Politecnico di Milano
School of Industrial and Information Engineering

Master of Science in Computer Engineering

Proximity Detection Using Bluetooth Low Energy Technology to Identify Seated People

Supervisor: Prof. Fabio Salice
Co-Supervisor: Andrea Masciadri

Author: Daniel Rosato
Student Id.: 851785

December 2018

Abstract

This work presents a non-invasive monitoring system to recognize if a person is seated in a chair using Bluetooth Low Energy (BLE) technology. The system is composed by devices that can be easily attached to a piece of furniture: a transmitter module or beacon that continuously broadcasts a Bluetooth Low Energy signal, and a client which collects the RSSI value of the received messages. All this is analyzed in a central server that evaluates the attenuation of the BLE signal within the 2.4GHz band, correlating it to the presence of a human body in the proximity of the device. This is performed using an unsupervised machine learning algorithm that learns the behaviour of the BLE signal.

The BLE signal is noisy by nature, i.e. it is affected by several environment variables. To prevent this from damaging the predictions, a filter was introduced to remove the noise from the signal. More over, to further improve the estimations, more beacons are added, providing more information and a clearer picture of the state of the chair. To reduce the noise and incorporate the measurements of different beacons, a Kalman Filter with sensor fusion is implemented.

Sommario

Questo lavoro presenta un sistema di monitoraggio non invasivo per riconoscere se una persona è seduta in una sedia utilizzando la tennologia Bluetooth Low Energy (BLE). Il sistema è costituito da dispositivi che possono essere facilmente collegati a una mobilia: un modulo trasmettitore o beacon che emette continuamente una segnale BLE e un cliente che raccoglie il valore RSSI dei messaggi ricevuti. Tutto ciò viene analizzato in un server centrale che valuta l'attenuazione del segnale BLE entro la banda di 2,4 GHz, correlandolo con la presenza di un corpo umano nelle vicinanze del dispositivo. Questo viene fatto usando un algoritmo di machine learning non supervisionato che impara il comportamento della segnale BLE.

La segnale BLE è rumorosa per natura, cioè è influenzato da diverse variabili ambientali. Per evitare che ciò danneggi le previsioni, è stato introdotto un filtro per eliminare il rumore della segnale. Inoltre, per migliorare ulteriormente le stime, vengono aggiunti più beacon che forniscono maggiori informazioni e un quadro più chiaro dello stato della sedia. Per ridurre il rumore e incorporare le misure di diversi beacon, viene implementato un filtro Kalman con fusione di sensori.

Acknowledgements

I would like to thank Prof. Fabio Salice and PhD. fellow Andrea Masciadri for guiding me through this process.

To my family and friends, thank you for your support. I could not have done it without you.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

There is a growing concern about the increasing number of people over 60 years old. By the year 2050, the number is expected to reach nearly 2.1 billion people. An increase of around 56% from an estimated 900 million today according to the United Nations report [18]. This translates into several challenges for the future of society.

The overload in the health care system represents one of those big challenges. It is only natural that with an increased elderly population, more resources will be needed to provide good care for them. With the number increasing so much, it is certain that a solution needs to be provided, and it has to be affordable and practical.

An emphasis in prevention has been surging in the past few years. The ability to monitor and assist people in their everyday lives, especially at their own homes as a measure to reduce health care costs in hospitalizations. A "proactive health care system" [7]. One that helps motivate life long healthy behaviour and health self awareness.

This is where technology comes in. It provides ways to collect data that can be analyzed later to detect patterns in the behaviour of the people being monitored.

This set of new technologies are based on a new paradigm called ambient intelligence, which aims at empowering people's capabilities by providing digital environments that are sensitive, adaptive and responsive to human needs [21]. This set of assisted living technologies based on ambient intelligence are called Ambient Assisted Living (AAL) tools.

1.1 Motivation

A lot of research has been performed on improving the state of the art with regards to tools, methods and algorithms related to AAL [21]. Making it easier to recognize anomalies on people's routines and trigger the appropriate response. By doing so, the response time to any problem they might encounter gets significantly reduced because of early detection, and that way help prevent any life threatening situation.

In most of the cases the home anomalies triggered by accidents or illness, cause a change in the behaviour of the people being monitored which results in inactivity. An example would be to monitor the amount of time spent in a particular room. In this case it is important to detect if the person is being seated or lying on the floor to trigger an alarm.

Another important thing to keep in mind is that senior citizens tend to forget and hate to carry positioning devices. Therefore, non intrusive solutions will always be preferred. Having said that, by monitoring the daily life of people at home, using non-intrusive environmental sensors, it is possible to infer the Activities of Daily Living that the person performed [28]

The focus of this work is to present a non intrusive solution for proximity detection of a human being, more specifically, applied to the case of identifying if a person is seated in a chair using **Bluetooth Low Energy** technology and therefore take advantage of all the benefits of the technology mentioned above.

The Bluetooth Low Energy (BLE) signal is transmitted in the 2.4 GHz radio frequency. This means that the signal may be distorted by interference from specific elements in the environment such as water. It is important to note that a human body is about 60% water, it absorbs signals in the 2.4 GHz band. Taken this into account, by identifying such distortion it is possible to infer human proximity.

With the help of a beacon transmitter that uses BLE technology it is possible to measure the amount of distortion in the signal when a human being is present between the beacon and the client. By using the RSSI (received signal strength indicator) as a measurement value, these changes can be determined. When a person stands between the devices, the signal is not completely shielded, because Bluetooth signals are omnidirectional. However, the RSSI decreases.

In this thesis, a system is designed around the measurements collected from the BLE client and beacon(s). This system is composed by a web server that is implemented to store and process all the information sent from the devices.

The BLE signal is noisy, it is affected by several environment variables. A filter is introduced to remove the noise from the signal. More over, to further improve the estimations, more beacons are added, providing more information and a clearer picture of the state of the chair. To reduce the noise and incorporate the measurements of different beacons, a Kalman Filter with sensor fusion is implemented.

Finally a machine learning algorithm is used to analyze and learn the behavior of the signal and make the according predictions about whether someone is seated or not.

The main contribution of this work is to demonstrate that Bluetooth Low Energy Technology can be used as a proximity detection solution to identify seated people.

The thesis is organized as follows:

- **Chapter 2** contains a review of the current state of the technologies and concepts to be used during the development of this work along with important theoretical background needed to understand the topics of the project.

- **Chapter 3** describes how all the components of the system interact with each other as well as justifying their inclusion in it.

- **Chapter 4** gives a detailed description of the theory and implementation regarding the Kalman filter used in the project. Also it expands on the theory and implementation behind the unsupervised learning algorithm used in the system.

- **Chapter 5** contains several experiments performed to test the behaviour of the system and provides the evaluation and analysis of the results obtained.

- **Chapter 6** describes the structure and architecture behind the web API and interface used to integrate all aspects of the system.
- **Chapter 7** gives an overall summary of the work done in the project as well as tackle limitations and possible future work.

Chapter 2

State of the Art

This chapter describes the background concepts and methods used in developing this thesis. An overview of the theory related to this work as well as the current state of the research regarding some of the topics.

2.1 Proximity Detection

Proximity detection of a human being has been widely studied in the past [6]. It has always been the topic of research and practical applications. This is to be expected given that it can be applied to a wide variety of scenarios.

The approach to proximity detection varies considerably depending on the methods and tools used. Two major approaches can be identified, these are classified as intrusive and non intrusive.

While an intrusive solution consists of attaching a wearable device to the user, a non intrusive solution on the other hand performs the detection without interfering with the user's activities. Here, the focus will be placed on non intrusive solutions.

Visible light cameras have been widely used to tackle this, especially in the fields of security and monitoring systems. This approach has several limitations given that its performance is affected by nonuniform illumination, shadows and low external light in the evening and night [9]. Not to mention that the setup of a monitoring system can get quite expensive, requires a lot of infrastructure and may irritate the person being monitored.

As an alternative to visible light cameras, thermal or light infrared cameras are also used to overcome the issues mentioned above. The drawbacks with this approach are that its performance is undermined by low contrast, low image resolution and high temperatures experienced during the day [9]. Also, as regular cameras, they suffer from the fact that a setup can get quite expensive, requires a lot of infrastructure and may irritate the person being monitored.

Capacitive sensors are also used for proximity detection. A common application is to detect seat occupancy in cars as an integral part of the airbag safety system. These sensors are efficient but they require electrodes to be placed in the surface layers of the seats [4]. They can be very accurate, although it can be a costly operation. In this scenario it is unlikely that the seat of the car gets replaced but the application this project aims for, is to find an easy to install and remove solution. And this is more of a permanent solution attached to the seat.

Other examples of capacitive sensors used for proximity detection can be found in Zeeman et al. [30] and Togura et al. [27]. These sensors are very noisy, since they are susceptible to changes in the environment like temperature and humidity. Also, they are not very power efficient solutions.

On the contrary, Bluetooth technology, more specifically Bluetooth Low Energy, has not been explored much for non intrusive proximity detection.

2.2 Bluetooth Low Energy

Bluetooth Low Energy (BLE) is a wireless personal area network technology, used for data transmission among devices. It is intended to be a low power alternative to regular Bluetooth.

BLE enables very low-power applications to run on batteries for months or even years. They use 50-99% less power than regular Bluetooth, which means the average beacon can last up to 2 years. This is thanks to the fact that BLE remains in sleep mode constantly except for when a connection is initiated. The actual connection times are only a few milliseconds, unlike Bluetooth which would take roughly 100 milliseconds. The reason the connections are so short, is that the data rates are so high at 1 Mb/s. BLE beacons also have a 60-80% cheaper cost of operation than competing standards.

While regular Bluetooth has higher data rates, it does consumes battery life quickly and costs a lot more. On the contrary, BLE is used in applications that do not require transmission of large amounts of data and because of this it has a battery life of years at a cheaper cost.

One example of BLE used as a non intrusive human motion detector, using the attenuation of the BLE signal caused by a human body, can be found in Sugino et al. [25].

No other efforts in the literature were found regarding the use of BLE as a non intrusive solution, much less as a tool to detect if a person is seated on a piece of furniture.

2.3 Recursive algorithms

There are several problems regarding algorithms that process all available data as a single batch:

- Memory limitations: all the observations need to be stored in memory.
- Computational load: it requires large amounts of computational power to perform the algorithms steps, specially if they require large matrix inversions.
- Time delay: to make quick decisions sometimes a less precise but immediate model can be more useful than one that is available only after processing large amounts of data.
- Inability to track the model: if the underlying behaviour of the data is non-linear or time varying, the model will only have local validity and will need to be updated and adapted to the most recent data.

On the other hand in recursive algorithms the parameter estimates are adjusted recursively over time in the following way:

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \text{ correction term}$$

They are needed for on-line application problems, where the underlying system is stationary, but the parameters need to be corrected as new data arrives. Another context in which recursive algorithms are useful is adaptive identification problems, where the underlying system is time variant or non linear, therefore the parameter estimates must be adapted in order to follow the system in its current region of operation.

Some of the applications can be found in adaptive control and fault detection, where the model of the system is continuously updated to rapidly detect significant changes in its behaviour.

Recursive algorithms provide several advantages over their batch counter parts. Here are the most important ones:

- Low computational requirement for efficient usage in online applications
- Modest memory requirement since not all data is stored in memory, and it does not grow over time.

2.4 Optimal Filtering

The term optimal filtering refers to a set of methods that can be used to estimate the state of a time varying system which is indirectly observed through noisy measurements.

"A filtering problem consists in reconstructing or estimating a non measurable variable based on the knowledge of other, measurable and uncertain variables" [20].

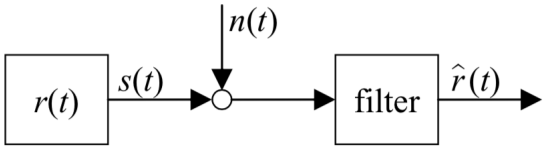


Figure 2.1: General filter problem (Taken from [20])

Figure 2.1 shows a generalized filtering problem where the latent variable $r(t)$ is different from the measured variable $s(t)$ although they are related in some way so that the filtering can be applied.

There are two types of filtering problems:

- Basic filtering where $r(t) = s(t)$. The variable to be estimated is the same as the measured one.
- Generalized filtering where $r(t) \neq s(t)$. The measured variable $s(t)$ depends on $r(t)$ but does not coincide with it.

There are two approaches to filtering. These are state-space models such as the Kalman filter, and input-output models such as Kolmogorov and Wiener.

2.4.1 Bayesian Filtering

Bayesian statistics takes past information, also known as the prior, into account at the time of calculations. The Bayesian approach finds the probability that an event is true given that a past event has happened.

Bayesian filtering refers to the Bayesian way of formulating optimal filtering.

In this scenario, the state of the system refers to a collection of dynamic variables that fully describe the system. The noise in the measurements reflects that they are uncertain, therefore, even if we knew the state of the system, the measurements would not be deterministic, but a distribution of possible values. The time evolution of the state is modeled as a dynamic system which is affected by a process noise, that reflects the uncertainty in the dynamics.

It can be said that the unknown quantity is a vector $\{x_0, x_1, x_2, ...\}$ which is observed through a set of noisy measurements $\{y_1, y_2, ...\}$. This is shown in Figure 2.2.

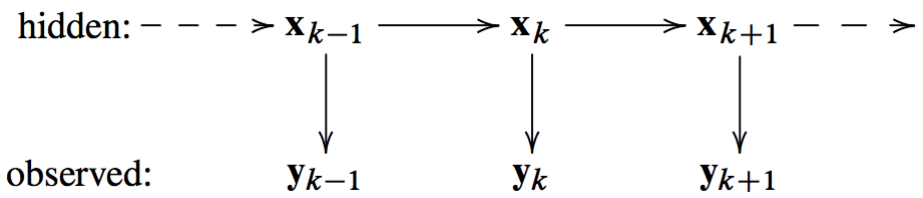


Figure 2.2: Hidden states observed through noisy measurements (Taken from [22])

The purpose is to estimate hidden states $X = \{x_0, ..., x_n\}$ from the observed measurements $Y = \{y_1, ..., y_n\}$. Which means that in the Bayesian sense, the joint posterior distribution of all states given all the measurements, needs to be calculated. This can be performed using Bayes rule:

$$P(X|Y) = \frac{P(Y|X)p(X)}{P(Y)}$$

Where:

- $P(X)$ is the prior distribution defined by the dynamic model.
- $P(Y|X)$ is the likelihood model for the measurements.
- $P(Y)$ is the normalization constant defined as:

$$P(Y) = \int P(Y|X)P(X)dX$$

The main drawback to this approach is that each time a new measurement is collected the full posterior distribution needs to be recalculated.

This can be avoided by restricting the class of dynamic models to probabilistic Markov sequences, thus removing the issue of having to compute the full posterior distribution at each time step. But this is outside the scope of this project, since the Kalman Filter approach also solves this problem with assumptions more suited to the nature of the main objective outlined in this thesis. This will be expanded upon in later chapters.

2.5 Probability Distributions

A probability distribution describes the probability of a given random variable to take any value within the sample space.

It is important to note that the probabilities for all values of a discrete random variable is known as a discrete probability distribution. On the other hand the probabilities for all values of a continuous random variable is known as a continuous probability distribution. The probability of each value must be grater than zero and the sum of all probabilities must equal one.

For a discrete random variable X the sum of all probabilities in the sample space can be written as:

$$\sum_{u} P(X = 1) = 1$$

For a continuous random variable:

$$\int_{u} P(X = 1)du = 1$$

2.5.1 Gaussians

A Gaussian or normal distribution is a continuous probability density function. It can be described entirely by two numbers, its mean (μ) and variance (σ^2). It is defined as:

$$f(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$$

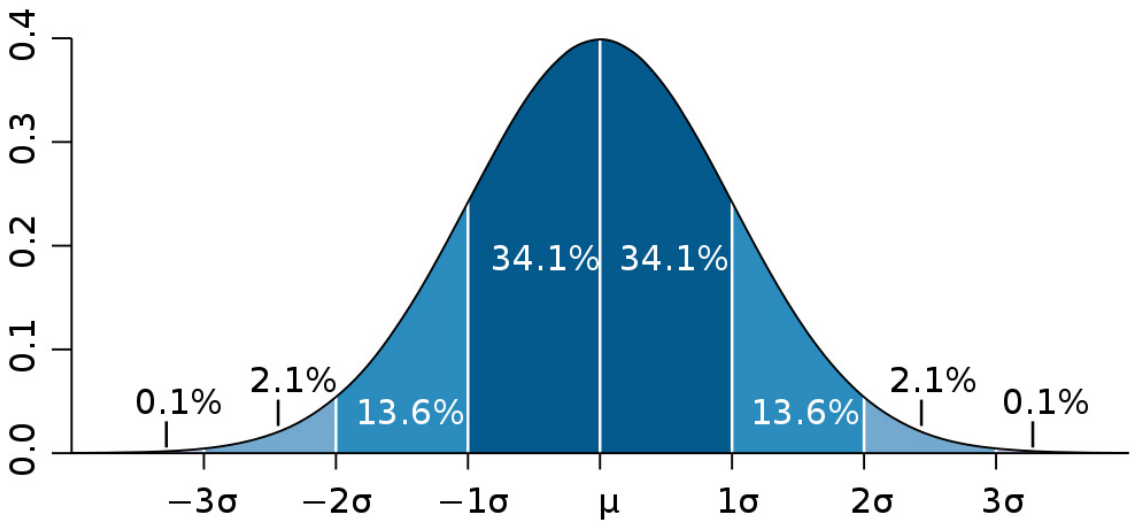


Figure 2.3: Bell curve of a Gaussian distribution taken from [19]

This type of probability density function appears in the real world all the time. Its shape can be seen in Figure 2.3. A vast amount of natural phenomena behaves according to this distribution.

The area under the curve gives the probability of the values selected. By calculating the integral from the starting point to the final point, the probability of those values can be found.

$$\int_{x_0}^{x_1} \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]dx$$

It is important to note that the notation for a normal distribution for a random variable X is:

$$X \sim N(\mu, \sigma^2)$$

Which means that an infinite number of possible values can be described by two numbers. Which makes the distribution and any operation performed with it, computationally efficient.

The standard deviation is a measure of how much the data deviates from the mean. As can be seen in Figure 2.3, for Gaussian distributions, 68% falls within one standard deviation ($\pm 1\sigma$) of the mean, 95% falls within two standard deviation ($\pm 2\sigma$) and 97.5% falls within three ($\pm 3\sigma$). A high variance can be interpreted as low certainty or very inaccurate belief.

An important property of Gaussian distributions is that the sum of two Gaussians results in a Gaussian as well.

Normally when multiplying two non linear functions, the result is a different non linear function. But in the case of multiplying two Gaussians, the result is a Gaussian too.

The product of two independent Gaussians is given by:

$$\mu = \frac{\sigma_1^2\mu_2 + \sigma_2^2\mu_1}{\sigma_1^2 + \sigma_2^2}$$
$$\sigma^2 = \frac{\sigma_1^2\sigma_2^2}{\sigma_1^2 + \sigma_2^2}$$

The sum of two Gaussians is given by:

$$\mu = \mu_1 + \mu_2$$
$$\sigma^2 = \sigma_1^2 + \sigma_2^2$$

The previous equations show how operations with Gaussians can be very easy to perform.

2.6 Statistical Learning

"The science of learning plays a key role in the fields of statistics, data mining and artificial intelligence, intersecting with areas of engineering and other disciplines" [5].

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E" [14].

Given a set of measurements, statistical learning refers to a set of approaches that try to estimate a function f which describes the behaviour of the data.

In other words, given a set of quantitative responses Y and p predictors $X = (X_1, ..., X_p)$, it is assumed that there is some relationship between them that can be written in the general form of:

$$Y = f(X) + \epsilon$$

Where ϵ is a random error term with mean zero.

This estimation is realized either to perform predictions about future values of the data or to perform inference, i.e. understand how the predictors relate to the response.

Most statistical learning problems fall into two categories: supervised and unsupervised.

2.6.1 Supervised

In the supervised scenario, for each observation there is an associated response. The purpose then is to find a model that describes the data, i.e. find an association between the observation and the response. This is done for the purpose of predicting future behaviour or to better understand the relationship between the observations and the response.

There are two main categories for supervised learning. These are regression and classification. The difference between them can be found in the type of response variable of the problem at hand. If the response variable is a quantitative one, a regression model is more suitable, on the other hand if the response variable is categorical, then a classification algorithm would be more appropriate.

2.6.2 Unsupervised

Unsupervised learning describes the scenario where no response is associated to the observations taken. It is called unsupervised because of the lack of a response variable that can supervise the analysis.

The tool to use in an unsupervised scenario is called cluster analysis. The goal in this case is to determine whether the observations fall into relatively distinct groups, i.e. partition the observations into clusters so that those assigned to the same cluster have a higher similarity than those assigned to a different one.

There are three types of clustering algorithms: combinatorial algorithms, mixture modeling and mode seeking. The focus here will be placed on combinatorial algorithms.

Combinatorial algorithms evaluate the data without a direct reference to an underlying probability model. In this approach, each observation is uniquely labeled by an integer $i \in \{1, ..., N\}$ and a pre specified number of clusters $K < N$ is determined and labeled with an integer as well $K \in \{1, ..., K\}$. Each observation is assigned to exactly one cluster using a similarity function.

2.7 Web Technologies and Methodologies

Web applications were initially built using a simple, monolithic architecture. At this time all of the interactions were done using HTML and Javascript for HTTP. As the

web grew larger, the need for integration and reuse grew as well, making the exposure of alternative ways of interacting with websites by providing different APIs, more popular. This increased even more with the rise in popularity of mobile applications in the late 2000s, which helped elevate APIs to become the go to solution of web development.

2.7.1 Service Oriented Architecture

"There are many benefits that come with web services, such as promoting reuse and higher level of abstractions" [1].

A service oriented architecture (SOA) consists in loosely coupled and highly autonomous services focused on solving business needs. It is preferred that all services have clearly defined contracts and use the same communication protocol.

Usually the API contract is designed and built first, followed by the development of all the clients that will consume it. This tackles the problem of having multiple interfaces for the same application, such as a mobile application, a desktop website, a mobile website etc... This can be seen better in Figure 2.4.

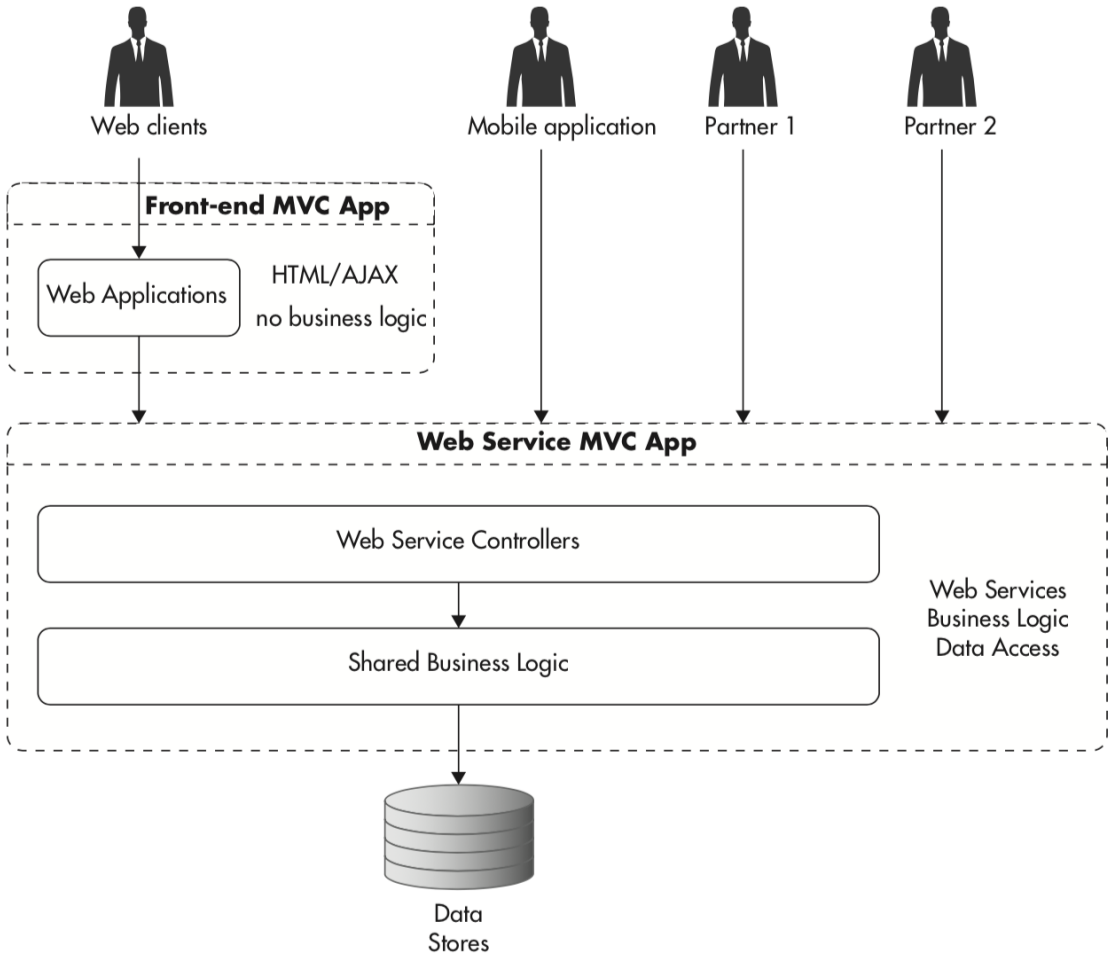


Figure 2.4: SOA architecture application with multiple clients taken from [1]

In Figure 2.4 all the clients use the same API when interacting with the application.

It is important to note that all of the business logic now resides inside of the web services layer, removing it from the clients. Which in turn makes the modification or replacement of clients easier since all they need to do is know how to consume the API.

From the scalability point of view, this approach promotes loose coupling and separation of concerns which helps in scaling clients and services independently of each other.

