions of the firmware. To collect the BLE information we will implement a simple program, using the library offered by Arduino, send its own beacons with a dummy value and collect beacons from other boards and send by USART the beacon received with the signal strength of that reception.

Now we will check the Bluetooth chipset characteristics. First of all we can see that we have an integrated isotropic PIFA antenna, like the ones used in mobile devices. The following are some of the other features of the antenna. First we see that it offers full support for Bluetooth 4.2, this version includes BLE and also the most widespread currently in most Bluetooth devices. We can also see that the antenna does not measure more than 1 centimeter in length and also only receives with a power of 0.3 W and transmits with 0.4 W. With these characteristics we will have a range more reduced than in the case of other devices, but sufficient to obtain valid results.

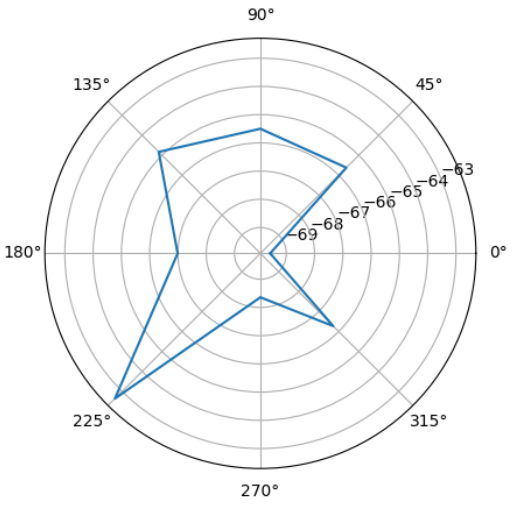


Figure 2.7: 360-degree antenna behavior

Then we will check how the antenna behaves in different positions. For this test we will

place the two devices at a distance of 100 centimeters, without interferences, with both antennas in direct line of sight and calculate the received signal strength. Repeat these readings by rotating the emitter 45 degrees each time until 360 degrees are obtained. We will place the results obtained in a graph 2.7 to be able to observe the differences.

As we can see from the results, the antenna behavior is different depending on how the transmitter and receiver are aligned. Except for the 225 degree case, in all other cases the antenna performance is quite stable with some slight variations. We can also observe that the worst signal reception occurs when both antennas are both in the same position. Even with all this we are going to assume for our experiments that we have an omnidirectional antenna following the manufacturers specifications.

2.3.2. New Node

With all this we already have the new node architecture adapted for Bluetooth Low Energy, both in the case of nodes connected to the current, such as autonomous with battery. Even for cases where the entire integrated node will use Bluetooth Low Energy to also indicate the readings in real time, as for architectures where the I2C bus is eliminated.

In the end, in this case, the functionalities added to the node do not go beyond announcing the values read by Bluetooth Low Energy, and to be able to query the advertised. This does not mean that we do not have more possibilities. In the current model, since we are more focused on using mainly GAP and not establishing connections, the functionalities are very limited. But in some models it could and would be recommended to establish a connection and use all the functions of Bluetooth Low Energy.

It is true that in the architecture proposed at the end of this chapter there is no connection between multiple devices, this is because a peripheral device can only be connected to a central device. In the case of a large number of Bluetooth Low Energy devices aiming at contact tracing, if a connection were necessary for detection, it would not be possible for all the devices to trace each other.

But in the case of, for example, the mobile node with a telephone, the beacon Bluetooth Low Energy and the contact tracing application and even the node model without I2C, it is possible to establish a connection without any problems. Furthermore, it would be advisable to establish a connection where possible. In this case, the information that could be exchanged would be more extensive. The central node could even transmit information back to the peripheral node. Moreover, in this case the communication is more robust, while normally the devices continuously scan for GAT packets, there is a lifespan where if a GAT packet is detected again it is discarded.

2.4. Phone App

Another element that we will require is a phone application that communicates directly via Bluetooth Low Energy, without using Google or iOS API implementations. This last element is more focused on the case of contact tracing or for the moving node model. In this case we will have a simple application that detects the Bluetooth GAP packets to detect the nodes, and can also connect to one of the nodes, in case the need arises.

Due to the difficulties and complexity of mobile application development, the application is only implemented for use on Android versions 12 or higher.

The application has several objectives, but the main ones are to scan the Bluetooth Low Energy devices that are in range of the phone and also to collect the tokens issued by the device. It also adds the ability to send information to the Bluetooth Low Energy device with which it is communicating.

In Figure 2.8 we can see in more detail the two main parts of the phone application. First we have the first screen that scans the Bluetooth Low Energy devices that are in range of the phone and displays them.

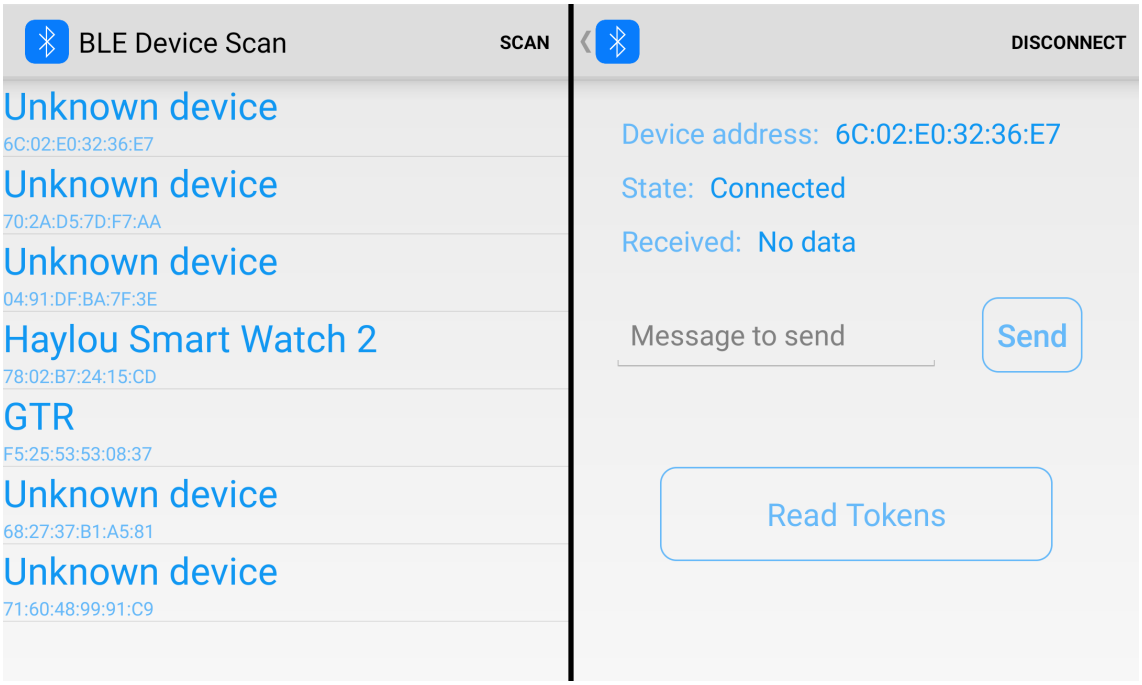


Figure 2.8: Display of the two main screens of the application

The second screen is the most important, once we have indicated the device is when we have the most important functionality in this case. To be able to read the token that is announcing at this moment the device. If it is a contact tracing device it would be the token that identifies it, if it is a node the contamination values that it has read.

We can see that there is also the possibility to send a message to the device. This functionality is available to make small tests to the contact tracing model where the phone application works together with a Bluetooth Low Energy beacon.

CHAPTER 3. DATA SET

Next we will evaluate how robust and good is the ability to apply Bluetooth Low Energy to digital contact tracing. In more detail how accurate our contact risk calculations can be using the intensity and quality of the Bluetooth Low Energy signals. The first step we will follow is to build our own data set of readings between two Bluetooth Low Energy devices by observing the signal strengths in multiple scenarios. With this data set we will be able to observe and study the behavior of the Bluetooth Low Energy to evaluate its capabilities as contact tracing.

3.1. Bluetooth Implementation

For the implementation that we will use to build the data set it will be a very minimal and simple version compared to the large amount of functionality that Bluetooth Low Energy offers. In our case, we will follow a similar operation to the current COVID-19 applications.

The code present in the transmitter and receiver will be the same in both cases and both have the same behavior. We can see a pseudocode of the implementation in the figure 3.1. In both cases we will use only GAP, with the beacons emitted to announce the existence of our device, we can obtain the signal strength. In order to do that we will create a GATT service with a special ID for our devices, so they can identify each other. This GATT service will include a characteristic, this characteristic will be like the key both devices exchange in the GAEN apps. We will add this service to the information we deliver in GAP, with this when we announce the device we also send the value of our key due to the GATT service announced by GAP. In our case as we only want to evaluate BLE we will not generate cryptographic keys and the token is a value that is incremented every second.

With this we already have our devices emitting beacons that would allow them to identify each other and collect the Rssi of the signal they receive. But for the operation of BLE there is a problem, the GAP scanning discards the GAPs already received for 30 seconds. So what we do is that when we receive a GAP message, we collect its information and then we discard all the GAP devices previously identified.

This code will allow our devices to continuously send and collect the GAP beacons they

```
1: Create GATT Characteristic C with UUID IdC
2: Create GATT Service S with UUID IdS
3: Attach Characteristic C to Service S
4: GAP Advertise S
5: while true do
6:     Scan for GAP
7:     if GAP UUID == IdS then
8:         Collect GAP info
9:         Send GAP info to USART
10:        Clean GAP collected info
11:    end if
12: end while
```

Figure 3.1: Pseudocode of our bluetooth implementation

have received and simultaneously send the data of this beacons by USB to a laptop for collection. With this we solve the problem of waiting time for pick up the GAP information again, which means more data in the same time and now we can collect a large number of GAP packets with an Rssi, which is what we need to evaluate the performance of Bluetooth Low Energy.

3.2. Scenarios

For the selection of the data set we will evaluate 5 different scenarios at 4 different distances. Readings will be taken at 1m, 1.5m, 2m, 2.5m. In the figure 3.2 we can see a schematic representation of the 5 scenarios, which are as follows:

1. Transmitter and receiver with direct vision
2. Transmitter and receiver aligned with receiver rotated 90 degrees horizontally
3. Transmitter and receiver aligned with receiver rotated 180 degrees horizontally
4. Emitter and receiver aligned with the receiver rotated 180 degrees horizontally and vertically
5. Emitter and receiver aligned, without rotation and with an equidistant obstacle.

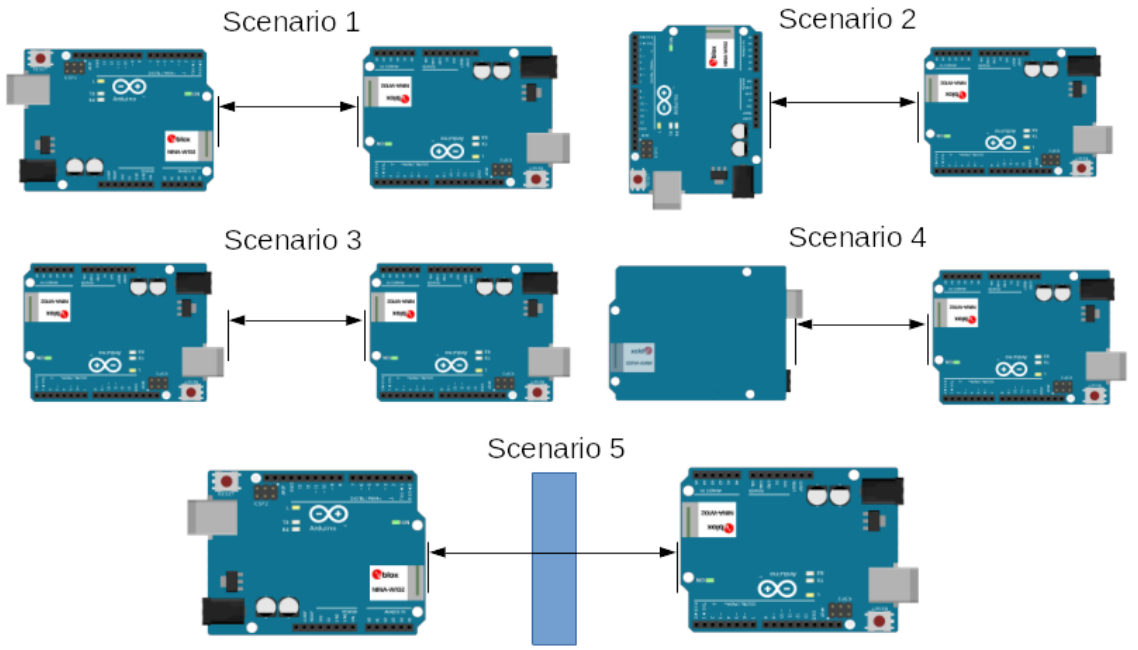


Figure 3.2: Visual of the different scenarios

All readings have been produced in as realistic an environment as possible, with more Bluetooth devices, WiFi networks and other elements using the 2.4GHz band in close proximity to the transmitter and receiver. For the readings, the transmitter and receiver were placed and held for 20 minutes, exchanging beacons with each other and recording the intensity received with each beacon.

For the readings both the transmitter and receiver are on a flat surface with no objects between them that can alter the transmission and reception as shown in figure 3.3. Once emitter and receiver are at the desired distance and position, the metric tape is removed to eliminate any further interfering factors and the samples are collected for 20 minutes.

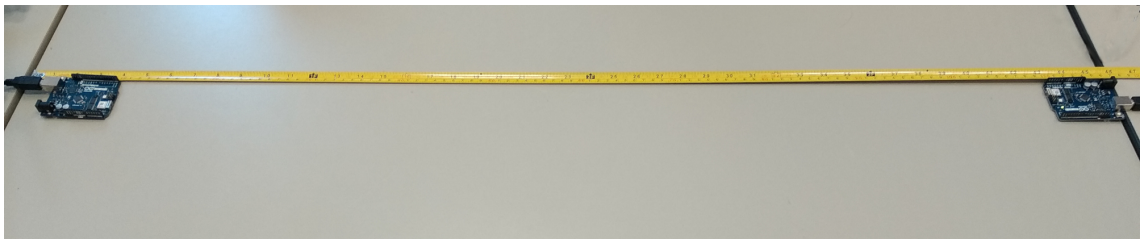


Figure 3.3: Image of the real environment for scenario 1 at 1m

We also have two extra set of readings, the first one is scenario 1 at a distance of 3 meters. In these particular set, as a result of the high occupancy of the 2.4Ghz band, the packet loss increases very considerably and no further readings were taken for the data set. This is because, compared to the other sets, the number of packets lost in the later scenarios was over 85% and made it very difficult to obtain valid results. We will see the data loss with more detail in the following section. The second set is a set obtained from the emitter. In this case the transmitter, during the time period of a test, records each time a packet has been sent. Later this set is used as a reference to be able to check how many packets have been lost during the readings.

With all this we end up getting a set of approximately 6000 beacons per scenario, with a total of 5 scenarios and 4 different distances, we get a total of 120000 beacons in different situations.

3.3. Results

As a preliminary step we are going to look at the results we have obtained, without processing them or treating them in any way. We will look at the number of packets lost, the differences between different distances and some specific scenarios to see how the Rssi values behave[1]

3.3.1. Data Loss

We have a graph in the figure 3.4 where we can observe the packet loss, including the special case of 3m distance. First of all, we see that in the case of 3 meters the packet loss is very different from that found in the rest of the scenarios and distances. We also see that the percentage of lost packets remains between 30% and 50% in the other scenarios and distances. There are cases, as for example scenario 1 that 2.5 m obtains less losses than in the rest of the scenarios, these variations can be the result of small variations in the amount of interference in the 2.4 Ghz band or the behavior of the electromagnetic waves.

