INSTITUT FÜR
NACHRICHTENTECHNIK

Prof. Dr.-Ing. Gerhard Bauch

Master's Thesis

November 2016

Ranging and Positioning in Indoor Environment using Ultra-Wideband Radios

Author: Pablo del Hoyo
Supervisor: Prof. Dr.-Ing. Gerhard Bauch

TUHH
Technische Universität Hamburg-Harburg

I, Pablo del Hoyo Rodríguez, hereby declare that I have written this Master's Thesis independently, and that I have not made use of any aid other than those acknowledged in this master's thesis. Neither this Master's Thesis, nor any other similar work, has been previously submitted to any examination board.

Summary

Resumen

Este documento incluye el Trabajo de Fin de Máster de Pablo del Hoyo. En primer lugar se introduce la base teórica del proyecto. Como se detalla más adelante, el surgimiento de las tecnologías con conciencia de localización ha provocado que las redes inalámbricas de sensores adquieran mayor importancia. Se estima que este tipo de redes crezcan exponencialmente, siendo empleadas en una amplia variedad de campos como la medicina, el comercio, etc. Dentro de estas redes, el propio estado de los dispositivos cobra especial importancia. En este contexto, el estado de un dispositivo es entendido como un conjunto de parámetros del entorno que son medidos. En concreto, las coordenadas del dispositivo serán muy utilizadas.

Determinar las coordenadas de un nodo en una red es un problema muy común en el ámbito de las telecomunicaciones. Este problema no es de fácil solución si no hay una comunicación directa entre los nodos empleados para localizar dicha posición. Esta situación se produce a menudo en interiores, donde la presencia de obstáculos como paredes dificulta la comunicación directa. La situación se puede empeorar si los nodos de la red están conectándose y desconectándose continuamente de la red.

Los sistemas con señales de banda ultra ancha son utilizados para posicionar. Este tipo de señales tiene unas características que las hacen muy adecuadas en entornos interiores sin ser especialmente cara su implementación. Además, las señales de banda ultra ancha obtienen una resolución de centímetros a la hora de localizar los dispositivos. Se han propuesto multitud de algoritmos para resolver el problema de la localización de los nodos en la red de una forma precisa. La mayoría de ellos determinan las coordenadas de un agente (un dispositivo cuya distancia es desconocida a priori) comunicándose con varios "anchors" (anclas en inglés, dispositivos cuya posición es conocida). Sin embargo, los agentes pueden comunicarse entre sí, produciendo algoritmos más precisos y robustos gracias a redes de localización cooperativas.

Además, varios dispositivos reales serán utilizados para estudiar este problema en detalle. En este proyecto se emplean los dispositivos de Pozyx, desarrollados por Pozyx Labs. Disponen de una serie de sensores, incluyendo una antena de banda ultra ancha que permite determinar las coordenadas de los dispositivos. Se desarrollará una serie de experimentos que revelarán la precisión de los algoritmos de localización en un escenario real, bajo diferentes circunstancias. De esta manera se podrá estudiar el error producido durante las medidas. Finalmente, las conclusiones más relevantes y las posibilidades del

trabajo futuro son presentadas.

Este trabajo está escrito en inglés y fue desarrollado como estudiante de intercambio en el programa Eramus 2015-2016 en la universidad técnica de Hamburgo (TUHH).

Palabras clave: UWB, interior, medidas, error, localización, posicionar, algoritmos de localización, Pozyx.

This document contains the work regarding the Master's Thesis of Pablo del Hoyo. First, the theoretical background will be introduced. As it will be detailed, the arising of location awareness technologies is provoking the increase of the importance of the Wireless Sensor Networks (WSN). It is estimated that these kind of networks will experience an exponential growth, becoming very common in a vast number of applications, including those related with medicine, tracking, commerce, etc. Within these networks, the own status of the participating devices is strongly relevant, understanding status as a set of parameters of the environment that are measured. Particularly, the position or coordinates of a device are specially important.

Determining the coordinates of a node is a common problem in a network. The problem can be more complicated if a direct link is not available between the nodes. This situations is usually produced in indoor environments. The problem gets more complex if the nodes are always connecting and disconnecting from the network.

For positioning, the Ultra-Wideband (UWB) communication systems are proposed. It has several features that make it very suitable for the indoor environment, with an easy and low cost implementation. Therefore, UWB signals provide a positioning accuracy of centimetres. In addition, a lot of algorithms have surged to determine the status of a node within the network precisely. Most of the algorithms determine the position of an agent (a device whose position is previously unknown) communicating with some anchors (devices whose position is already known). However, agents can also communicate between each other, resulting in a cooperative work to determine their position. This yields to better algorithms in terms of accuracy and robustness.

Moreover, real devices will be employed to study this problem in detail. This project uses a Pozyx tag, developed by Pozyx Labs. These devices contain sensors, including an UWB antenna to determine the coordinates of the devices. A set of experiments with these devices will reveal the precision of the positioning algorithms in a real case scenario. This study is employed to correct the influence of the ranging errors. Finally, the most relevant conclusions and future work is presented, in order to continue with a technology that may provide many localization utilities in the future.

Keywords: UWB, indoor, ranging, positioning, location algorithms, Pozyx

Contents

List of Figures

List of Tables

Acronyms

AoA Angle of Arrival

AWGN Additive White Gaussian Noise

GPS Global Positioning System

GUI Graphical User Interface

ISM Industrial, Scientific and Medical

I2C Inter-Integrated Circuit

LOS Line of Sight

LS Least Squares

MAI Multiple Access Interference

NBP Nonparametric Belief Propagation

NLOS Non Line of Sight

SS Signal Strength

TDoA Time Difference of Arrival

ToA Time of Arrival

USB Universal Serial Bus

UWB Ultra-Wideband

UWC Ultra-Wideband Chip

WSN Wireless Sensor Networks

Chapter 1

Introduction

This document was written during the Summer Semester 2016 at the Technische Universität Hamburg-Harburg (TUHH), while the student took part in the Erasmus Program as an exchange student at Universidad Politécnica de Madrid (UPM).

Location awareness is already a crucial feature of wireless sensor networks and will also play a key role in future wireless networks. Next generation networks like 5G or Smart-Port-related sensor networks will extensively utilize location information. Especially in GPS denied environments, obtaining accurate and reliable location information is extremely challenging. Different localization schemes have been proposed in the past decades, including cooperative and non-cooperative, Bayesian and non-Bayesian, and many others. Lately, it was shown that cooperation between agent nodes can significantly improve the performance of localization schemes. Both accuracy and robustness can be increased significantly.

Ultra-Wideband (UWB) signals are shown to have beneficial properties for ranging in indoor environment. IEEE 802.15.4 standardizes a physical (PHY) layer for UWB transmissions. The Institute of Communications has purchased several 802.15.4a compliant UWB radios, which are augmented with inertial sensors, gyroscopes and pressure sensors. All parts are assembled on an Arduino compatible board. Off-the-shelf localization and tracking algorithms are implemented on the devices. Though, different algorithms with different requirements and localization algorithms should be implemented and compared in realistic environment. Particularly, some specific devices will be used in order to develop a real case scenario and performing a ranging measurement campaign.

In literature, numerous localization and tracking algorithms have been proposed. In terms of computational complexity, the least squares approach turns out be a good choice. This approach however shows inferior localization accuracy in networks with sparse infrastructure. In these networks, belief propagation turns out to be significantly better choice in terms of localization accuracy. The increased accuracy though comes at the price of increased complexity.

This Master's Thesis contains the basics of the algorithms related to the location within indoor environments with the most relevant information to understand how they have been developed and designed. In addition, a characterization of the error presented during the ranging measurements is established. The document presents the work of the student, and also shows the achievements accomplished by him. As explained, a set of devices have been used to implement a real case scenario.