2.7.2 Multilayer Architecture

A multilayer architecture consists in dividing functionality into a set of layers. Lower level components expose APIs for the higher level components to consume. It is important that lower layers do not rely on functionality provided by higher layers. An example of this architecture can be seen in Figure 2.5.

Layers enforce structure and reduce coupling as lower level components become simpler and less coupled with with the rest of the system. It also allows for easy replacement of lower layer components as long as they fulfill the same API contract.

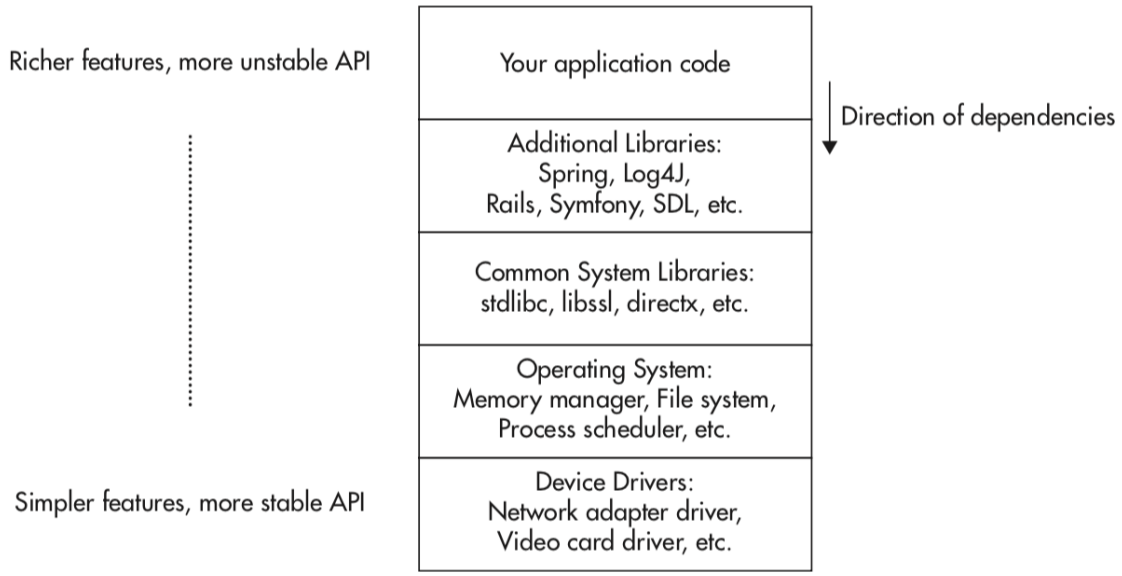


Figure 2.5: Multilayer architecture example taken from [1]

2.7.3 REST Services

Representational state transfer or REST is a resource oriented architectural style that was developed in the early 2000s. Its signature is its simplicity and lightweight which has made it the standard for today's web application integration.

REST services use URLs to uniquely identify resources. It uses HTTP methods to define the type of interaction the client will have with the resource.

- GET method is used to fetch information about a resource or its children.
- PUT method is used to replace an entire resource or a list by providing a replacement.

- POST method is used to create or update a resource.
- DELETE method is used to remove resources.

2.7.4 JSON

Javascript Object Notation is a lightweight data interchange format. It is self describing and easy to understand.

One of its main features is that it can be easily transferred to a web server due to the fact that it is text only. This also allows it to be used as a data format by any programming language.

Its structure is formed by:

- A collection of value/pairs
- An ordered list of values

Figure 2.6 shows an example of a JSON text.

```
{
  "array": [
    1,
    2,
    3
  ],
  "boolean": true,
  "color": "#82b92c",
  "null": null,
  "number": 123,
  "object": {
    "a": "b",
    "c": "d",
    "e": "f"
  },
  "string": "Hello World"
}
```

Figure 2.6: JSON example

Chapter 3

Architecture

This chapter expands a little on what the nature of the system's requirements are. For this, it is imperative to explore how the RSSI of the beacons behave over time in the desired setting.

3.1 Initial setup

For the roles of beacon and client, the NodeMCU esp32 development board was chosen. The esp32 are small, low cost, low power and very versatile micro controllers. As stated on the offitial website [3]:

The esp32 has a hybrid Wi-Fi and Bluetooth Chip, meaning it can perform as a complete standalone system or as slave to a host MCU. They feature ultra low power consumption since they are engineered for mobile devices, wearable electronics and IoT applications. They are also capable of functioning reliably in industrial environments, with temperatures ranging from -40^0C to $+125^0$C.

This is just a brief summary of the capabilities of the micro controller. The important take away from this is that they can perform as a BLE beacon as well as a BLE client, and also connect to Wi-Fi and send the measurements collected as an HTTP request.

The development boards have USB ports that can be used to power the micro controller. Each of the components used are powered with power banks using this port. They are attached to the chair using simple double sided tape. For the initial setup both beacons are placed one in front of the other underneath the chair. As shown in Figure 3.1.

This architecture is selected because the closer the beacon and the client are to each other, the greater the attenuation caused by a human body will be, making it easier to identify. This can be seen in Figure 3.2, where it can be observed that as the beacon moves further and further away from the client, the distortion caused by a human body decreases.

This is by no means a market ready solution. For the objective at hand a better design could have been developed but **the focus of the project lies in the analysis and processing of the RSSI** of the beacons, and for that, this design will do just fine.

The RSSI can be measured with the Friis transmission equation:

$$P_r = P_t * (G_t * G_r) * c^2/(4\pi Rf)^2 \tag{3.1}$$

Figure 3.1: Initial placement of beacon and client

Equation 3.1 gives the power received by an antenna from another antenna that is transmitting a known amount of power at a distance under ideal conditions.

In this setup, the beacon sends periodically to the client the RSSI between them, calculated with the Friis transmission equation. The client in turn sends the measurement to a web server via an HTTP request.

From now on the RSSI of the beacon will be referred to as 'the signal'.

3.2 The signal

The signal is measured in dBm, which means decibels relative to one milliwat. It is important to note that once the components are set in place, their position does not change. This means that the signal should remain constant as long as there is no noise and no one is seated on the chair.

Figure 3.3 shows the first 200 measurements taken from a beacon placed under the chair. It can be appreciated that the data is noisy and has some really pronounced peaks.

Figure 3.4 shows how the signal gets distorted when someone sits down on the chair. This distortion comes from the presence of a human body placed between the beacon and the client.

Figure 3.4 also shows that by learning how to identify this distortion, it would be possible to achieve the goal of proximity detection.

It is clear by looking at the previous mentioned figures, that it is important to try to reduce the noise from the signal as much as possible to differentiate between

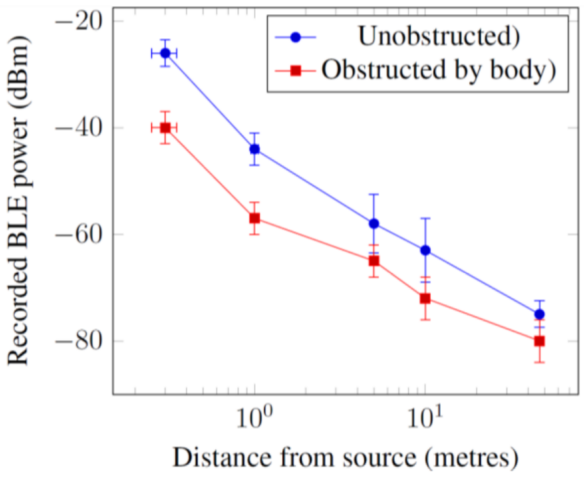


Figure 3.2: RSSI with and without body distortion (taken from [15])

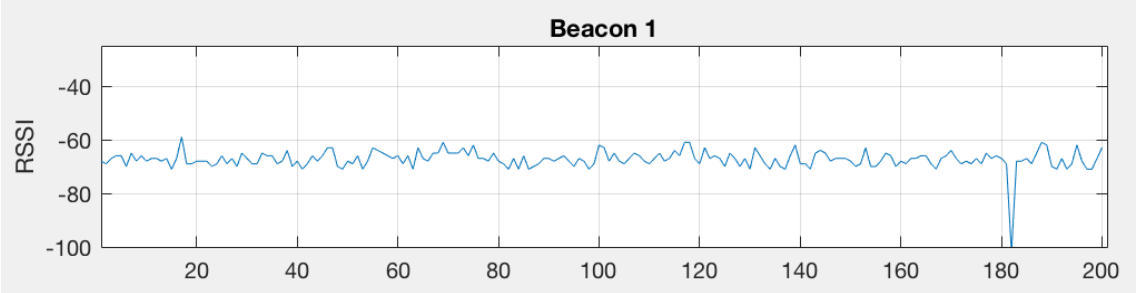


Figure 3.3: Signal of the Beacon

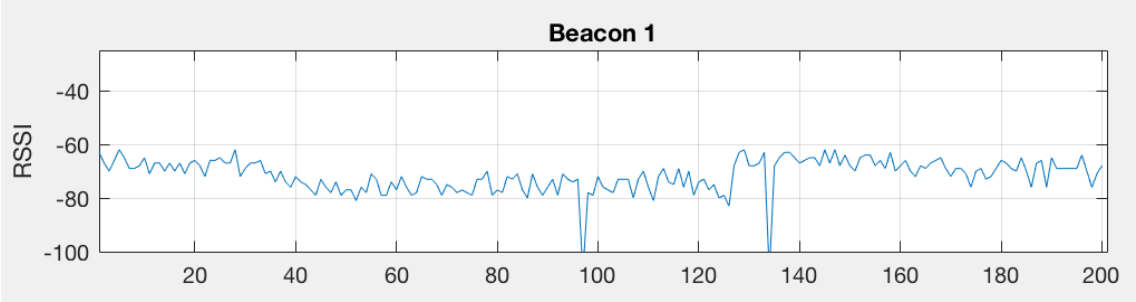


Figure 3.4: Signal of the Beacon when someone sits down

distortion by noise and distortion by human presence.

3.2.1 Managing the noise

A very noisy signal is difficult to analyze and will cause problems at the time of differentiating between someone being seated or not. As mentioned before, noise reduction is essential for the achievement of the main goal of the project.

For this objective, a Kalman Filter is implemented. Its purpose is to filter out the noise from the signal.

The Kalman filter is an algorithm that uses the state-space approach to filtering and takes a series of measurements observed over time to produce estimates of unknown variables. It combines the limited knowledge available on how a system

behaves with the noisy observations taken by the sensors to produce the best possible estimate of the state of the system.

It was developed by Rudolf Emil Kálmán, and it is used in a wide variety of fields. This is due to the fact that it is computationally light and very fast, which makes it a very powerful tool.

The filter as a tool for noise reduction can be proven to be very effective as it is shown in Marselli et al. [13]. Where different versions of Kalman filters are implemented to successfully reduce the noise of microsensor signals.

Another example of noise reduction using a Kalman filter can be found at Ki Hwan Eom et al. [2]. Here, it is used to reduce the noise in a multi sensor RFID system.

As the project moves forward, more beacons will be added to the chair in order to try to get a clearer picture of the signal. A very important feature of the filter is that it allows to merge the different measurements from the different beacons into one single state estimation with better accuracy. This is known as sensor fusion.

Examples of this are abundant in the literature, given that it is a major feature of the filter, and its applications are plentiful.

One of the examples can be found in Shu-Li Sun et al. [26]. Where a new multi sensor fusion criteria is presented and applied to a radar tracking system.

Another example of sensor fusion with Kalman filter can be found in J.Z. Sasiadek et al. [23]. In this work, the filter is applied to fuse position signals from the Global Positioning System (GPS) and Inertial Navigation System (INS) for the use of autonomous mobile vehicles.

3.2.2 Learning the behaviour

As mentioned before, the key part of the project is to identify or learn the distortion caused by a human's body being present between the beacon(s) and the client. For this task, a machine learning algorithm is required.

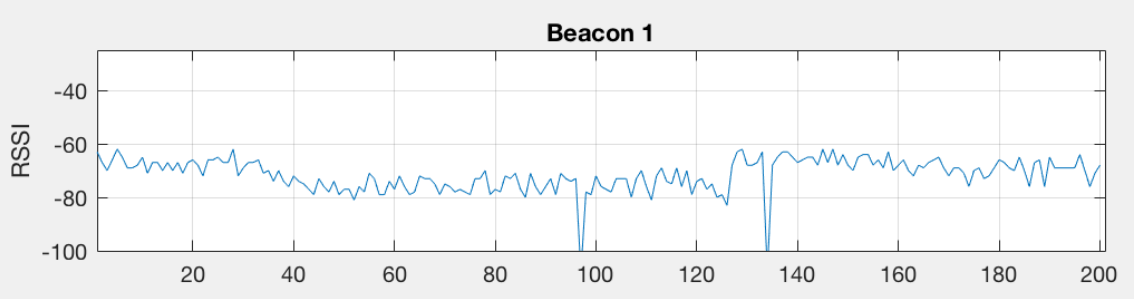


Figure 3.5: Distorted signal of the Beacon from chair 1

Figure 3.5 and Figure 3.6 show the effects of a person being seated in different chairs. It can be appreciated that the amount of distortion is different. In Figure 3.5 the distortion is around -10 dBm and in Figure 3.6 the distortion is around -20 dBm.

In this case, an unsupervised algorithm is best suited to the project. This is because of the nature of the problem at hand, where each chair has a different structure, and therefore the amount of distortion caused by a person sitting down is different for each one.

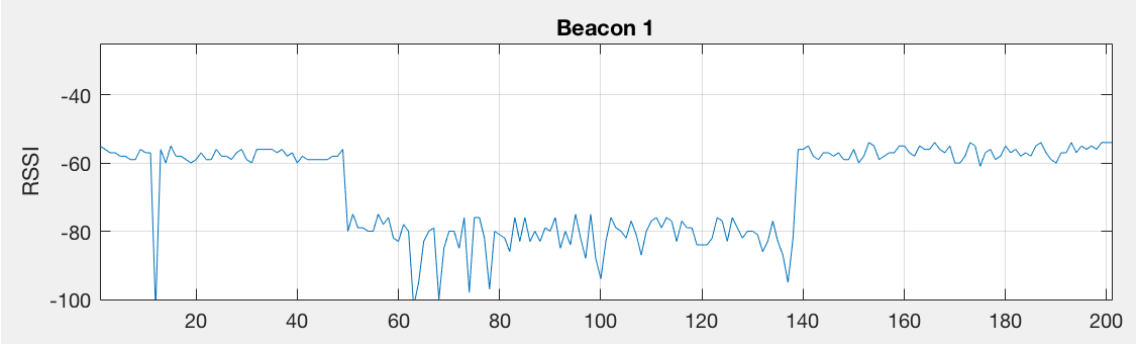


Figure 3.6: Distorted signal of the Beacon from chair 2

This means that for a supervised learning algorithm, a training set must be elaborated with samples taken for when no one is seated and for when someone is, more importantly, this must be performed for every chair the system needs to be installed in, given that the environment also plays a role. This is not practical, nor feasible at a large scale when there are multiple chairs in play.

The advantages of an unsupervised algorithm are that not only it does not need a training set, but it will get better over time. This is not the case for the supervised one, which after the training phase, its performance will not increase anymore.

Another thing to consider is that the problem of identifying if someone is seated or not, is really a classification problem. That is, given a set of inputs a prediction of a qualitative variable, whether someone is seated or not, is required. Predicting a qualitative response for an observation can be referred to as classifying that observation, since it involves assigning the observation to a category.

There are several unsupervised learning methods available in the literature but the one chosen for this project, thanks to its simplicity, is K-means clustering.

The K-means cluster algorithm falls into the combinatorial classification and there are plenty of examples of the algorithm being used in the literature given that it has been around for over 50 years since its inception. One example can be found at Kanungo et al. [11]. Another one can be found at H.P. Ng et al. [17], where it is used for image segmentation.

For the purposes of this project, the algorithm will be used to assign the observations coming from the BLE signal to its respective cluster. It is a rather straightforward application except for the fact that the algorithm will be applied over an online system, i.e. new measurements will keep arriving and the algorithm must be able to process one observation at a time. For this, a recursive version is necessary.

This recursive version is known as "Online K-means" and its implementation can be seen in [12]. Another example of a slight modification in the algorithm so that it is more suitable in high traffic scenarios can be seen in [24]. Providing evidence of the possibility of using the algorithm for the purposes intended in this project.

3.2.3 Web API and interface

A web application programming interface (API) will receive the measurements sent from the BLE client. Every measurement will be stored in a database.

Data visualization has become very important in today's world. This is the reason a web interface is developed to manage and monitor the system and its

performance.

The web interface will communicate with the web API and display all the information in an effective way, to allow for proper analysis.

Further details about the web API and interface can be found at a later chapter in this document.

3.3 Final architecture

Putting together all the pieces previously described, the final architecture as shown in Figure 3.7 is composed of:

- 1 or more beacons
- 1 client
- Kalman Filter
- Online K-means Algorithm
- Web API
- Web interface

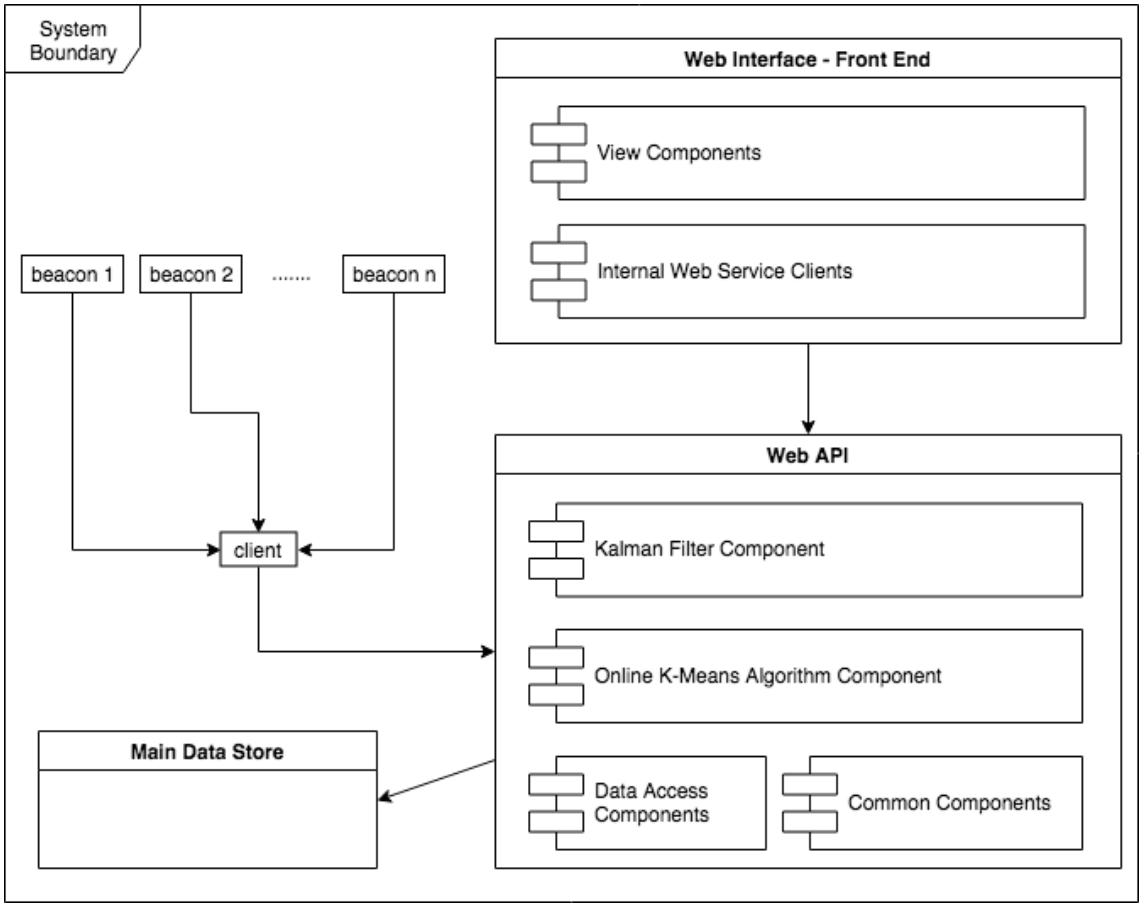


Figure 3.7: Final architecture of the system

The client, after getting the measurement from the beacon(s), sends it to the Web API, where it is processed by a Kalman filter to reduce the noise as much as possible and merge all the measurements together. The output of the filter is then fed to the machine learning algorithm so that it learns the behaviour of the signal and perform the predictions about whether someone is seated or not. This is all displayed by the web interface.

Chapter 4

Methods for Identification

This chapter contains a description of all the tools used to perform the identification process. It expands on the theory behind each one, how they are implemented and how they are used in the system.

4.1 Optimal Kalman filter (r=0)

This project will tackle the basic filtering problem which consists of having both variables $r(t)$ and $s(t)$, shown in Figure 2.1 coincide.

The following is taken from [20].

The Kalman Filter can be considered part of the Bayesian filter family. The filter takes the state-space approach to filtering.

Consider the following stochastic system:

$$S : \begin{cases} x(t+1) & = Fx(t) + v_1(t) \\ y(t) & = Hx(t) + v_2(t) \end{cases}$$

Two noise terms appear in the formulation.

- $v_1(t)$ which corresponds to the noise affecting the process.
- $v_2(t)$ which corresponds to the noise affecting the measurements.

Where S is the classic linear system depicted in Figure 2.1.

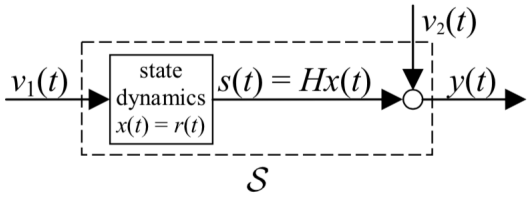


Figure 4.1: Kalman filter approach (Taken from [20])

With the following assumptions:

- $v_1(.) \sim WGN(0, V_1)$ where $V_1 \succeq 0$
- $v_2(.) \sim WGN(0, V_2)$ where $V_2 \succeq 0$
- $v_1(.)$ and $v_2(.)$ are independent

- $x(1) \sim G(0, P_1)$ where $P_1 \succeq 0$

As mentioned in the previous chapter, the Kalman filter is very good at estimating the true value of the signal and removing the noise from it.

Important thing to note about the filter is that it relies upon the assumption that all the inputs come from a Gaussian distribution.

For the design of the filter the signal will be treated as a Gaussian distribution and so will be the noise.

It is important to note that even though the Gaussian distribution comes up in nature all the time, not everything is described by it. It is perfectly possible to have a phenomenon that does not belong to a Gaussian distribution. Although, quoting the central limit theorem which states, "when independent random variables are added, their properly normalized sum tends toward a normal distribution" [29] and since the nature of the project consists on analyzing large amounts of measurements of the signal, it is safe to assume that the central limit theorem will hold.

Before expanding on how the filter is designed, a short digression on the role Gaussians play in the calculations of the filter comes first in this chapter. The following borrows heavily from this great book [10].

Figure 4.2 shows the Gaussian distribution generated from the beacon data presented in Figure 3.3. Its mean and its variance are $\mu = -67.32$ and $\sigma^2 = 12.05$ respectively.

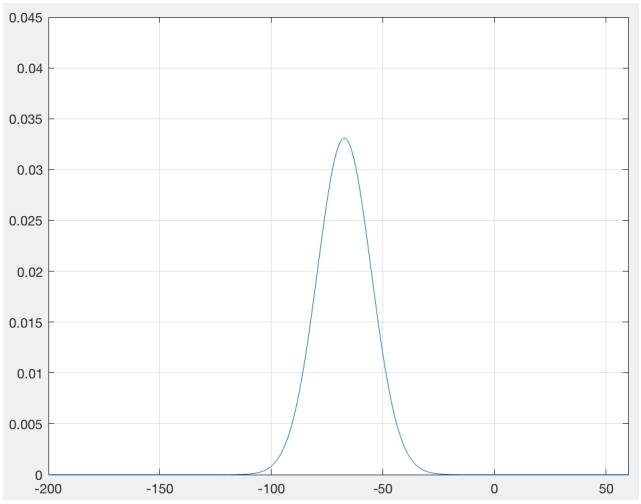


Figure 4.2: Gaussian distribution generated from Figure 3.3

One way to interpret Figure 4.2 is to look at the height of the bell curve. Since the area under the curve is the probability of the signal having a certain value, it is possible to infer that it is very likely that the signal will take on the value of -67.32 dBm given that the highest point of the bell sits in that value, in contrast with, for example a value of -100 dBm.

Figure 4.3 shows the Gaussian distribution generated from the beacon data presented in Figure 3.4. Its mean and its variance are $\mu = -71.52$ and $\sigma^2 = 37.60$ respectively.

The first thing to observe in Figure 4.3 is that the bell curve is lower, which means there is less certainty about the value of the signal. This is due to the higher variance of the data. The distribution is more spread out over all the possible values.

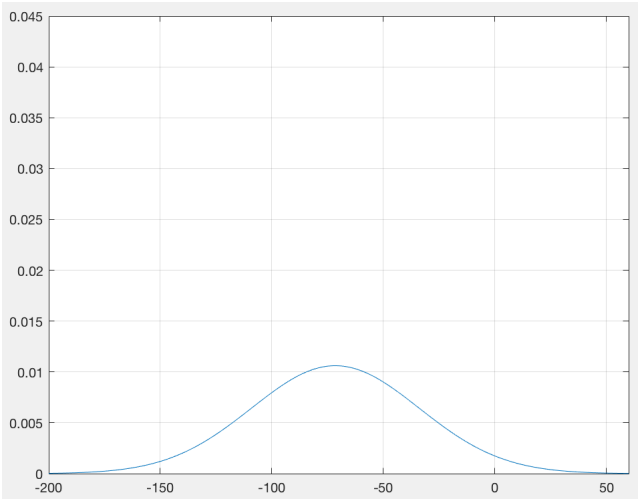


Figure 4.3: Gaussian distribution generated from Figure 3.4

A key property of this type of distribution to take into account is that the sum of two Gaussians is a Gaussian as well. The same applies for the multiplication of two Gaussians.