[1] All the data is available in the following git repository(https://gitlab.com/nienna/ble-dataset)

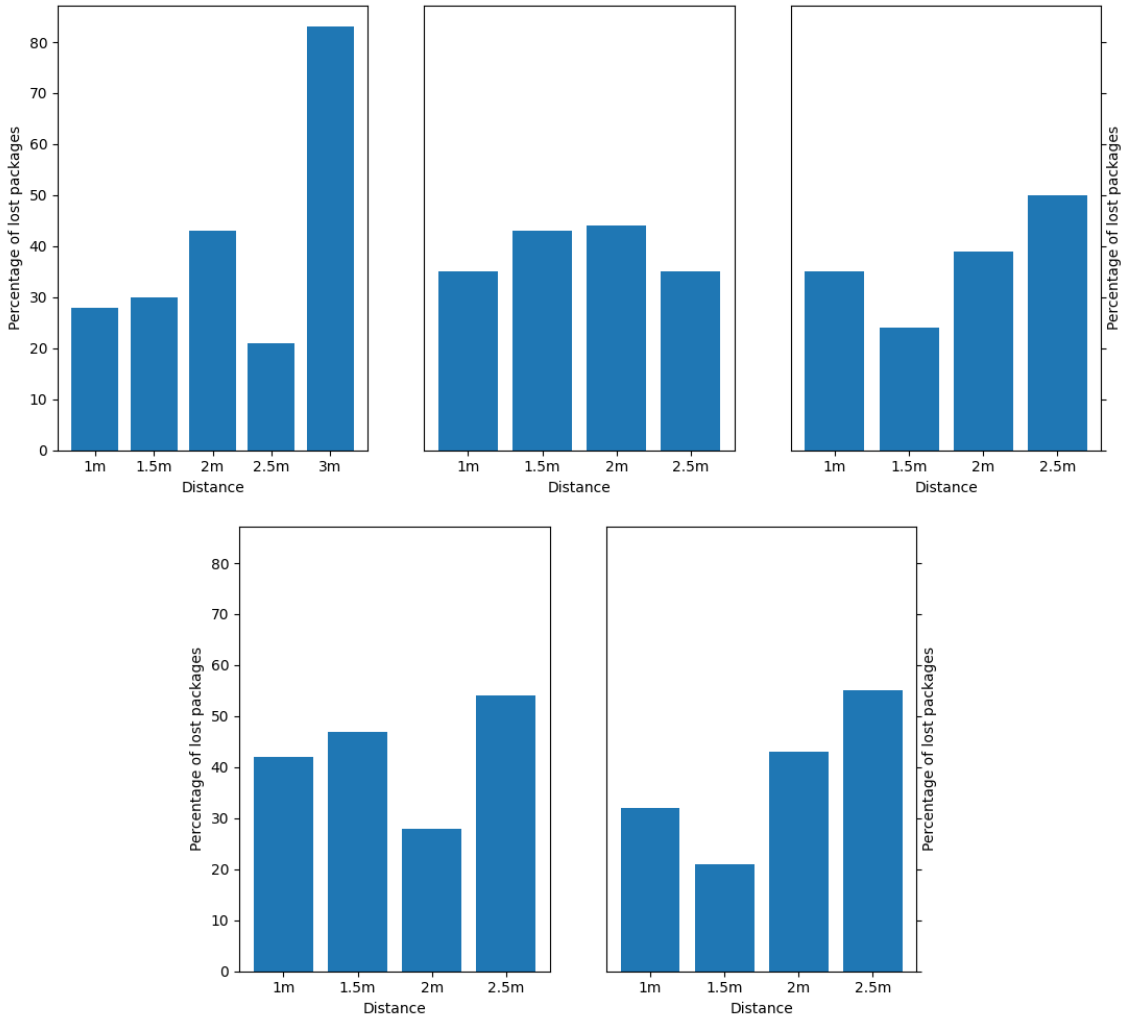


Figure 3.4: Percentage of packets lost in the different scenarios

To confirm the effect of the interferences we have made some readings in the same conditions as in the data set but in an environment with the least possible amount of interferences in the 2.4 Ghz band and we have checked the maximum and minimum rssi value. In figure 3.5 we can see the comparison of the received rssi, where the dark colors represent the scenario where the interference has been reduced to the maximum.

In this case we can clearly see that, despite being at the same distance and in the same conditions, the amount of interference in the 2.4 Ghz band affects very considerably the Rssi values. We can see how, in the cases without interference, we obtain much better signals than in the case where interference is present.

3.3.2. Distance comparison

Now let's take the data at different distances to see if we have differences between the multiple distances. In this case we will only look at a few graphs but all data collected is available in a git repository. In this case we will observe two cases, the behavior in scenario 1, in the figure 3.6, the behavior in scenario 4, in the figure 3.7, and the behavior in scenario 5, in the figure 3.8, at different distances.

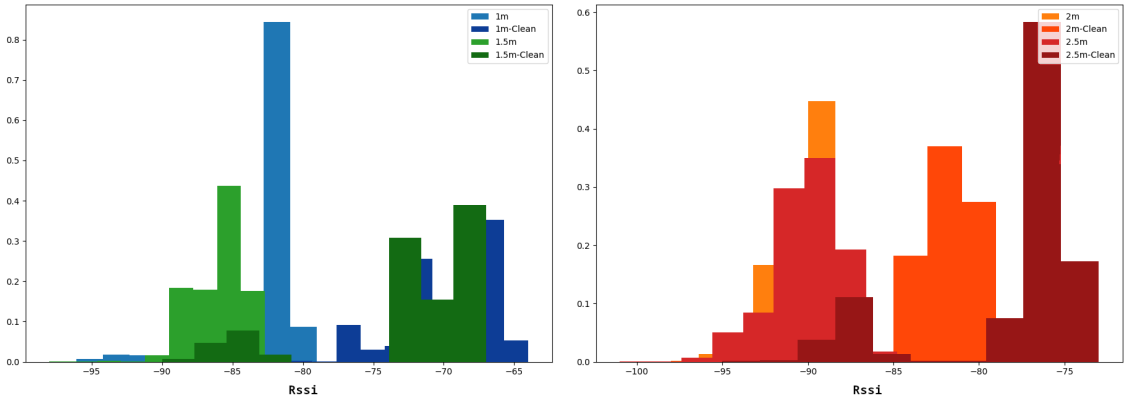


Figure 3.5: Comparison of the percentage of packets received based on rssi

Figure 3.6: Beacons collected for scenario 1 at different distances

We can see a relationship between the Rssi received and the distance between the transmitter and receiver. This confirms that the shorter the distance, the lower the Rssi and the better the reception quality. Another relationship in the Rssi is the position of the transmitter and receiver, as well as the presence of objects between them, affects the quality of the received signal. With all this we see that there is a relationship between the quality of the signal or the Rssi that allows us to indicate or perceive a distance or distance ranges.

3.3.3. Specific scenarios

To finish observing the readings received we will look at some specific readings where we can observe some behaviors to must be taken into account. A more detailed study of these characteristics behaviors of Bluetooth Low Energy and their corrections if necessary

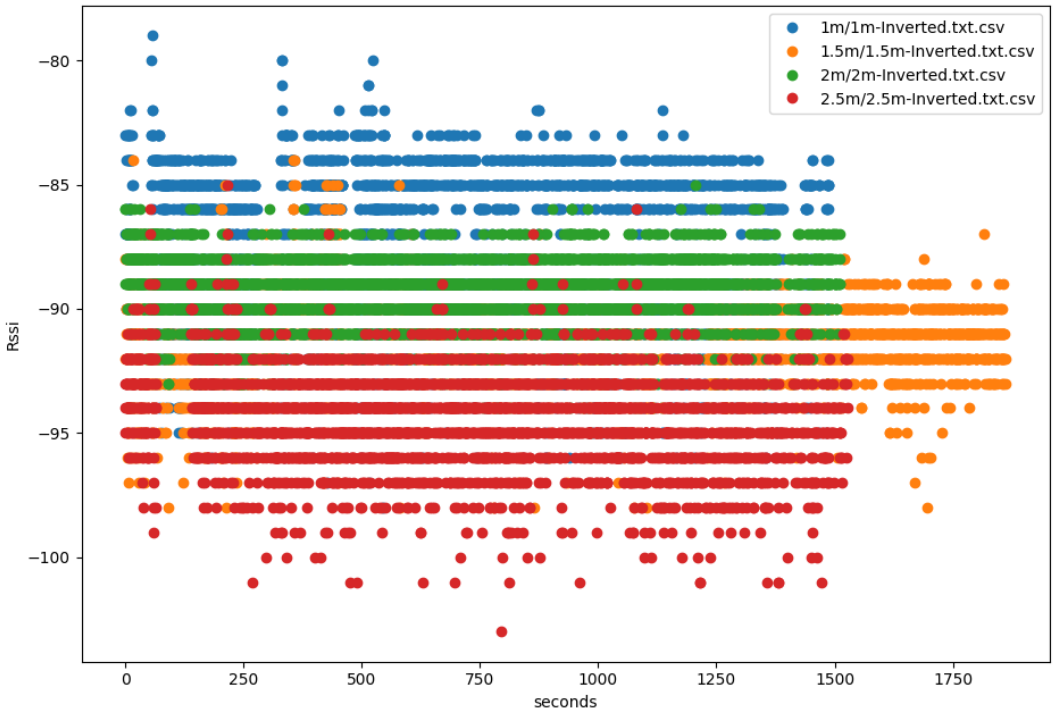


Figure 3.7: Beacons collected for scenario 4 at different distances

Figure 3.8: Beacons collected for scenario 5 at different distances

will be made in more detail in the section 3.5..

Let's start with one of the simplest scenarios, direct vision between the transmitter and the receiver (scenario 1) and at a distance of 1m, we can observe the readings in the figure 3.9. We can see that most of the readings are stable, within the range between -80 and -83 dBm, which would be expected in the case where the distance is smaller, the receiver and the transmitter are aligned. But we can also observe a cloud of outliers in the range greater than -80,-85 dbm.

We also can observe a behavior with obstacles, in this case scenario 5 at 2.5 meters available in figure 3.10. Here we can also observe that we have a cloud of outliers, but in this case the range is much less precise than in the previous case. Now we have a much larger range, produced by the difference in space and the interference produced by the obstacle between the transmitter and the receiver. But also, this set of outliers is not constant over time, what in one part of the readings we would consider outliers, in the other part is more diffuse and difficult to classify.

The last scenario we will look at is the special case of scenario 1 at a distance of 3m, as shown in the figure 3.11. First of all, we can observe that the number of packets is smaller compared to the other two cases and there is a very limited number of outliers. We can also distinguish how the beacons are distributed in blocks over time, we can distinguish one between 1400 and 1600 seconds and another between 200 and 400 seconds.

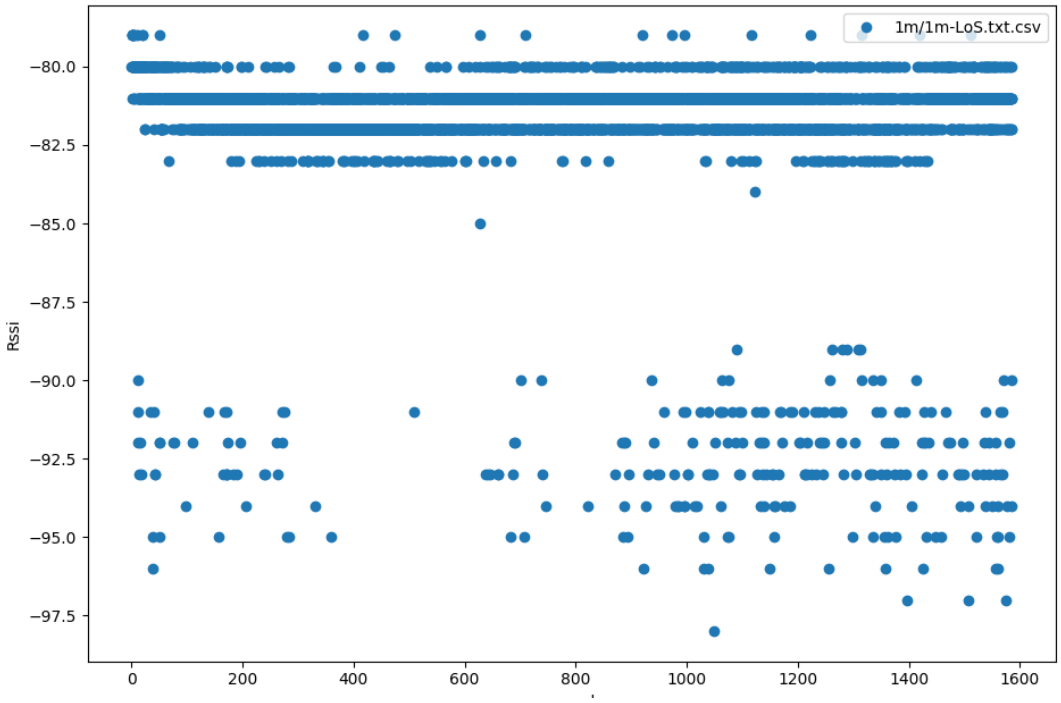


Figure 3.9: Beacons collected for scenario 1 at 1m

Figure 3.10: Beacons collected for scenario 5 at 2.5m

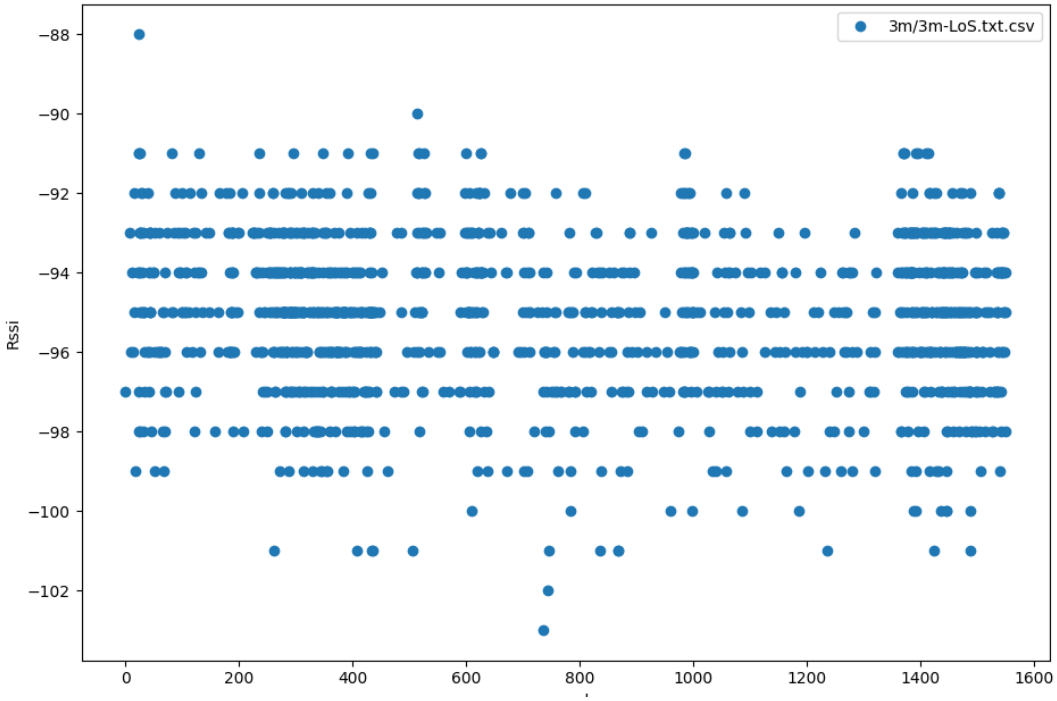


Figure 3.11: Beacons collected for scenario 1 at 3m

3.4. Testing Sets

The main idea of the data collected so far is to have a reference set to be able to calibrate and use as reference for the contact tracing mechanisms that we are going to use. In this case we are going to use sets with a shorter period of time, in this case they will be 5 minutes long and the scenarios will be the following:

- Scenario 1 at a distance of 1.25m
- Scenario 2 at a distance of 1.60m
- Scenario 4 at a distance of 2.10m
- Scenario 1 at a distance of 2.60m

The readings collected from these 4 scenarios can be found in Figure 3.12. We have chosen these scenarios first to check the calibrations with distances for which we do not have any data. Secondly because, as we can see, the data have very similar behaviors between them.