1.1 Objectives

Considering this, the student has the task to implement the following localization and ranging campaigns on the given hardware:

- Measurement campaign to statistically describe the indoor localization ranging error.
- Localization
  - internal device algorithm
  - belief propagation

In more detail the following tasks have to be performed:

- Literature study to understand the concepts of belief propagation in the context of cooperative localization.
- Literature study to understand the difficulties of indoor ranging using estimations of the propagation time of the wireless signal using UWB.
- Conduct a measurement campaign to obtain statistical data of the indoor ranging error.
- The above-mentioned schemes should be implemented on the Pozyx nodes (Arduino compatible tags). The student has to get familiar with the hardware and the corresponding programming language.
- Evaluation should be conducted regarding the following figures:
  - positioning accuracy
  - and complexity
- A comprehensive documentation of code and scripts has to be provided.
- Summary of the results in a detailed documentation.

1.2 Structure

This document is structured as follows:

- Chapter 1 is this one: contains the overall description of the Thesis, its objectives and structure.
- Chapter 2 explains the background of the project, the main concepts which have to be taken into consideration in order to understand the functionality of the localization algorithms and the previous theoretical investigation associated with the project.
- Chapter 3 describes the methodology of the project, focusing on the Pozyx hardware employed in the research, as well as the software and the environment.
- Chapter 4 reproduces the practical experiments of the project, giving for each one an explanation, an objective and results.
- Chapter 5 deals with the possible future work and the conclusions of the investigation, providing some guidelines that may be followed in similar future projects.
- Appendix A contains some relevant parts of the code.
- Appendix B contains the tests which conform the experiments.
- Bibliography.

Chapter 2

Background

This chapter presents the major concepts which are related to the considered positioning algorithms. Firstly, the physical constraints are evaluated, which emphasize the benefits of UWB signals for ranging in indoor environments. Secondly, the importance of the location-aware technology is highlighted. This is needed to understand the theoretical foundation of the positioning techniques used in this project. Finally, the main source of localization errors for these algorithms is discussed.

2.1 Concepts

Some of the terms explained in this section will be used in the entire document. A list of concepts is defined so that the reader could consult any of these terms. If more than one term is defined for the same case, the *italic* terms are preferred. All of these terms are related somehow with the indoor **positioning** algorithms or the estimation of the **ranging** error or the Pozyx devices.

- **Anchor**: a device whose position is *a priori* known. It is always fixed.
- ***Agent*** or **tag**: a device whose position is *a priori* unknown. It is usually mobile.
- **Arduino**: manufactured board used to mount the Pozyx devices on.
- **Bias**: offset in ranging errors.
- **Node**: a participant in the network, either an anchor or an agent.
- ***Position***, **Location** or **State**: in this context, coordinates of a node.
- **Positioning**: process of estimating the position of an agent within the network.
- **Pozyx device**: a device mounted onto an Arduino which can be employed to develop a real case scenario.
- **Ranging**: process of estimating the distance between an agent and a node.

2.2 Ultra-wideband Radios

The UWB signals have some features that make them very suitable not only for telecommunication systems in general but also for **ranging**. Among these characteristics, UWB radios have absolute bandwidths of more than 500 MHz or relative bandwidths larger than 20%. Larger bandwidth provides a high reliability and improves the ranging accuracy, respectively. High reliability is produced as a consequence of the frequency diversity: as the signal contains many frequency components, it is probable that at least one of them reaches its target. Large bandwidth increases the accuracy due to high time resolution. High reliability and location accuracy also reduces the small scale fading since the information is spread over the different frequency components. This also makes UWB signals robust to narrowband interferences from other systems. Consequently, communications systems with higher robustness are achieved [SGS05].

UWB signalling is especially suitable for sensor networks because it allows centimetre ranging accuracy, with low-power consumption and low-cost implementation. UWB signals are employed in a vast area of applications such as logistics, security, medical, search and rescue, communications, etc. They are particularly suited for short to medium range communications, where the requirements of large bandwidth, robustness in multipath scenarios and centimetre accuracy may be demanded.

The attractiveness of UWB is higher in networks which require localization capabilities. The good time resolution facilitates to easy detection of the multipath components even in cluttered environments, mitigating the errors produced by this phenomena. The ranging accuracy scales with the bandwidth of the signals [GG05]. In addition, distance estimation accuracy can be improved in NLOS conditions (see Section 2.3.1 for more information).

A new physical layer for low rate communications combined with accurate ranging has been standardized in IEEE 802.15.4a. The physical layer defines three sets of frequencies in the following ranges: between 0.5 GHz and 1 GHz, between 3 GHz and 5 GHz, and between 6 GHz and 10 GHz. The data rates and transmission ranges of different standards can be observed in the Figure 2.1.

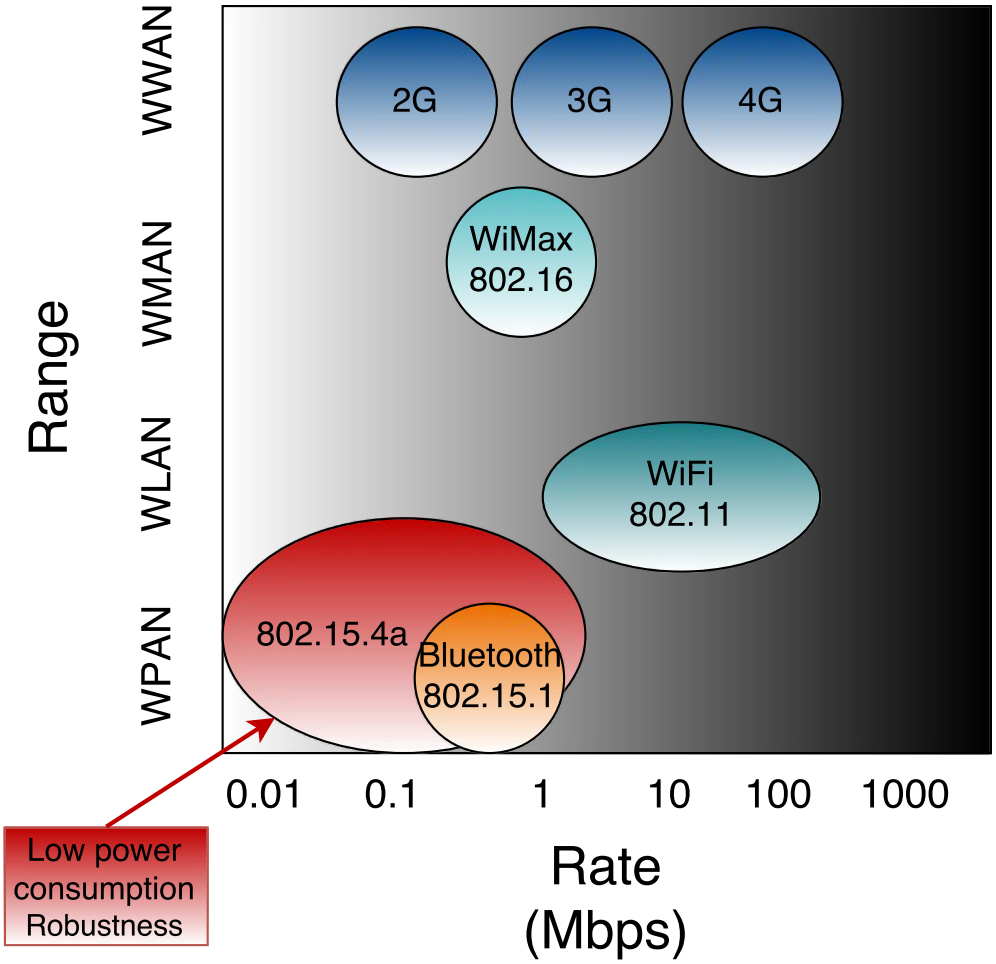

Figure 2.1: 802.15.4a comparison with another wireless standards.