Imagine that Figure 4.3 represents the belief the system has over the current value of the signal at time t as a new measurement arrives, and Figure 4.2 represents the belief that the system had over the previous value of the signal at time $t-1$.

There is a way of using the previous belief to improve the current one. The way to do that is to multiply the Gaussians together. The result of the multiplication can be observed in Figure 4.4.

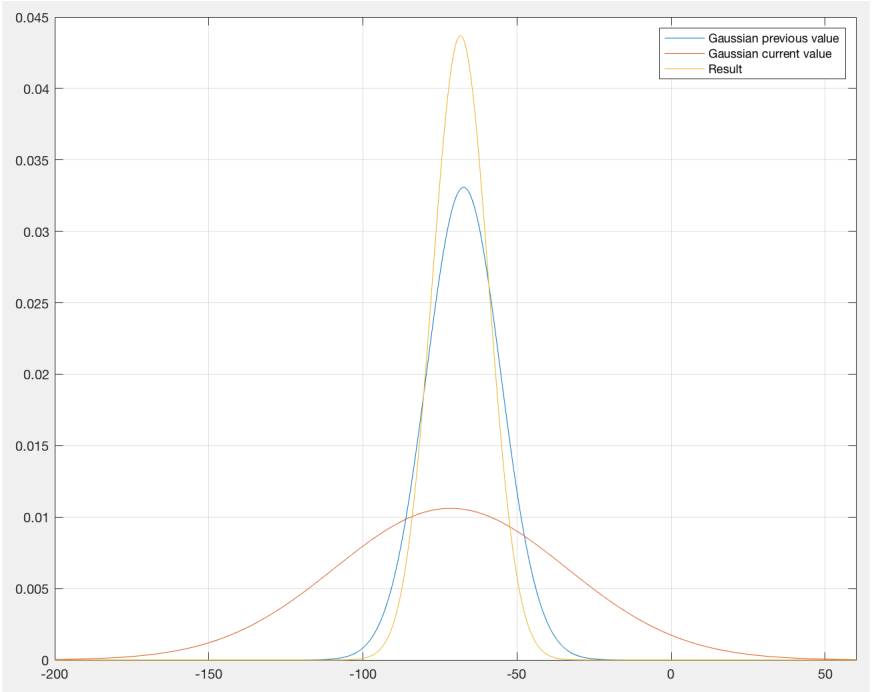


Figure 4.4: Result multiplication of Gaussians

The major thing to take away from Figure 4.4 is that the resulting Gaussian, shown in yellow, has a higher bell curve than the other two. This means that the

belief in the new resulting value is higher.

At its core, this is what the Kalman filter does, recursive operations with Gaussians. And this is only possible due to the properties mentioned before, that the sum and multiplication of two Gaussians result in another Gaussian.

Each Gaussian can be described completely by only two values, its mean and variance. Because of this, performing calculations with Gaussians is very efficient, which is the main reason why the Kalman filter is computationally feasible.

Taken into consideration everything described before, there will be several assumptions made about the nature of the signal and the noise.

For the design of the filter the signal will be treated as a Gaussian distribution and so will be the noise.

4.1.1 Derivation of the Filter

As mentioned before, this project makes use of the basic filtering setting, so the optimal Kalman filter is needed.

The main objective is to estimate the state $x(t)$ ($r(t)$ in Figure 2.1) based on $y^t = \begin{bmatrix} y(t) & y(t-1) & \dots & y(1) \end{bmatrix}^T$ ($s(t)$ in Figure 2.1)

Here is the derivation of the filter, taken from [20]:

$$\begin{aligned} \hat{x}(t|t) &= E[x(t)|y^t] = E[x(t)|y^{t-1}, y(t)] \\ &= E[x(t)|y^{t-1}] + E[x(t)|e(t)] \\ &= \hat{x}(t|t-1) + E[x(t)|e(t)] \end{aligned}$$

Here $e(t) = y(t) - \hat{y}(t|t-1)$, is the innovation carried by the observation at time t. Obtained as the difference between the observation and its optimal estimate given the past data up to time t.

The term $E[x(t)|e(t)]$ can be calculated using Bayes formula:

$$E[x(t)|e(t)] = \Lambda_{x(t)e(t)} \Lambda^{-1}_{e(t)e(t)} e(t) \tag{4.1}$$

There are two terms in Equation 4.1 that need to be calculated, more specifically:

$$\Lambda_{x(t)e(t)} \tag{4.2}$$
$$\Lambda_{e(t)e(t)} \tag{4.3}$$

Equation 4.2 can be derived as follows:

$$\begin{aligned} \Lambda_{x(t)e(t)} &= E[x(t)e(t)^T] \\ &= E[x(t)(y(t) - \hat{y}(t|t-1))^T] \\ &= E[(x(t) - \hat{x}(t|t-1) + \hat{x}(t|t-1))(Hv(t) + v_2(t))^T] \\ &= E[(v(t) + \hat{x}(t|t-1))(Hv(t) + v_2(t))^T] \\ &= E[v(t)v(t)^T]H^T + E[v(t)v_2(t)^T] + E[\hat{x}(t|t-1)v(t)^T]H^T + E[\hat{x}(t|t-1)v_2(t)^T] \\ &= P(t)H^T + \cancelto{0}{E[v(t)v_2(t)^T]} + \cancelto{0}{E[\hat{x}(t|t-1)v(t)^T]H^T} + \cancelto{0}{E[\hat{x}(t|t-1)v_2(t)^T]} \\ &= P(t)H^T \end{aligned}$$

Equation 4.3 can be derived as follows:

$$\begin{aligned}\Lambda_{e(t)e(t)} &= E[e(t)e(t)^T] \\ &= E[(y(t) - \hat{y}(t|t-1))(y(t) - \hat{y}(t|t-1))^T] \\ &= E[(Hv(t) + v_2(t))(Hv(t) + v_2(t))^T] \\ &= HE[v(t)v(t)^T]H^T + \cancelto{0}{HE[v(t)v_2(t)^T]} + \cancelto{0}{E[v_2(t)v(t)^T]H^T} + E[v_2(t)v_2(t)^T] \\ &= HP(t)H^T + V_2\end{aligned}$$

In the end the following is obtained:

$$\begin{aligned}E[x(t)|e(t)] &= \Lambda_{x(t)e(t)}\Lambda^{-1}_{e(t)e(t)}e(t) \\ &= P(t)H^T[HP(t)H^T + V_2]^{-1}e(t)\end{aligned}$$

$K_0(t) = P(t)H^T[HP(t)H^T + V_2]^{-1}$ is known as the Kalman gain.

So the optimal filter ends up as:

$$\hat{S_0} : \hat{x}(t|t) = \hat{x}(t|t-1) + K_0(t)e(t) \tag{4.4}$$

The term $P(t)$, which is the variance of the state estimation error, needs to be derived recursively as well. For which a recursive expression it is needed for the state estimation error $v(t)$.

$$\begin{aligned}v(t+1) &= x(t+1) - \hat{x}(t+1|t) \\ &= Fx(t) + v_1(t) - F\hat{x}(t|t-1) - Ke(t) \\ &= Fv(t) + v_1(t) - K(t)[Hv(t) + v_2(t)] \\ &= [F - K(t)H]v(t) + v_1(t) - K(t)v_2(t)\end{aligned}$$

Now this expression can be used to derive the recursive expression for $P(t)$.

$$\begin{aligned}P(t+1) &= E[v(t+1)v(t+1)^T] \\ &= E[([F - K(t)H]v(t) + v_1(t) - K(t)v_2(t))([F - K(t)H]v(t) + v_1(t) - K(t)v_2(t))^T] \\ &= [F - K(t)H]P(t)[F - K(t)H]^T + V_1 + K(t)V_2K(t)^T\end{aligned}$$

The recursive equation for $P(t)$ is a Difference Riccati equation (DRE).

From this derivation, all the variables needed in order to implement the Kalman filter, are identified:

- x: the set of state variables to be estimated.
- F: state transition matrix.
- P: co-variance of the state estimation error.
- V_1: process noise matrix.
- y: the set of measurements collected.
- H: measurement function.
- V_2: measurement noise matrix.

4.1.2 Design of the filter

With the identified parameters, it is now only a matter of choosing their value so that the filter can perform the function it is needed for.

State variable

The signal contains one observed variable, which is the RSSI of the beacon(s). This is measured in dBm. This would constitute the state variable $\mathbf{x}$ of the filter.

$$\mathbf{x}(t) = [x_1(t)]$$

State transition matrix

The state transition matrix $\mathbf{F}$ is the matrix that will transform the previous state into the new state.

The combination of $\mathbf{Fx}$ is the model that will describe the behaviour of the data. In this project, the beacons are held in place, they do not change positions, therefore they will always be at the same distance with respect to the client. Which means that the RSSI should be constant when nothing is interfering with it. Therefore:

$$\mathbf{x}(t) = \mathbf{x}(t-1) \tag{4.5}$$

Equation 4.5 is the equation that determines how the model will behave. Since there is only one state variable, in order to model the signal to be a constant line, the state transition matrix will be $\mathbf{F} = 1$, so that:

$$\hat{\mathbf{x}}(t) = \mathbf{F}\mathbf{x}(t-1)$$

conforms with Equation 4.5.

Covariance of the state prediction error

The covariance of the state prediction error P is the Discrete Riccati Equation (DRE) and it is calculated recursively by the filter as just derived in the previous section. But it still needs to be initialized. This is usually done as:

$$P(1) = P_1 = var[x(1)]$$

The initialization of P is dictated by how much confidence there is in the initial value chosen for the state variables. A good initial estimate translates into a small value for $P(1)$.

Process noise matrix

The process noise matrix V_1 will be a scalar because there is only one state variable chosen for the filter. This matrix denotes how much the model changes between steps of the algorithm.

A high process noise means that the filter will follow the measurements more closely. Likewise, if the process noise is small, this means there is a lot of certainty in the model, and the filter will assign more weight to the estimates at the time of

updating the new state. This is done through the Kalman gain, which contains the term V_1 in the equation of its calculation.

Since there is a lot of confidence in the model chosen before, given that the signal should be constant, a small value to V_1 will be assigned. In this case 0.01 is small enough.

This seems like an arbitrarily chosen value, but it will do for now. In further sections the selection of this value will be expanded upon.

Set of measurements (y)

The vector of measurements $\mathbf{y}$ collects all the RSSI of all the beacons. In this case, just one.

$$\mathbf{y}(t) = [y_1(t)]$$

Measurement function

The measurement function $\mathbf{H}$ defines how to move from the state variables to the measurement space.

So that when $\mathbf{Hx}$ is performed, the result is in the same space as $\mathbf{y}$. Important to note that this is needed to calculate the innovation $e(t) = y(t) - H\hat{x}(t|t-1)$.

For now the model will remain for the the case when there is only one beacon and one client, i.e., one measurement and one state variable. The relationship between the state and the measurements, since both measure the same variable (RSSI), is described by the following equation:

$$y_1(t) = x_1(t) + v_2(t) \tag{4.6}$$

And for that, the measurement function is $\mathbf{H} = 1$.

$$\mathbf{y}(t) = \mathbf{Hx}(t) + v_2(t)$$

so that it conforms with Equation 4.6.

Measurement noise matrix

The measurement noise matrix V_2 describes the variance of the measurements. As stated before the focus for now will be in the case for only one beacon operating, therefore the measurement noise matrix becomes a scalar, i.e., the variance (σ^2) of the signal of the beacon.

The system will have a calibration phase where a large enough amount of measurements will be taken before the filter enters operation. This measurements will be used to calculate the variance of the signal.

4.1.3 Results for the single beacon filter

With all the parameters defined, the single beacon filter is implemented.

- $x_0 = y(0)$ the state variable is initialized with the first measurement.
- $\mathbf{F} = 1$.

- $\mathbf{P}_1 = 4$ since the initialization is performed with the first measurement there should be a certain amount of confidence in it.
- $V_1 = 0.01$ there is a lot of certainty in the model.
- $\mathbf{H} = 1$.
- $V_2 = 5.98$ variance calculated from the first 100 measurements collected of the beacon's signal shown in Figure 3.3. This is done to emulate the calibration part of the system which will collect a predefined amount of measurements to calculate the parameters of the filter.

Figure 4.5 shows the results for the data shown in Figure 3.3.

It is important to note in Figure 4.5 that the filter does a good job of eliminating the noise from the signal, and since the process noise is set to a small amount $V_1 = 0.01$, it follows the model more closely leaving the state estimation as a semi constant line, which is the desired behaviour.

Another thing to note in Figure 4.5 is the variance of the state estimation error. It converges to ~ 0.249 which indicates that there is confidence in the state estimate.

Bad initialization

Figure 4.6 shows what happens when the filter is set with a bad initial estimate for the state variable $x_0 = 0$. Important to note that since a bad initialization is chosen, the initial value of $P(1)$ is increased to 10. It also shows that the filter still finds the signal very quickly and P converges again to ~ 0.249.

This is an important characteristic of the filter. Specially given the nature of the problem of this project. This solution should work independently of the chair chosen, and the value of the signal can differ significantly depending on the structure of the chair. The fact that the filter can still find that value regardless of a bad initialization is an important step forward.

4.1.4 Sensor fusion

In a problem that handles uncertainty it is always a good idea to try to include more information when possible. In this case more beacons.

The filter provides a way to combine measurements from different sensors. Results from the previous section were taken using a single beacon architecture, from now a second beacon is added. In order to do this there are some filter's parameters that need to be adjusted. Now that there is a second measurement:

$$\mathbf{y}(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix}$$

The new measurement has the same relationship with the state variable that was defined in the previous section.

$$y_1(t) = x_1(t) + v_2(t)$$
$$y_2(t) = x_1(t) + v_2(t)$$

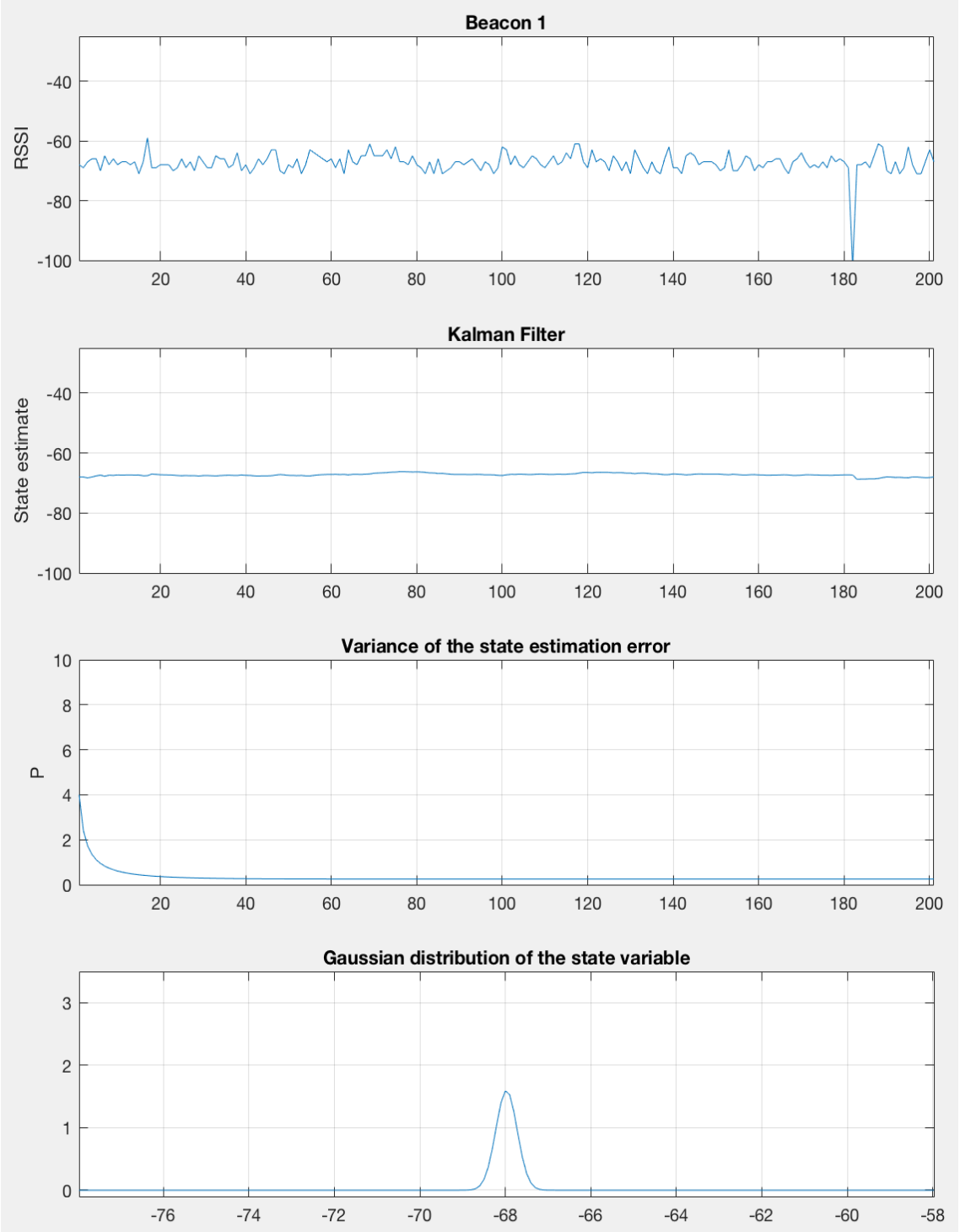


Figure 4.5: Single beacon Kalman Filter performance

This needs to be included in the filter. The way to do it is by adjusting the measurement function $\mathbf{H}$ so that it transforms $\mathbf{x}$ which is a vector of length 1, into a vector $\hat{\mathbf{y}}$ of length 2.

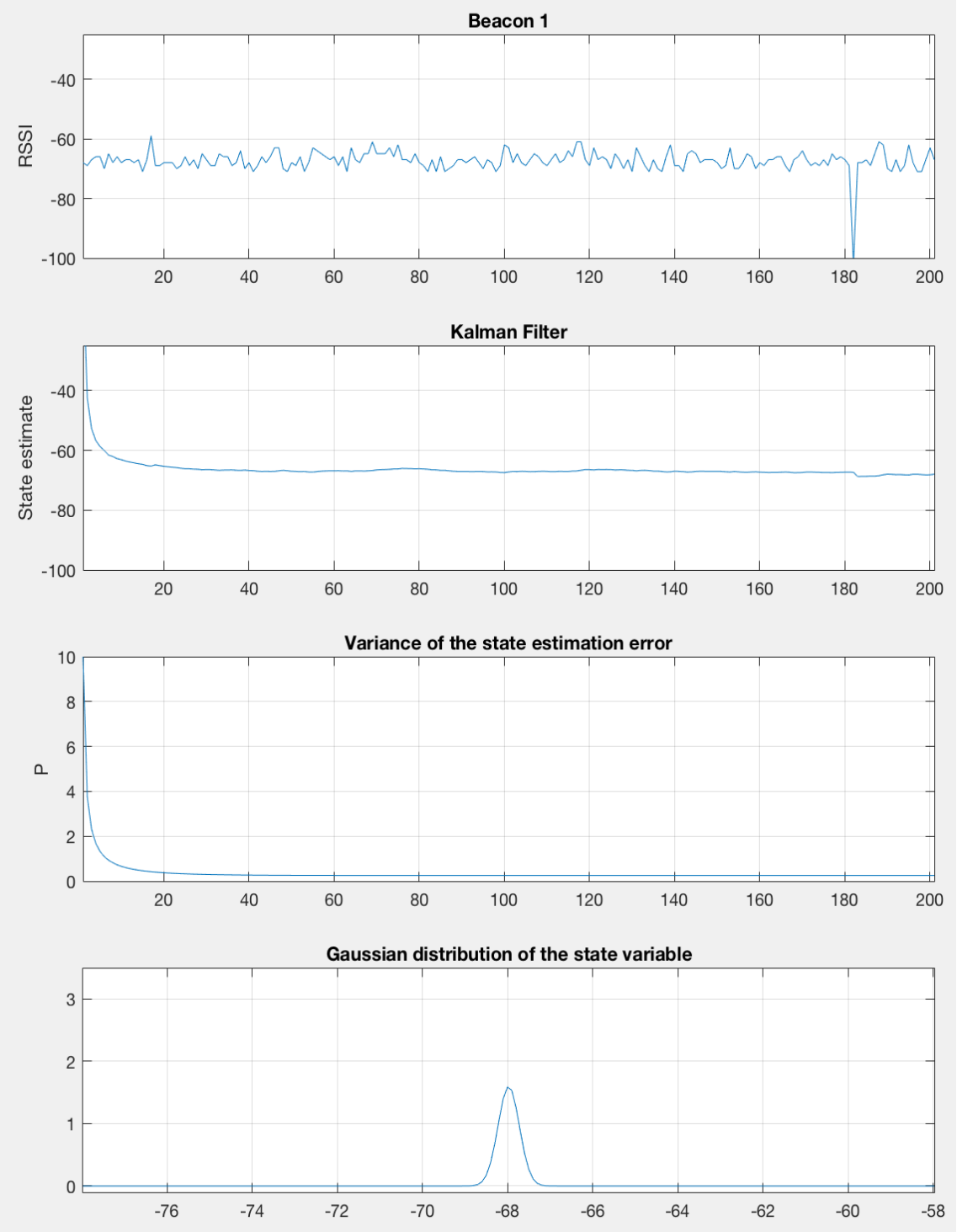


Figure 4.6: Single beacon Kalman Filter performance bad initialization

H will change to have the following structure:

$$\mathbf{H} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

so that

$$\hat{\mathbf{y}}(t) = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \hat{\mathbf{x}}(t)$$

When the innovation $e(t)$ is calculated, a subtraction between our current state estimate and each of the new measurements is performed.

$$e(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \end{bmatrix} \hat{\mathbf{x}}(t)$$

That is because both beacons measure the same variable, which is the RSSI. If this was not the case, $\mathbf{H}$ would need to be modified accordingly.

Another variable that needs to be adjusted is the measurement noise matrix V_2. Since now there are 2 measurements, the matrix becomes a 2x2, where the elements of the diagonal are the variance for each beacon, and the off diagonal slots contain the co-variance between them.

$$\mathbf{V}_2 = \begin{bmatrix} \sigma_1^2 & \sigma_{12}^2 \\ \sigma_{21}^2 & \sigma_2^2 \end{bmatrix}$$

Data was gathered for a second beacon. The first 200 measurements are shown in Figure 4.7

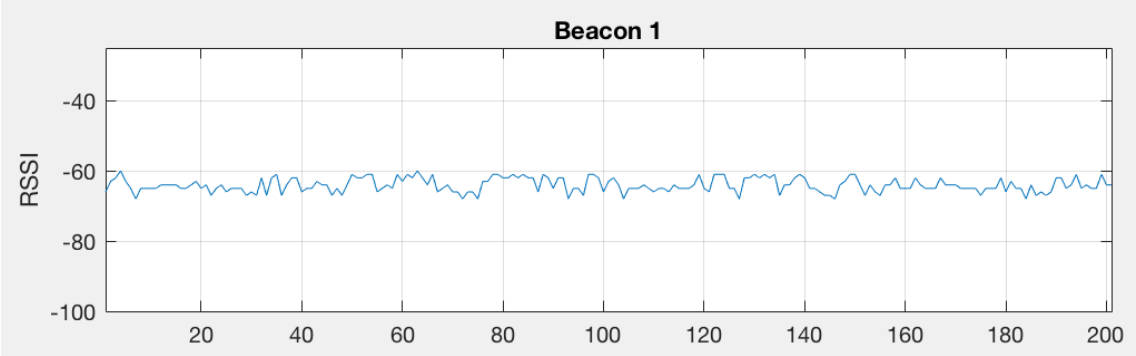


Figure 4.7: Signal of the second Beacon

Calculated from the first 100 measurements of the data obtained for both beacons, from Figure 3.3 and Figure 4.7, the matrix ends up like this:

$$\mathbf{V}_2 = \begin{bmatrix} 4.42 & -1.52 \\ -1.52 & 5.98 \end{bmatrix}$$

It makes sense that the co-variance between the signals is not zero, given that one of the variables that distorts it are others 2.4GHz signals.

With this new set of parameters, here is how the filter performs:

- $x_0 = y_1(0)$ the state variable is initialized with the first measurement of the first beacon.
- $\mathbf{F} = 1$.
- $\mathbf{P}_1 = 4$ there is still a moderate amount of confidence in the state variable initialization.
- $V_1 = 0.01$ there is still a lot of certainty in the model.
- $\mathbf{H} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$.

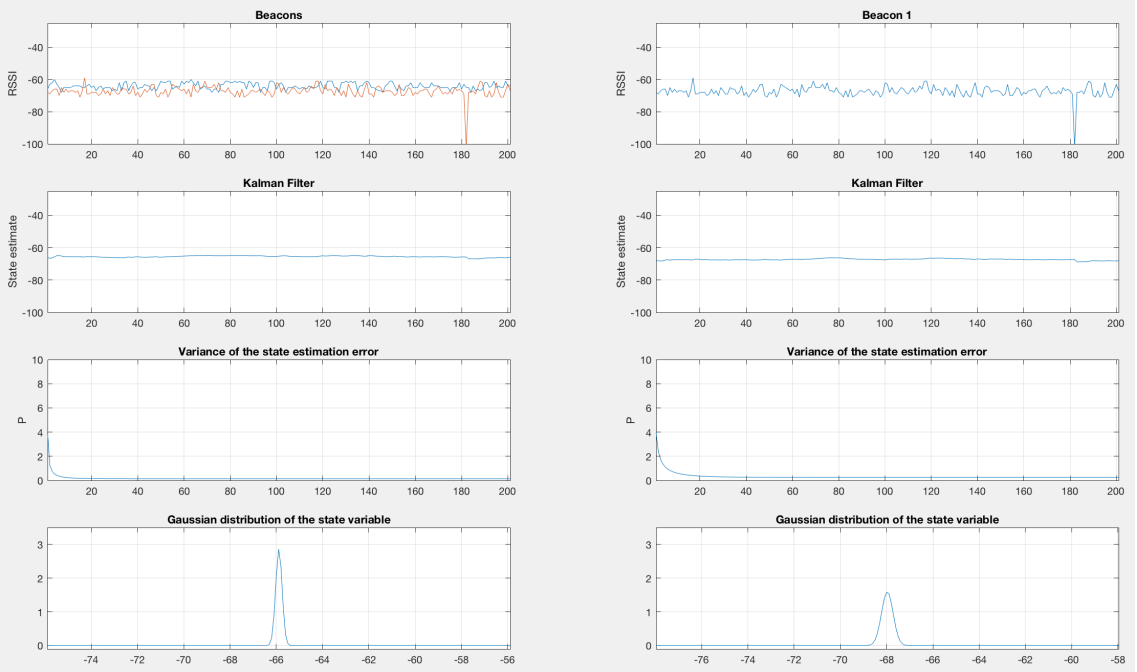


Figure 4.8: Implementation with multiple beacons (left) vs single beacon (right)

- $V_2 = \begin{bmatrix} 4.42 & -1.52 \\ -1.52 & 5.98 \end{bmatrix}$.