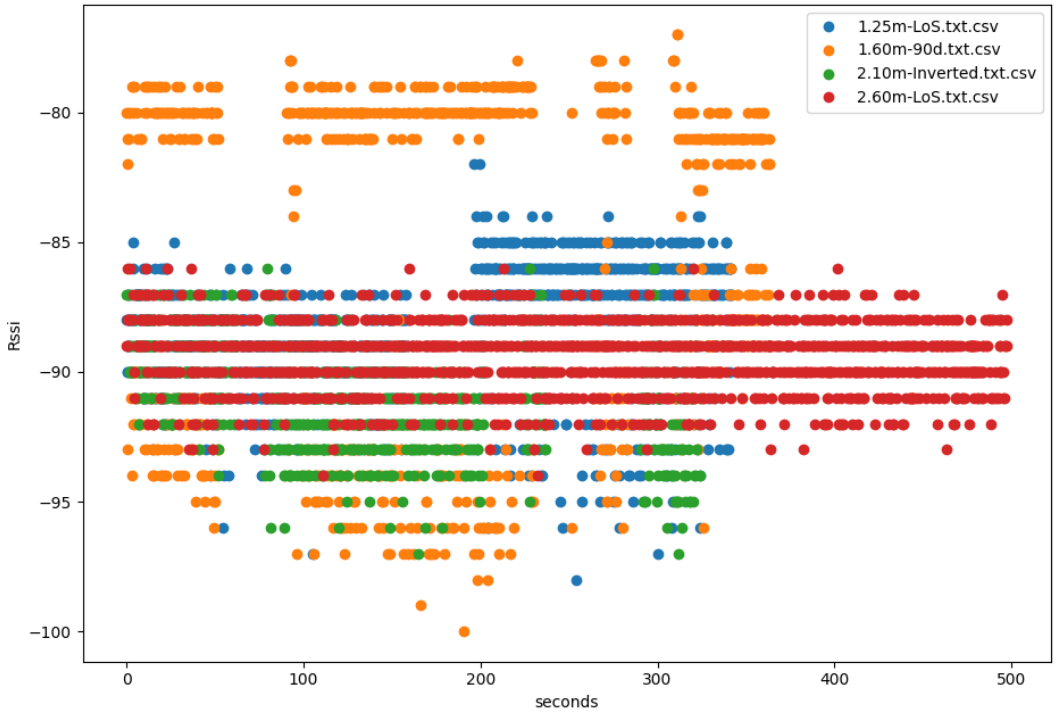


Figure 3.12: Beacons collected for the testing sets

First of all the 2.60m and 2.10m readings have a very similar signal quality between them, this is due to the difference of positions between them. On the other hand 1.60 due to the rotation between transmitter and receiver has outliers that could result in inferring that it is at a distance of less than one meter. Finally we have the 1.25 m readings, in this case they behave similarly to the 1 m readings in scenario 1, but in this case, due to the outliers and their distribution, it is much more difficult to differentiate them from the other three readings.

We will use these 4 scenarios to test the accuracy and reliability of contact tracing as these scenarios easily overlap with each other. This makes it very difficult to differentiate between them using the Rssi and allows us to test how good Bluetooth is at accurately determining contact tracing in complex and difficult scenarios.

3.5. BLE Preprocessing

Once we have the collected data set we can start working with it. But as we have observed in the previous sections, the collected data present some problems, outliers to be eliminated, a lot of variation, etc. Next we will look at these problems and see what ways we have to treat our data to be able to work with them more effectively.

3.5.1. Characteristics of BLE signals

As we have observed in the previous chapter when reviewing the results obtained from our dataset we can discern a set of characteristics in the behavior of Bluetooth Low Energy.

First of all, our data are not linear, the signal quality or Rssi is indicated in dBm, decibel-milliwatt, a unit of ratio where decibels are expressed in relation to one watt. To express an arbitrary power P(mW) as x(dBm) we may use the following formula.

$$x = 10\log_{10}\frac{P}{1(mW)}$$

Conversely, we can express a level of dbm, X as mw with the following expression

$$P = 1(mW)10^{\frac{x}{10}}$$

With this formula we can see the range in mW over which our signals are in dBm with -80 being the highest value and -102 the lowest value. In this case we are between 10pW and 0.1 pW of received signal strength. We can add that it is difficult to detect packets with a signal strength below 0.1 pW or -100 dBm as this is considered the minimum value at which a signal can be received from a wireless network.

The second characteristic is that Bluetooth Low Energy, by design and as we can see from the data received, in some cases more than others, works by bursts. In the case of Bluetooth Low Energy the main way to detect a Bluetooth Low Energy device is through the Inquiry function. The inquiry function searches for other Bluetooth Low Energy devices in range and collects information about these devices. This information includes the MAC, the clock and its Rssi among others. The problem is that Bluetooth Low Energy is intended to be Low Power, and unlike other wireless networks this communication is not periodic. In addition to that, there are other Bluetooth Low Energy behaviors, such as the need to synchronize the frequencies between transmitter and receiver since Bluetooth Low Energy uses pseudo random hopping frequency for transmission.

These two elements together are the main determinants of the signal being received by bursts. To alleviate this problem when generating our dataset, we have decided not to use Inquiry and to rely on existing contact tracing applications. The devices will transmit beacons as a continuous warning that the receiver will continuously pick up. This reduces

the problem present with inquiries but some cases still occur, even if they are more reduced where the reception of the signal is distributed in blocks.

In this case we must use a filter to pre-treat our signal, but due to the non-linearity of our signal the Kalman filter which is one of the most used to filter signals would not have the best possible behavior. With the characteristics of the signal we are receiving we will use a particle filter using Monte Carlo methods.

3.5.2. Montecarlo Particle Filter

The particle filter is a sequential montecarlo method, although it is normally used in virtual reality systems, we can adapt it to our signal. In this case we will rely on a similar adaptation of particle filter for bluetooth positioning that search the main objective as us, obtain an approximate distance from Rssi[14]. The formula we will follow for our particle filter, based on the previous implementation, is as follows:

$$x_i(t) = x_i(t-1) + n_x \Delta t$$

$$y_i(t) = y_i(t-1) + n_y \Delta t$$

$$n_x \sim \mathcal{N}(0, \sigma_{pos})$$

$$n_y \sim \mathcal{N}(0, \sigma_{pos})$$

Where $\mathcal{N}(\mu, \sigma_{pos})$ represents a Gaussian distribution with μ mean and σ typical deviation, and Δt is the time interval between iterations.

With the previous formula we will implement our own Monte Carlo filter in python to filter the signals we have collected in our dataset. As in this case we will work with non-motion scenarios we can implement a simple version of the filter. If we want to take into account displacements or other characteristics it would be necessary to add these considerations to the particle filter formula.

The last element we need to take into consideration is to determine the number of times we apply montecarlo methods in our filter. After doing some small tests the final value for the number of times montecarlo methods are applied in the filter is 150. This value has been set because in the first place with few iterations the results were slightly lower. In addition to this the running times of our filter are not very long, and the time to run the filter 150 times does not exceed 5 seconds. As in this case the time is not a limiting factor, it has been decided to run a high number of times to obtain a good filtering.

With all this determined, we will now filter all the signals we have collected in our data set and filter them. The comparison between the filtered and unfiltered signals of some cases can be seen in Figure 3.13, 3.14, 3.15 and 3.16. As we can see our filter works, the outliers are discarded and the minimum and maximum value of the Rssi is much more limited. It is true that the performance of the filter is better in some cases than in others, but we can consider that we have assumed our objective of filtering the signal, reducing outliers, etc.

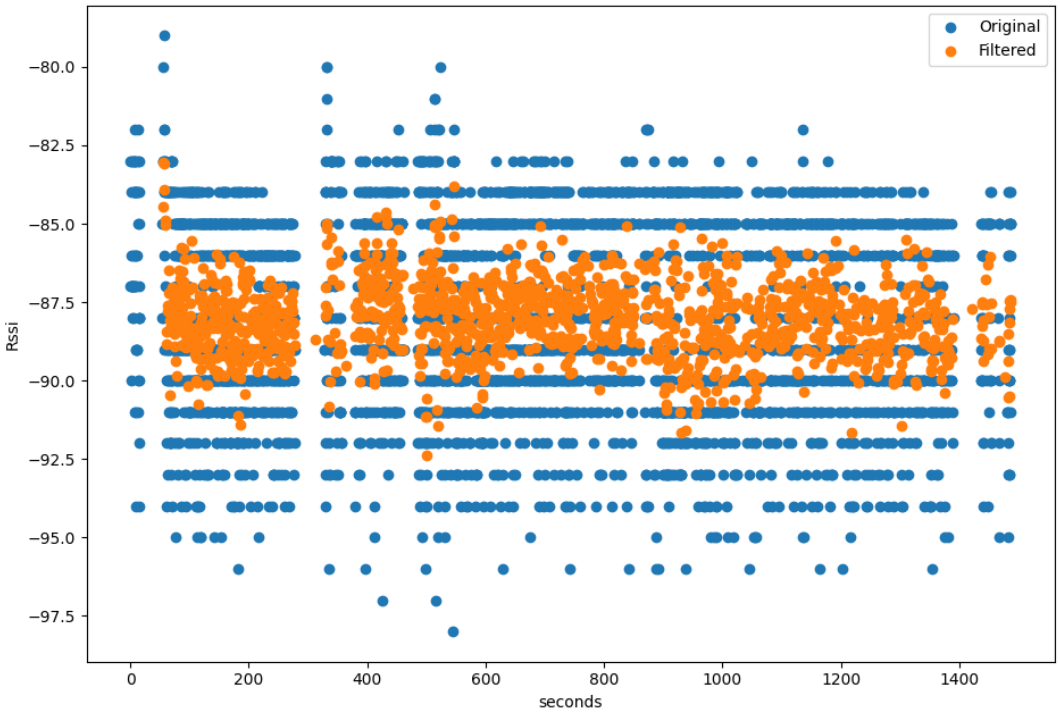


Figure 3.13: Comparison between filtered and unfiltered signal for scenario 4 at 1m

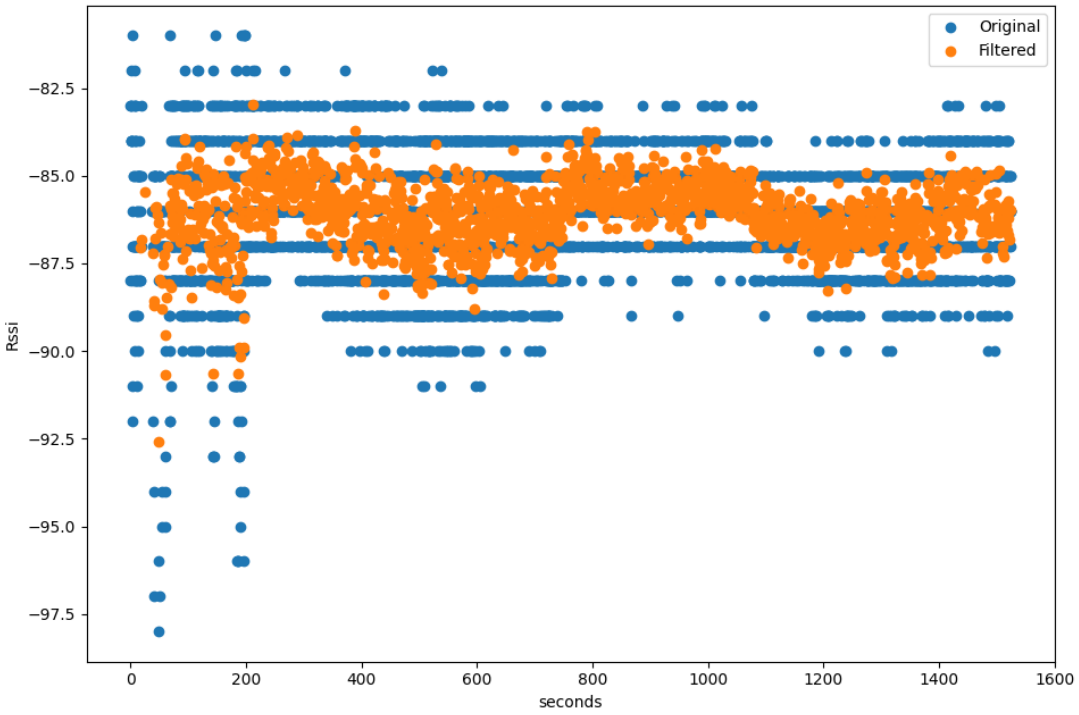


Figure 3.14: Comparison between filtered and unfiltered signal for scenario 1 at 1.5m

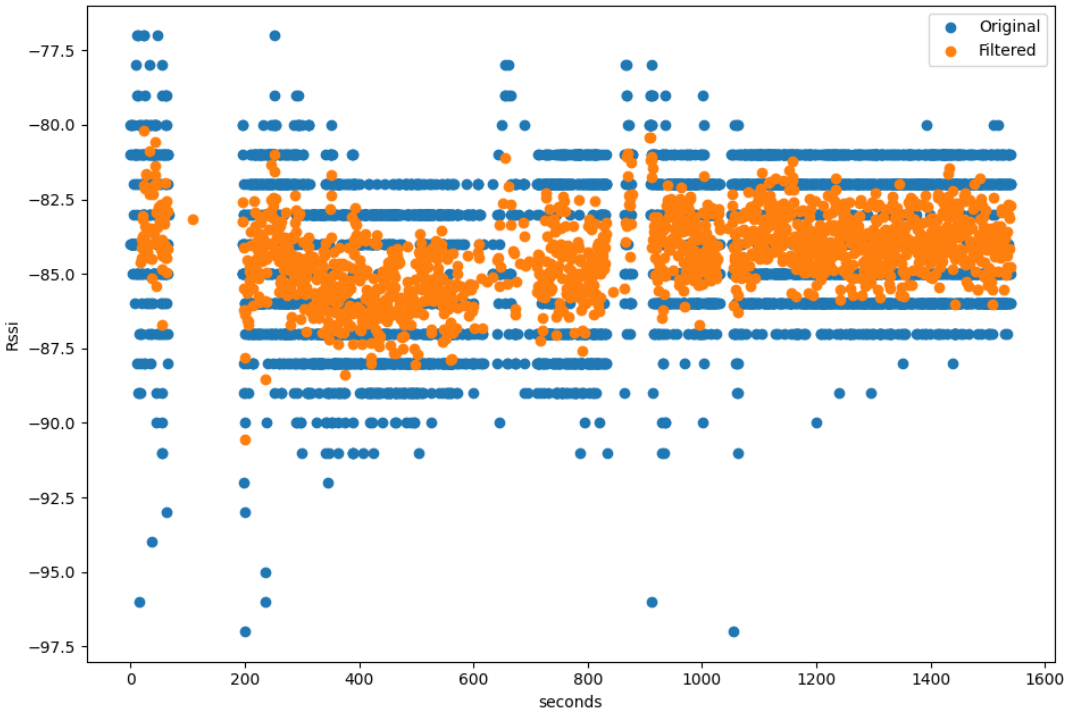


Figure 3.15: Comparison between filtered and unfiltered signal for scenario 3 at 2m

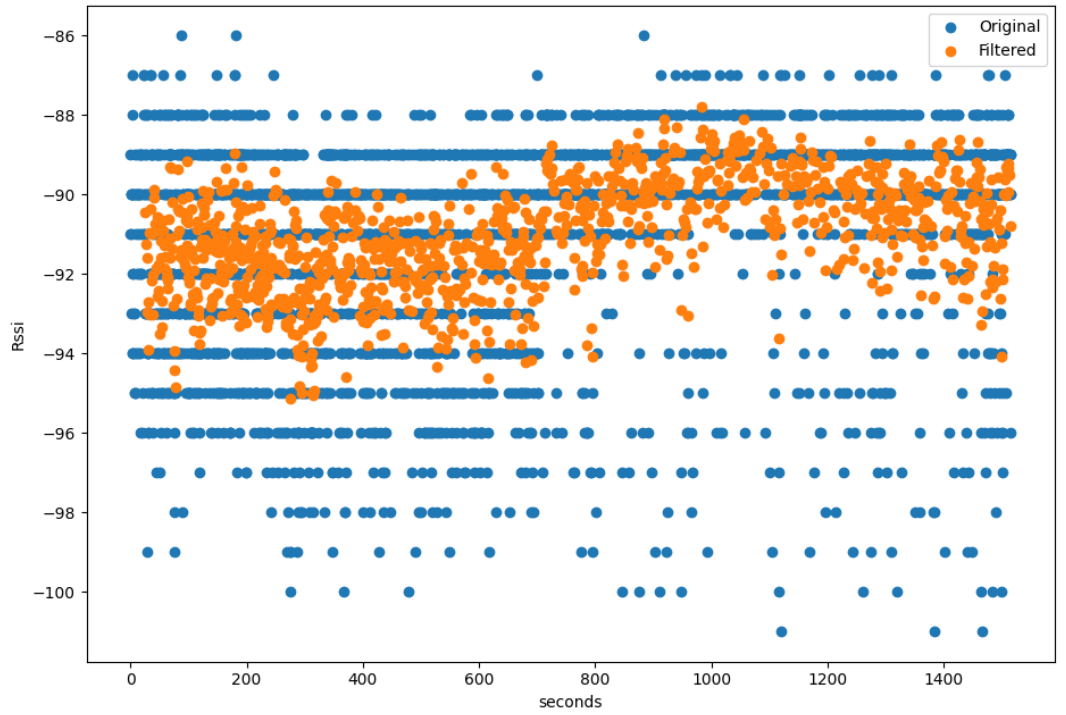


Figure 3.16: Comparison between filtered and unfiltered signal for scenario 5 at 2.5m

CHAPTER 4. DETECTING CONTACTS

Now that we have enough data about the signal between a Bluetooth Low Energy transmitter and receiver and we have cleaned and filtered the signals we will evaluate the first method of contact tracing. As we have seen previously, the contact is consider depending on the quality of the signal, and then is mainly determined by the distance between transmitter and receiver. With this in consideration, we can calculate the distance between transmitter and receiver, and determine a level of risk based on this.