2.3 Location-aware Technologies

Location-aware technologies are a key factor for wireless networks composed by several untethered agents. In these kinds of networks, the transmitted information may depend on the location where it is measured. This makes the **positioning** at nodes fundamental. These networks have an ad-hoc nature. In ad-hoc networks nodes disconnect or connect from or to the network. Consequently, a small amount of infrastructure is desirable. For that purpose, the nodes may share individual information to locate themselves within the network. In addition, undesirable conditions such as interferences or noise should be considered. The cooperative location algorithms are shown to produce great reliability and robustness against these circumstances [RMP03].

Location awareness is an essential feature in the Wireless Sensor Networks (WSN). Location-aware technologies are very suitable to obtain very accurate location estimates within the sub-meter scale context and under GPS denied conditions. The self-localization capability of the nodes must be needed if the own state of the nodes cannot be determined by a central administrator. Usually in this context, state is understood as a couple of coordinates, but it may include others such as orientation, the altitude and so on.

The localization process is usually divided in two stages: the measurement phase, where the agents obtain some position information by communicating with neighbouring anchors or agents; and the location-update phase, when the agents calculate their own position based on those measurements [NPC05]. The first phase consists of sending packets between neighbouring nodes. A receiver can extract localization information relative to the transmitter using some physical signal metrics with the first arriving path. The possibilities for gathering this kind of information are explained in detail in Section 2.3.1. Regarding the measurement phase, this project tries to characterize the uncertainty of the measurement that can arise due to the noise, multipath propagation, and NLOS conditions. The impact of these effects will be analyzed with a **ranging** campaign measurement.

In addition, the cooperative localization is gaining more and more attention. The main difference with respect to the conventional and cooperative localization is depicted in Figure 2.2. In this example, when an agent obtains distance estimates with respect to three anchors, trilateration is applied to determine its own position. The agents have received the position of the anchors during the first stage. If each agent cannot independently determine its own position based on the information gathered from the anchors, they could cooperatively find their positions. Thus, cooperative localization can increase localization accuracy and coverage [HWW09]. Summarizing, cooperative localization techniques provide methods to determine the position of some nodes with information from other agents.

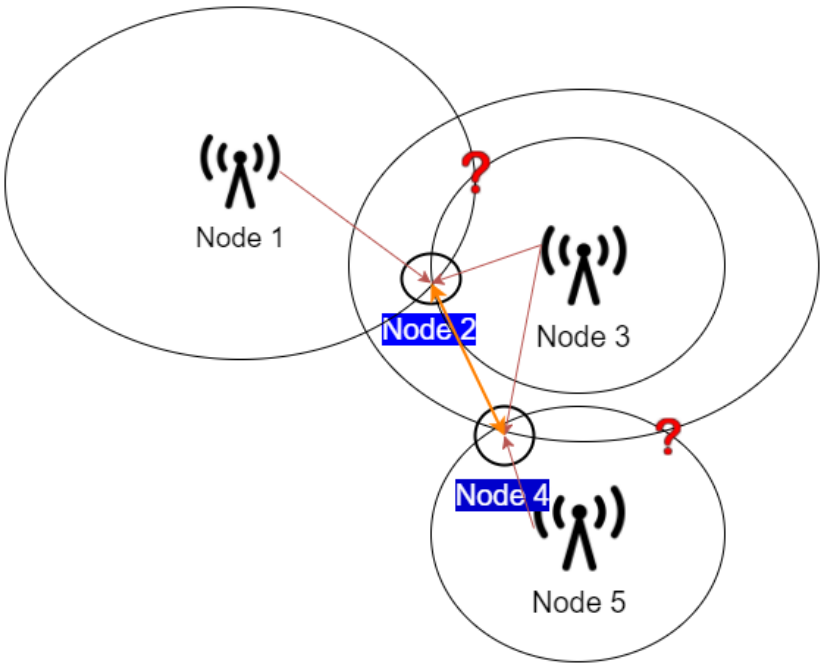

Figure 2.2: Cooperative localization. The agents (nodes 2 and 4) are not capable of determining their positions by only using the distances respect to the anchors. They may also be located in the spots marked with a question mark because not every agent reaches every anchor. However, if both agents cooperate and range directly (orange arrow) between each other, they unambiguously determine their coordinates.

2.3.1 Signal Metrics

The position of a node is determined given the information exchanged between an agent and either an agent or and anchor [DDW09]. The position of the receiver can be inferred with this information, using the metrics presented in this section.

Time-based Approaches

The separation between two nodes is inferred by the signal propagation delay. Assuming that the signal travels at a speed of c, the distance $\hat{d}$ can be estimated as:

$$\hat{d} = c \cdot \hat{t} \tag{2.1}$$

where $\hat{t}$ is the measured duration of the signal propagation.

These schemes provide good results because of the high time resolution if UWB signals are employed. They are less costly than AoA systems (see 2.3.1) systems and need less information than Signal Strength (SS) based systems (see 2.3.1).

In more detail, if both clocks of the transmitter and receiver are synchronized, the first transmitting node can include a timestamp which allows the receiver to determine $\hat{t}$, based on Time of Arrival (ToA).

There are mainly three methods for measuring the signal propagation delay:

- One-Way ToA: a node determines the time of arrival of a received a signal (timestamp t_2) from another reference node (timestamp t_1). If both of them have a common clock, the time of flight t_f is the difference between them ($t_f = t_2 - t_1$). Unlike the Signal Strength (SS) technique, the result is improved with the relative bandwidth but it is significantly affected by synchronization errors [HB01].

- Two-Way ToA: no common clock is required. Instead the round trip time ($t_{RTT} = 2t_f + t_d$) is measured. t_d is a known processing time and t_f the time of flight. To calculate the time of flight, a first node transmits a packet to a second one, which answers with an acknowledgement after a processing delay t_d. However, a relative clock drift between the nodes may influence the estimation accuracy.

- Time Difference of Arrival (TDoA): in this case, all the reference nodes are synchronized. Multiple signals are broadcasted from synchronized nodes (anchors) with all-time known positions. The agents can measure the Time Difference of Arrival (TDoA) from all of them. This solves the synchronization problem when the response delay is high. Another solution is to broadcast a reference signal. The anchors measure then the ToAs and share the information (typically through a wired connection) to calculate the TDoA. These techniques eliminate the need of common synchronized clocks of the nodes at the expense of more information exchanges. Receivers nodes still need synchronization.

Signal Strength

The distance between two nodes is calculated measuring the received signal power at one node with a path-loss model. This model can either be theoretical or empirical. The power received will be lower in the receiver when the distance between two nodes increases. This technique needs at least three anchors and requires a path loss model. There are many models to implement this technique which may provide different ranging accuracies [TPC06]. A common model is given by the equation below:

$$P_{rec}(d) = P_o - 10n_p \times \log \frac{d}{d_o} + S \tag{2.2}$$

where P_{rec} is the received signal power, P_o is the power measured at a reference distance d_o (a priori information), n_p is the path-loss exponent, and S the large scale fading variations (usually modelled as a Gaussian random variable).

Angle of Arrival

This technique is based on the measurement of relative pair of angles of the nodes from several reference nodes. Agents employ antenna arrays, which are located in a fixed orientation (see Figure 2.3). The potion of an agent can be inferred from these angles. This technique may be also known as direction of arrival.

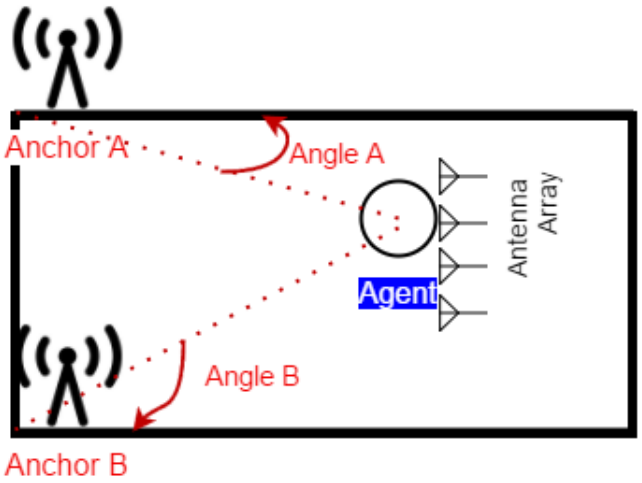

Figure 2.3: Angle of Arrival (AoA) technique. Agent infers its position from the angles received by Anchor A and Anchor B, respectively.