Figure 4.8 shows side by side the sensor fusion implementation vs the single beacon one that was developed in the previous section. It is important to note that not only does the left filter converges faster, but it also achieves a lower variance of the state estimation error P (**$\sim$0.138** vs **$\sim$0.249**).

The new measurement has provided new information about the behaviour of the true signal and therefore improved the estimations.

In this case it is safe to say that **two beacons are better than one**.

Bad initialization

Again a bad initial estimate for the state variable $x_0 = 0$ is set and the initial value of $P(1)$ is increased to 10. Figure 4.9 shows how the sensor fusion filter performs in such conditions.

Figure 4.9 shows side by side the sensor fusion implementation vs the single beacon one in a bad initialization context. The sensor fusion filter also finds the signal very quickly in the event of a bad initialization. But again it is important to note that not only does the left filter converges faster, but it also achieves a lower variance of the state estimation error P (**$\sim$0.138** vs **$\sim$0.249**). The same values achieved with a good initialization and almost as fast as before.

4.1.5 Time varying signal

The results shown so far belong to the case when no one is seated and the signal is only altered by noise in the environment. Now when someone sits down in the chair, the behaviour of the signal changes. Its value decreases because there is a human body present between the beacons and the client as shown in Figure 3.4.

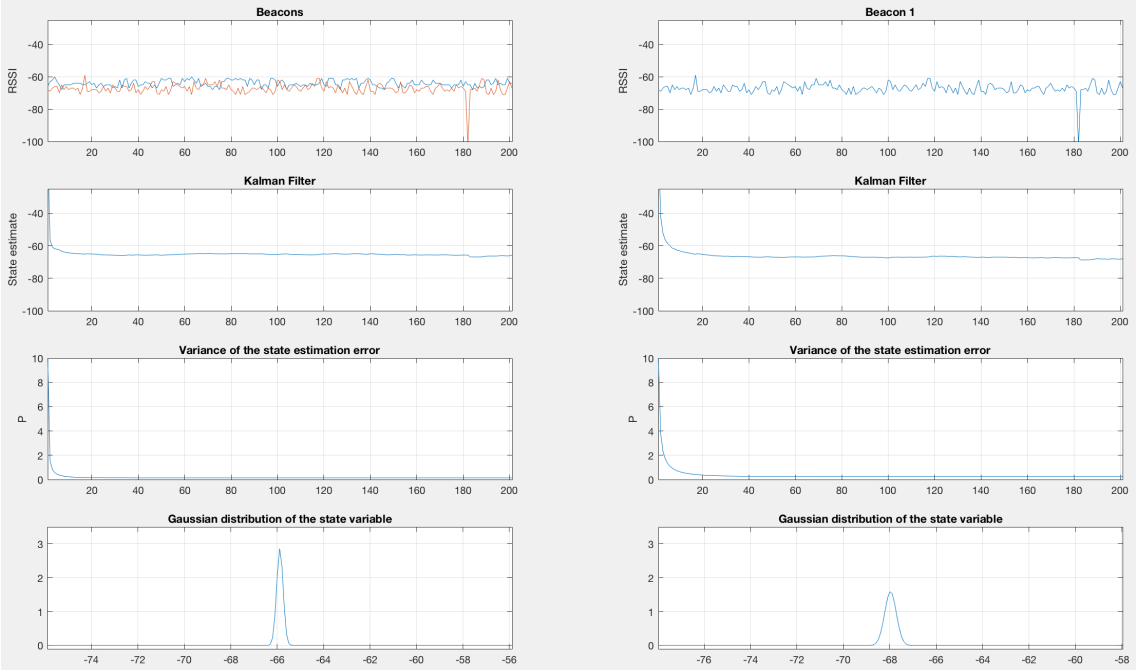


Figure 4.9: Multiple beacon Kalman Filter performance bad initialization

Figure 4.10 shows how the multiple beacon filter performs when the signal changes due to someone sitting down on the chair. It is important to observe that the filter lags behind the signal.

This is to be expected given that the process noise of the filter was set to a very low value ($V_1 = 0.01$). That way the filter still believes that the estimations of the model are much better than the measurements.

This is not good for the purposes of the problem at hand. It is imperative that the filter be more reactive to the changes in the behaviour of the signal, therefore the process noise needs to be adjusted accordingly.

Adjusting the process noise

To find a good value for the process noise, several experiments were performed. These experiments consisted in seeing how the filter behaves in the context of someone being seated, with a higher process noise.

First, the process noise is increased to $V_1 = 0.05$ and the results are shown in Figure 4.11

In this case the distortion is not as pronounced because of the structure of the chair. Therefore it does not affect the system as much, and the filter even though is behind the signal, it is not by a large margin.

Increasing $V_1 = 0.05$ does improve the filter's performance as can be seen in Figure 4.11, it is more reactive, but to get a better sense of the problem, a different data set will be used. This one was taken from a different chair where the distortion is more pronounced than in the one shown in Figure 3.4 and Figure 4.11. This can be seen in Figure 4.12.

In Figure 4.12 the lag of the signal can be better appreciated. In this example it does affect the system given that there is a big margin in the distortion of the signal, and it takes the filter around 25 iterations to find the new value. Figure 4.13

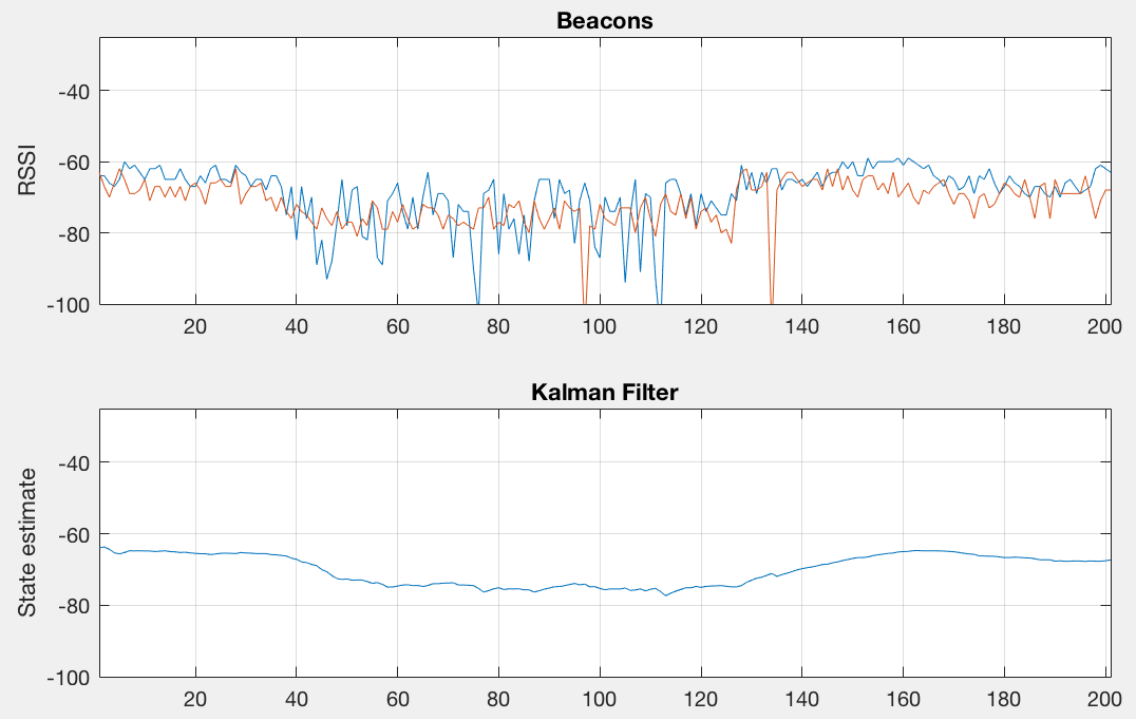


Figure 4.10: Performance of sensor fusion filter when someone sits down

shows the results of increasing V_1 to 0.05.

In Figure 4.13 the improvement of increasing the process noise can be greatly appreciated. It takes the filter around 10 iterations less to reach the new value of the signal.

It is important to note that with a higher process noise, the variance of the state estimation error also increases. This time converging to ~ 0.291 with $V_1 = 0.05$ vs ~ 0.123 when $V_1 = 0.01$. This is an increase of 236%.

Now V_1 is increased to 0.1.

In Figure 4.14 the improvement of increasing the process noise from $V_1 = 0.05$ to $V_1 = 0.1$ is marginal. Also the convergence value of P increased to ~ 0.428. This is an increase of 147% compared to when $P \sim 0.291$ for $V_1 = 0.05$. This increase is not really justified in terms of performance improvement, therefore a value of $V_1 = 0.05$ is selected.

There is one more thing to evaluate to see if the new value for V_1 is acceptable.

So far the filter has only been evaluated when the signal changes. With an increase process noise, the filter will follow the measurements more closely, which means that it will be more susceptible to noise. This is why it is important to evaluate the filter's performance in a noisy environment when no one is seated.

For this, a different data set will be used. The one shown in Figure 4.15.

Figure 4.15 shows a noisy signal when no one is seated. The spikes in the RSSI of the beacons are due to a noisy environment. The panel below shows the performance of the filter with $V_1 = 0.01$ vs a filter with $V_1 = 0.05$.

Even though the filter with $V_1 = 0.05$ is more susceptible to the spikes shown in Figure 4.15, it is not by a large margin compared to the filter with $V_1 = 0.01$.

In light of these results, it is safe to conclude that a process noise of $V_1 = 0.05$ is a good choice for the Kalman filter implemented in this project.

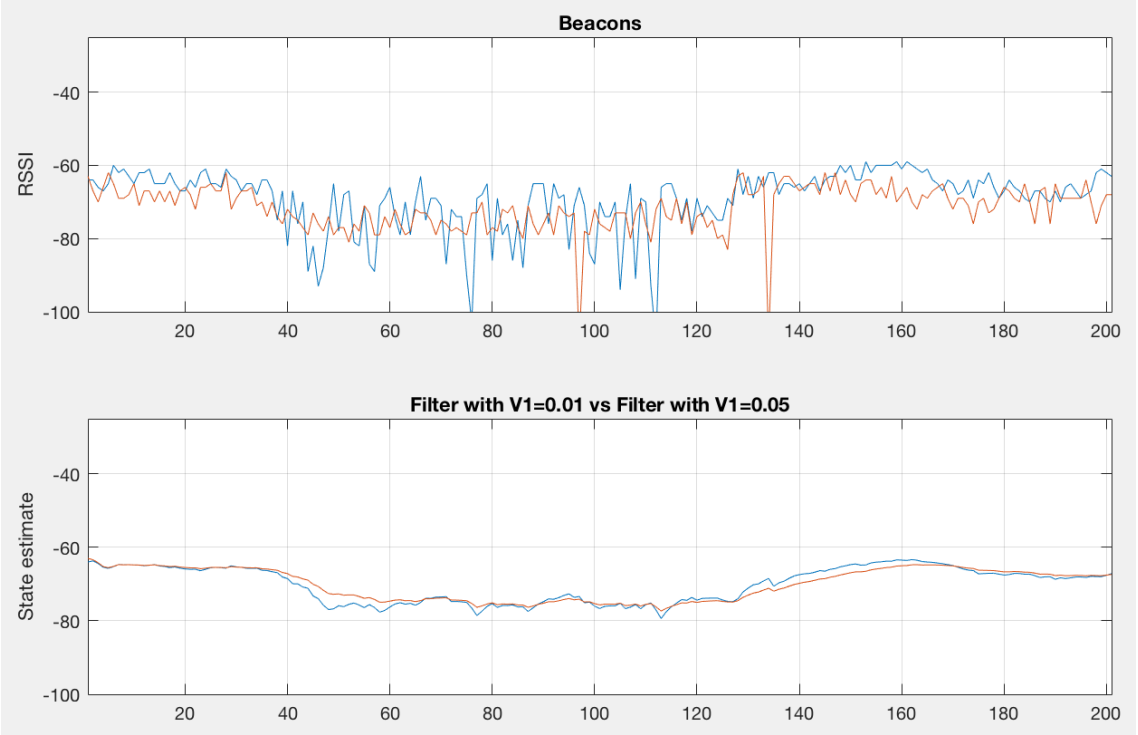


Figure 4.11: Sensor fusion filter with $V_1 = 0.05$ (blue) vs filter with $V_1 = 0.01$ (orange)

4.2 Unsupervised Learning Algorithm

The chosen algorithm must not affect the usability of the system, i.e. the calibration phase must be user friendly in a way that does not require too much effort and time from the users.

A calibration phase that is fast and straightforward is a key point that makes the difference between the system being feasible in a production environment or not.

Taking into consideration the previous points, a supervised learning algorithm would be out of the question. This is because a training set would be required. In this scenario, since the structure of the chair affects the distance between the beacons and the client and therefore the distortion of the signal caused by a human body, a training set would have to be created for each different model of chair on which the system is installed. On top of that, since the environment also plays a role in the amount of noise and because of this, in the variance of the signal, a training set would have to be created in the case that the chair changes environment to ensure best possible results.

If a supervised algorithm were to be chosen, the training set creation would have to be performed during the calibration phase, i.e. asking the user to sit down and stand up several times while data is collected. This is far from user friendly and when extended to multiple chairs in multiple environments, becomes a tiresome and sometimes impossible task.

It is clear that an unsupervised algorithm is the best possible choice in this case. An algorithm that requires no previous training and is able to learn in real time, therefore become better as more samples are collected, would make the calibration phase conform with the usability standard needed.

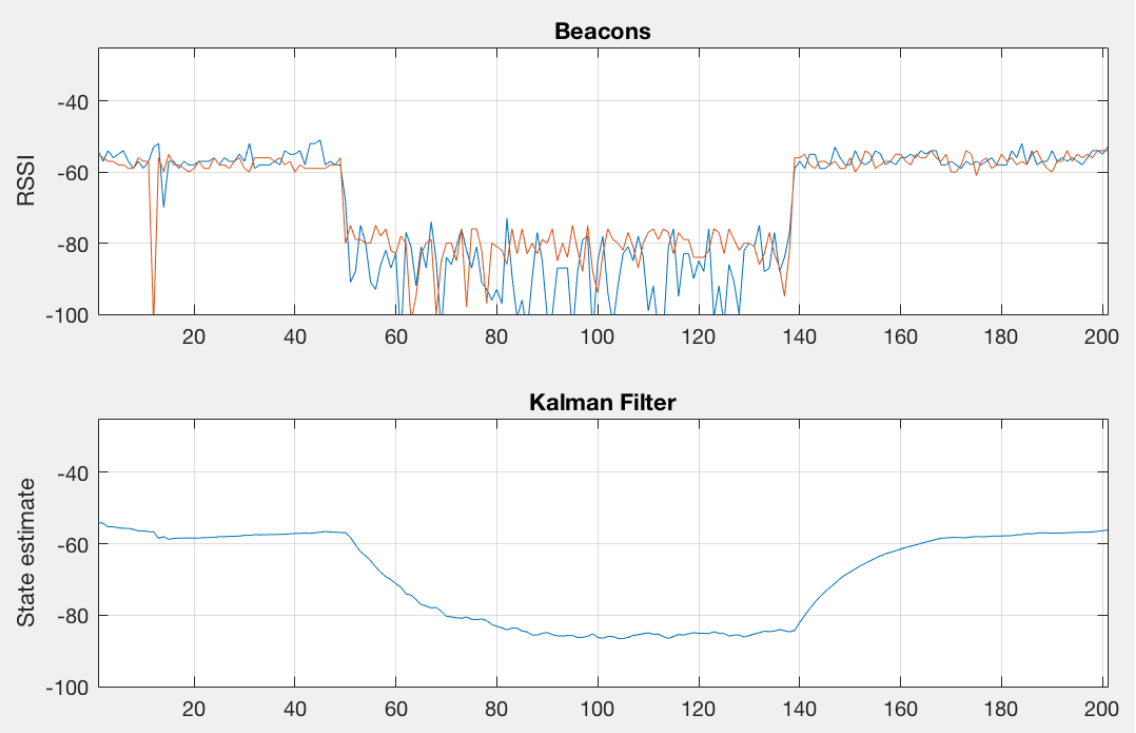


Figure 4.12: Performance of sensor fusion filter when someone sits down in chair 2

As the unsupervised machine learning algorithm for the system, k-means clustering was chosen due to all the points previously discussed.

4.2.1 K-means clustering

The following borrows heavily from this great book [5].

"The K-means algorithm is one of the most popular iterative descent clustering methods" [5]. It is designed for problems on which all the variables are quantitative.

A predetermined number of clusters $K < N$ is defined and each one is labelled with an integer $k \in \{1, .., K\}$. Each new measurement is assigned to its respective cluster using an *encoder* $k = C(i)$.

A mathematical loss function is then defined and further minimized through a combinatorial optimization algorithm. Given that the goal is to assign close points to the same cluster, the loss function becomes:

$$W(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(i')=k} d(x_i, x_{i'}) \tag{4.7}$$

This equation characterizes the extent by which observations assigned to the same cluster tend to be close to one another. It is referred to as the within cluster point scatter.

Cluster analysis by combinatorial optimization consists in minimizing W over all possible assignments of the N data points over the K clusters. The problem with this is that as N increases, the number of possibilities grows exponentially, making such optimization only feasible on small data sets. For this reason, practical clustering algorithms limit themselves to analyzing a small fraction of possible encoders $k =$

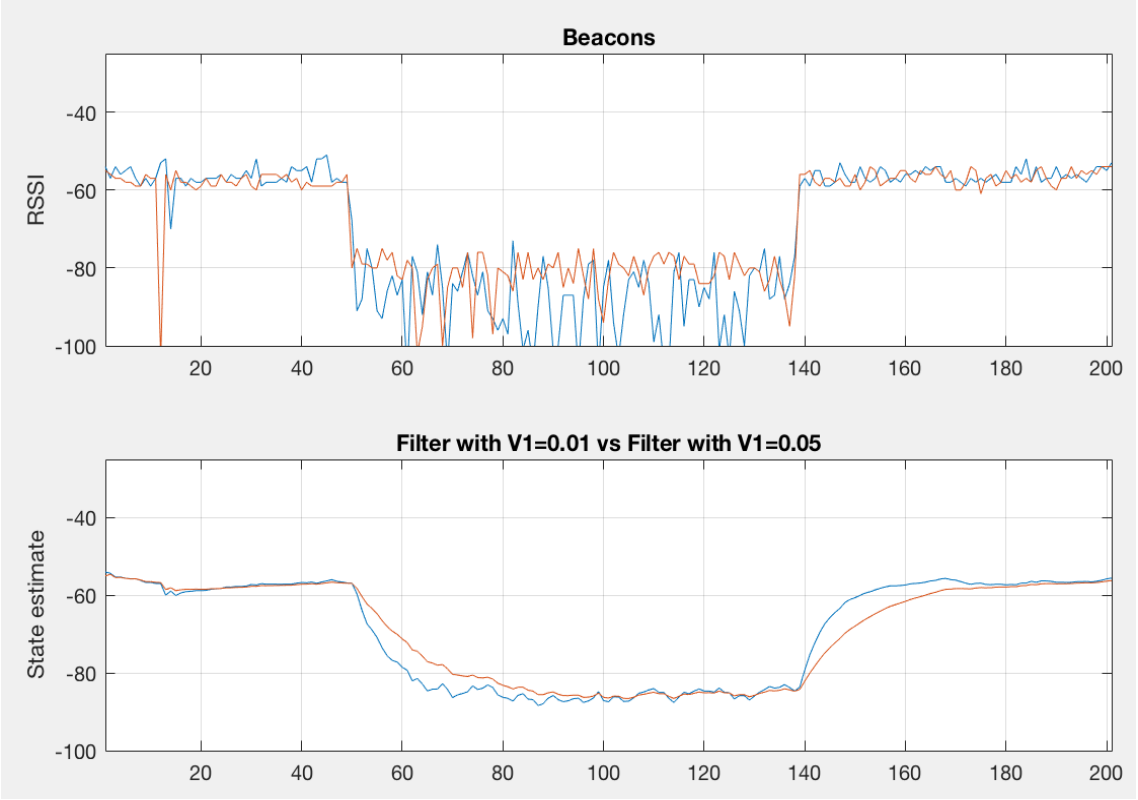


Figure 4.13: Sensor fusion filter with $V_1 = 0.05$ (blue) vs filter with $V_1 = 0.01$ (orange)

$C(i)$, in order to try to find a subset that contains the optimal one or at least a good suboptimal one.

In these strategies an initial partition is specified, i.e. the clusters are assigned an initial value. At each iterative step this assignment is changed such that the value of the cost function is improved with respect to the previous value. Because of this, these algorithms are susceptible to converging to local optima.

To evaluate how similar each of the observations are with one another, the Euclidean distance is chosen as a similarity measure.

$$d(x_i, x_{i'}) = \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 = ||x_i - x_{i'}||^2 \tag{4.8}$$

Which leaves the within cluster scatter point equation as:

$$W(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(i')=k} ||x_i - x_{i'}||^2 \tag{4.9}$$

$$= \sum_{k=1}^{K} N_k \sum_{C(i)=k} ||x_i - \bar{x}_k||^2 \tag{4.10}$$

In Equation 4.10, $\bar{x}_k = (\bar{x}_{1k}, ..., \bar{x}_{pk})$ is the mean vector associated with the kth cluster and $N_k = \sum_{i=1}^{N} I(C(i) = k)$. The loss function is minimized by assigning the N observations to the K clusters in such a way that the average dissimilarity of the observations with respect to the cluster mean, is minimized.

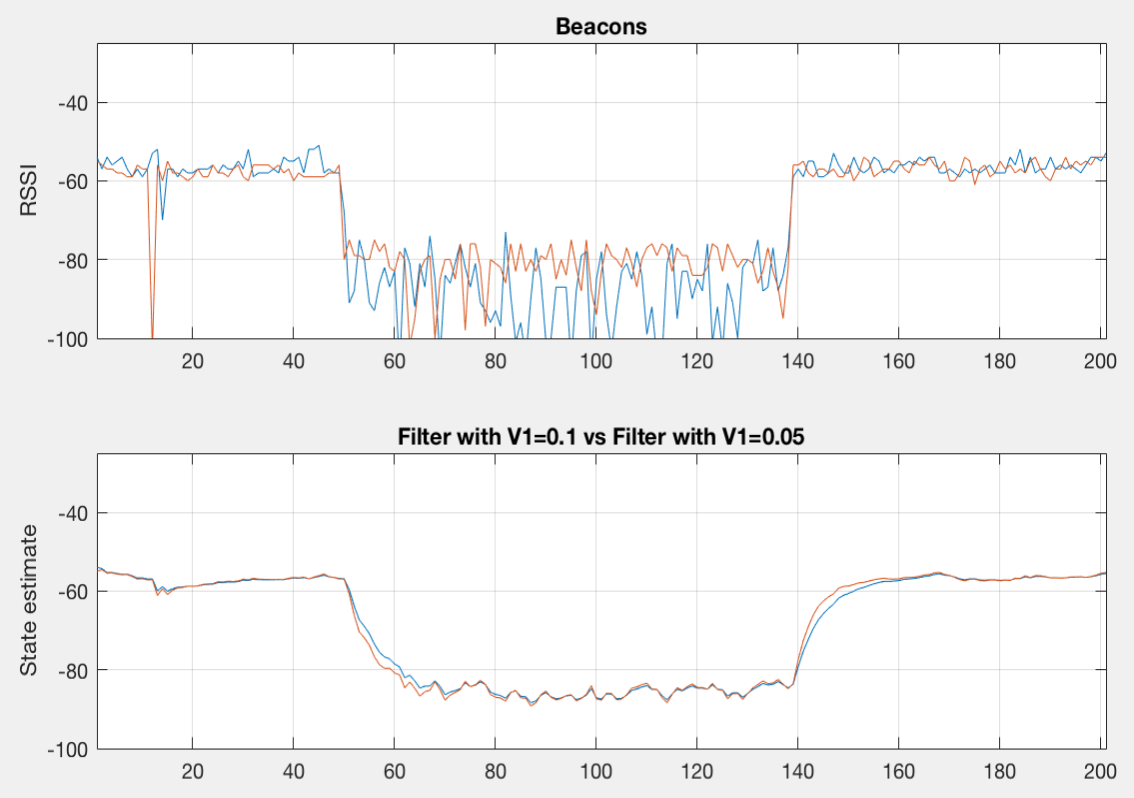


Figure 4.14: Sensor fusion filter with $V_1 = 0.05$ (blue) vs filter with $V_1 = 0.1$ (orange)

The main goal is to find an encoder $C^*(i)$ that achieves the goal described before of minimizing the cost function.

$$C^* = \min_{C} \sum_{k=1}^{K} N_k \sum_{C(i)=k} ||x_i - \bar{x}_k||^2$$

It is important to note that for any set of observations S.

$$\bar{x}_S = \operatorname*{argmin}_{m} \sum_{i \in S} ||x_i - m||^2 \tag{4.11}$$

Therefore C^* can be obtained by solving:

$$\min_{C,\{m_k\}} \sum_{k=1}^{K} N_k \sum_{C(i)=k} ||x_i - m_k||^2 \tag{4.12}$$

Equation 4.12 can be minimized with Algorithm 1.