4.1. From Rssi to Meters

From the data collected we know that there is a relationship between distance and the quality of the received signal. This behavior can be contrasted with those of other experimental results where a distance is obtained from a Bluetooth signal. In this case there is a known formula where Rssi is related to distance and can be found in several academic papers[14][10]. The formula relating both elements is the following:

$$P_L(d)[dB] = P_L(d_0) + 10 * n \log \frac{d}{d_0} + X_{\sigma_L}$$

Where $P_L(d)$is the received signal power in a distance d, $P_L(d_0)$ is the same power but in a reference distance d_0, n is the path loss exponent and X_{σ_L} represents the noise using a random variable. This formula is generic and needs to be adapted to our needs. In this case we want to know the distance based on a received signal and this formula indicates the intensity that we will receive at a specific distance. For this we isolate the value d since this is the value we do not know and want to obtain. The formula we will obtain and the steps we follow are:

$$P_L(d) - P_L(d_0) - X_{\sigma_L} = 10 * n \log \frac{d}{d_0}$$

$$\frac{_L(d) - P_L(d_0) - X_{\sigma_L}}{10 * n} = \log \frac{d}{d_0}$$

$$10^{\frac{_L(d) - P_L(d_0) - X_{\sigma_L}}{10 * n}} = \frac{d}{d_0}$$

$$d_0 * 10^{\frac{P_L(d) - P_L(d_0) - X_{\sigma_L}}{10 * n}} = d$$

Now we can obtain the distance based on the received Rssi as long as we have an Rssi and a reference distance. We will apply a few more small changes to the formula, in this case our signal is already filtered, so the noise present in it is considerably reduced and we are not searching a precise distance. With this in consideration we can eliminate X_{σ_L} to simplify the mathematical calculations, since is to take into account the noise that is produced in the communication. The final formula we will use to calculate the distance according to the signal quality is described below.

$$d = d_0 * 10^{\frac{P_L(d) - P_L(d_0)}{10 * n}}$$

The next step is to obtain the reference values of d_0 and $P_L(d_0)$. In this case we are going to use four different values for d_0 and $P_L(d_0)$ and check which one gets the best result and in which cases.

In this case we will have 4 sets of values to check how well the distance calculation works. We will use the data from the five scenarios for each distance for this purpose. We will take all the Rssi values that we have collected once filtered and calculate the mean value, this will be the value that we will use as $P_L(d_0)$. In the table 4.1 we can observe the four pair of values we are going to use to calculate distances.

$P_L(d_0)$	d_0
-84.698533 dBm	100 cm
-87.484866 dBm	150 cm
-89.971363 dBm	200 cm
-90.147241 dBm	250 cm

Table 4.1: The four pair of values to calculate distance

As we can see, the Rssi are reduced the greater the distance, but as we increase the distance the separation between them is reduced. The difference in Rssi between 2 meters and 2.5 meters is much smaller than between 1 meter and 1.5 meters.

4.2. Results

Next we will check how well the distance calculation is obtained with the 4 values to calibrate. As a start let's evaluate how well the distance is calculated against parts of the sets we have used to obtain the pairs of values for the calculation. In this case we will use a different set of each of the 4 distances. Scenario 1 for 1m, scenario 4 for 1.5m, scenario 2 for 2m and scenario 5 for 2.5m have been used.

In the figures 4.1, 4.2, 4.3 and 4.4 we can see the results of the conversion. Each point represents the distance to which the received beacon has been converted, the red line is the average distance given by the conversion of all the readings and the green one is the real distance to which the readings have been taken.

In this case we can observe several things, first of all, that in most scenarios the distance calculation works relatively well, scenario 4 has a very different behavior that we will evaluate below. Especially if we decide to use the values of 2.5m, although the 1.5 and 2 also get pretty good results, and the worst performer is the 1m. Taking into account the worst pair of values the margin of error we have is 50cm. Approximately this 50 cm is the space occupied by a seated person.

Another detail to keep in mind is that depending on the distance and the scenario some value pairs work better than others. In the case of 2.5m, it gives very good results with readings at 1m, but falls behind compared to 2m when used to calculate its own readings.

We can see from the previous tests, that in scenario 4 the distance calculation fails completely, with cases where the calculation fails by up to more than 100cm. So let's check this scenario in more detail. For this we will visualize the results of calculating the distance for the other 3 cases, 1m, 2m and 2.5m. In figures 4.5, 4.6 and 4.7 we can see the results of calculating the distances as in the previous figures.

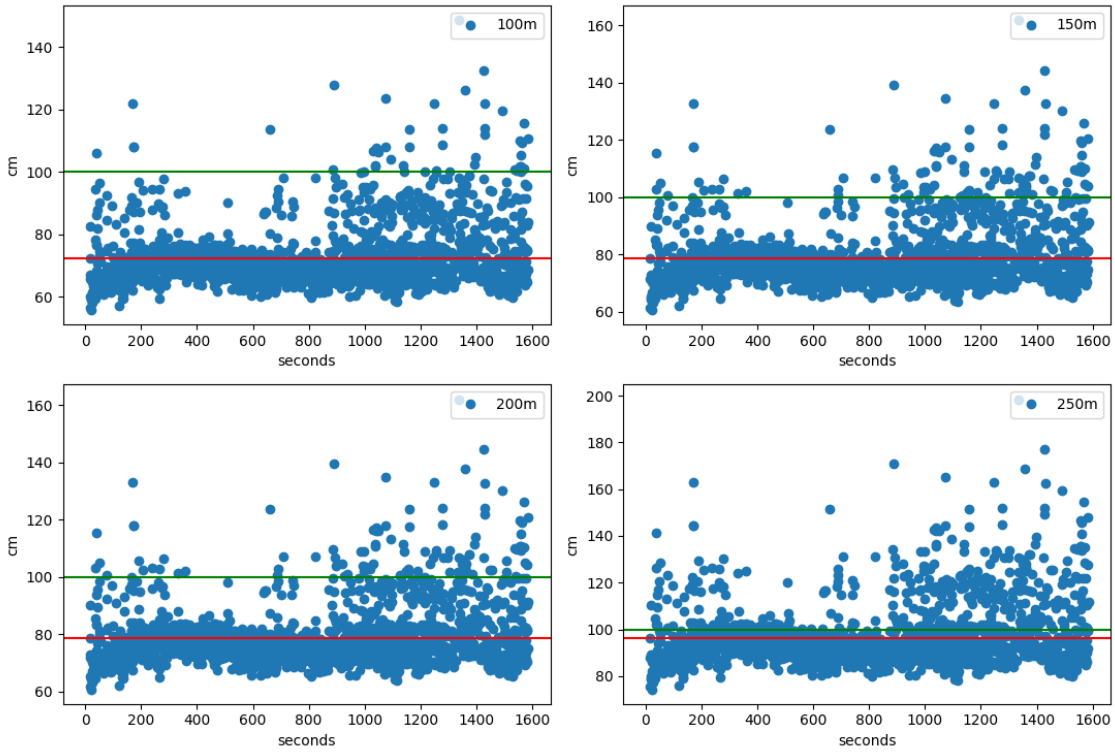


Figure 4.1: Results of calculating the distances with the four pairs for 1m in scenario 1

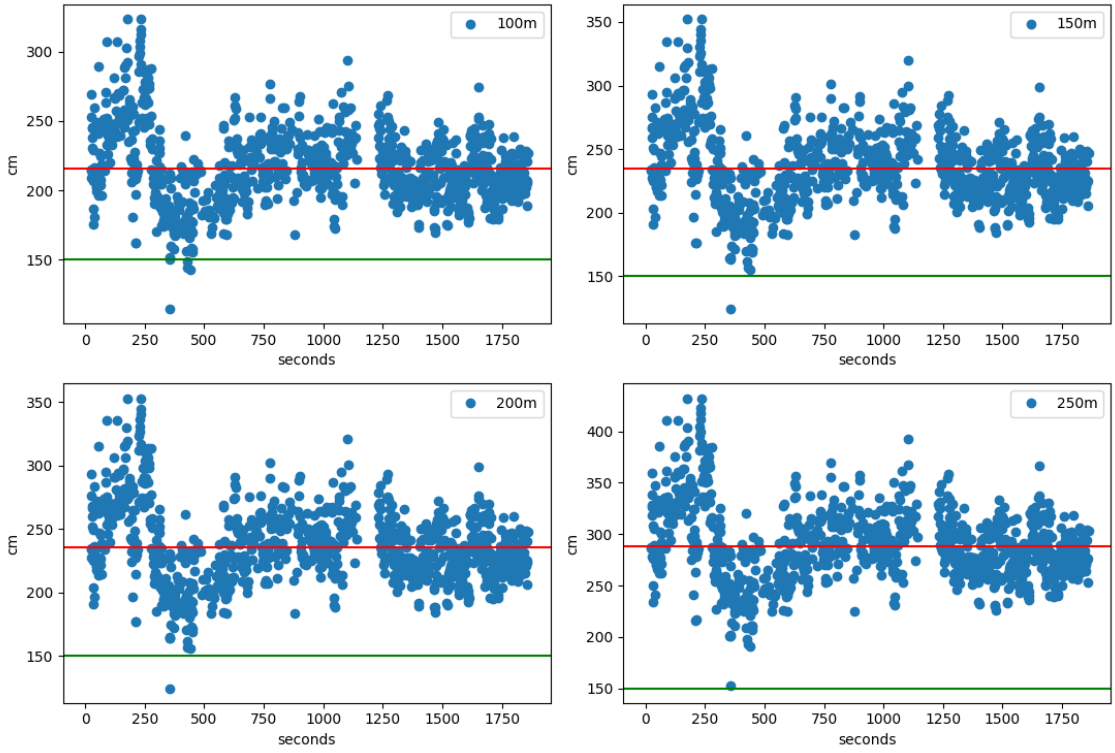


Figure 4.2: Results of calculating the distances with the four pairs for 1.5m in scenario 4

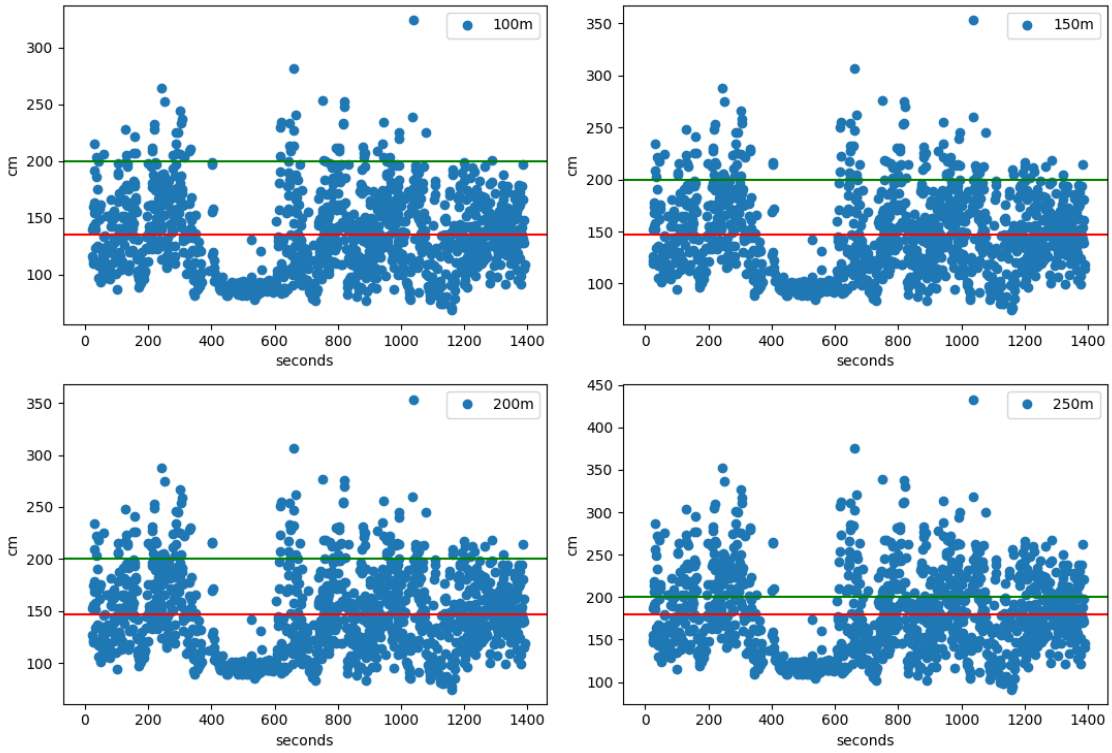


Figure 4.3: Results of calculating the distances with the four pairs for 2m in scenario 2

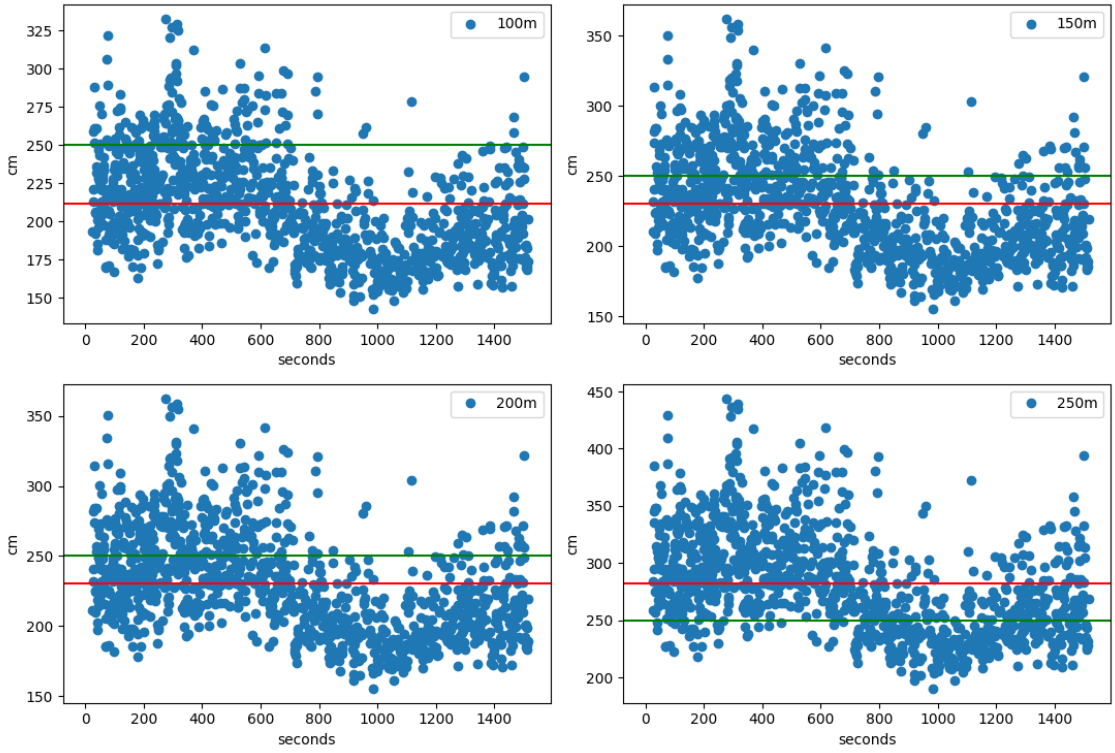


Figure 4.4: Results of calculating the distances with the four pairs for 2.5m in scenario 5