Connectivity

This technique considers a node to be within the range of communication of a reference transmitter if it is able to connect it. If a node receives a signal from a transmitter, it will be inside its coverage area. This area can be modelled as a circle, centred at the transmitter position. As a counterpart, disconnectivity may also provide information about the

position of a node given that it cannot receives a signal from a known transmitter. The possible positions could be confined d by the intersection of these coverage circles, when the node is connected to more reference nodes. This process can be seen in the Figure 2.4 below:

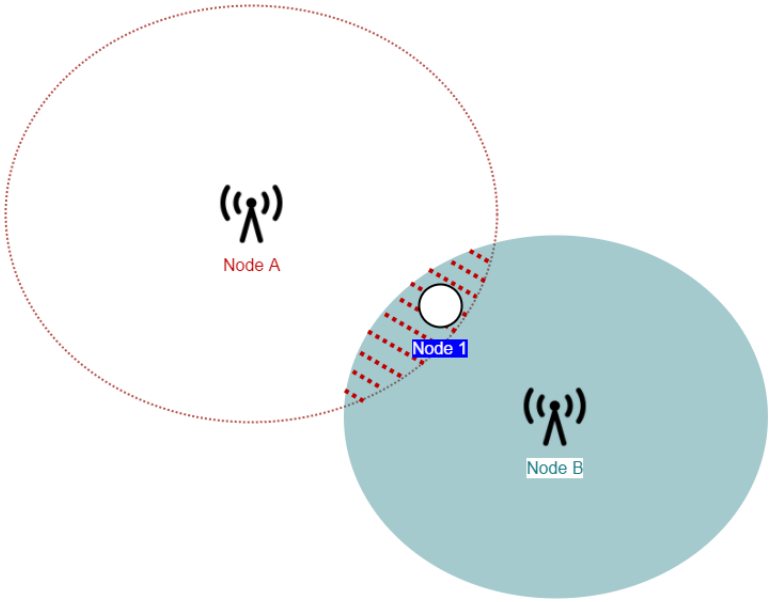

Figure 2.4: Positioning based on connectivity. The Node 1 infers from Node A an area (red) for its possible position. Get the same information from Node B (blue). Its actual position will be inside the intersection between these areas (dashed lines).

Hop-count

This technique assumes an average distance d_{NN} between nodes. The distance between the reference node and the receiver $\hat{d}$ can be calculated as $\hat{d} = N \cdot d_{NN}$, where N is the number of times a signal has passed through a node to reach the receiver. In Wireless Sensor Networks (WSN), the nodes can only communicate with nodes inside its coverage area. The nodes outside this area may only been reached by routing the information with intermediates, using multiple hops. The distance to them can be estimated by counting the number of nodes.

Pattern Matching

Pattern Matching or fingerprinting compares the received signal characteristics to a set of a database. This characteristics can include Time of Arrival (ToA), Angle of Arrival (AoA) and Signal Strength (SS) or a deterministic waveform. A fixed beacon for every node throughout the entire network is operated to get a pattern database. The resulting algorithm will be employed to determine the most closely match to an incoming signal. The distance is finally inferred by the pattern which matches best.

2.4 Sources of Errors

The sources of errors that may degrade the accuracy of the ranging techniques are studied in this section. The signal metrics can be affected by undesired circumstances.

The main source of errors while ranging are usually related to the multipath propagation, Multiple Access Interference (MAI), direct path excess delay and direct path blockage. The reflected signals fade and complicate the detection of the direct path [DDW09]. An example can be observed in the Figure 2.5 below:

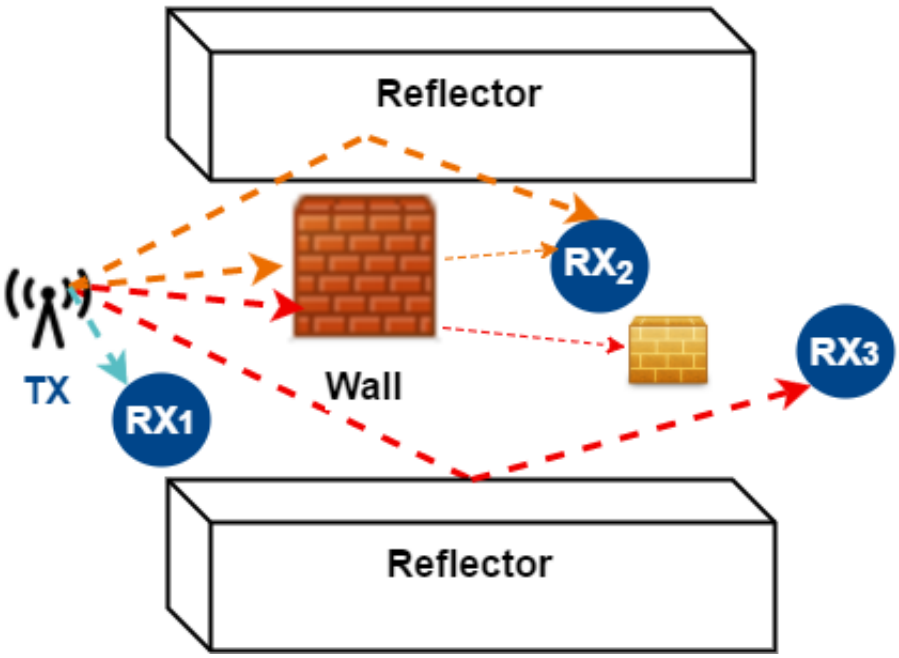

Figure 2.5: Source of ranging errors. The transmission to RX_1 is considered as **direct path** because the signal transmission has a constant and known electrical permittivity medium. There is no LOS blockage. The transmission to RX_2 is considered as **direct path excess delay** because the first arriving path has been attenuated. The transmission to RX_3 is considered as **direct path blockage**. LOS occurs when the first arriving path corresponds to the direct path. NLOS occurs when the direct path has an excess delay or it is blocked [DDW09].

2.4.1 Multipath Propagation

Multipath fading occurs when different signals arrive at the receiver thorough different paths. This causes constructive or destructive interferences. As a result, the detection of the direct path component can be difficult [WS98]. The received signal is given by the superposition of all riving paths. The UWB signals alleviate this problem as explained in Section 2.2.

The received waveform $s(t)$ of a given pulse $p(t)$ is given by:

$$s(t) = \sqrt{E_p} \sum_{l=1}^{L} \alpha p(t-\eta) + n(t) \tag{2.3}$$

with L multipath components, α and η amplitude and delay of each component, and E_p the average received energy. This channel response assumes an unitary energy pulse $p(t)$ with duration T_p and $n(t)$ as Additive White Gaussian Noise (AWGN).

To minimize the error, the ToA of the first path when η is equal to η_1 has to be derived.

Direct Path Excess Delay. The direct path excess delay is a consequence of the partial obstruction of a direct path component. The signal will be affected not only by the distance but also by the material of some obstacles. The signal delay is produced since the propagation is slower depending on the material. The propagation of electromagnetic waves is slower compared to air in some mediums. This produces a direct path excess delay. These waves may even be blocked, which it is known as a direct path blockage. This error ($\Delta\tau$) can be estimated with the following expression:

$$\Delta\tau = (\sqrt{\varepsilon_R} - 1)\frac{d_W}{c} \tag{2.4}$$

where ε_R is the relative electrical permittivity of the material and d_W is the thickness of the obstacle.

Direct Path Blockage. In this case there is no direct path component at all. Only the NLOS components of the signal are received. This results in overestimated distances. NLOS propagation occurs when the direct path is completely blocked or excesses a detection time delay. Only reflections of a signal reach the receiver. The distance travelled under these circumstances is different to the straight line distance. The main challenge is to mitigate and detect this condition. Some pattern recognitions algorithms can be employed when there is no information about the NLOS error. For example, the information is gathered from some beforehand known locations. The NLOS delays can be inferred due to the high variance of the delays compared to the previous LOS measurements.