While steps 1 and 2 of 1 reduce the value of the cost function ensuring convergence, it is perfectly possible to converge to a local minimum as mentioned before.

A possible solution to the local minimum problem would be to run the algorithm several times with different initialization for the clusters. Although for the purposes of this project that solution cannot be implemented. A different method will have to be used, this will be expanded upon further in later sections of this chapter.

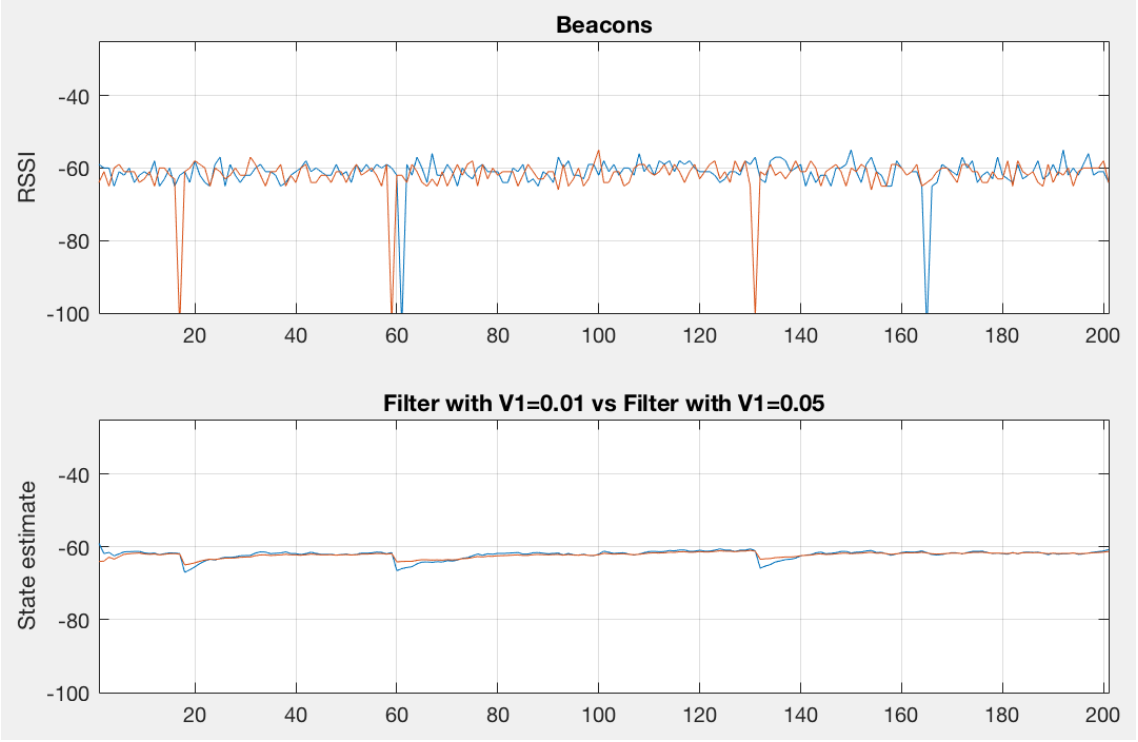


Figure 4.15: Sensor fusion filter with data set 3 with $V_1 = 0.05$ (blue) vs filter with $V_1 = 0.01$ (orange)

4.2.2 Online K-means clustering

Given that the algorithm needs to be applied over an online system, where new measurements will keep arriving to be processed, a recursive version will have to be implemented. This is because the algorithm must be able to process one measurement at a time.

A key part of the algorithm is the calculation of the clusters' centers. This is done by averaging all the measurements assigned to that particular cluster based on the similarity measure selected. The following borrows heavily from [24].

To find a recursive version of the algorithm, a good place to start is to analyze

Algorithm 1 K-means Clustering taken from [5]

1. For a given cluster assignment C, the total cluster variance (4.12) is minimized with respect to $\{m_1, ..., m_k\}$ yielding the means of the currently assigned clusters (4.11).

2. Given a current set of means $\{m_1, ..., m_k\}$, (4.12) is minimized by assigning each observation to the closest (current) cluster mean. That is,

$$C(i) = \underset{1 \leq k \leq K}{\operatorname{argmin}} ||x_i - m_k||^2 \tag{4.13}$$

3. Repeat steps 1 and 2 until assignments do not change.

how the mean $\mu^{(n)}$ of n observations $\{x_1, ..., x_n\}$ is calculated.

$$\begin{aligned}
\mu^{(n)} &= \frac{1}{n}\sum_{j=1}^{n} x_j \\
&= \frac{1}{n}\sum_{j=1}^{n-1} x_j + \frac{1}{n}x_n \\
&= \frac{n-1}{n}\mu^{(n-1)} + \frac{1}{n}x_n \\
&= \mu^{(n-1)} - \frac{1}{n}\mu^{(n-1)} + \frac{1}{n}x_n \\
&= \mu^{(n-1)} + \frac{1}{n}(x_n - \mu^{(n-1)})
\end{aligned}$$

In the end we have that the calculation of the new cluster center for a given cluster after a new observation arrives can be written as:

$$\mu^{(n)} = \mu^{(n-1)} + \frac{1}{n}(x_n - \mu^{(n-1)}) \tag{4.14}$$

With Equation 4.14 the algorithm becomes a recursive one, given that it only depends on the last observation to get the new value of the center.

The algorithm for the Online K-means ends up as follows:

Algorithm 2 Online K-means taken from [12]

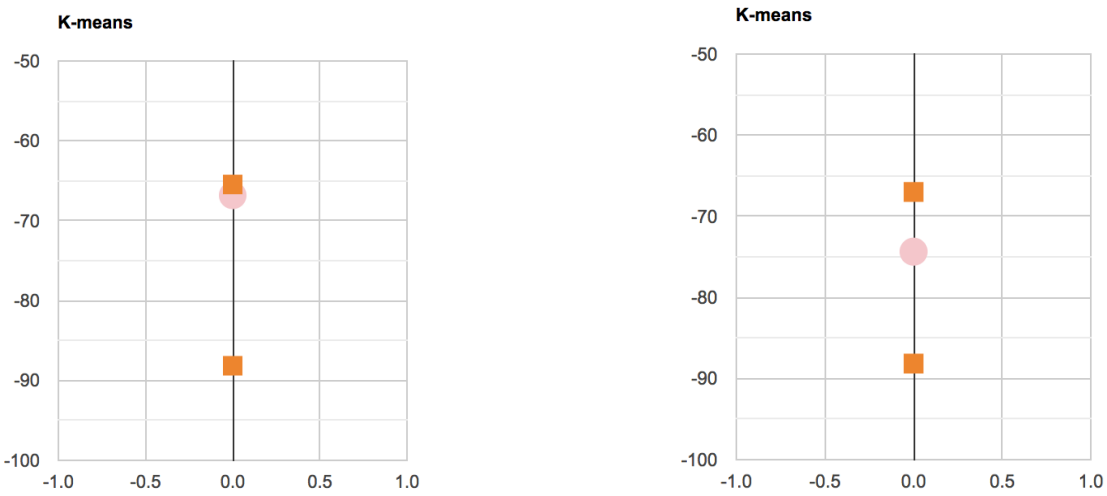
$i \leftarrow 1$
for $x_j \in \{x_1, ..., x_n\}$ **do**
  if $j \leq k$ **then**
    $\mu_i \leftarrow x_j$ ▷ Initialize centroids
    $n_i \leftarrow 1$
    $i \leftarrow i + 1$
  else
    $\mu_i = argmin_j ||x_j - \mu_l||^2$ ▷ Determine winner centroids
    $\mu_i \leftarrow \mu_i + \frac{1}{n_i+1}(x_j - \mu_i)$ ▷ Update winner centroid and n_i
    $n_i \leftarrow n_i + 1$

With Algorithm 2 the system is able to assign new measurements to its respective cluster relatively quickly without having to analyze the whole dataset again. Thanks to this modification the algorithm becomes feasible in the context of this project.

4.2.3 Initializing the clusters

As mentioned before, in K-means clustering, the number of clusters needs to be defined beforehand and each of their centers has to be initialized as well.

Since the algorithm is being applied to an online system, it is not possible to run it with different cluster initializations. Because of this, it is needed to apply previous knowledge about the signal and the problem to select not only how many clusters but also good initializations for each one, so that the algorithm does not yield a local minimum.



(a) Initial state

(b) Final state

Figure 4.16: Cluster initialization with 10 times standard deviation separation. Setup with $\mu_2 = \mu_1 - 10 * \sigma_s$. The vertical axis represents the intensity of the signal measured in dBm. The squares represent the cluster centroids. The circle represents the new measurement, it is red if it is assigned to cluster one, green otherwise. In Figure 4.16b someone is seated, yet the measurement is still assigned to cluster one.

Given the nature of the problem explained in previous chapters, it is possible to observe that two clusters are needed. A cluster for "not seated" and one for "seated".

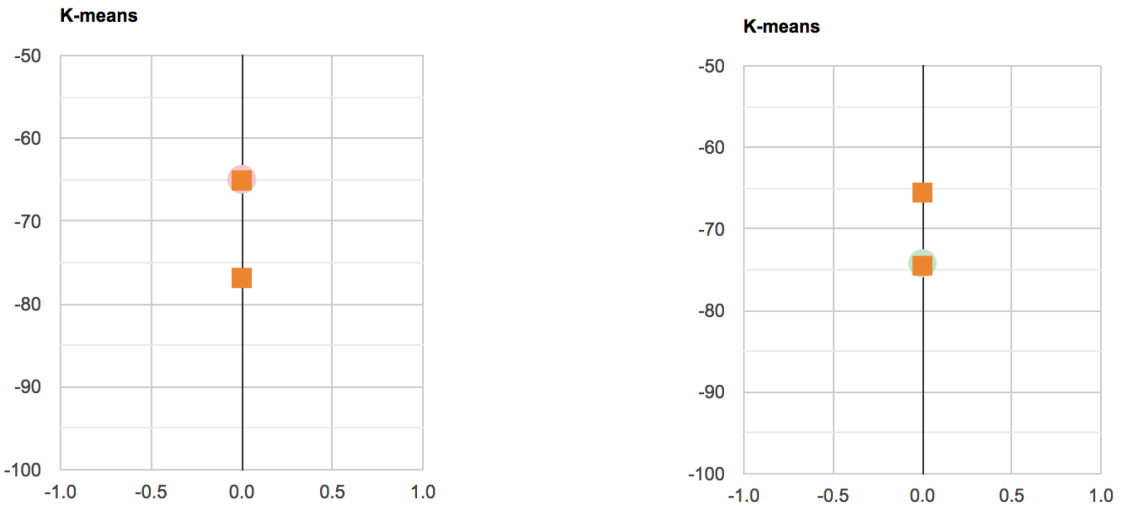
By observing how the signal behaves, this can be seen in Figure 3.4 and in Figure 4.12, a strategy for cluster initialization is created. By using the calibration phase, the mean of the signal (μ_s) when no one is seated is calculated and it is used as the centroid for the "not seated" cluster.

For the second cluster, which is the one for "seated", a little experimenting was needed to find a good initialization value. The main idea is to use the standard deviation of the signal (σ_s) calculated during the calibration phase. As seen before, different chair structures yield different distortions of the signal.

Taking this into consideration, an initialization value that is as close to the first centroid as possible, but not so much as to absorb the "not seated" measurements, is needed.

The first experiment was devised by placing the second centroid ten standard deviations away from the first one. This resulted in a very accurate algorithm at first when no one was seated, although after someone did, for the example seen in Figure 3.4 where the distortion is not that pronounced, the observations where still closer to the first centroid and therefore misclassified as "not seated". This is because the second centroid was placed too far away from the first one. This can be seen in Figure 4.16.

A second experiment was performed using five standard deviations away from the first centroid. This resulted again in a very accurate algorithm at first when no one was seated, but after someone sat down the algorithm was still able to assign the measurement to the second cluster unlike with the previous configuration. This can be observed in Figure 4.17. Also it is important to note how the second cluster centroid ended in a different position than where it initially started as can be seen



(a) Initial state

(b) Final state

Figure 4.17: Cluster initialization with 5 times standard deviation separation. Setup with $\mu_2 = \mu_1 - 5 * \sigma_s$. The vertical axis represents the intensity of the signal measured in dBm. The squares represent the cluster centroids. The circle represents the new measurement, it is red if it is assigned to cluster one, green otherwise.

in Figure 4.17b, which means that the algorithm is reactive as it was intended.

It may seem that a good initialization for the centroids can be obtained by using five standard deviations as observed in the second experiment, but it is important to find the smallest distance between the clusters that would yield the best possible results. This is again due to the fact that there might be a chair with a structure that translates in very low distortions in the signal when someone sits down, therefore a third experiment was devised with three standard deviations away from the first centroid. This can be seen in Figure 4.18.

The third experiment resulted again in a very accurate algorithm at first when no one was seated as well as after someone sat down, given that the algorithm was still able to assign the measurement to the second cluster.

A final experiment was performed using two standard deviations from the first centroid. This resulted in several misclassifications at first when no one was seated, although after someone sat down the algorithm stabilized placing the cluster centroids in their final and optimal position. This can be observed in Figure 4.19.

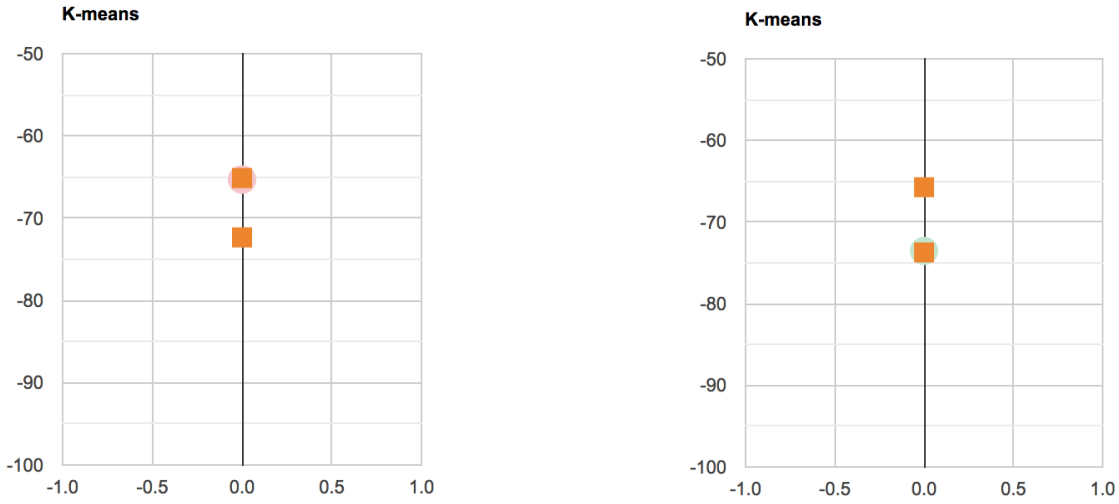
After taking into consideration the results of the experiements previously described, it is concluded that the second centroid should be initialized to three times the standard deviation of the signal (σ_s) lower than the value of the first centroid. This can be seen in Equation 4.16.

$$\mu_1 = \mu_s \text{ (centroid of cluster 1 "not seated")} \tag{4.15}$$

$$\mu_2 = \mu_1 - 3 * \sigma_s \text{ (centroid of cluster 2 "seated")} \tag{4.16}$$

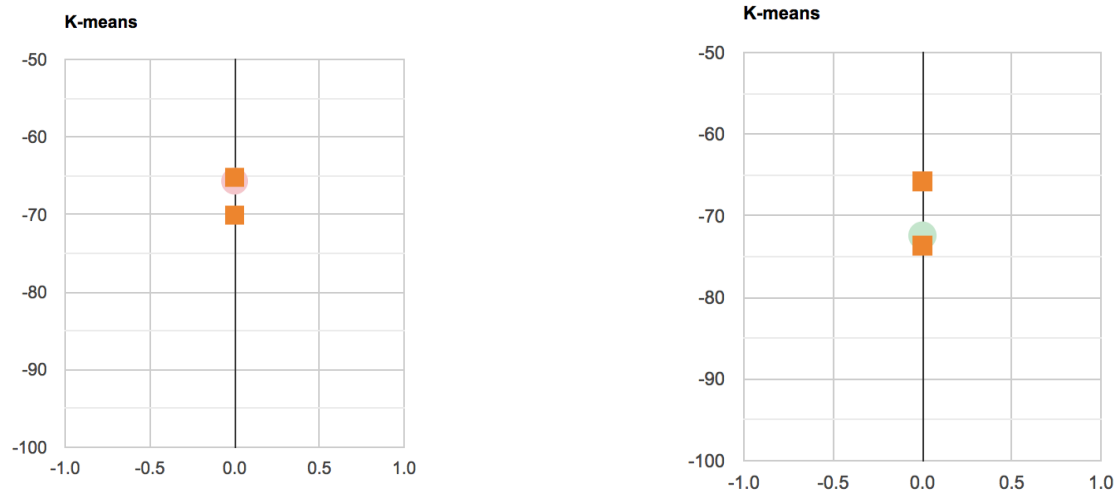
As mentioned before, by using the Euclidean distance as a similarity measure, when a new measurement arrives, it will be assigned to the cluster that has its centroid closer to the new observation.

If the centroid closer to the observation belongs to the first cluster, the measurement will be assigned to it and labeled as "not seated", while on the other hand,



(a) Initial state (b) Final state

Figure 4.18: Cluster initialization with 3 times standard deviation separation. Setup with $\mu_2 = \mu_1 - 3 * \sigma_s$. The vertical axis represents the intensity of the signal measured in dBm. The squares represent the cluster centroids. The circle represents the new measurement, it is red if it is assigned to cluster one, green otherwise.



(a) Initial state (b) Final state

Figure 4.19: Cluster initialization with 2 times standard deviation separation. Setup with $\mu_2 = \mu_1 - 2 * \sigma_s$. The vertical axis represents the intensity of the signal measured in dBm. The squares represent the cluster centroids. The circle represents the new measurement, it is red if it is assigned to cluster one, green otherwise.

if it is closer to the second centroid, it will be assigned to the second cluster and labelled as "seated". After the assignment is performed, the new positions of the centroids are calculated based on Equation 4.14.

Chapter 5

Experiment Results

This chapter evaluates the performance of the system in real world scenarios.

To get a good sense of how the system behaves it is important to perform the experiments using multiple chairs of different structures as well of conducting them with different configurations.

The micro controllers were installed in each of the chairs where the experiments took place. All of the data was sent to a web server where it was stored in a database for further processing.

After the data was collected, a dataset, containing more than a thousand samples, for each chair and configuration was generated by exporting the records into comma separated (*.csv*) files.

Each file contained the values for the RSSI of the signal with a timestamp of when that observation was sampled. An example can be seen at Figure 5.1.

```
rssi,created_at
-60,2018-05-22 23:02:59.660762
-61,2018-05-22 23:03:00.427054
-60,2018-05-22 23:03:01.542599
-59,2018-05-22 23:03:02.661885
-61,2018-05-22 23:03:03.883411
-58,2018-05-22 23:03:04.994033
-59,2018-05-22 23:03:06.113222
-57,2018-05-22 23:03:07.264673
-57,2018-05-22 23:03:08.798328
```

Figure 5.1: Comma separated file with data collected for analysis

One file was generated for each beacon installed in each chair.

A set of controlled experiments were designed to test the system. As mentioned before, diversity is key, i.e., using different configurations, to get a good picture of how the system behaves and the accuracy of its predictions. For this, two very different chairs were chosen.

The first chair can be seen in Figure 5.2 and the second in Figure 5.4.

For each chair a set of 9 experiments were performed. Each chair was tested with one beacon, then two beacons and finally three beacons. In each of these three scenarios, three different configurations were tested, with a process noise $V_1 = 0.01$, with a process noise $V_1 = 0.05$ and without a filter altogether.

- Chair 1 with one beacon Figure 5.6. The results can be seen in Table 5.1.

Figure 5.2: Chair No. 1

Figure 5.3: Chair No. 1 client

Figure 5.4: Chair No. 2

Figure 5.5: Chair No. 2 client

- Chair 1 with two beacons Figure 5.7. The results can be seen in Table 5.1.
- Chair 1 with three beacons Figure 5.8. The results can be seen in Table 5.1.
- Chair 2 with one beacon Figure 5.9. The results can be seen in Table 5.2.
- Chair 2 with two beacons Figure 5.10. The results can be seen in Table 5.2.
- Chair 2 with three beacons Figure 5.11. The results can be seen in Table 5.2.

Figure 5.12 shows all the metrics side by side for better comparison and analysis of the results for chair 1.

Figure 5.13 shows all the metrics side by side for better comparison and analysis of the results for chair 2.

5.1 Analysis

In this chapter, the results of the experiments obtained in the previous section, are discussed.

Before continuing with the analysis it is important to define the nature of the action, i.e. what does it mean to be seated. For the purposes of this project, if a person sat in the chair for less than three seconds, it cannot be counted as seated.

Figure 5.6: Chair No. 1 with one beacon

Figure 5.7: Chair No. 1 with two beacons

1 beacon				
	Precision	Sensitivity	Accuracy	Specificity
$V_1 = 0.05$	82%	86%	88%	90%
$V_1 = 0.01$	77%	80%	79%	79%
No Filter	85%	99%	93%	88%
2 beacons				
	Precision	Sensitivity	Accuracy	Specificity
$V_1 = 0.05$	91%	92%	93%	93%
$V_1 = 0.01$	85%	93%	91%	89%
No Filter	84%	95%	90%	87%
3 beacons				
	Precision	Sensitivity	Accuracy	Specificity
$V_1 = 0.05$	94%	93%	95%	96%
$V_1 = 0.01$	88%	89%	90%	91%
No Filter	93%	93%	93%	93%

Table 5.1: Results for Chair 1

This is why in all the experiments the person sat down for a lot more than three seconds. Having said that, the results expressed here do take into consideration

Figure 5.8: Chair No. 1 with three beacons

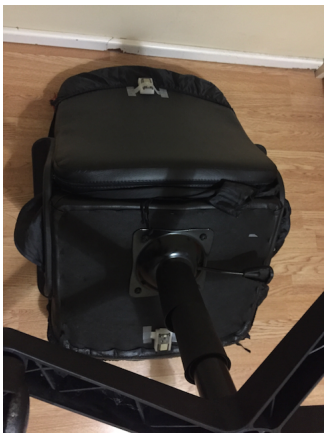


Figure 5.9: Chair No. 2 with one beacon

Figure 5.10: Chair No. 2 with two beacons

those first three seconds when the person just sat down to calculate the accuracy. This is done to keep the results as pure and bias free as possible, but it is something to keep in mind. The harmful effects of the delay, or smoothing of the signal, is an issue that will be touched upon during the analysis, and this affects the overall accuracy of the system, but as mentioned before, some of this decline in performance can be accepted due to the nature of the action and the overall goal of this project. This means that the false negatives, i.e. the sensitivity, is not that important as a

Figure 5.11: Chair No. 2 with three beacons

1 beacon				
	Precision	Sensitivity	Accuracy	Specificity
$V_1 = 0.05$	57%	99%	63%	25%
$V_1 = 0.01$	55%	96%	62%	33%
No Filter	66%	82%	69%	55%
2 beacons				
	Precision	Sensitivity	Accuracy	Specificity
$V_1 = 0.05$	90%	95%	93%	91%
$V_1 = 0.01$	82%	87%	85%	83%
No Filter	80%	100%	90%	83%
3 beacons				
	Precision	Sensitivity	Accuracy	Specificity
$V_1 = 0.05$	94%	76%	89%	97%
$V_1 = 0.01$	91%	95%	93%	91%
No Filter	86%	98%	92%	87%

Table 5.2: Results for Chair 2

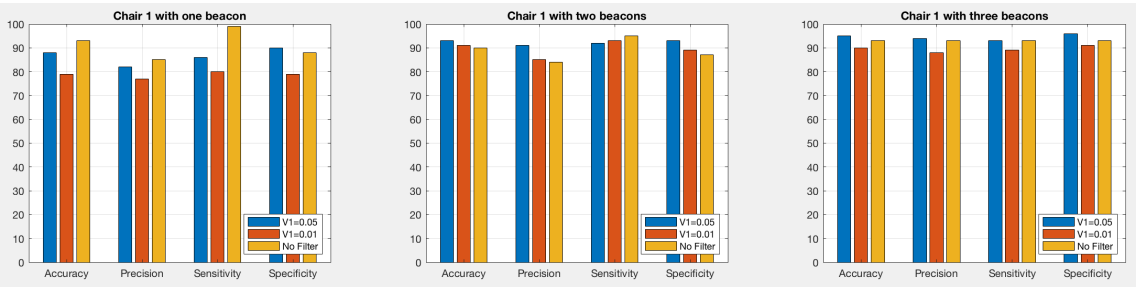


Figure 5.12: Chair No. 1 results

measure in terms of the goal of identifying if someone is seated instead lying down in the floor for long periods of time.

The first thing to note is that the structure of the chair plays an important role on the selection of the configuration. This is due to any material that can interfere with the signal such as metallic parts of the likes that can be seen in the base of Figure 5.4. This is expressed in the difference of accuracy obtained in the single

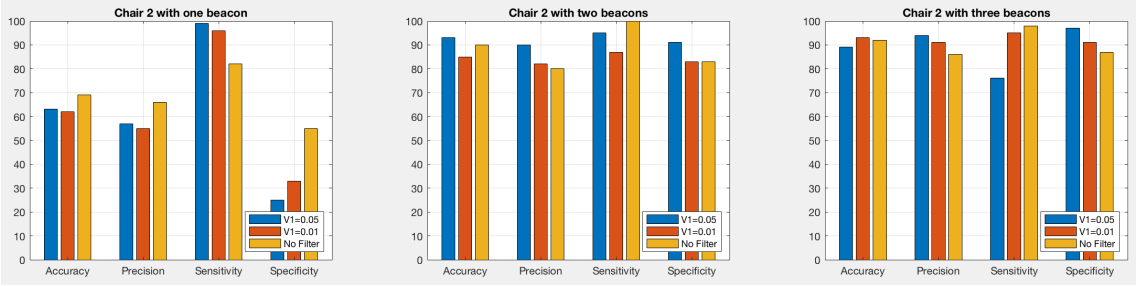


Figure 5.13: Chair No. 2 results

beacon configuration in Figure 5.12 and Figure 5.13. Not all configurations work the same for all chairs.