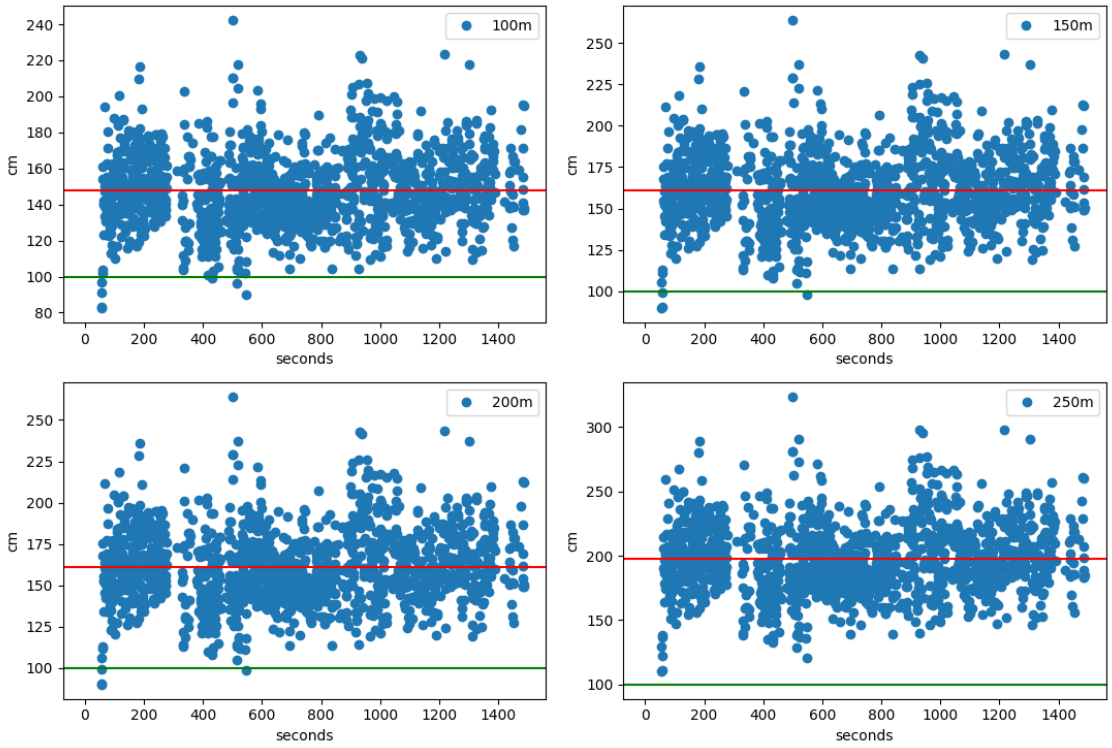


Figure 4.5: Results of calculating the distances with the four pairs for 1m in scenario 4

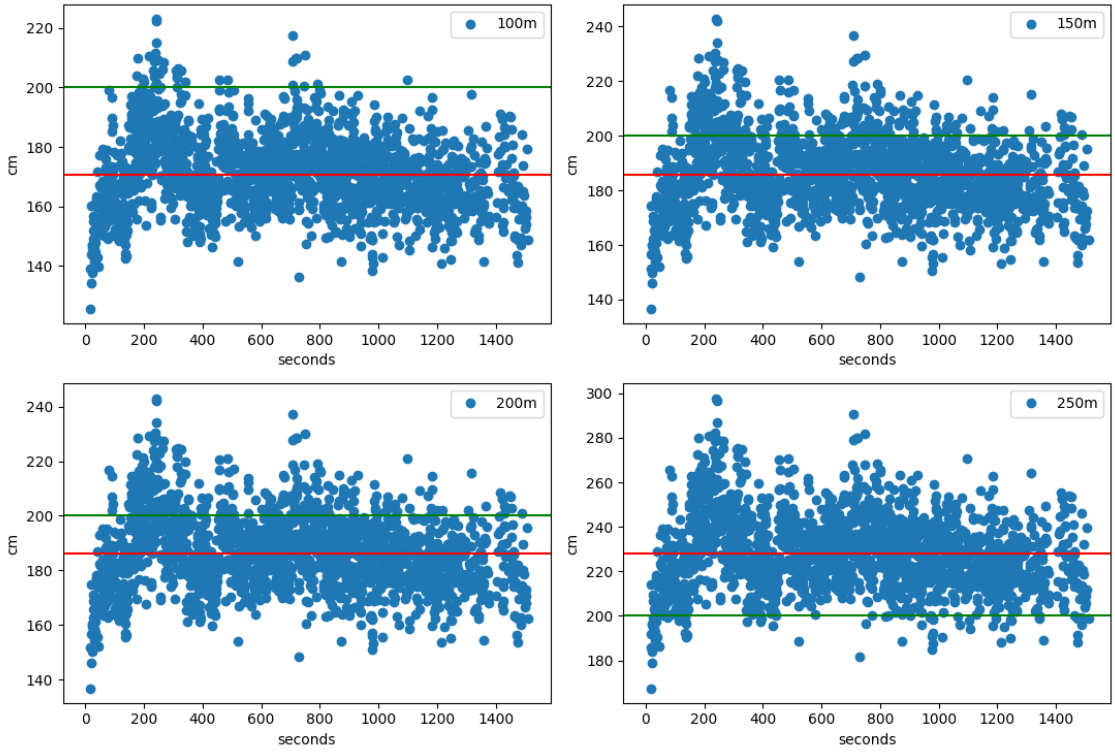


Figure 4.6: Results of calculating the distances with the four pairs for 2m in scenario 4

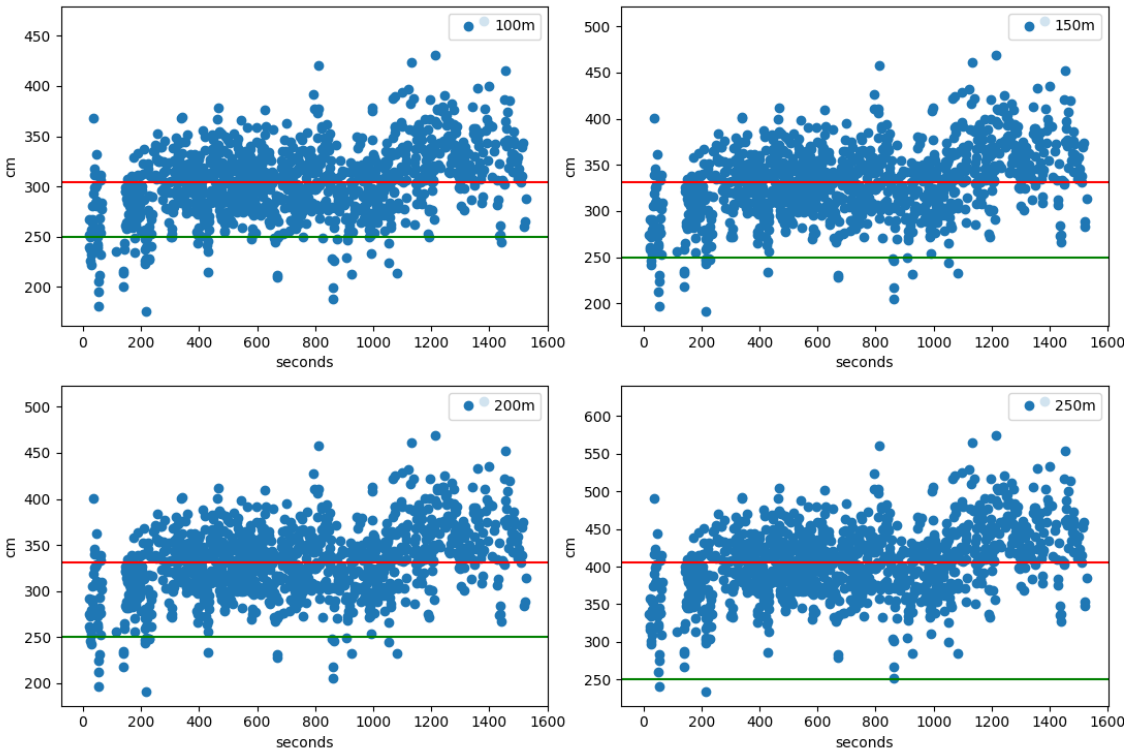


Figure 4.7: Results of calculating the distances with the four pairs for 2.5m in scenario 4

We can observe without a doubt that in this case scenario 4 is much more difficult to obtain a correct distance, in this case produced by the difference of positions between the transmitter and the receiver. It is difficult to go below a margin of error of less than 40 cm, it is true that we can see it in some scenarios with a specific pair of values, but we do not see values as close as in the other scenarios.

All and the difficulty we can also see that the results so erroneous that we have seen, are an isolated case of 1.5m. In these cases we obtain margins of error, if it is true that high in some cases, not so extreme. Another detail is that once the distance increases we see that the results of the distance are better, there is less error, this ranges between 70 and 50cm if we use the pair of values that gives better results.

In this scenario we can also see another behavior to take into consideration. In this case the calibration with the 1m distance is the one that obtains the best results in some cases. The one that in the previous scenario was always the worst performance, in this case it is the one that gets the best results.

With all this we can determine that scenario 4 is a scenario that has a special behavior to the others, that is more difficult to evaluate, but that does not prevent us from obtaining a valid distance and that would allow us to obtain a valid traceability.

As a final evaluation of the reliability and robustness of our mechanism for calculating the distance with Bluetooth Low Energy we will test it against the test sets. We can see the results obtained in the following figures 4.8, 4.9, 4.10 and 4.11.

In this evaluation we can see two great distinctions. The two sets under scenario 1 are much more difficult to calibrate compared to the other two readings where we have obstacles or where they are not in line of sight. In the second and third readings, one of the pairs of values results in the real value with very little difference, less than 10 cm. And with the exception of a few specific cases, the rest of the calibrations are also quite accurate. With a margin of error similar or less than those obtained in the first tests.

We will now evaluate the results we have obtained in the two cases where the transmitter and receiver are aligned and with direct vision. In this case we can see that the conversion results are less accurate.

In the case of 1.25m, none of the pairs of values stands out for obtaining accurate values. It is true that the error margins of calibrating with one meter are considerably smaller than those of calibrating with the other three pairs of values. In this case, we again have extreme values of up to 1m margin of error if the 2.5m value grid is used. However, the values when calculating with the best value we have, we obtain margins smaller than 50cm.

Finally, the values at 2.6m have also extreme behaviors when calculating the distance values. In this case we have a similar behavior to the 1.25m, but in contrast who calibrates better in this case is the pair of 2.5m values. While with 2.5 meters we calculate values with acceptable error margins of about 30 centimeters. With the rest of the values the margin of error shoots up and the calculation of the distance fails by more than 1m.

These results show that we have similar distance values to those already seen when confronted with the test tests. For each specific case we have some cases that work better than others but most of them obtain acceptable results. In addition we have certain extreme scenarios where we can only obtain good approximations of the distances with very specific pairs of values.

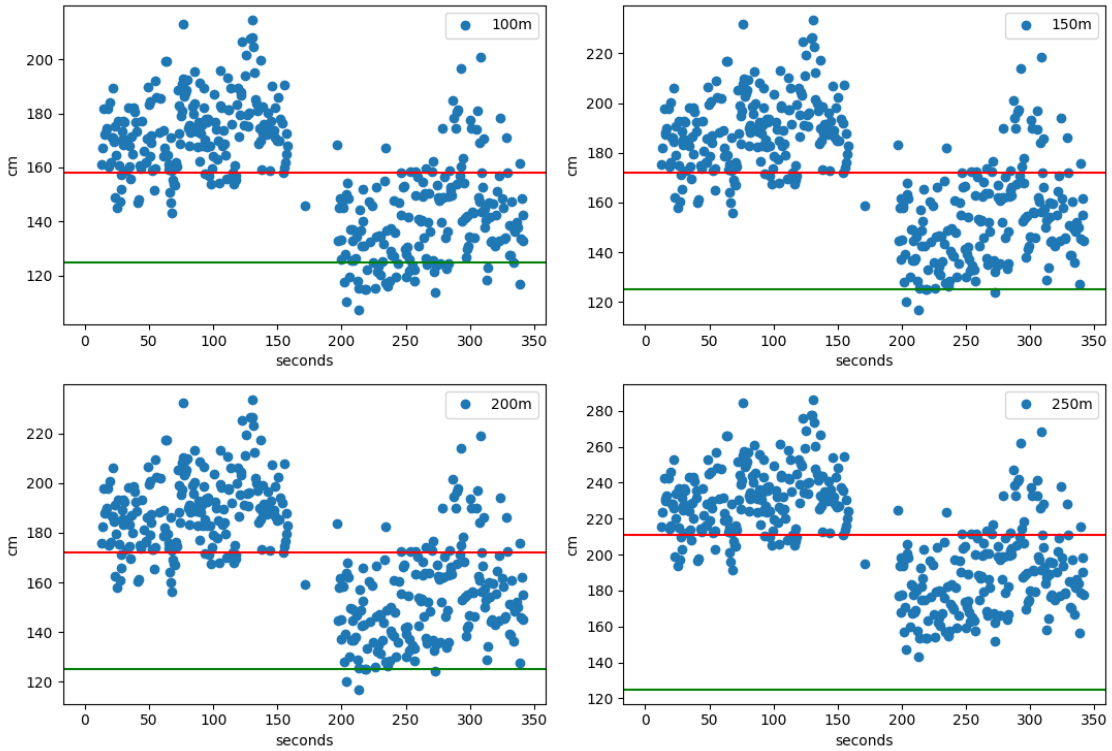


Figure 4.8: Results of calculating the distances with the four pairs for 1.25m in scenario 1

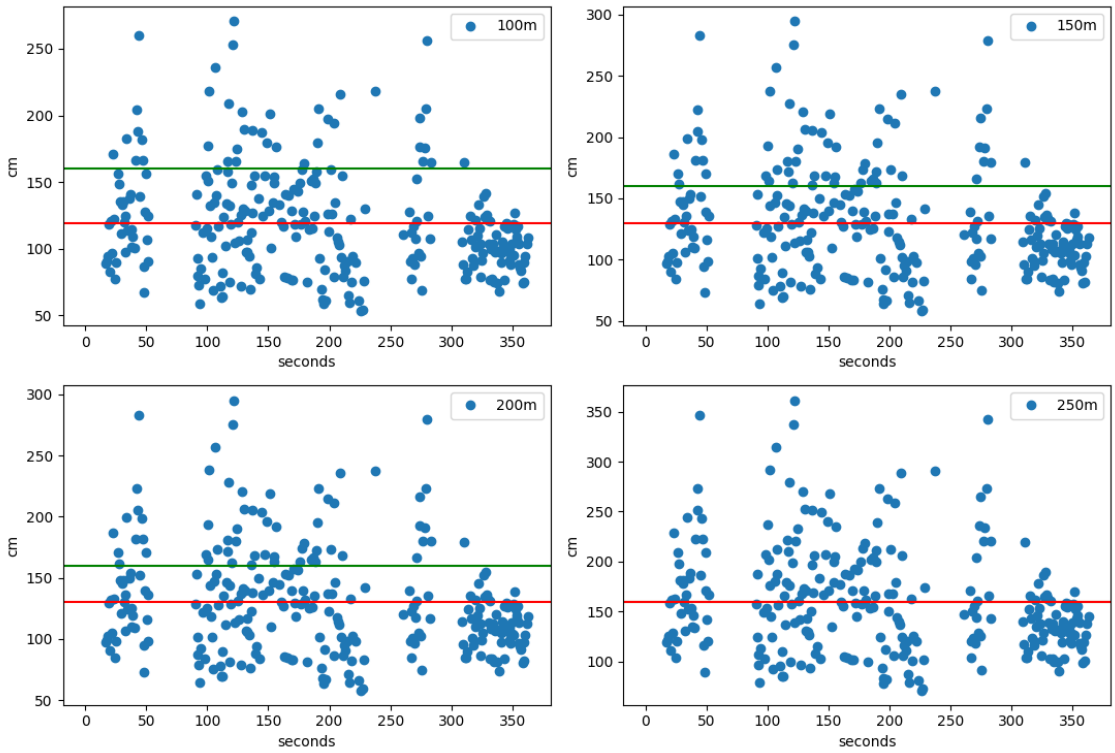


Figure 4.9: Results of calculating the distances with the four pairs for 1.6m in scenario 2