2.4.2 Clock Drift

In time-based systems, accurate estimates are a prerequisite [SY04]. These estimates are not only influenced by the transmission medium frequency channel, but also by the drift of the antenna clock.

The local time of a node is given by the following equation:

$$C(t) = (1+\delta)t + \mu \tag{2.5}$$

where δ is a drift relative to the correct time and μ is an offset.

As a result, the time difference in the internal clocks used to calculate an estimated distance is given by the expression:

$$\hat{\tau} = C_2(t) - C_1(t) = \tau(1+\delta) \tag{2.6}$$

where δ plays a major role.

2.4.3 Multiple Access Interference (MAI)

Other nodes may interfere with a reference signal. The UWB signals have to coexist with different systems. As a result, some narrowband and multiple user interferences are expected.

The narrowband interferences are given by the expression:

$$i(t) = \sqrt{2I}\alpha_I \cos(2\pi f_i + \phi_I) \tag{2.7}$$

being I the average received power, f_I the center frequency of the narrowband interference, α_I the amplitude of the fading associated with the interference and ϕ_I the phase of the interference.

The multiple user interference depends on the transmitted signals structure and the channel characteristics. These effects can reduced using medium access control.

2.4.4 Errors in Different Ranging Techniques

The major degradation of *time-based* techniques is due to the multipath propagation. This condition provokes the superposition of various paths of the incoming signal under several changing conditions (including attenuation and delay). Thus, the detection of the direct path is not easy. If it is not detected, a positive bias to the ranging measurement will occur.

The *Signal Strength (SS)* techniques have been very used in Wireless Sensor Networks (WSN) because the hardware requirements are less demanding. However, the main disadvantage is that the large bandwidth of UWB signals is not actually exploited. To alleviate this, sometimes it is combined with ToAs techniques. Environments with many obstacles are often not very suitable for Signal Strength (SS) techniques because the multipath propagation and direct path blockage attenuates the signal. Consequently, shadowing is the major source of errors in Signal Strength (SS) based systems.

The main disadvantage of *Angle of Arrival (AoA)* techniques is the cost and size of the arrays. In addition, the number of paths may be very high. A complication arises from NLOS conditions due to the complexity of direct path detection [Lie07]. Intuitively, the positioning accuracy of this technique highly depends on the number of antennas. Consequently, it is not suitable for systems with ad-hoc nature or large number of nodes. Thus, AoA technique is not scalable.

Chapter 3

Methodology

This chapter explains the functionalities of the Pozyx devices and how they communicate. First, a brief description of a Pozyx device is given. Based on this, the functions of the device are detailed. Finally, the environment is shown.

3.1 Pozyx Devices

Several Pozyx devices are employed in a positioning system. A Pozyx device is composed by several sensors. It is mounted onto an Arduino-compatible board and provides several functionalities that are controlled using the Arduino. Particularly, contains an UWB chip which generates UWB signals and an antenna that radiates them. The UWB signals can be used to get distance estimates to another node. It can also interact with some fixed anchors or agents to determine its position. With Pozyx devices, it is possible to send custom data packages to wirelessly interact with the registers of the remote Pozyx device. This allows the control of that Pozyx device. The register of a Pozyx device may contain either a function or some data [Poza]. Thus, the receiver node may be remotely accessed and be given some instructions. These instructions are transmitted using an UWB signal (see Section 2.2).

The picture 3.1 shows a real Pozyx device, with a micro-controller unit and an Ultra-Wideband Chip (UWC).

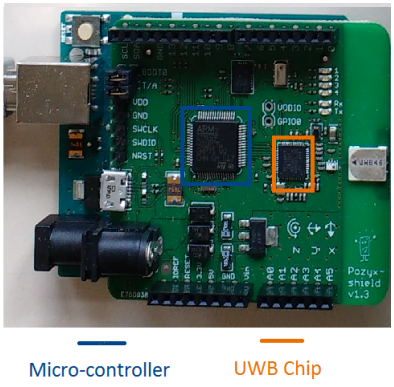

Figure 3.1: A real Pozyx device.

The schematic Figure 3.2 shows the functionalities and units of a Pozyx device. The main part of the board is the micro-controller unit that provides all the Pozyx device localization functionalities. The functions are contained in the registers. The micro-controller communicates with all the on-board sensors. It performs **ranging**, **positioning** and calibration algorithms. The micro-controller also provides an interface to the external Arduino through the Universal Serial Bus (USB), Inter-Integrated Circuit (I2C) protocol and an interrupt line. For a lower processing time, the micro-controller runs a real-time operating system. This achieves that all the functionalities are executed with little delay and high reliability. The board contains various sensors which are out of the scope of this project. The board is also equipped with an UWB chip that provides the wireless transmission and ranging capability of the Pozyx device.

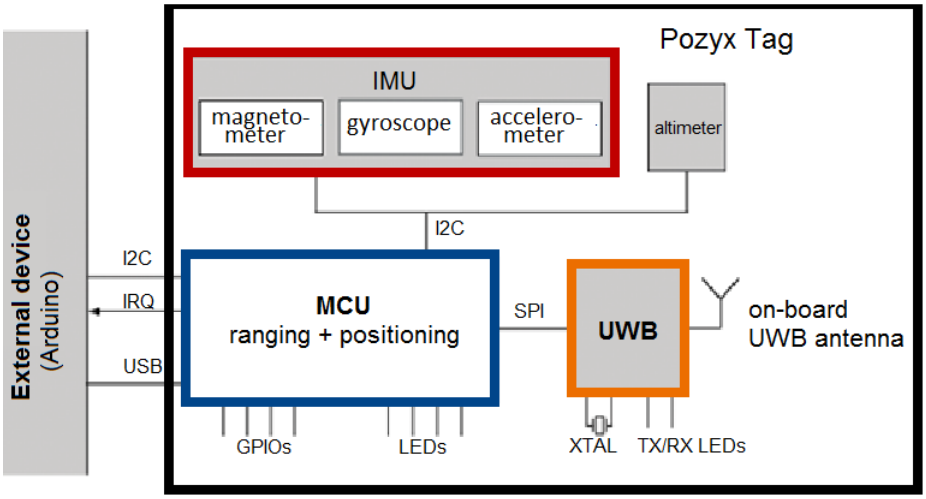

Figure 3.2: Pozyx device. Schematic Figure.

For the **ranging** objectives of this project,

- A Matlab interface will be programmed to initiate ranging with Pozyx devices. This interface will show the most relevant information about the ranging error in real time.
- Several ranging experiments will be conducted to get some statistical information of the distribution of the ranging errors depending on various circumstances.
- The difference between the actual distance and the estimated by the Pozyx device will be studied.

For the **positioning** objectives of this project,

- A Matlab interface will be programmed to customize and calibrate a scenario.The selection of distinct positioning algorithms will be available. Position estimates in real time of a node will be displayed in the map of the scenario.
- The Poxyz localization algorithm, as well as the Nonparametric Belief Propagation (NBP) algorithm will be available. The NBP algorithm is considered as a black box and its programming is out of the scope of the project [ATIW05]. The different location results will be observed in a plot.

3.2 Ranging

For the ranging experiments, a Graphical User Interface (GUI) for Matlab is programmed. It enables ranging between a pair of Pozyx devices. The GUI provides the functionalities to restrict the time or the number of ranging samples. A figure of histogram of ranging estimates is displayed.

3.2.1 Prerequisites

The program will enable a communication between the Arduino board and a computer which is equipped with Matlab. Two parts will be considered in order to make the installation: the Matlab side and the Arduino side (see Figure 3.3).

Figure 3.3: Interface. Both sides of the program.