It can be observed in Figure 5.12 that all configurations provide a high accuracy.

The one with the lowest accuracy is the single beacon with the Kalman filter with a process noise of $V_1 = 0.01$. The fact that this is the case is related to the reactivity of the filter. Since the process noise is not very high, it takes a while for the estimation of the state to reach the new value of the signal, resulting in miss classifications during this adjustment. This is reflected in the precision, where miss classifications due to noise are highly reduced, but this is lost because miss classifications due to the estimate lagging behind the signal (someone stood up but the system takes a while to notice) are introduced. The same principle applies for the lower value in sensitivity too, someone just sat down and the system takes a while to notice, resulting in several false negative predictions. This is better observed in Figure 5.14.

Figure 5.14: Chair No. 1 with 1 beacon configuration with $V_1 = 0.01$ lag example

Things get better with a process noise of $V_2 = 0.05$. As seen in Figure 5.12, the accuracy increases to 88%, almost a 10% increase with respect to the previous mentioned configuration. This is due to the reactivity of the filter. Because of this, there are less false negatives and false positives due to the signal lagging behind, increasing the precision and the sensitivity as well. This reactivity can be seen in Figure 5.15.

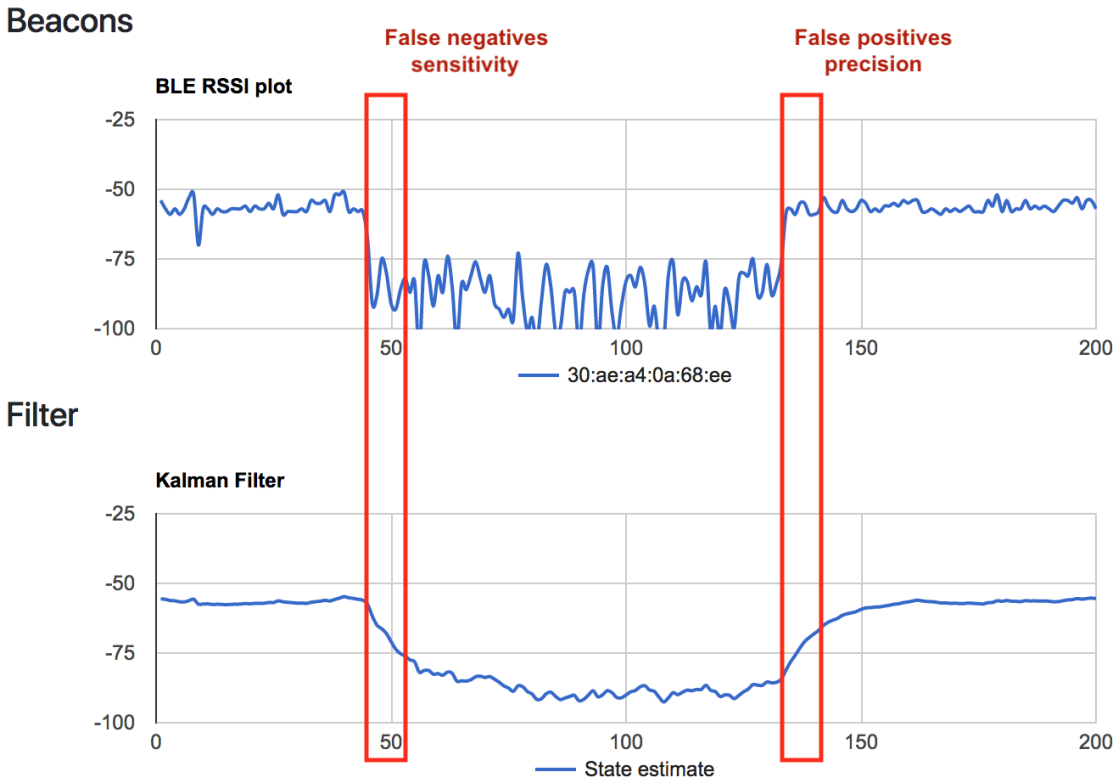


Figure 5.15: Chair No. 1 with 1 beacon configuration with $V_1 = 0.05$ lag example

The accuracy achieved when there is no Kalman filter associated, seen in Figure 5.12, is actually higher than with the filter. This is due to the fact that without the filter the algorithm is extremely reactive to any variations in the signal. This may seem like an advantage, but this is the result of the environment not having a lot of noise. In the case that the signal is affected by noise like in the next experiment, then the accuracy will decrease, making the algorithm exposed and susceptible to false positives. Figure 5.16 shows a miss classification due to this.

In the case of **two beacons** for Chair No. 1, it can be observed in Figure 5.12 that the performance increases considerably. Not only across the accuracy of all the configurations, except for the one without the filter, but also in the precision and sensitivity.

In this experiment the signal was affected more by noise and this is where the filter shines. Not only the accuracy is better than without the filter, but the precision increases considerably. The precision of the configuration with $V_1 = 0.05$ is almost 10% greater than without the filter as can be seen in Figure 5.12.

As with the one beacon scenario, the accuracy of the filter with a higher process noise is again higher than the filter with a lower one in the two beacons scenario.

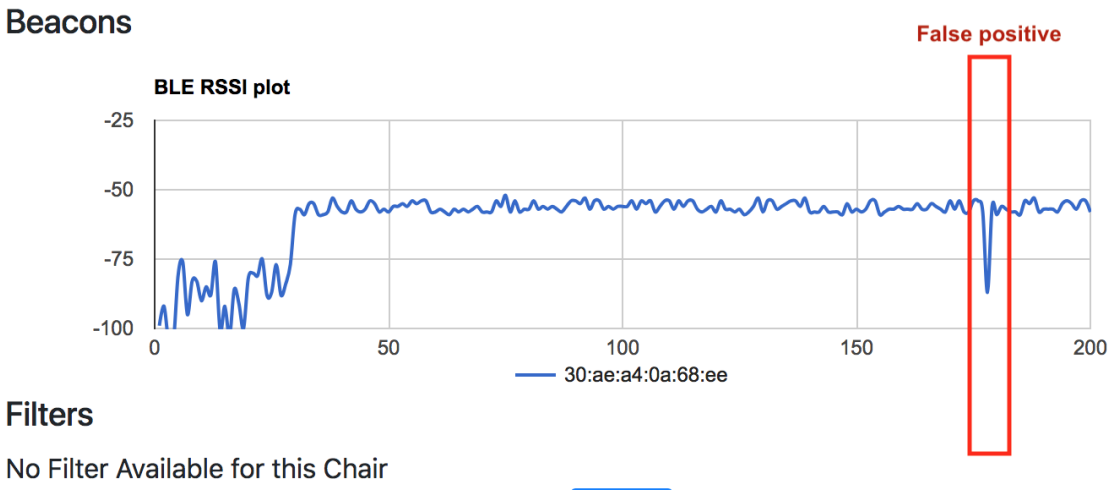


Figure 5.16: Chair No. 1 with 1 beacon configuration with no filter false positive example

Continuing with the experiments for Chair No. 1, it can be appreciated that the performance increases again when a new beacon is added.

With three beacons the accuracy for the configuration with the filter with a $V_1 = 0.05$ increases although not by much. The same occurs with the other values, precision, sensitivity and specificity.

It can be concluded that a configuration of two beacons is more than enough to get good predictions, although an improvement can be obtained by adding a new beacon to the configuration.

Moving on to the experiments performed in Chair No. 2 Figure 5.4, the important role the structure of the chair plays, can be observed in the first set of experiments. Where the performance of the system is really affected and therefore returned very poor results for the single beacon configuration as can be seen in Figure 5.13.

This is in part because the chair is larger than the first one, making the distance between the client and the receiver larger as well. Also in part because of the metal base that supports it, given that as mentioned before, metallic surfaces can distort the 2.4GHz BLE signal. This chair poses a big challenge for the system due to this factors, and as can be seen in Figure 5.13 in subsequent experiments, the system is flexible enough to get around these issues.

The first experiment with a single beacon configuration for Chair No. 2 returned an accuracy of around 60% or less for all the three experiments performed. This is due to all the reasons explained before.

The high values in sensitivity are due to the fact that the system has a hard time differentiating between someone seated or not, and almost all the predictions err on the side of no one being seated.

A significant improvement can be observed changing slightly the position of the beacon and adding a new one to the configuration as well, as can be seen in Figure 5.10. The results improve dramatically. The system can get a clearer picture of the signal and its distortion.

The configuration with two beacons and a filter with a process noise of $V_1 = 0.05$ outperforms the other filter in all categories. It only trails behind the configuration

with no filter with regards to the sensitivity, although not by much. This can be traced back again to the issue of the reactivity of the filter.

The configuration with no filter is more reactive, but more susceptible to false positives. That is why the filter outperforms it considerably in the precision category with a difference of almost 10%.

The experiments with three beacons returned high values in all categories. The experiments show a little improvement in comparison to the 2 beacons experiments.

It is important to note that these are the only experiments where the lower process noise filter outperforms, in terms of accuracy, the one with a higher V_1.

This is because the filter with a value of $V_1 = 0.05$ scored a very low value in sensitivity, i.e. it had a very high value of false negatives due to the third beacon not being placed in the best possible position due to the structure of the chair as seen Figure 5.11. The signal behaved in an erratic way affecting the results of the filter. Therefore the filter with a lower process noise, that did not follow the measurements more closely yielded better results in this particular scenario. Although it is important to note that not by a large margin.

Even in the face of these challenges, the system is robust enough to perform well.

The sampling rate used for the experiments, and the system in general, is of one observation every half a second, i.e. 0.5 s, which is more than enough to test the hypothesis of the thesis. Having said that, the issue of the delay in the signal can be further improved if the need would arise to do so, by just increasing the sampling rate. The higher the sampling rate the more observations the filter will have per second, allowing it to find the new value of the signal faster, although in the same amount of iterations as before. This would not necessarily improve the metrics previously displayed in Figure 5.12 and Figure 5.13 because even though the system is faster, it would need the same amount of observations to reach the desired result.

Chapter 6

Web Architecture

In order for the project to be scalable and flexible enough to be feasible and practical, a multilayer service oriented architecture was needed to provide the proper support structure for the complete process.

The less computation the micro controllers do, the easier it is to replace them. As mentioned before, the hardware solution for this project was developed with the only purpose of analyzing the feasibility of using BLE technology for proximity detection, it is by no means a market ready solution, therefore the system was developed in a way so that the hardware side, i.e. the BLE client and BLE server, could be easily replaced. This is the reason why a web server that stores all the measurements and performs all the computations is created.

For the purposes of this project, the advantages of having a multilayer service oriented architecture out weight those of other types of applications. Among those advantages are: the fact that it can be accessed anywhere in the world, is highly robust, the use of web services and an API makes it highly flexible and resilient to changes in other parts of the system, such as the interface or the hardware solution, and it is easier to scale if the need arises to do so.

There are several reasons why the decision of separating the UI from the API was made. The most important one is the fact that with the separation comes modularity which improves testing, readability and maintainability. Another reason is reusability, which means that the API can be reused by other applications. Also means that it is easier to change the display of the system should the need arise. This is an important feature given that data visualization has become essential to analysis.

For the web server and API, Ruby and Ruby on Rails were chosen for the language and framework. With those tools it is possible to setup a working web application quickly, which is exactly what is needed to validate the hypothesis of this project. It is important to note that as a framework, Ruby on Rails, has come a long way in terms of scalability and performance, it is by no means a limitation today, but if the need arises, there is always the possibility to migrate to a more performance oriented framework just as long as the API is implemented in the same way, everything else should fall in place.

For the web interface that will display the information stored in the server, React.js is chosen. An important feature of React.js is that it is based on components, each component has its own logic and controls its rendering, and the fact that they can be combined into bigger components makes the code highly reusable. The vir-

tual DOM is another of React.js main features, this is because DOM manipulation is one of the biggest performance bottlenecks in front end applications. React tackles this with a virtual DOM which is basically, a copy of the real DOM that lives in memory. React performs any changes really fast in the virtual DOM and then uses a highly efficient algorithm to determine which changes should be applied to the real DOM providing higher performance and a better user experience.

The use case for the system can be observed in Figure 6.1.

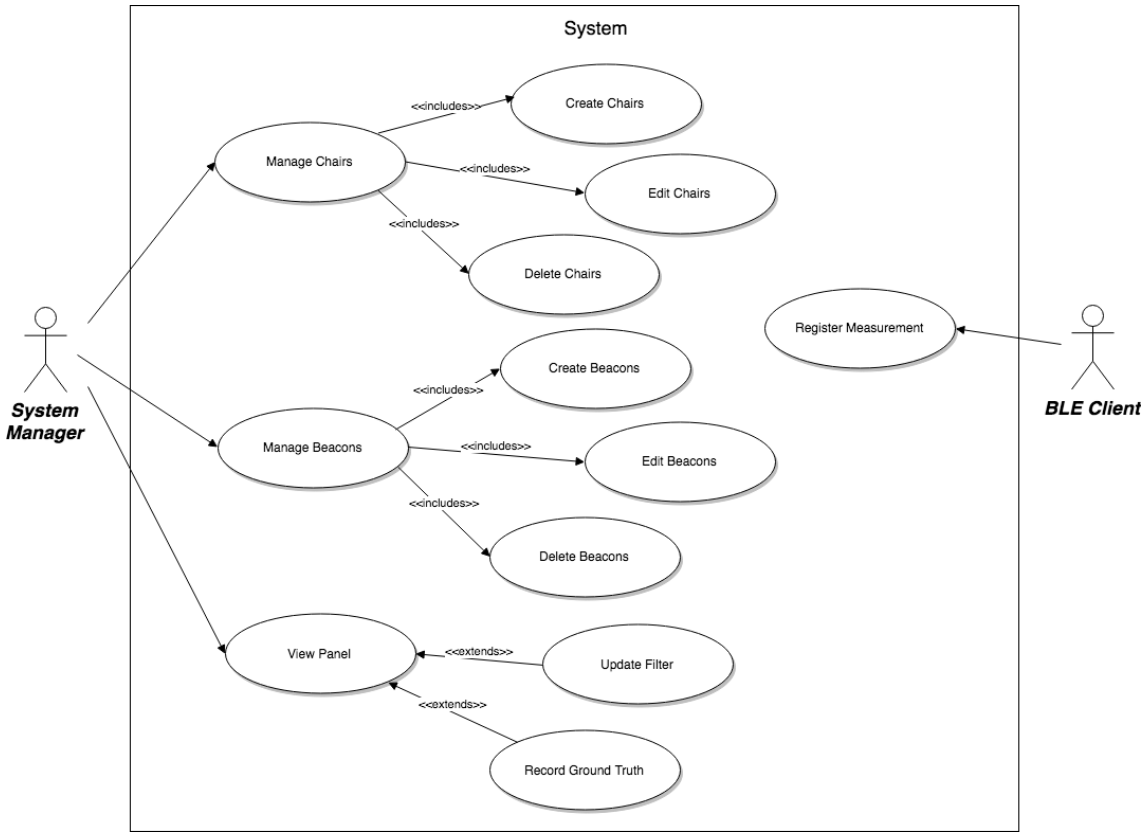


Figure 6.1: System Use Case

6.1 Data model

The application's data model can be seen in Figure 6.2. It displays all the models used in the development of the backend of the application.

6.2 Description of the application

The system is a web application that allows the user to manage and monitor all the information that gets generated by the micro controllers. The user can create chairs and beacons and also generate associations between them. By doing so, the system can determine which measurements belong to which chair, and therefore make predictions about whether someone is seated on it or not.

The application has an index page for chairs that shows all the chairs available in the system. This can be seen in Figure 6.3 although at this point no chair has

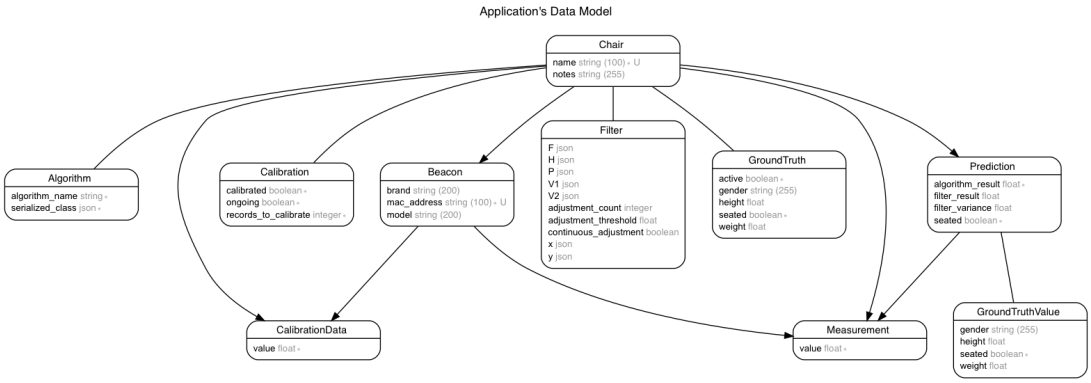


Figure 6.2: Data Model

been created.

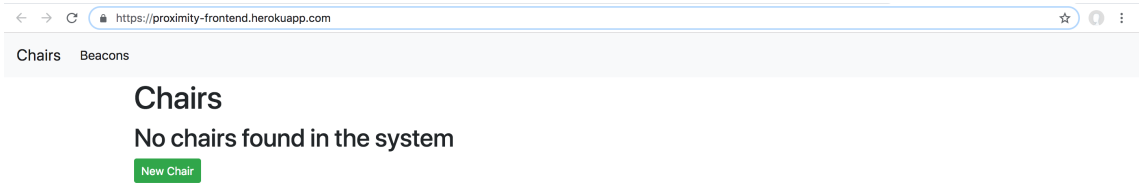


Figure 6.3: Index page for chairs (No chair created)

By clicking on the 'New Chair' button the user is redirected to the form page to create a chair shown in Figure 6.4.

In Figure 6.4 there is the option to apply the Kalman filter to the chair (Apply filter). This was made optional because the system was developed to be as flexible as possible. There might be situations where the chair with the filter under performs. But it was done mainly to allow for experimentation, to see how the system behaves without the filter as well. After the chair is created the user is redirected to the index page Figure 6.5.

The table shown in Figure 6.5 depicts the information for each chair, if it has a filter or not and if it has been calibrated.

The system also contains an index page for beacons that shows all the beacons available in the system. This can be seen in Figure 6.6 although at this point no

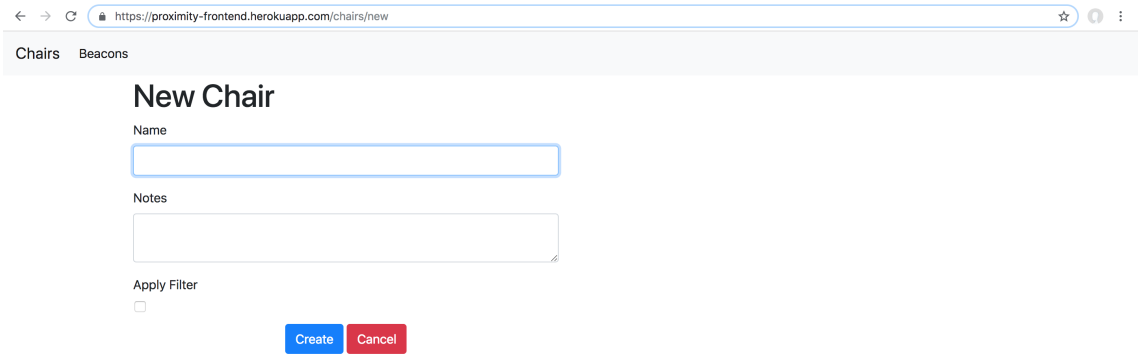


Figure 6.4: Page for the creation of a chair

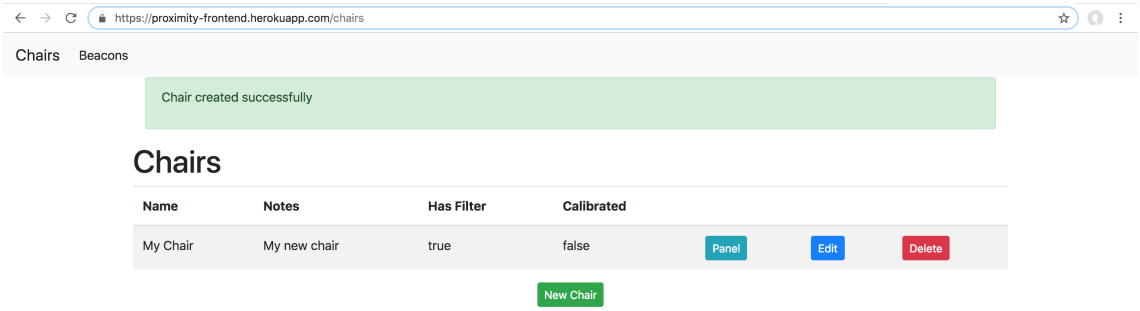


Figure 6.5: Index page for chairs

beacon has been created.

By clicking on the 'New Beacon' button the user is redirected to the form page to create a beacon shown in Figure 6.7.

Figure 6.7 shows the fields needed to create a beacon, only two of them are important, these are the mac address and the chair it belongs to. The mac address should be unique to each beacon given that is the field by which the system differentiates them. In order for the beacon to be used by the system, it needs to be

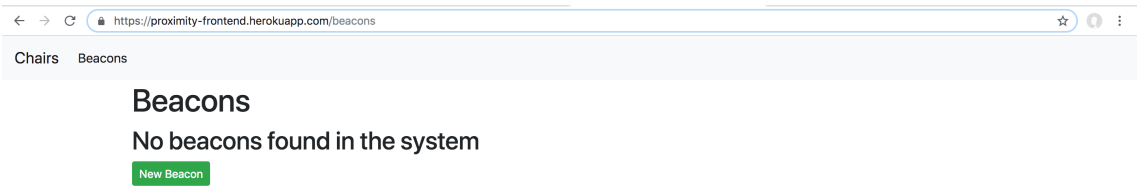


Figure 6.6: Index page for beacons (No beacon created)

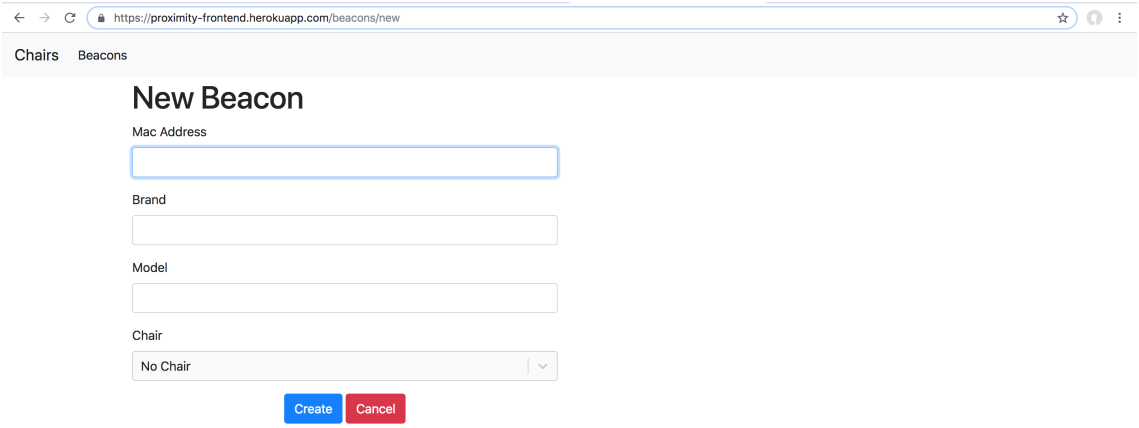


Figure 6.7: Page for the creation of a beacon

associated to a chair. After the beacon is created the user is redirected to the index page Figure 6.8.

A chair can have multiple beacons assigned to it Figure 6.9.

Once the chair and its beacons are set up, in order for the system to start making predictions, the chair must be calibrated. To calibrate a chair, the "Edit" button must be selected from the index page Figure 6.5. This will take the user to the edit page of the chair shown in Figure 6.10. In this page, the user can update

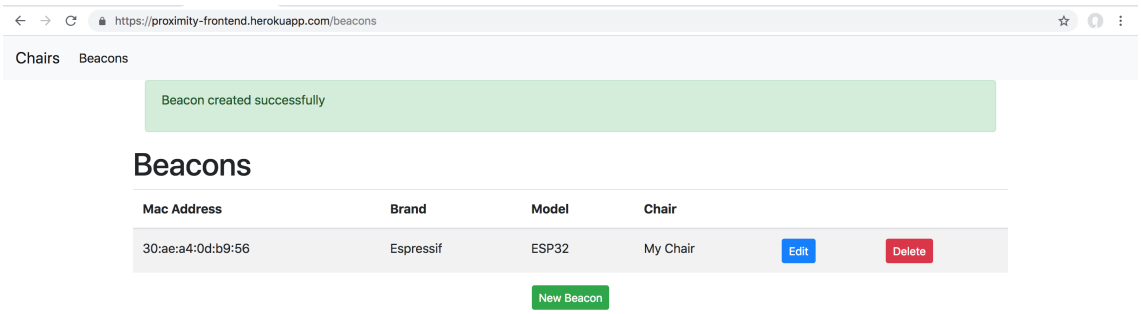


Figure 6.8: Index page for beacons

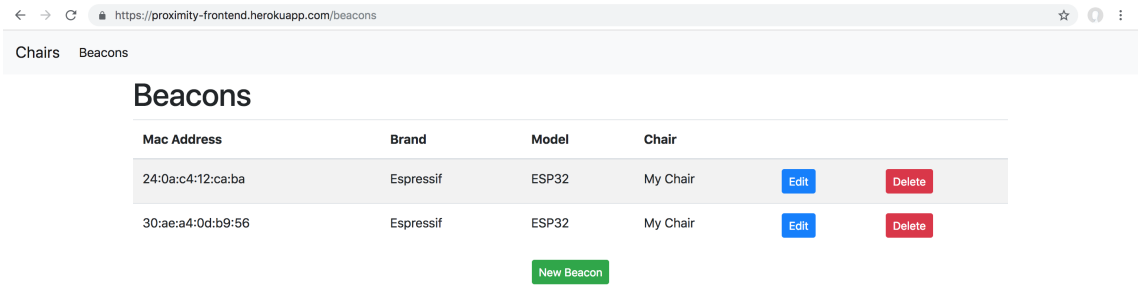


Figure 6.9: Index page for beacons (2 beacons assigned to the same chair)

any information regarding the chair, including add a or remove its filter as well as start the calibration phase. In Figure 6.10 it can be observed that the chair is not calibrated, there is an indicator to the right side of the screen "Calibrated: false" which will turn into true when the chair is calibrated. After pressing the "Start Calibration" button, the chair calibration begins, this is shown in Figure 6.11.