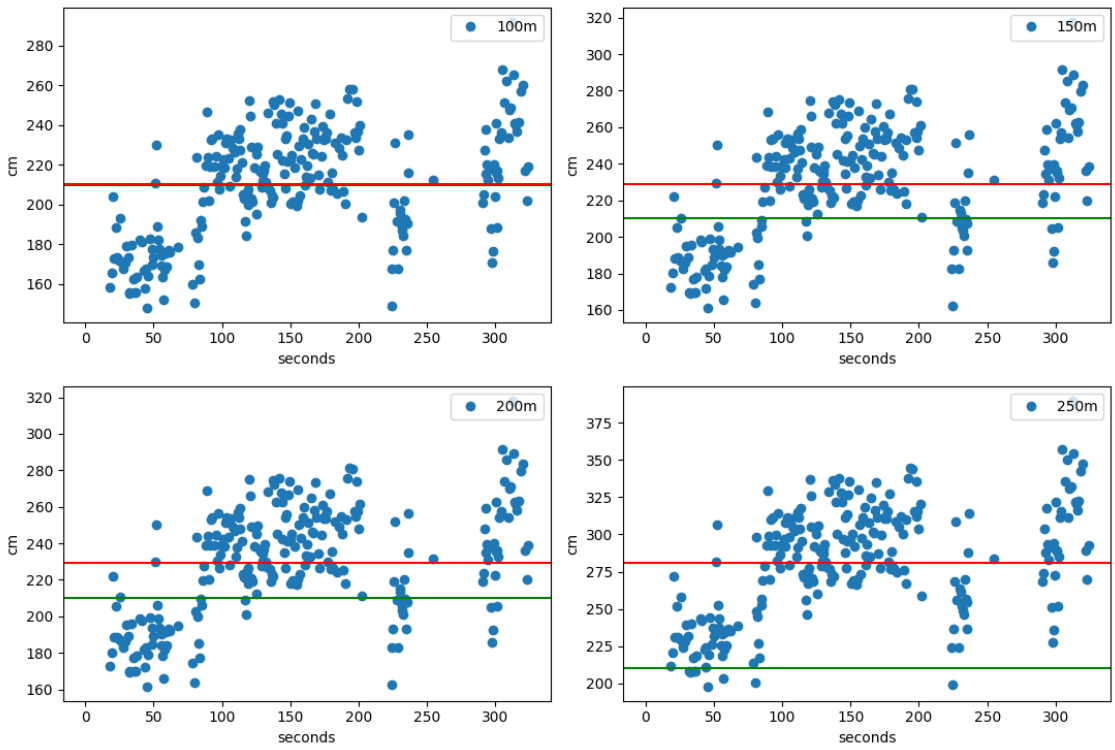


Figure 4.10: Results of calculating the distances with the four pairs for 2.10m in scenario 4

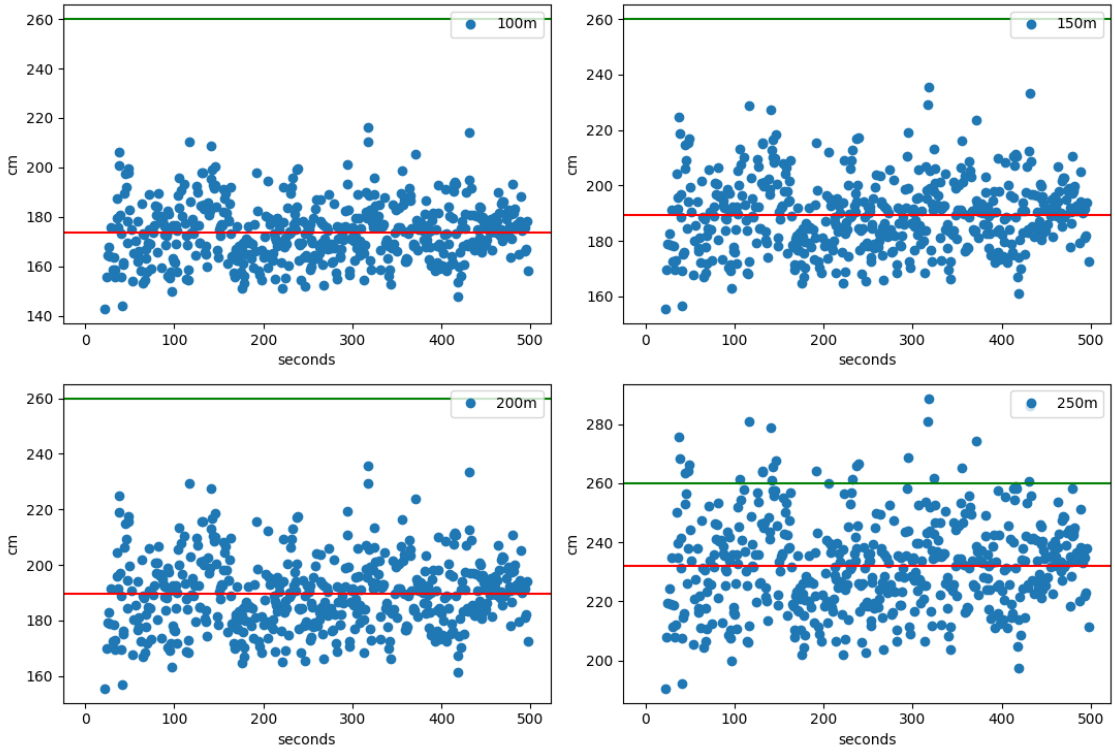


Figure 4.11: Results of calculating the distances with the four pairs for 2.6m in scenario 1

4.3. Classifier

Another mechanism that we will evaluate to calculate distances using Bluetooth Low Energy will be a machine learning classifier. In the case of contact tracing the precise distance is not necessary, it work in ranges of distances to determine risk of infection. For this we will classify the readings into different distance groups and evaluate with which classification method the best results are obtained. The first step of all is to determine which is the best classification method to use and what parameters we are going to use to classify. For this we are going to make a quick classification with the main classification methods:

- **Nearest Neighbours:** classifies finding a number of training samples close to the new sample and assigns a label with this information.
- **Support Vector Machines:** Construct a hyperplane or a set of hyperplanes to separate and classify the data.
- **Decision Tree:** Classifies the samples based using simple decision rules inferred from the training data.
- **Random Forest:** Classifies constructing a multitude of decisions trees and classifies based in the output of most trees.
- **Naive Bayes:** Applying the Bayes theorem with the naive assumption of conditional independence to classify the data.
- **Linear Discriminant Analysis:** Based on a generalization of Fisher's Linear Discriminant finds a linear combination to classify.
- **Quadratic Discriminant Analysis:** Like the LDA but without the assumption that the covariance of each class is identical.

We are going to classify using the Rssi values collected from all the distances and all the scenarios, where every block of distances will be one group to classify. We also are going to add more features of the signal to get a better results in the classification. The added features are going to be:

- **Data loss:** The percentage of packets lost in the transmission.
- **Max Rssi:** The maximum Rssi value collected.
- **Min Rssi:** The minimum Rssi value collected.
- **Dispersion:** The dispersion of all the Rssi values collected.

With all of this we use the collected and filtered data as training set and we use it against the main classification methods without tuning. We can see the results in the figure 4.12. In the quick test to evaluate which one of the classification methods performs better we divide in 4 blocks of distances. Near, that is based on 1m, Medium based on 1.5m, Far based on 2m and very Far with 2.5m o greater.

We can see that no one gets a very precise classification, but this is not very critical in this case. We were doing a quick run to determine which one of the classifications gets

the better preliminary results, to continue working with it. We can see that Random Forest is the best classification method in the preliminary tests followed closely by Quadratic Discriminant Analysis.

The first thing we can see is that the classifications are stable. In the great majority of cases the classification is the same for all the Rssi values. With some classifiers the 2.1m

Figure 4.12: Comparison of the Results with multiples classifiers

readings fluctuate between two adjacents ranges. Another characteristics we can see in this preliminary test is that we have problems to classify as a very far distance. This is an expect behaviour, if we remember the first readings of the Rssi without filter the signal we can observe that the readings for 2m and 2.5m are very close and intertwined. Now that we have two candidates for classification Random Forest and Quadratic Discriminant Analysis we are going to test with different sets, not only one per distance. And in the case of Random Forest tuning the parameters of the classifier. In the tuning of the Random Forest we are going to find the number of trees, the quality and minimum samples for a split and the max depth for the trees that offers a better results. After this we are going to see which one performs better.

4.3.1. Random Forest

The first thing we are going to do is to run the Random Forest classifier with the same parameters as in the quick test to see in more detail how correct the classification is. In this case we have a considerably small forest, with a total of 10 trees, a maximum depth of 5 nodes and only two samples are required for a split, figure 4.13 shows the results of classify the test sets.

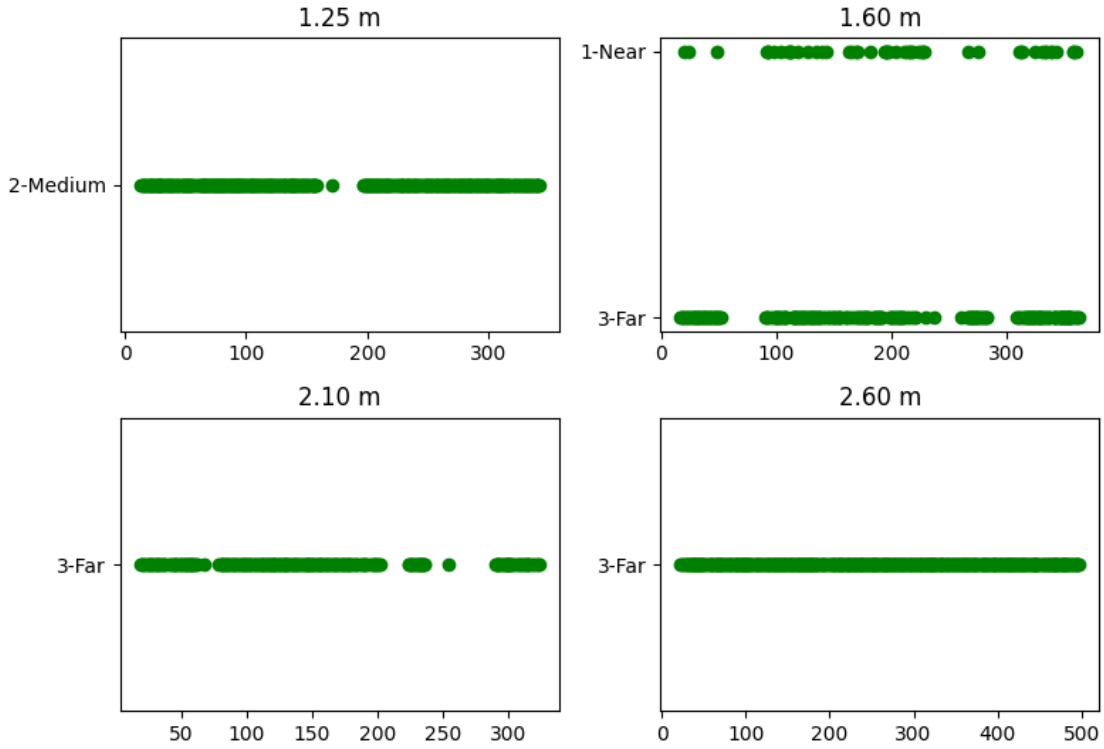


Figure 4.13: Initial results of Random Forest Classifier

We can see that when classifying the result is not as good as expected. First of all the 2.6m set is not classified as far away, but this is something we have not seen with any of the classifiers. Secondly, it presents difficulties in discerning the correct distance in the 1.6m case. This last problem is a behavior that, seeing the behavior of the signal at 1.6m, is expected to occur. In figure 4.14 we can see a comparison of the Rssi of both the 1.6m set and the 1m and 2m sets. In this case they are all on the same scenario and for the

classification 1m is classified as near and 2m as far. We can observe in this case how the 1.60m signal oscillates between a wide range of Rssi values and in some cases it is found in values lower than the 1m Rssi. With this in mind we will now try to adapt the Random Forest parameters to see if we are able to more accurately classify the 1.60m distance.

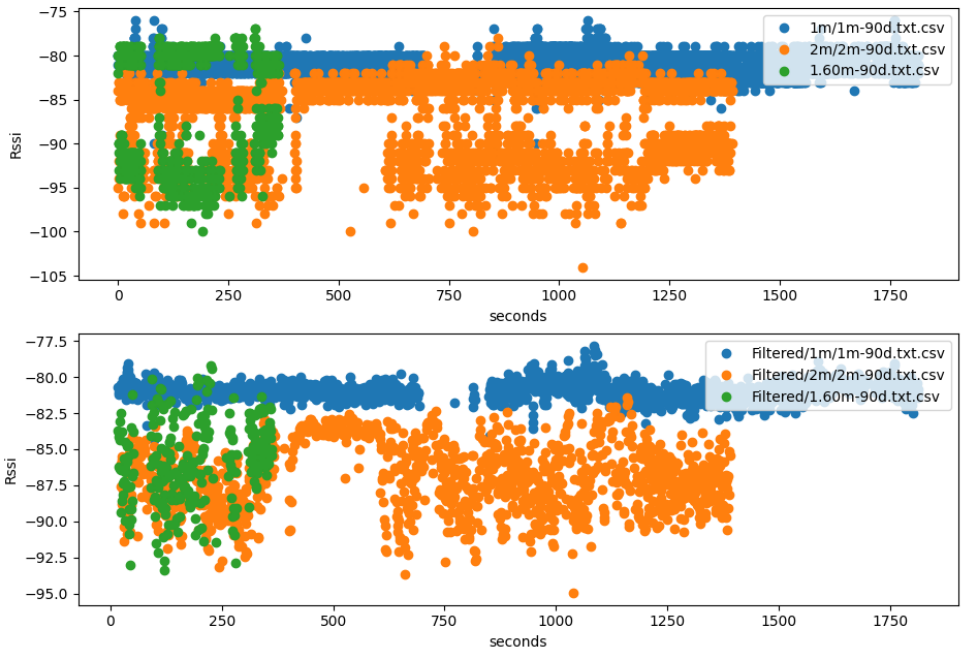


Figure 4.14: Comparison of filtered and unfiltered signals at 1m, 1.6m and 2m.

The parameters that we will modify in random forest to improve the classification are the following:

- The number of trees in the forest.
- The maximum depth of the tree.
- The minimum number of samples required to split an internal node.

After tuning, the best combination of parameters to obtain the best possible classification with Random Forest is the following. A total of 15 trees with a maximum depth of 10 nodes and with a minimum of 10 samples to split a node. The other combinations of parameters result in a misclassification of 1.6m, considering it to be in the 1m range, or presenting problems classifying 1.25 meters and being unable to determine whether it is in the 1.5m or 2m range. The final classification can be seen in Figure 4.15.

4.3.2. Quadratic Discriminant Analysis

Next we will evaluate and perform tuning for Quadratic Discriminant Analysis, in this case the tuning that can be done is more limited in comparison with Random Forest. First, as

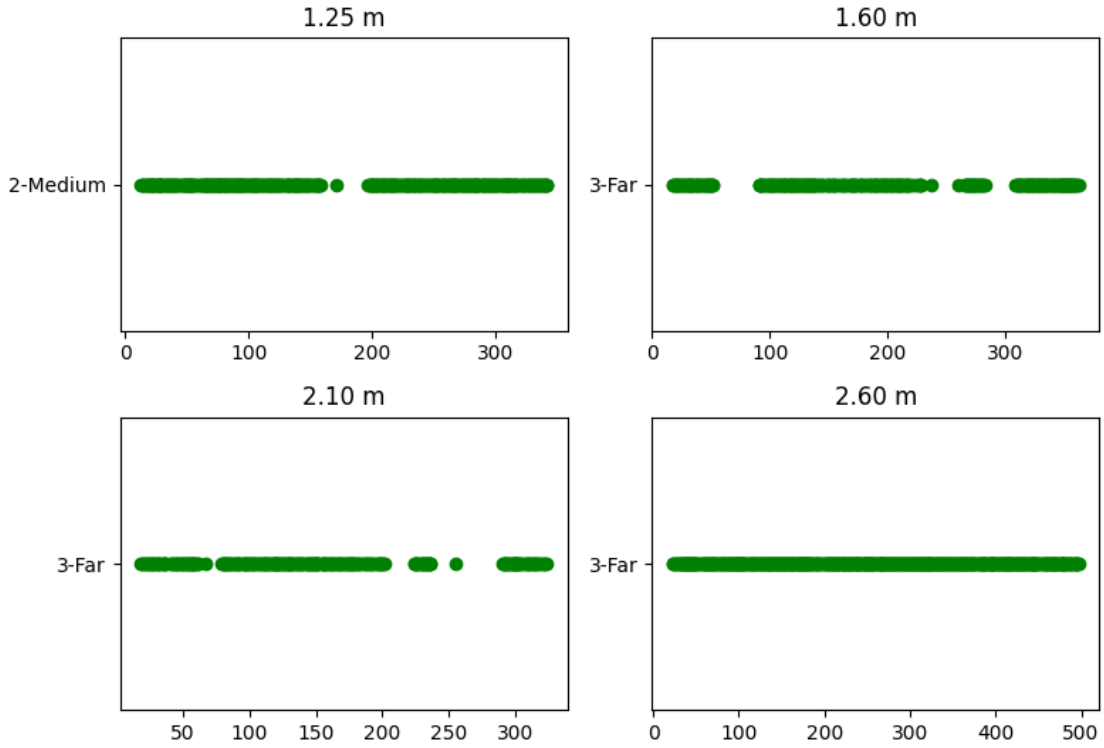


Figure 4.15: Tuned Classification of Random Forest.

with Random Forest, we will look at the results of the classification in the case of running it with the parameters of the preliminary results.