For the Matlab side it is required:

- Matlab R2016a (older versions may also work).
- Arduino Support for Matlab: package enable communication at an Arduino with Matlab. The installation steps are available at reference [Pozb].

For the Arduino side is required:

- Arduino Uno x1.
- USB-B Cable x1.
- Pozyx device for Arduino.
- The Arduino IDE and Pozyx Arduino Library: the installation steps are provided at reference [Poza].
- Pozyx anchor x1 (or another Pozyx device).
- USB-C Cable x1 (or another USB-B Cable and Arduino UNO).
- Laser Meter x1.

3.2.2 Problem Setup

A C program is uploaded onto the Arduino board to set the ranging experiments parameters. It makes use of a library which performs the ranging function. Using a USB connection, the code (appendix A.1.1) can be uploaded. The Pozyx device will start ranging and obtaining distance estimates to the remote node [Pozd]. Once the remote device register is called to estimate the distance from the source node calling it, it will answer accordingly. The range estimates are transmitted to Matlab with the Arduino. The communication process between the two Pozyx devices is shown in the Figure 3.4.

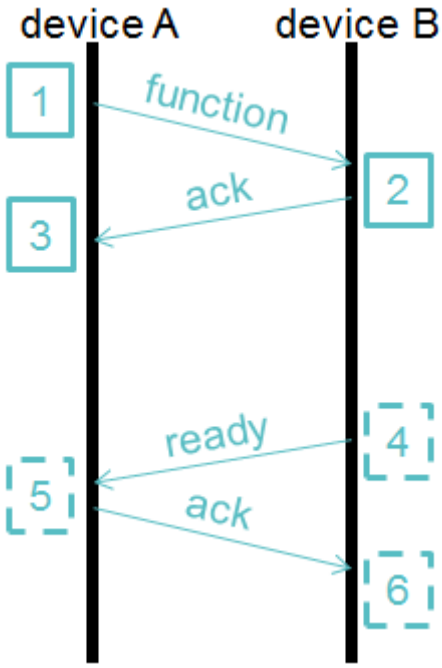

Figure 3.4: Pozyx devices communication. The device A initiates the process and calls a remote register function from the device B (Pozyx device transmission send). The function can represent either a ranging or positioning functionality. The device B will send an acknowledgement of the message if it is correctly received. The rest of the steps are optional and provide some reliability features in the communication.

When ranging between two nodes, the network id of the device B and the register function to be called are some of the input parameters. The Pozyx device system provides single-hop wireless communication between neighbouring Pozyx devices making use of the UWB signals. Every device is uniquely identified by its 16-bit network identification (id) stored in the Pozyx device network id register. This direction is represented in hexadecimal format with the pattern 0x12AB. To make wireless communication between two Pozyx devices possible, they must have the same UWB settings and the receiver must be turned on.

[TUH16b] is referred to provide more details about the programme and the files which compose it. To execute the interface (*read.m*), some restrictions about the scenario have to be taken into consideration:

- The destination must be specified in the program *ready_to_range_arduino.ino* as well as the UWB channel (1, 2, 3, 4, 5 or 7). This can be done by editing the following lines:

```
// Edit with your scenario:
uint16_t destination_id = 0x6670;
int uwb_channel = 2;
```

- There is a minimum separation distance that allows the ranging. Distances lower than 20 mm usually provide a range estimator of zero. Distances greater than 100 m in LOS conditions are often undetected.

- There is some processing time in order to present all the information. As a result, the current data being read and that one being represented have an offset. The ranging results are not strictly represented in real time. However, all the estimates are considered when the ranging is completed (if it is limited by samples or time).

The external library defines numerous functions and an interface which can be used to remotely control a Pozyx device. During the main loop, the node commands the remote Pozyx device to range.

The Figure 3.5 shows the process described above. First, the program is uploaded onto the Arduino board using the Arduino IDE. The program employs the functions defined in an external library with an specified destination node as well as some UWB settings. Once this process is completed, both devices are ranging (continuous process bar). Nevertheless, the results cannot be read unless the serial monitor of the Arduino IDE is used, or, as in this case, a Matlab interface for this purpose is executed. It sends an open command to the Arduino that will read this data (between steps one and two). Once it is finished, the connection between Matlab and the Arduino is closed.

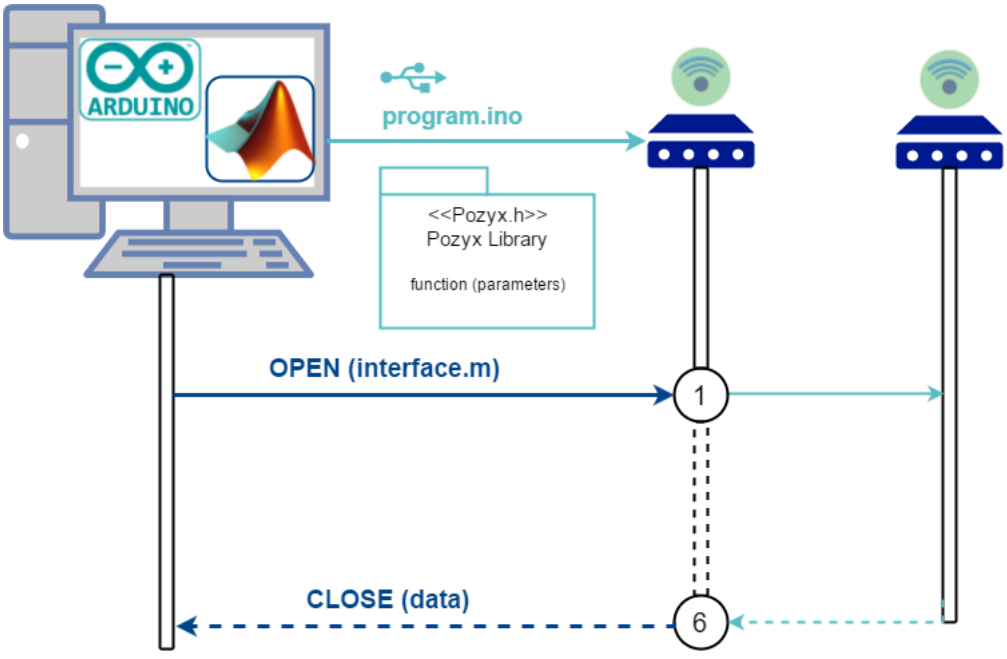

Figure 3.5: Pozyx ranging, program initializing the Arduino.

A screenshot of the interface is shown in the Figure 3.6:

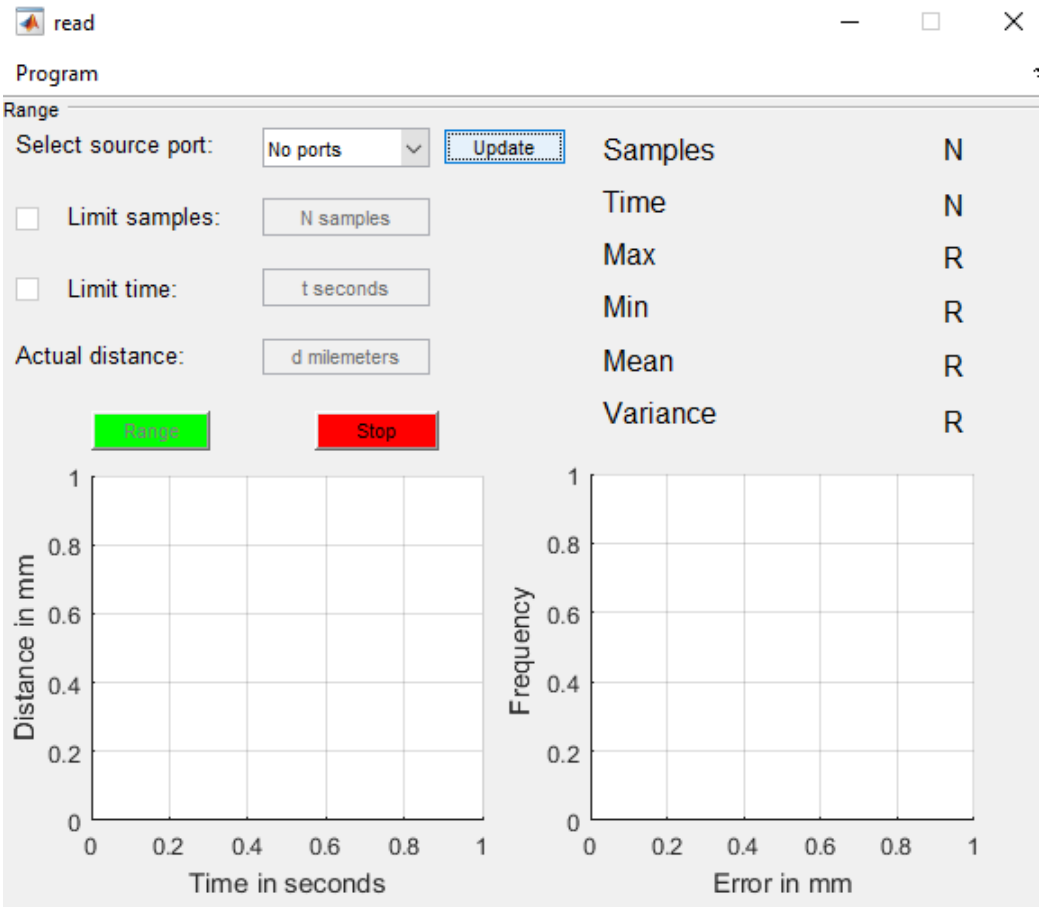

Figure 3.6: Pozyx ranging, interface of the program running on Windows 10.

The pop-up allows the selection of the USB port where the source Arduino board is connected (if available). The ports connected can be updated using the update button. It will show the new ports and remove those which are disconnected. The actual distance between the two devices can be set with the corresponding edit text box. The program initializes the Arduino with a baudrate of 115200, the destination id and UWB settings selected by the user before Matlab GUI opens a communication link with the Arduino. When the **Range** (see A.1.2) button is pressed, the node send this command to the remote Pozyx device (isDone == 0). It waits until a defined number of samples are successfully acquired (samplesAvailable == 1). The format of the answer is a matrix containing the number of the sample, the estimated distance and its time in each row. The matrix increases by one row for each measurement. This process (readData()) is continuously repeated (main loop) unless the number of samples or time for the test desired is reached (isDone == 1). In the meantime, a real time graph of the evolution of $\hat{d}(t)$ is shown, as well as the density function of the error $f(e)$ and its most relevant statistics, such as its mean, variance, maximum and minimum value. This process can be viewed in the Figure 3.7. Hence, the data read in the Matlab side is presented in a more intuitive way. The execution of the code can be stooped with the **Stop** button (isDone == 1).

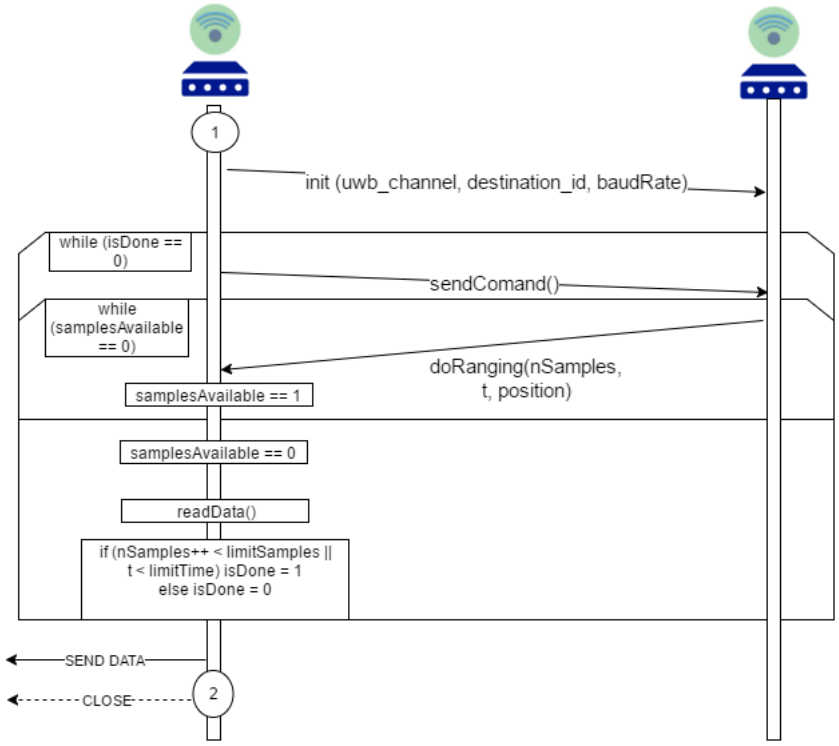

Figure 3.7: Pozyx ranging process

The actual distance between the Pozyx devices, denoted by d, was obtained through a laser meter, as shown as in the Figure 3.8.

Figure 3.8: Laser meter obtaining the actual distance.

At this point the Pozyx device obtains the estimated distance, which is called $\hat{d}$. The error is defined by the difference between this estimated distance and the real distance. Its evolution over time is shown following the next equation:

$$e(t) = \hat{d}(t) - d(t) \tag{3.1}$$

3.2.3 Output

Once the errors have been computed, the objective of the experiments is to compute how much these estimated distances diverge from the actual distance. The analysis of the distribution and statistical information of the error is performed subsequently.

For that purpose a test is defined. A test is nothing else but a ranging experiment under some fixed conditions such as the actual distance, the frequency channel, the LOS or NLOS conditions, and the antenna orientation. For each test the error over time $e(t)$ is shown. Based on the observed error, the density function is approximated by the histogram. Finally, a comparison of the error for every single test can be observed, gathering the variations and consequences while changing a condition when the rest of them are fixed.

The **ranging** experiments are performed in Experiment 4.1 and 4.2.

The GUI provides this analysis with the output data. The left plot of the Figure 3.6 shows the evolution of the distance over time, defined as $\hat{d}(t)$. Thus, the following graph represents the distance between the source and destination nodes calculated by the Pozyx UWB chip:

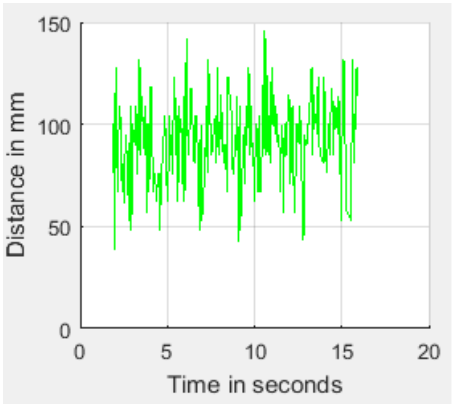

Figure 3.9: Pozyx, distance $d(t)$ over time.

The right plot of the Figure 3.6 represents the histogram $f(e)$ of the error distance defined as $e(t) = \hat{d}(t) - d(t)$. So the blue bars represent the number of appearances for each error:

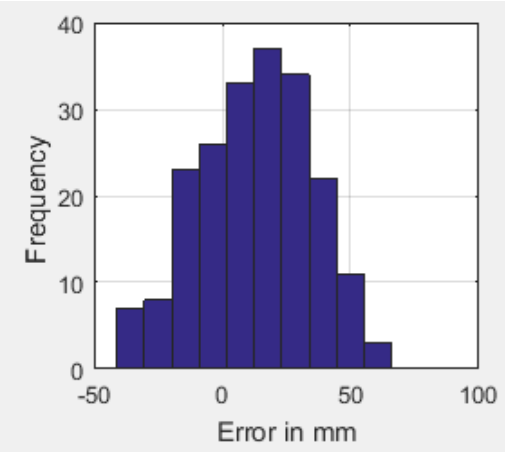

Figure 3.10: Pozyx, error histogram $f(e)$.