After calibration is complete, the system is ready to start making predictions, this can be seen by pressing on the "Panel" button from the index page Figure 6.5.

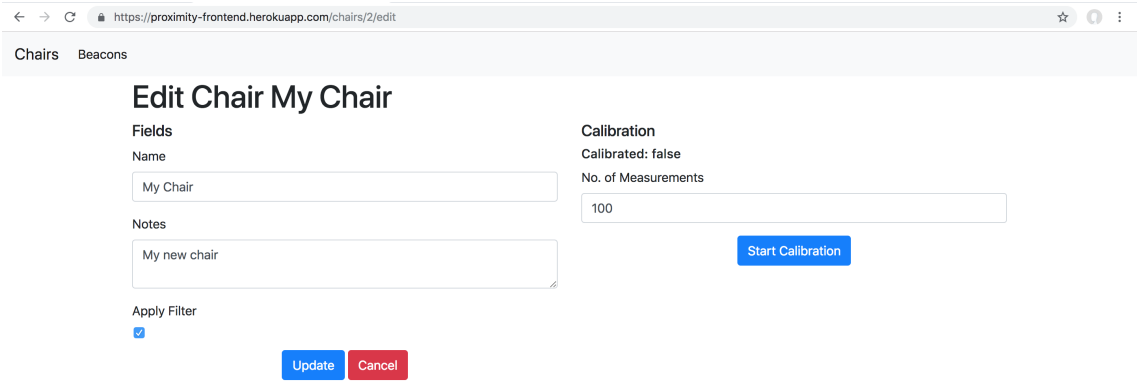


Figure 6.10: Edit page for chair

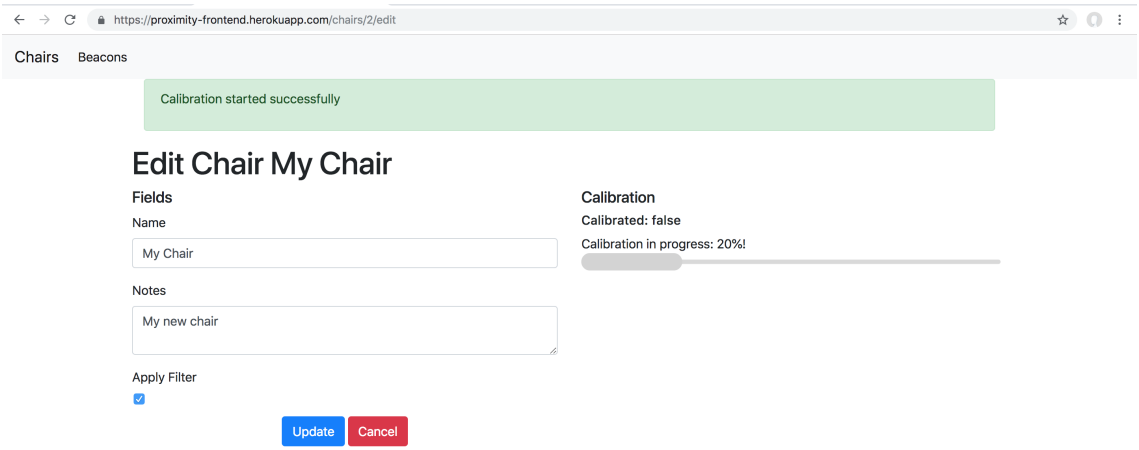


Figure 6.11: Edit page for chair with ongoing calibration

After pressing the button the user is redirected to the panel page for the selected chair Figure 6.12.

In this page Figure 6.12, all the information regarding the chair can be found. Whether someone is seated or not is displayed in the square at the top left corner of the screen (red for no one seated, green otherwise Figure 6.13). The first plot shows the signals coming from all the beacons assigned to the chair. The plot below it, shows the estate estimation of the filter based on the measurements obtained

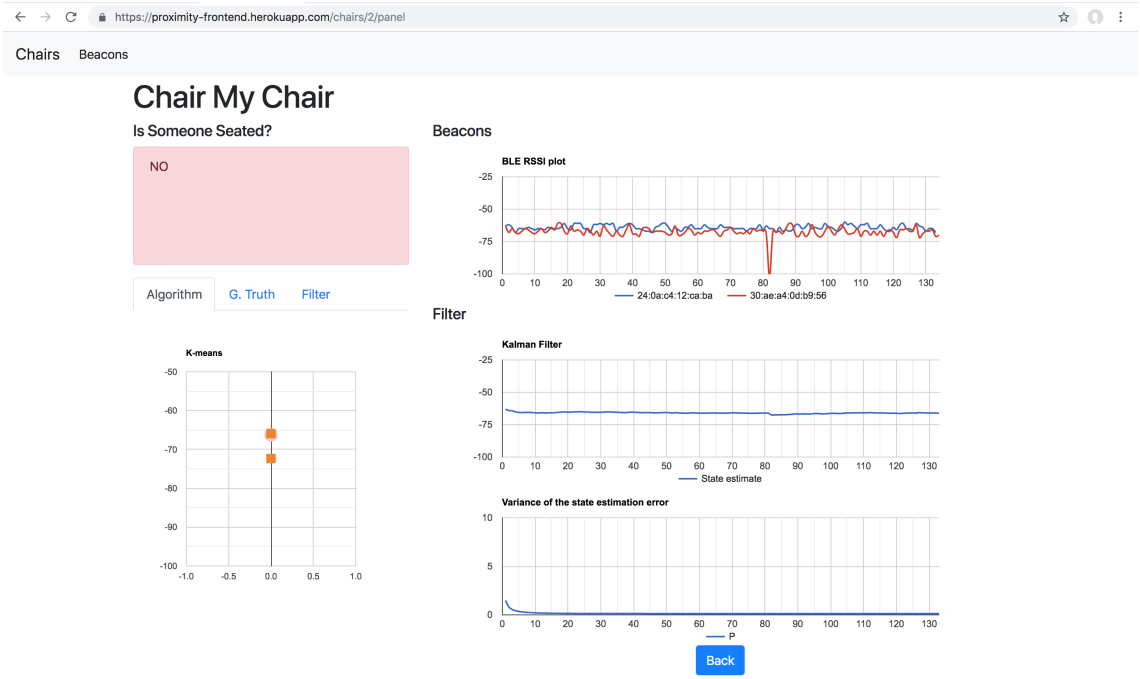


Figure 6.12: Panel page for chair

from both beacons. The last one shows the variance of the estate estimation error, which can be seen, as mentioned in the chapter for the Kalman filter, as how much confidence there is in its estate estimation.

Below the square that shows whether there is someone seated or not in Figure 6.12, the graph for the online K-means algorithm can be found. The squares represent the position of the centroids of each cluster and the circle represents the last measurement obtained.

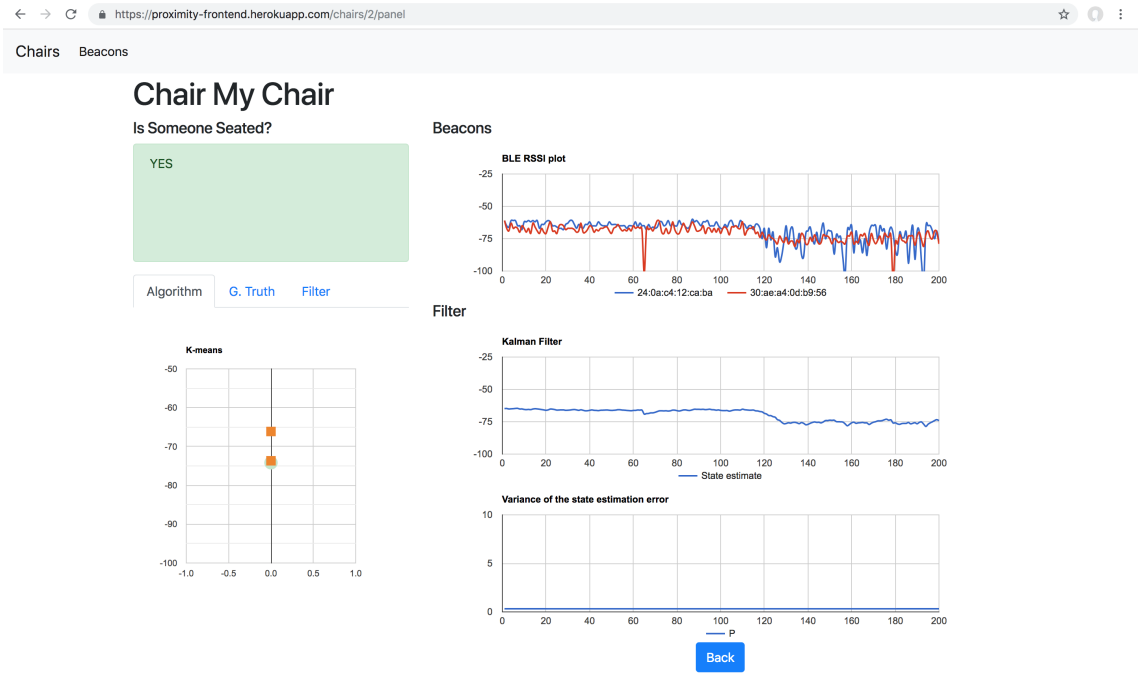


Figure 6.13: Panel page for chair when someone is seated

There is a section as well to collect the ground truth and evaluate how well the system is performing. This panel shown in Figure 6.14 where there previously was the k means cluster algorithm, is updated live as the new measurements are being collected. The values for precision, recall, specificity and accuracy are all being updated live so as to get a quick picture of how the system performs.

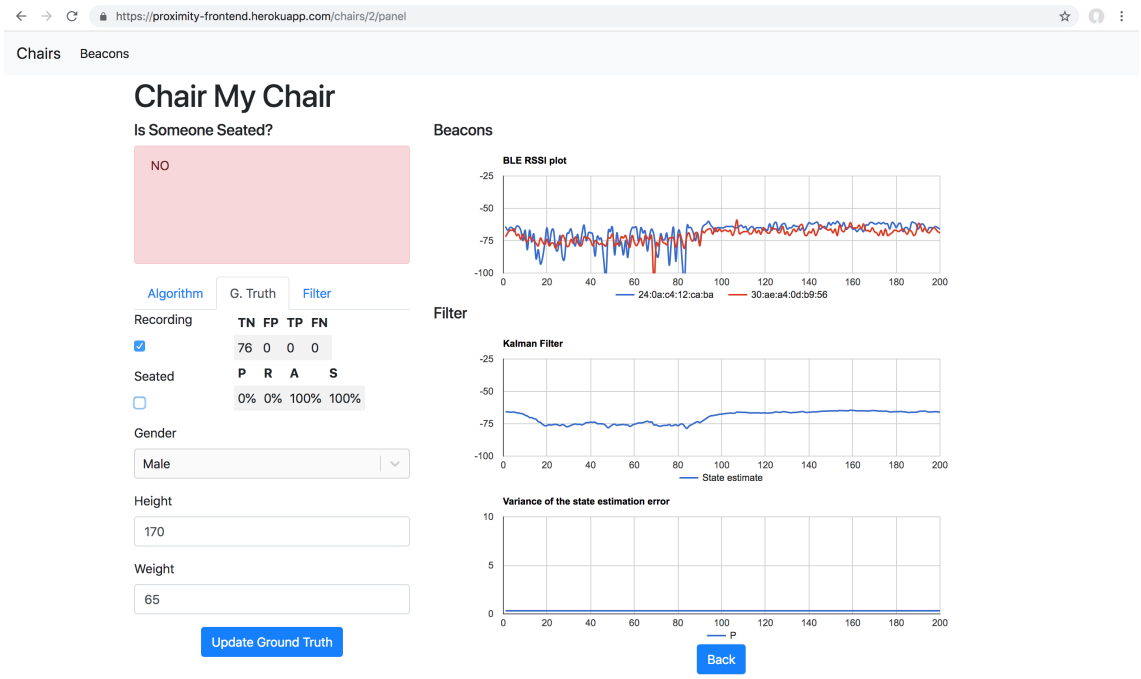


Figure 6.14: Panel page for chair collecting ground truth

There is one final panel shown in Figure 6.15 to adjust the process noise of the filter. This is to allow for experimentation during the test phase.

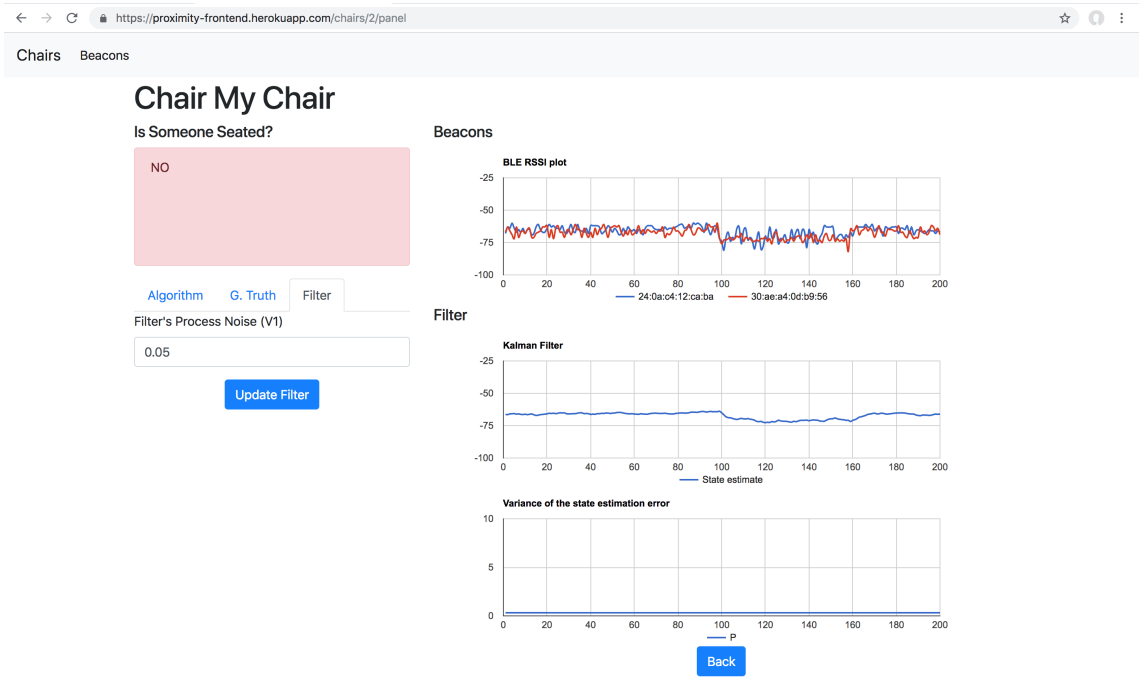


Figure 6.15: Panel page for chair filter panel

As mentioned before, the system is flexible enough to allow for a chair to not have a filter assigned to it as can be seen in Figure 6.16. In this case, if more than one beacon is assigned to the chair, the average of all measurements will be taken since there is no filter to perform sensor fusion. If the chair previously had a filter and it is removed, the chair will need to be calibrated again.

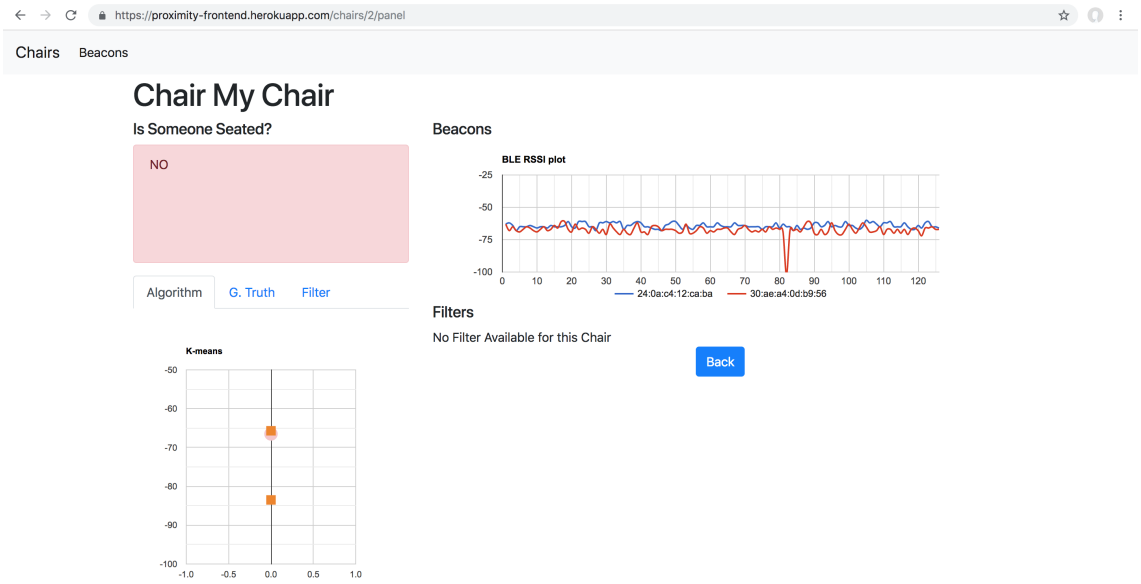


Figure 6.16: Panel page for chair without a filter

6.3 Sequence diagrams

The sequence diagrams are useful to get a clearer sense of the inner workings of the system. Here are presented the diagrams for each of the use cases previously mentioned. Most of them are straight forward CRUD operations but there are two that stand out. The first one is the "Register Measurement" seen in Figure 6.17 which depicts how a measurement is sent to the server, stored, and how it goes through the filter and through the algorithm to then become a prediction which is in turn stored as well.

The second one is the "View Panel" seen in Figure 6.24, which shows how the chair panel in the user interface fetches its data. It does it in an endless loop and it updates both the data from the beacons and the results stored in the predictions table that come from the filter and algorithm.