In the figure 4.16 is available the classification, in this case we see that QDA presents two problems. The difficulty to classify 2.6m in the plus 2.5m range, which is something we have observed in all classifiers. And the difficulty in determining whether the 2.10m set is in the 2m or 2.5m range.

In the case of Quadratic Discriminant Analysis we have two parameters that can be tuned:

- The class proportions, this parameter is inferred from the data.
- Regularizing covariance estimates by class

In the end the parameter that we can really modify in this case is the regularization of the covariance, the other parameter is already determined in the training data. In our case with a very small correction of 0.25 we corrected the main problem of Quadratic Discriminant Analysis, it now correctly classifies the 2.10m signal. If we increase the correction further the classification starts to present problems with other classifications, such as inaccurately classifying 1.25m in the 2m range. The results of the Quadratic Discriminant Analysis classification after tuning can be seen in Figure 4.17.

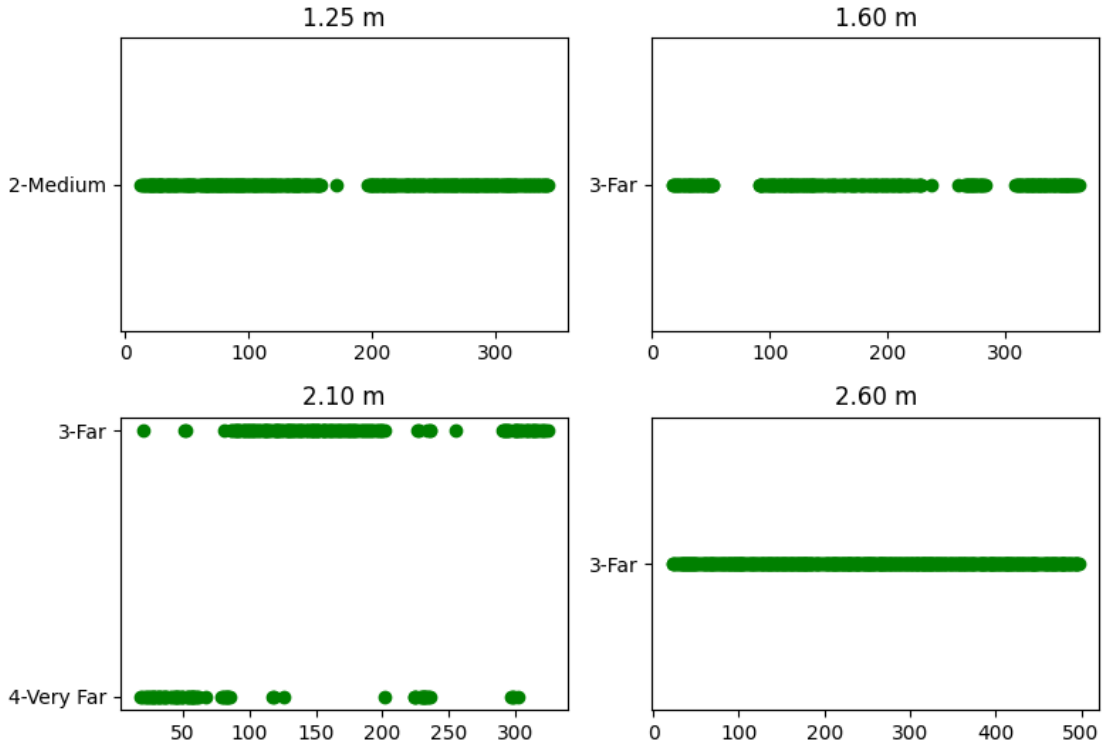


Figure 4.16: Initial results of Quadratic Discriminant Analysis Classifier

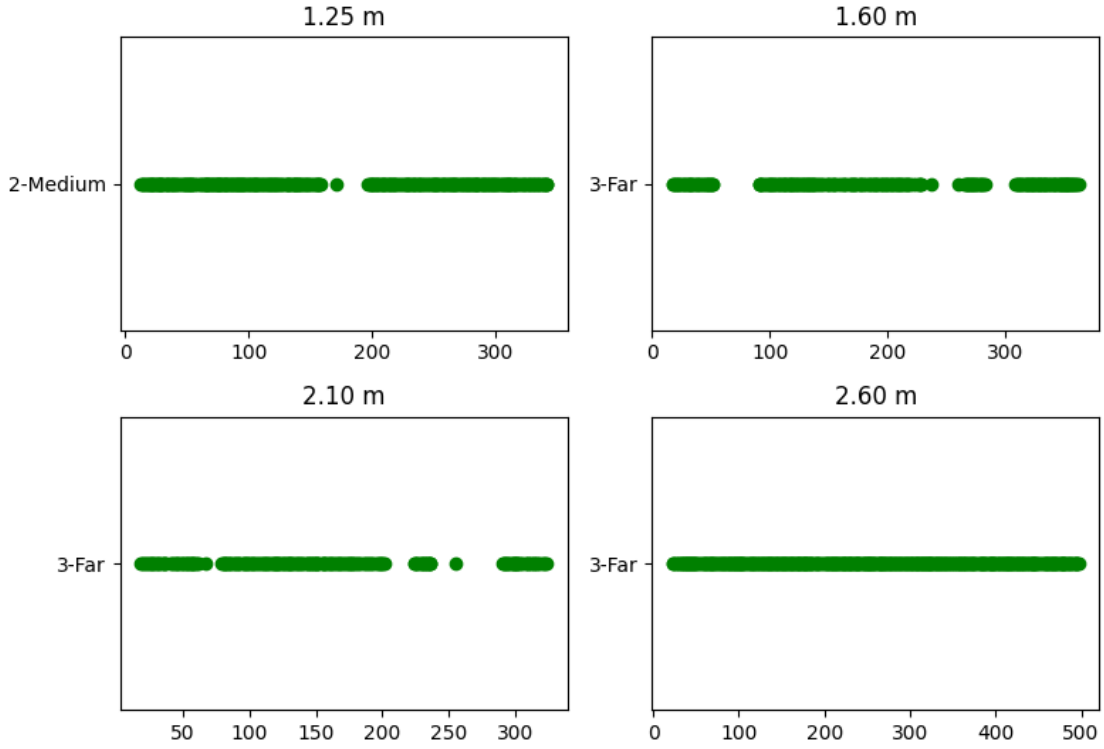


Figure 4.17: Tuned results of Quadratic Discriminant Analysis

4.4. Results

With the parameters of the two classification mechanisms that give the best results defined, let's see what results we get if the classes we classify on are different from the ones we have been using.

The first thing we will try to do is to classify into categories not by distance, but by how the signal quality is distributed. We will also check the behavior of the classification with only three classes, one for short distances, one for long distances and one for intermediate values. Another option is to classify only in two groups, if the distance is short, less than 2m, or large, more than 2m. In the table 4.2 we can see the classes for the different test based in the distances of the training sets. With these three combination of classes we will evaluate how good are the results we obtain with Random Forest and Quadratic Discriminant Analysis.

Meters	Rssi Distribution	Three Classes	Near or Far
1m	Next	Near	Near
1.5m	Near	Near	Near
2m	Far	Medium	Far
2.5m	Far	Far	Far
3m	Away	Far	Far

Table 4.2: The multiple classification sets

The first classification will be by visible blocks according to the distribution of the signal quality. In figure 4.18 we can observe the Rssi filtered at various distances and how we can distinguish 4 classifications based on this. With these four classifications we ran Random Forest and Quadrant Discriminant Analysis to observe the accuracy of their classification.

We run both classifiers with the parameters that we have obtained by tuning and collect the results that are in figure 4.19. We can observe that the behavior of the two classifiers is very similar. They consider that 1.60m, 2.10m and 2.60m are all in the same range, which would be correct for 2.10m and 2.60m, but neither of them is able to distinguish that 1.60m should be in another classification.

Another detail is that 1.25m presents problems in its classification, while Quadratic Discriminant Analysis is better able to determine where the signal is actually located. In both cases it oscillates between the 1.5m and the 2m group. In neither case does it oscillate between 1m and 1.5m which would be a more accurate classification.

Another option to distribute the distances would be to make three groups, one that is considered close, where it would be 1m and 1.5m and another far which would be 2.5m and 3m. While the distance at 2m would remain as an intermediate strip to avoid overlapping near and far.

The classifier results can be found in Figure 4.20. In this case we can see how Random Forest again presents some problems when classifying the 1.25m set and places it in the intermediate set when it should be considered as close as Quadratic Discriminant Analysis does.

The 2m set, and the 1.60 set could both be considered correctly classified. It is true that 1.60m should be classified more as proximate than intermediate. On the other hand, both

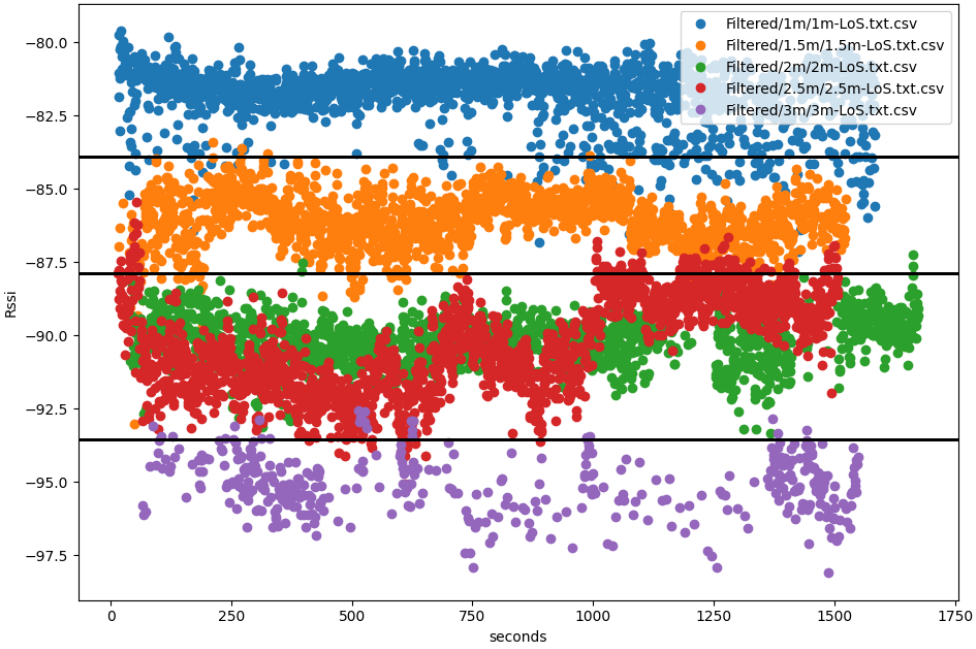


Figure 4.18: Signals filtered with the 4 blocks according to the Rssi visible

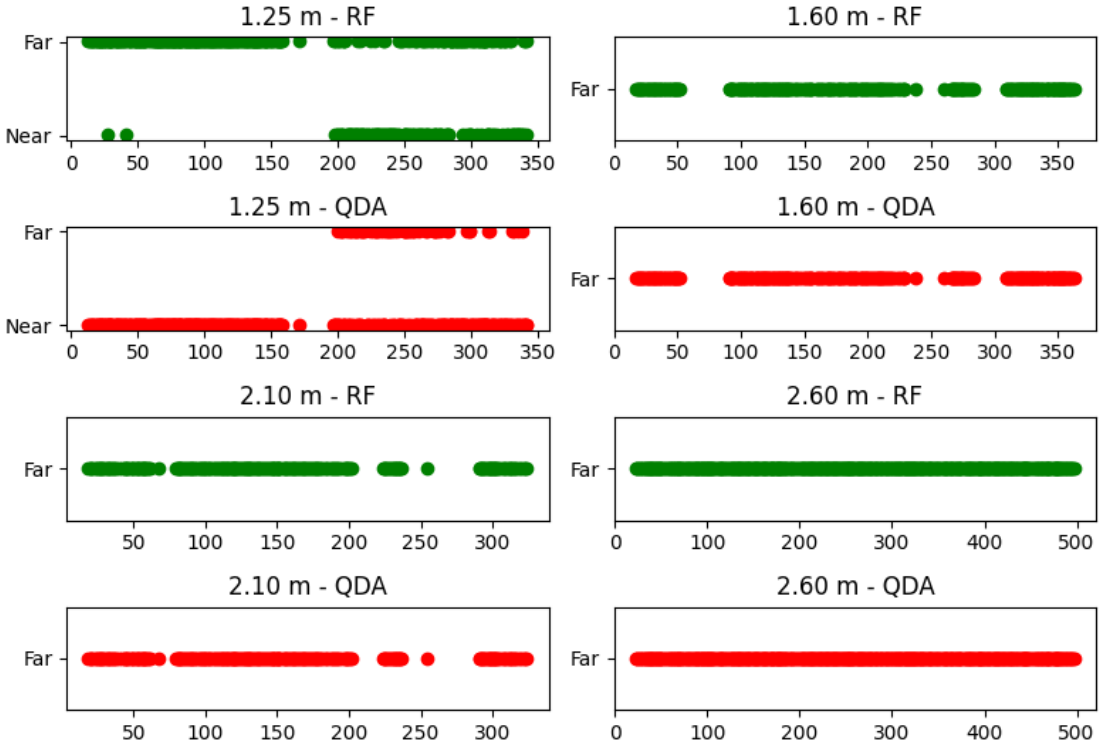


Figure 4.19: Classification results with classes by Rssi

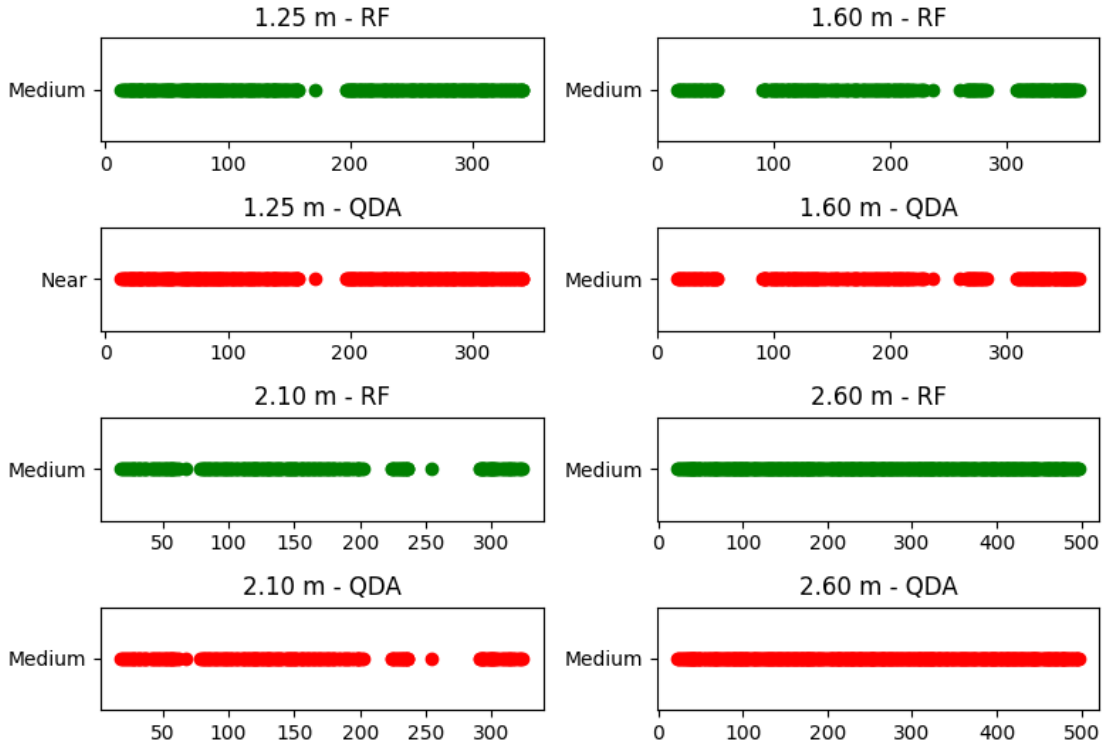


Figure 4.20: Classification results with 3 classes

classifiers consider that 2.60m is classified as distance in the intermediate range when its signal should be classified as far. The case of the 2.60m classifying problems we are observing, both in the preliminary tests and now will be covered in more detail in the conclusions.

The last step would be to convert our system with several classes for classifying to a binary one. Only with two classes being close, less than 1.75m, and everything else far, more than 1.75m. This case would be to stop classifying in a multi-class scenario and classify in only two options.

After obtaining the results of classifying with both methods visible in Figure 4.21, we can see how in this case Random Forest fails completely. It goes on to classify all signals as far away, on the other hand Quadratic Discriminant Analysis is able to discern that 1.25m is close, but fails again in the case of 1.60m.