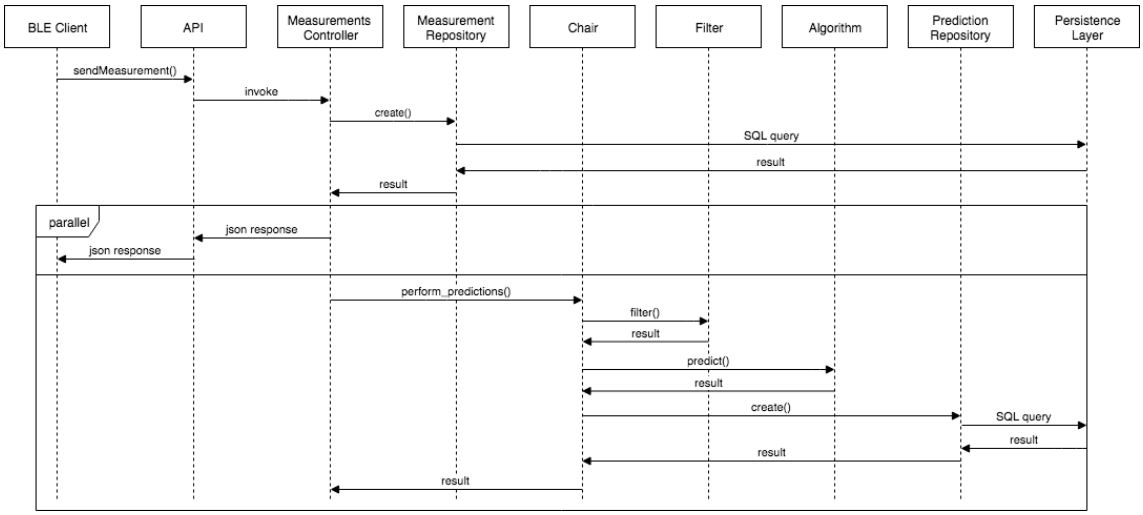


Figure 6.17: Sequence diagram for "Register Measurement" use case

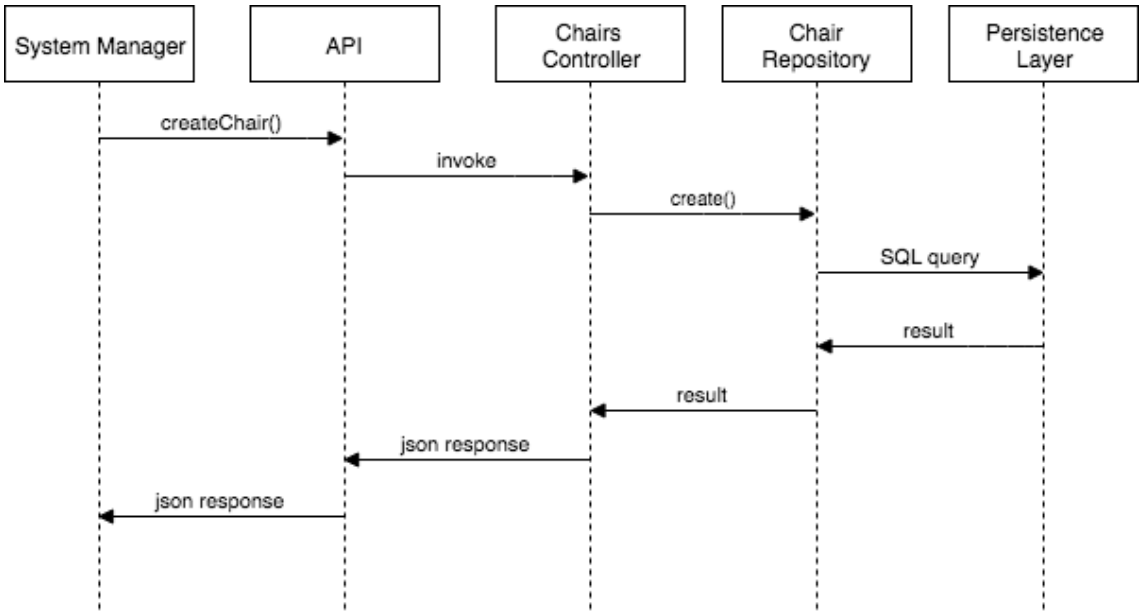


Figure 6.18: Sequence diagram for "Create Chair" use case

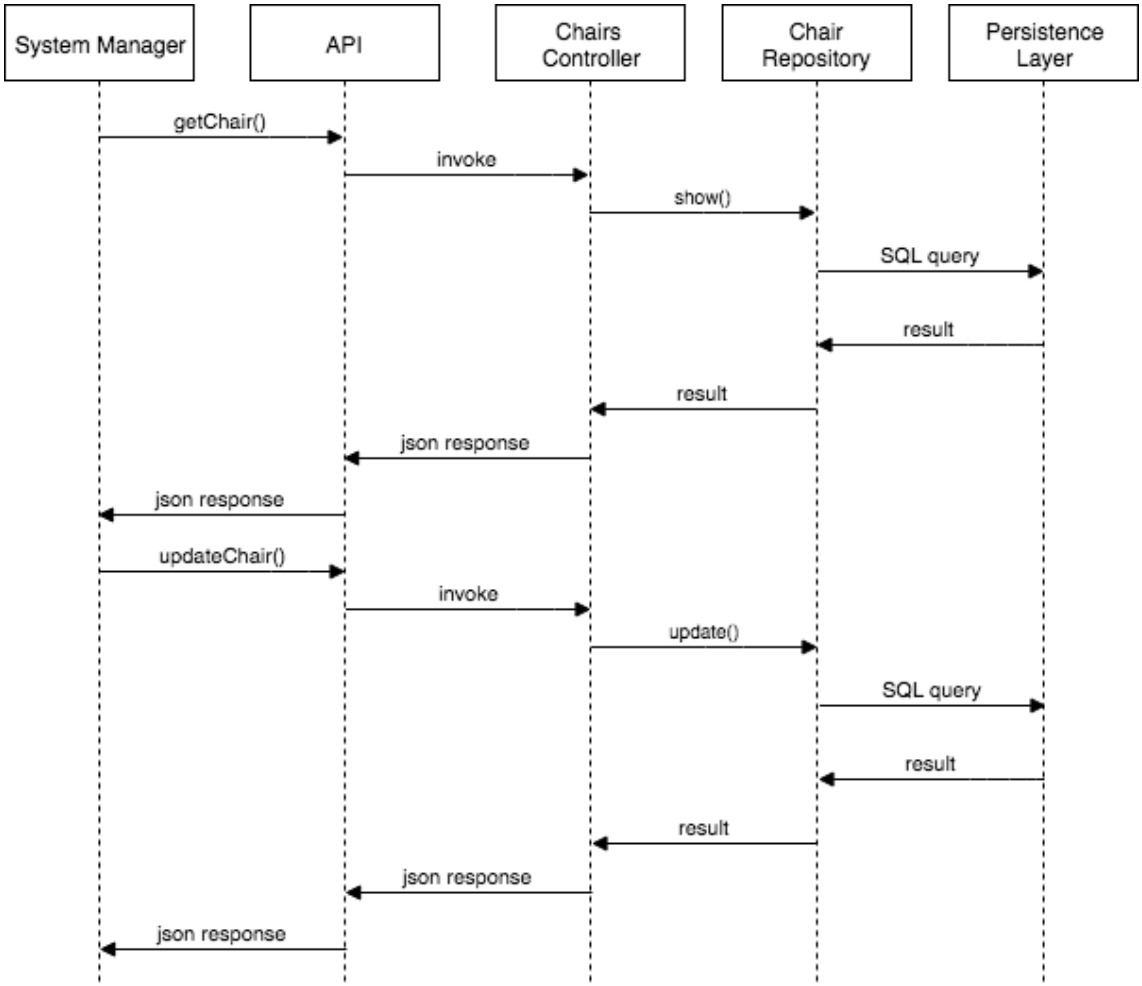


Figure 6.19: Sequence diagram for "Edit Chair" use case

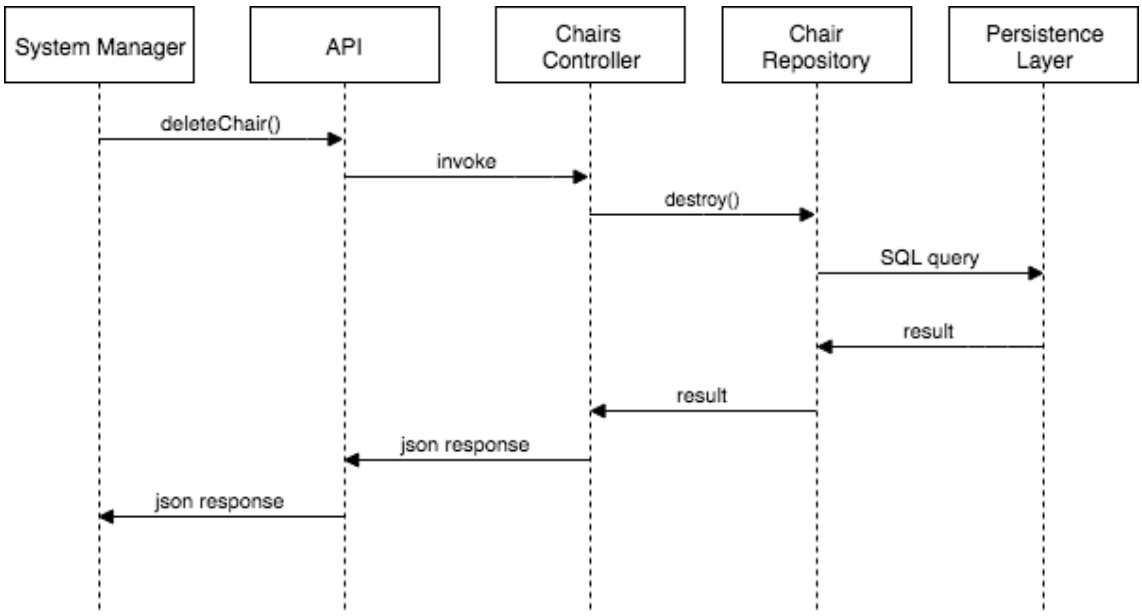


Figure 6.20: Sequence diagram for "Delete Chair" use case

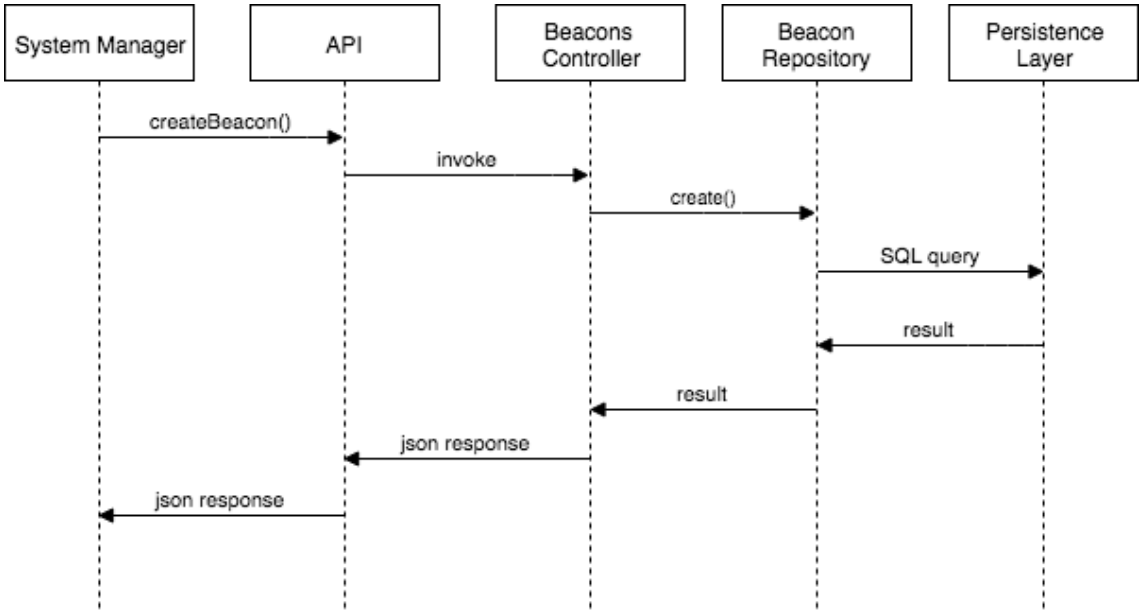


Figure 6.21: Sequence diagram for "Create Beacon" use case

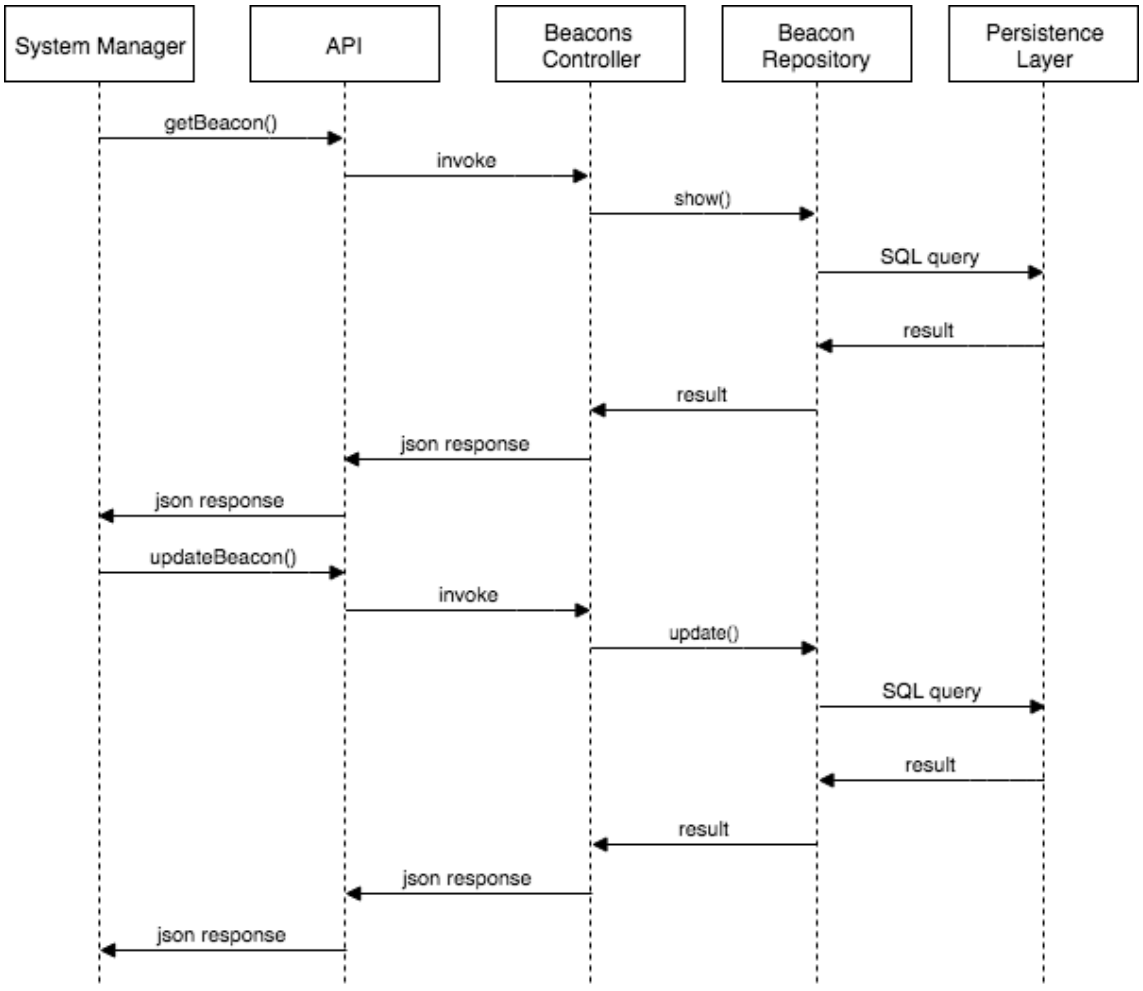


Figure 6.22: Sequence diagram for "Edit Beacon" use case

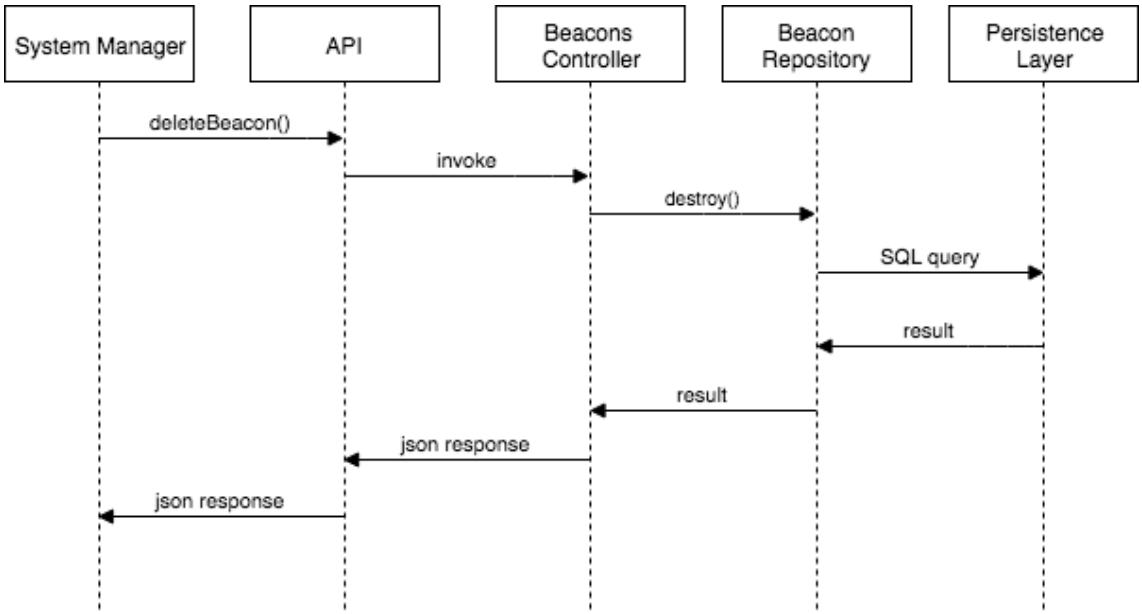


Figure 6.23: Sequence diagram for "Delete Beacon" use case

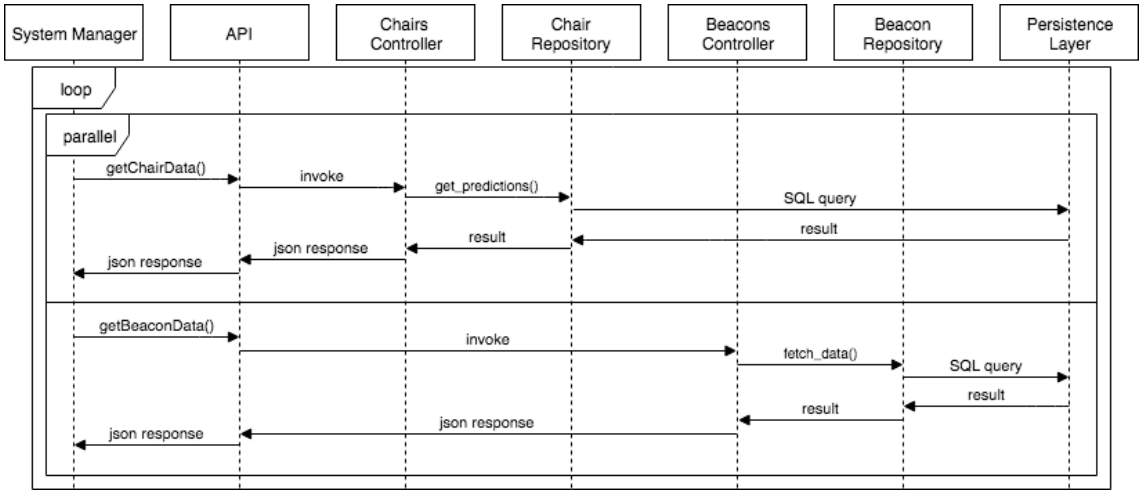


Figure 6.24: Sequence diagram for "View Panel" use case

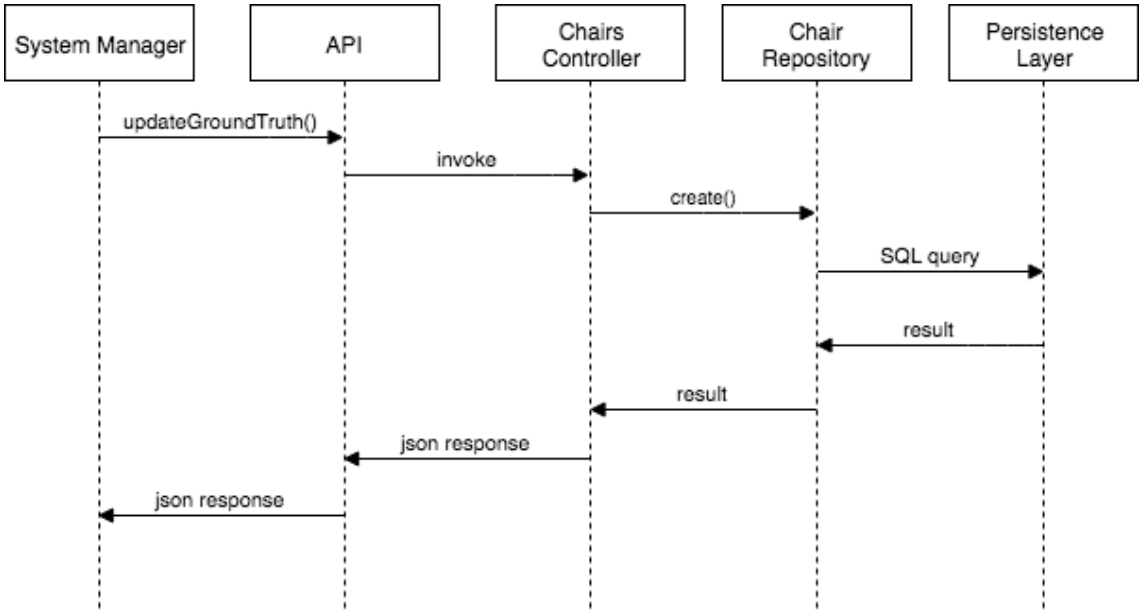


Figure 6.25: Sequence diagram for "Record Ground Truth" use case

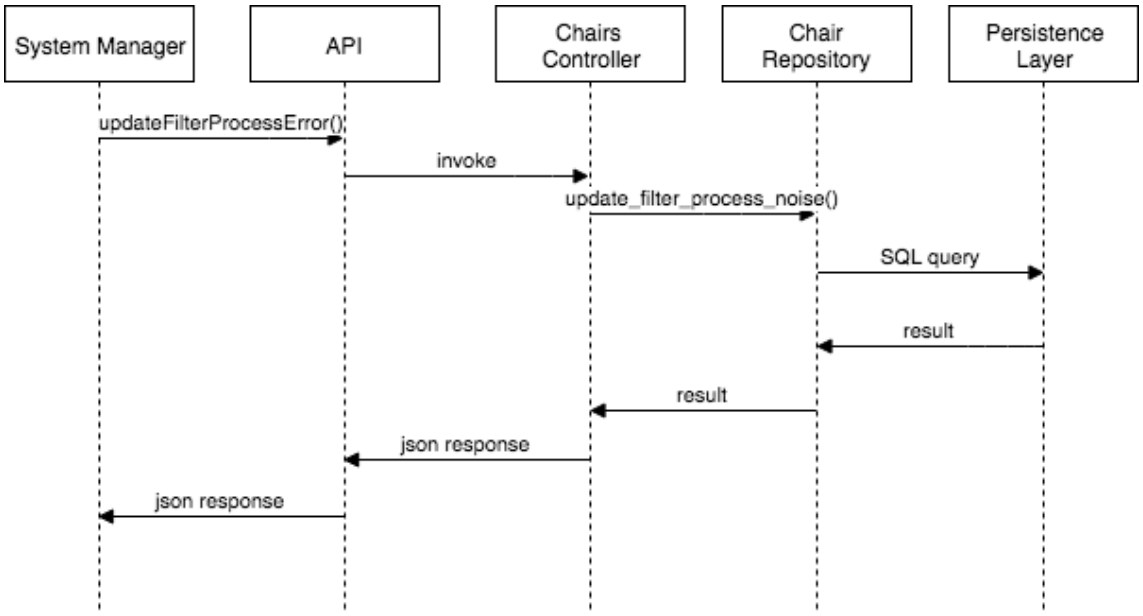


Figure 6.26: Sequence diagram for "Update Filter" use case

Chapter 7

Conclusion

This work sets out to try to evaluate the feasibility of using Bluetooth Low Energy technology as a non intrusive solution for proximity detection of a human being.

The system developed in this project can have many applications. One of them is in the framework of Assistive Technologies, retrieving information about patterns in the behaviour of people that need to be monitored. Information such as how long does a person spends sitting down.

For this particular task, BLE beacons are specially useful due to their low cost and low energy nature. Making them ideal in the domestic setting.

Using the RSSI calculated with the Friis transmission equation, a good understanding of what is happening around the devices can be constructed. This is due to the fact that when a human being is standing in between the client and the receiver, the signal gets distorted. These measurements are sent to a web server that takes care of analyzing them in real time.

The structure of the chair plays an important role and therefore more than one beacon per chair may be needed.

The BLE signal is noisy. It is affected by several environment variables. Most of them very common in an everyday setting.

The Kalman filter implemented in this project, not only reduces the noise of the signal, but also does a very good job at incorporating the measurements of other beacons.

Adding more beacons to the chair does improve the estimate of the state of the system. This was observed in the reduction of the variance of the state estimation error.

The filter proved to be a important part of the system for the objective of improving the precision of the predictions, made by the machine learning algorithm, in very noisy environments, i.e. reducing the amount of false positives.

A machine learning algorithm is used to learn the behaviour of the signal, in particular an unsupervised one.

An "online" variation of K-means was implemented to perform the task of learning the behaviour and making predictions. The algorithm performed well and it plays a key role in the system.

A web API was created to receive and store these measurements. During this process the value received by the API is then fed to the filter and to the machine learning algorithm.

To visualize and manage the system, a web interface was created. Here, all of the

information can be observed to monitor how all the moving pieces are performing, and make any adjustment as needed.

Based on the results obtained by the experiments performed, it can be concluded that **there is no single configuration that fits every scenario**. This is due to the nature of the problem. There are too many variables to take into consideration, especially given the fact that a single house commonly has many different types of chairs.

It is perfectly possible to encounter a situation where the process noise of the filter might need to be increased, or a new beacon added, or maybe remove the filter altogether.

This is why the main aspect of the system, built like this by design, is the fact that it can be configurable. All of the tasks described above can be performed by the person managing the system and tailor them to their specific needs.

Even though there is no configuration that performs well in every scenario, it is important to note that in all the tests performed, a good configuration has been found. This means that the system is flexible enough to yield good results even in an adverse context.

It can be concluded that the objectives set out in the project were accomplished and the use of BLE technology can be a viable solution for the proximity detection of a human being.

7.1 Limitations

The system performs well once a good configuration is found. Having said that, if after the calibration phase ends, the chair is moved, if this new setting varies a lot from the initial one, then the system will have to be re calibrated.

This is because the variance of the signal plays an important role in the creation of the filter. Plus the placement of the clusters of the K-means algorithm is key to its performance, due to the fact that the algorithm is susceptible to local minimum.

7.2 Future works

To tackle one of the issues mentioned above in the limitations section, the implementation of a Kalman filter that calculates recursively the value for V_2 may be needed.

Replacing the esp32 and power bank architecture is definitely an opportunity for improvement. Creating a custom solution could translate in the system being easier to install on a particular piece of furniture.

An algorithm that detects if a particular chair has been moved from its initial position would prove to be very useful, especially for implementing centralized architectures where there is only one client per room instead of one per chair.

Another opportunity for improvement would be to change from HTTP which is a TCP protocol, to UDP. This is because the client does not need to know about the response of the API, it only needs to send its information. By switching to UDP, the rate at which the measurements are collected can be increased.

References

[1] Arthur Ejsmont. *Web Scalability for Startup Engineers*. McGrawHill, 2015.

[2] Ki Hwan Eom et al. "Improved Kalman Filter Method for Measurement Noise Reduction in Multi Sensor RFID Systems". In: *Sensors* 11.11 (2011).

[3] *ESP-32 Module*. URL: `https://www.espressif.com/en/products/hardware/esp-wroom-32/overview`. (accessed: 12.10.2017).

[4] Boby George et al. "A Combined Inductive–Capacitive Proximity Sensor for Seat Occupancy Detection". In: *IEEE Transactions on Instrumentation and Measurement* 59.5 (2010).

[5] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, 2009.

[6] Li Hou et al. "Human detection and tracking over camera networks: A review". In: *IEEE International Conference on Pervasive Computing and Communications (PerCom)* (2016).

[7] Stephen S. Intille, Kent Larson, and Chuck Kukla. "Just-In-Time Context-Sensitive Questioning for Preventative Health Care". In: *Proceedings of the AAAI 2002 Workshop on Automation as Caregiver: The Role of Intelligent Technology in Elder Care* (2002).

[8] Gareth James et al. *An Introduction to Statistical Learning with Applications in R*. Springer, 2013.

[9] Eun Som Jeon et al. "Human Detection Based on the Generation of a Background Image by Using a Far-Infrared Light Camera". In: (2015).

[10] Roger R Labbe Jr. *Kalman and Bayesian Filters in Python*. URL: `https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python`. (accessed: 12.07.2018).

[11] T. Kanungo et al. "An efficient k-means clustering algorithm: analysis and implementation". In: *IEEE Transactions on Pattern Analysis Machine Intelligence* (2002).

[12] *Lecture Notes on Data Science: Online k-Means Clustering*. URL: `https://www.researchgate.net/publication/281295652_Lecture_Notes_on_Data_Science_Online_k-Means_Clustering`. (accessed: 12.08.2018).

[13] C. Marselli et al. "Application of Kalman filtering to noise reduction on microsensor signals". In: *Proceedings of the Colloque interdisciplinaire en intrumentation* (1998).

[14] Tom Mitchell. *Machine Learning*. McGrawHill, 1997.

[15] Alessandro Montanari et al. “A Study of Bluetooth Low Energy Performance for Human Proximity Detection in the Workplace”. In: *IEEE International Conference on Pervasive Computing and Communications (PerCom)* (2017).

[16] Andrew Ng. *Machine Learning Yearning*. 2018.

[17] H.P. Ng et al. “MEDICAL IMAGE SEGMENTATION USING K-MEANS CLUSTERING AND IMPROVED WATERSHED ALGORITHM”. In: *2006 IEEE Southwest Symposium on Image Analysis and Interpretation* (2006).

[18] World Health Organization. *World report on ageing and health*. World Health Organization, 2015.

[19] UNLV Department of Physics and Astronomy. *The Gaussian distribution*. URL: `http://www.physics.unlv.edu/~jeffery/astro/statistics/gaussian.html`. (accessed: 10.10.2018).

[20] Prof. Luigi Piroddi. *Model Identification and Adaptive Systems course slides*.

[21] Parisa Rashidi and Alex Mihailidis. “A Survey on Ambient-Assisted Living Tools for Older Adults”. In: *IEEE Journal of Biomedical and Health Informatics* 17.3 (2013).

[22] Simo Sarkka. *Bayesian Filtering and Smoothing*. Cambridge University Press, 2013.

[23] J.Z. Sasiadek and Q. Wang. “Sensor Fusion Based on Fuzzy Kalman Filtering for Autonomous Robot Vehicle”. In: *Proceedings of the 1999 IEEE International Conference on Robotics Automation* (1999).

[24] D. Sculley. “Web-Scale K-Means Clustering”. In: *Proceedings of the 19th international conference on World wide web* (2010).

[25] Kyohei Sugino et al. “Developing a Human Motion Detector using Bluetooth Beacons and its Applications”. In: *Information Engineering Express* 1.4 (2015).

[26] Shu-Li Sun and Zi-Li Deng. “Multi-sensor optimal information fusion Kalman filter”. In: *Automatica* 40.6 (2004).

[27] Takeshi Togura et al. “Long-Range Human-Body-Sensing Modules with Capacitive Sensor”. In: (2009).

[28] Fabio Veronese et al. “Realistic human behaviour simulation for quantitative ambient intelligence studies”. In: *Technology and Disability* 28.4 (2016), pp. 159–177.

[29] Wikipedia. *Central limit theorem*. URL: `https://en.wikipedia.org/wiki/Central_limit_theorem`. (accessed: 10.08.2018).

[30] Adriaan S. Zeeman et al. “Capacitive seat sensors for multiple occupancy detection using a low-cost setup”. In: *2013 IEEE International Conference on Industrial Technology (ICIT)* (2013).