The problem of misclassification of the 2.60m set is not present but in this case it is not present because of the distribution of the classification. 2.60m is misclassified in the 2m set, as now 2m, 2.5m and 3m are the same set the misclassification error cannot occur.

4.5. Conclusions

In this section, we have studied the behavior of Bluetooth Low Energy connectivity, and we see that both methods, mathematically determining a distance and the Machine Learning classifier, worked correctly. A functional conversion for digital contact tracing of the Bluetooth Low Energy signal quality to an approximation value of distance between the

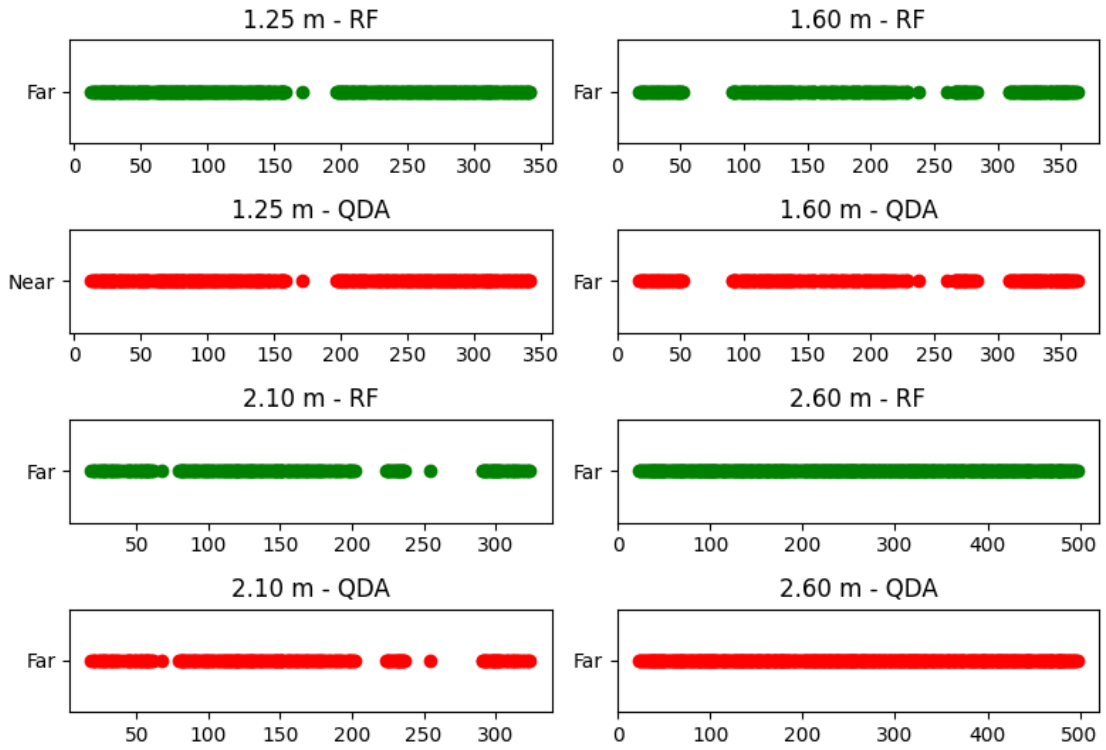


Figure 4.21: Classification results with 2 classes

transmitter and the receiver can be performed.

In most cases, we can determine the risk of contacts with the distance approximations we obtain. We have enough accuracy to determine if two people are sharing a table or if they are both in the same room. It is true that in some more complex scenarios this mechanism presents problems. For example in the case of 2.60m, which presents problems for all methods. Where it is considered a much smaller distance than it really is and the risk is only correctly determined in one of the four cases of the mathematical method.

The problem of the recurring misclassification of the 2.60m can be more clear if we observe in more detail in Figure 4.22 the Rssi values of the 2.6m set in contrast with the 2m and 3m training sets.

The Rssi values of the 2.60m set and the 2m set overlap, both have the same Rssi ranges. This is something we have already seen when comparing the 2m and 2.5m Rssi, both signals overlap. When calculating the distance of the 2.60m set we see how it is more normal that the calculated distance is closer to 2m than 2.60m.

In the case of the classifiers, these present another problem in the case of 1.60m readings. As we have already seen, the 1.60m signal distribution presents problems when it comes to being classified as the Rssi oscillates between the blocks of 1m and 2m. In the case of the mathematical method, since the entire signal is evaluated jointly, this problem is mitigated.

If we compare in more detail Random Forest and Quadratic Discriminant Analysis as classifiers both obtain very similar results. It is true that when starting to change the distribution of classes Quadratic Discriminant Analysis behaves better and maintains more valid classification results while Random Forest fails completely in some cases. Therefore we can

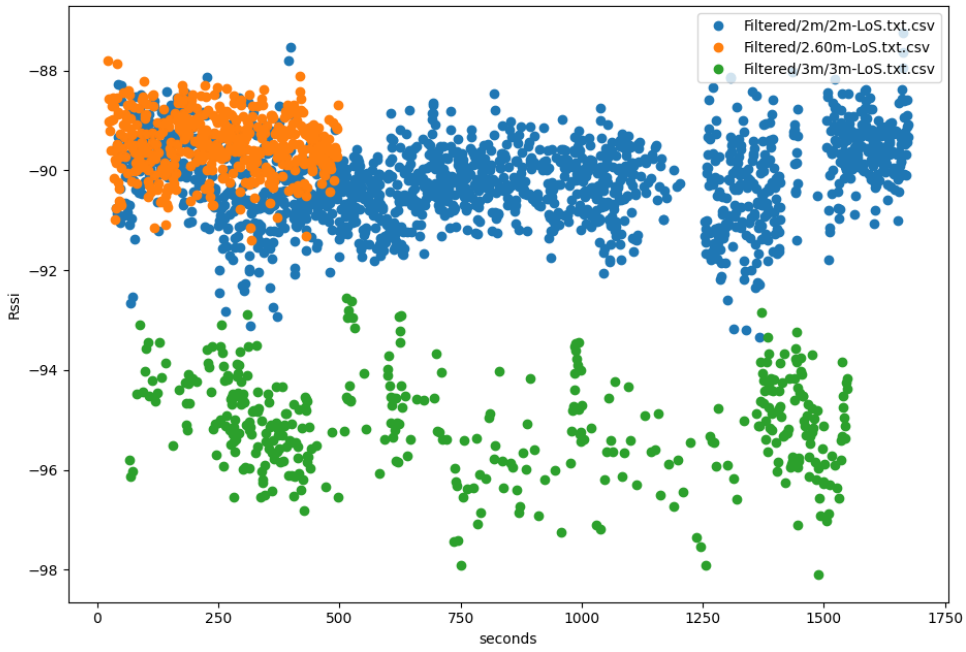


Figure 4.22: Comparison of 2.60m set with the 2m and 3m training sets

say that in these scenarios Quadratic Discriminant Analysis classifies better and is more flexible with the class distribution.

Another possibility would be to discard the classification with several classes. Determining only two possible ranges near and far, in which case other classification mechanisms such as Support Vector Machines could be available, which in this case have not passed the preliminary tests since they are not designed to classify into more than two classes. But in this case the possibility of determining with granularity different levels of risk of contact would be lost. And looking at the current results, the classification risk would have to be greatly improved and would not make any substantial difference in the case of the mathematical method.

It is quite possible that a much more complete data set with a higher granularity would obtain a much better results. It would give us more different Rssi values and in multiple scenarios, which would allow a better calculation of the reference values for the mathematical method and more data to feed the qualifier. Another approach in the case of classifiers would be, in contrast to classify each of the received Rssi beacons, to group them in clusters and classify with the average value to avoid the problems presented by the 1.60m test.

Despite the problems with both methods, along with the overlapping characteristics of some BLE signals, both methods are able to validly determine an acceptable risk value. The mathematical method determines it more accurately in most cases, but the classifier, having more information than just the Rssi, may be able to obtain more details from the contact.

CHAPTER 5. CONCLUSIONS

After evaluating multiple methods to determine the ability of Bluetooth Low Energy to determine risk values based on distance and signal quality we can determine that Bluetooth Low Energy is functional and that it is a valid technology for digital contact tracing.

We have been able to determine that the integration of Internet of Things elements into digital contact tracing architectures is feasible and without increasing costs and implementation difficulties. With this improvement, more information is available to assess the risk of contact. You can have knowledge of the level of air quality, temperature, humidity, etc. which are factors that, in the case of COVID or other diseases, affect their ability to transmit.

After evaluating multiple methods to determine the risk of contact based on the approximate determination of distance. We can also conclude that Bluetooth Low Energy is a valid and functional technology to be used in a robust way for contact tracing, is unable to establish a precise distance but this is not a problem for the contact tracing. The case of classifiers is a method that presents more difficulties in contrast to the mathematical method. But both mechanisms are able to reliably determine risk levels or contact possibilities based on the quality of the received signal.

One of the main limitations in this case is the characteristics of the antenna used for Bluetooth Low Energy testing. Its range is considerably small, in reference to the standard, and it does not have the capabilities that most devices would have. This has limited the study to distances no greater than 3 meters.

Despite its limitation in the case of range, this does not present a major problem. In the case of risk of contact once beyond 3 meters, this is greatly reduced. In the case of the Internet of Things node, packet loss over time is also not that much of a concern. The communication with the node should not be as stable as that of the users as the environmental values do not vary as quickly.

In addition, in the case of Bluetooth Low Energy, there is potential to obtain a more accurate and detailed tracing of contacts with a further development of its methods and with the application of some improvements.

One of the possible improvements would be, seeing that the mathematical method works better, but lacks information beyond signal quality, a combination of the two methods. We have observed that the mathematical method works better if the reference value is closer to the value of the signal to be evaluated. With this we could first use a classifier to determine the approximate distance and then use the mathematical method with that information.

Another option is to know in advance the situation between sender and receiver. In the figure 5.1 we can see some preliminary results of a Naive Bayes classifier, which determines not the range of the distance, but the orientation between transmitter and receiver.

A future work would be to study whether a combination of the different methods would lead to improvements in better assessing the distances of contagion. Or if more information available for the mathematical method implies that it obtains better or more stable results.

With this and in contrast to other available technologies. We can say that Bluetooth Low Energy is the best suited compared to the other alternatives, such as GPS, Wifi or classic Bluetooth. Considering the current situation of infrastructure and common use of mobile

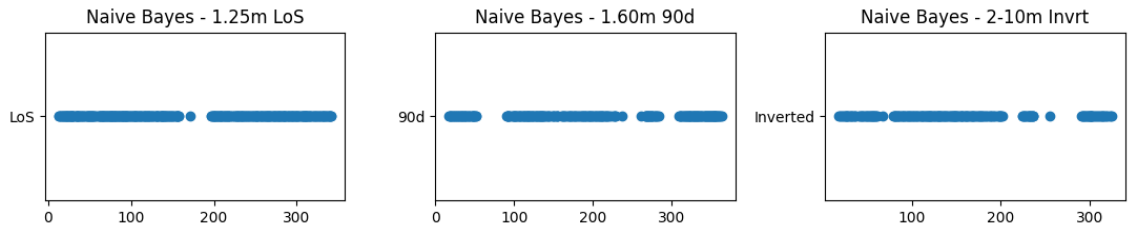


Figure 5.1: Preliminary results of Naive Bayes in classifying the orientation of the transmitter and receiver.

devices. Bluetooth Low Energy is not only the only technology with low power consumption, with sufficient accuracy and that does not require previous infrastructure, it is also able to assess risks based on signal quality and distance. And it is a technology that can considerably improve this performance with slight adaptations and still has room for improvement.

BIBLIOGRAPHY

[1] *Arduino Nano 33 IoT — Arduino Documentation*. URL: https://docs.arduino.cc/hardware/nano-33-iot (visited on 06/13/2022).

[2] "ATmega4808/4809 Data Sheet". In: (2020), p. 551.

[3] *Bluetooth Core Specification 4.0*. URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/ (visited on 02/15/2022).

[4] *Exposure Notifications: Helping fight COVID-19 - Google*. Exposure Notifications: Helping fight COVID-19 - Google. URL: https://www.google.com/intl/en_us/covid19/exposurenotifications/ (visited on 05/29/2022).

[5] Pau Ferrer-Cid et al. "Sampling Trade-Offs in Duty-Cycled Systems for Air Quality Low-Cost Sensors". In: *Sensors* 22.10 (Jan. 2022). Number: 10 Publisher: Multidisciplinary Digital Publishing Institute, p. 3964. ISSN: 1424-8220. DOI: 10.3390/s22103964. URL: https://www.mdpi.com/1424-8220/22/10/3964 (visited on 06/11/2022).

[6] Luca Ferretti et al. "Quantifying SARS-CoV-2 transmission suggests epidemic control with digital contact tracing". In: *Science* 368.6491 (May 8, 2020). Publisher: American Association for the Advancement of Science, eabb6936. DOI: 10.1126/science.abb6936. URL: https://www.science.org/doi/10.1126/science.abb6936 (visited on 05/29/2022).

[7] *File:A schematic of app-based COVID-19 contact tracing (Fig. 4 from Ferretti et al. 2020).jpg - Wikipedia*. URL: https://commons.wikimedia.org/wiki/File:A_schematic_of_app-based_COVID-19_contact_tracing_(Fig._4_from_Ferretti_et_al._2020).jpg (visited on 05/29/2022).

[8] *High Accuracy Distance Measurement Demo For Keyless Entry Systems*. Bluetooth® Technology Website. May 24, 2021. URL: https://www.bluetooth.com/bluetooth-resources/high-accuracy-distance-measurement-hadm-demo-for-keyless-entry-systems/ (visited on 05/10/2022).

[9] Douglas J. Leith and Stephen Farrell. "Coronavirus Contact Tracing: Evaluating The Potential Of Using Bluetooth Received Signal Strength For Proximity Detection". In: *arXiv:2006.06822 [cs, eess]* (May 19, 2020). arXiv: 2006.06822. URL: http://arxiv.org/abs/2006.06822 (visited on 02/01/2022).

[10] Tom Lovett et al. "Inferring proximity from Bluetooth Low Energy RSSI with Unscented Kalman Smoothers". In: *arXiv:2007.05057 [cs, eess, stat]* (July 9, 2020). arXiv: 2007.05057. URL: http://arxiv.org/abs/2007.05057 (visited on 02/01/2022).

[11] Jessica Morley et al. "Ethical guidelines for COVID-19 tracing apps". In: *Nature* 582.7810 (June 2020). Bandiera_abtest: a Cg_type: Comment Number: 7810 Publisher: Nature Publishing Group Subject_term: SARS-CoV-2, Technology, Policy, Public health, pp. 29–31. DOI: 10.1038/d41586-020-01578-0. URL: https://www.nature.com/articles/d41586-020-01578-0 (visited on 02/01/2022).

[12] *Nina-W10 Datasheet*. URL: https://content.arduino.cc/assets/Arduino_NINA-W10_DataSheet_%28UBX-17065507%29.pdf.

[13] Leonie Reichert, Samuel Brack, and BjÖRN Scheuermann. "A Survey of Automatic Contact Tracing Approaches Using Bluetooth Low Energy". In: *ACM Transactions on Computing for Healthcare* 2.2 (Mar. 2021), pp. 1–33. ISSN: 2691-1957, 2637-8051.

DOI: 10.1145/3444847. URL: https://dl.acm.org/doi/10.1145/3444847 (visited on 06/21/2022).

[14] Javier Rodas, Carlos J. Escudero, and Daniel I. Iglesia. “Bayesian filtering for a bluetooth positioning system”. In: *2008 IEEE International Symposium on Wireless Communication Systems*. 2008 IEEE International Symposium on Wireless Communication Systems. ISSN: 2154-0225. Oct. 2008, pp. 618–622. DOI: 10.1109/ISWCS.2008.4726130.

[15] Siraporn Sakphrom et al. “A Simplified and High Accuracy Algorithm of RSSI-Based Localization Zoning for Children Tracking In-Out the School Buses Using Bluetooth Low Energy Beacon”. In: *Informatics* 8.4 (Sept. 25, 2021), p. 65. ISSN: 2227-9709. DOI: 10.3390/informatics8040065. URL: https://www.mdpi.com/2227-9709/8/4/65 (visited on 05/31/2022).

[16] *Specifications – Bluetooth® Technology Website*. URL: https://www.bluetooth.com/specifications/specs/ (visited on 05/29/2022).

[17] *UNO WiFi Rev2 — Arduino Documentation*. URL: https://docs.arduino.cc/hardware/uno-wifi-rev2 (visited on 03/15/2022).