Finally the right data pannel of the Figure 3.6 displays the most relevant information of $f(e)$, including the mean and the variance:

Samples	100
Time	8.904
Max	146
Min	29
Mean	78.35
Variance	350.634

Figure 3.11: Pozyx, data of error $e(t)$ over time.

The measurements are exported to an Excel sheet to make an easier comparison with pivot tables and to process all the data as shown as in Figure 3.12 and Figure 3.13.

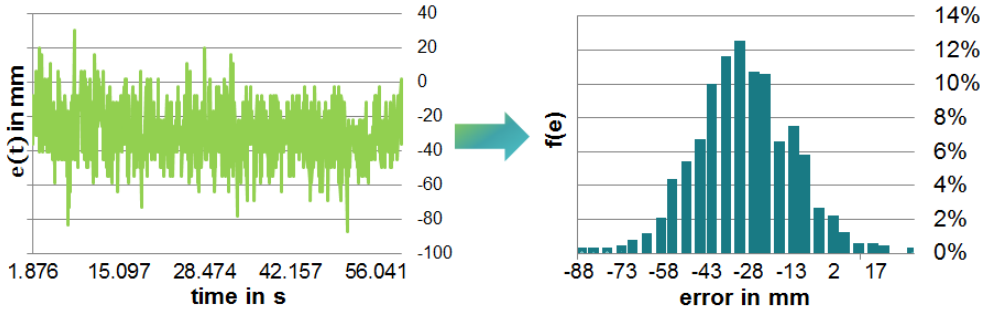

Figure 3.12: Pozyx, $e(t)$ and $f(e)$. Apparently $e(t)$ shows a Gaussian distribution of the errors. Using the histogram function, the distribution of the error $f(e)$ is gotten. The distribution is actually a Gaussian-like function. The arrow represents some kind of software analysis (Excel or Matlab).

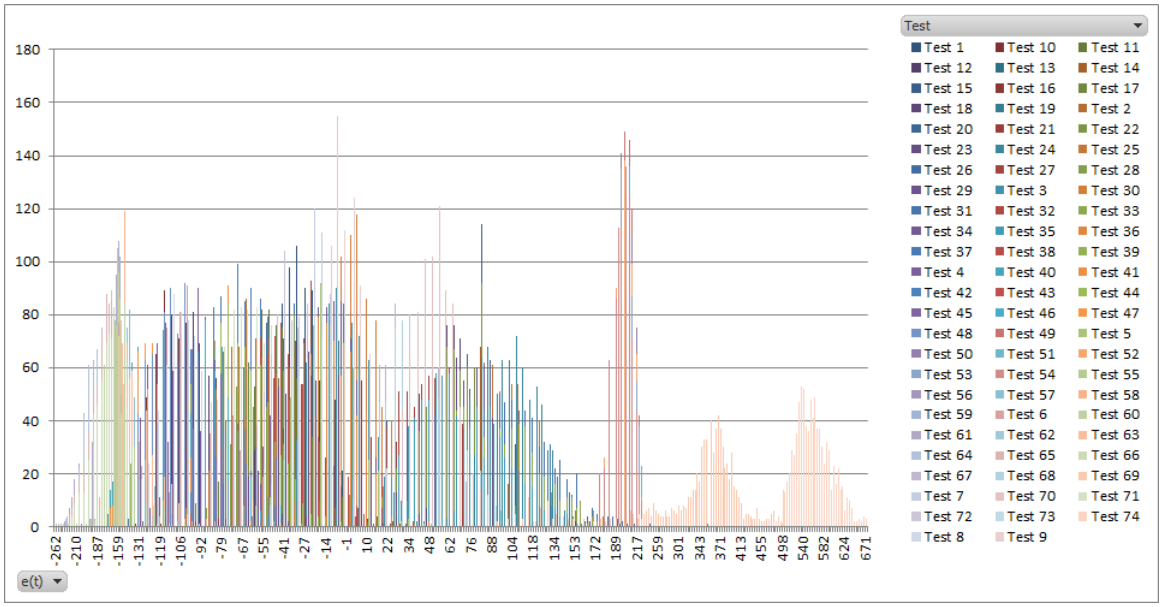

Figure 3.13: Pozyx ranging experiment, appearances for each error (in mm) and test in Experiment 1 (Section 4.1).

3.3 Positioning

For the **positioning** experiments, a Graphical User Interface (GUI) for Matlab is programmed. A real network deployment can be set-up easily. The GUI provides a drop-down menu where two positioning algorithms can be chosen in order to compare the estimated position. It can also localize an agent in real time. These experiments provide information required for the positioning algorithms.

3.3.1 Prerequisites

The program will establish a communication link between the Arduino board and the computer. Two parts will be considered in order to make the installation: the Matlab side and the Arduino side (see Figure 3.3).

For the Matlab side it is required:

- Matlab R2016a (older versions may also work).
- Arduino Support for Matlab: package enable communication at an Arduino with Matlab. The installation steps are available at reference [Pozb].

For the Arduino side is required:

- Arduino Uno x1
- USB-B Cable x1
- Pozyx device for Arduino
- The Arduino IDE and Pozyx Arduino Library: the installation steps are provided at reference [Poza]
- Pozyx anchor x [3, 6]
- USB-C Cable x [3, 6]

3.3.2 Problem Setup

Similarly to the **ranging** program (see Section 3.2), a C program has to be uploaded onto the Pozyx node (appendix A.2.1). The addresses of the devices and the number of anchors have to be set. Using a USB connection, the code is uploaded onto the Arduino board. The Pozyx device will be positioned through its own black box algorithm. and the anchors will be calibrated. The calibration can also be manual.

The positioning function behaves similarly to the ranging function, such as the Figure 3.4 showed. Unlike the ranging process, two functions are employed for positioning: the calibration function, which enable the agent to localize the anchors specified; and the the black box positioning function, which returns the estimate position of the agent during a number of samples specified, and also a list of distances estimates to the anchors.

[TUH16a] is referred to provide more details about the programme and the files which compose it. Some restrictions about the scenario have to be taken into consideration before running the main Matlab file (*localize.m*):

- The identifiers of the anchors must be specified in the program *positioning_arduino.ino*. The number of anchors in the scenario must be comprised between three and six. The external library does not support fewer or more anchors for calibration purposes, so it is out of the scope of the program to work with another amount of anchors. This can be done by editing the following lines:

```
// Edit with your devices:
uint8_t num_anchors = 4;
uint16_t anchors[4] = {0x6F13, 0x6F17, 0x6F23, 0x6F24};
```

- Only one agent in the scenario is possible. This is due to the calibration algorithm, which considers the specified addresses as an anchor address in any case. If it is an agent, the distance estimates can be obtained as well as its estimated position. This means that the agent would behave indistinctly to an anchor. Consequently, only non-cooperative algorithms are possible.

- The first specified anchor defines the origin of the coordinates map as the position (0,0). The second anchor defines the x axis and the third anchor the y axis. It is desirable to put them as perpendicular as possible. This has to be taken into consideration if only positive coordinates and a more intuitive map is desired. It is also strongly recommended to separate all the devices at least a minimum distance (around 20 mm). If this is not considered, some errors in calibration may appear. Nevertheless, it is contemplated negative coordinates and non-perpendicular located anchors. [Pozc] is referred in order to get more information about how to put the anchors.

- Coverage between all the nodes involving the scenario is assumed. The Pozyx devices have a range coverage of about 100 meters in LOS conditions. Larger separations have been not contemplated.

- The process of the calibration is automatically done. The documentation explains that the positions of the anchors can be manually set if the coordinates of the anchors are specified (in mm) and some code is uncommented as the following lines show:

```
// only required for manual anchor calibration.
  int32_t anchors_x[4] = {0, 1000, 0, 2000};
  int32_t anchors_y[4] = {0, 0, 1000, 2000};
// ...
// comment out the doAnchorCalibration if you are using
   manual mode
  SetAnchorsManual();
```

A screenshot of the interface is shown in the Figure 3.14:

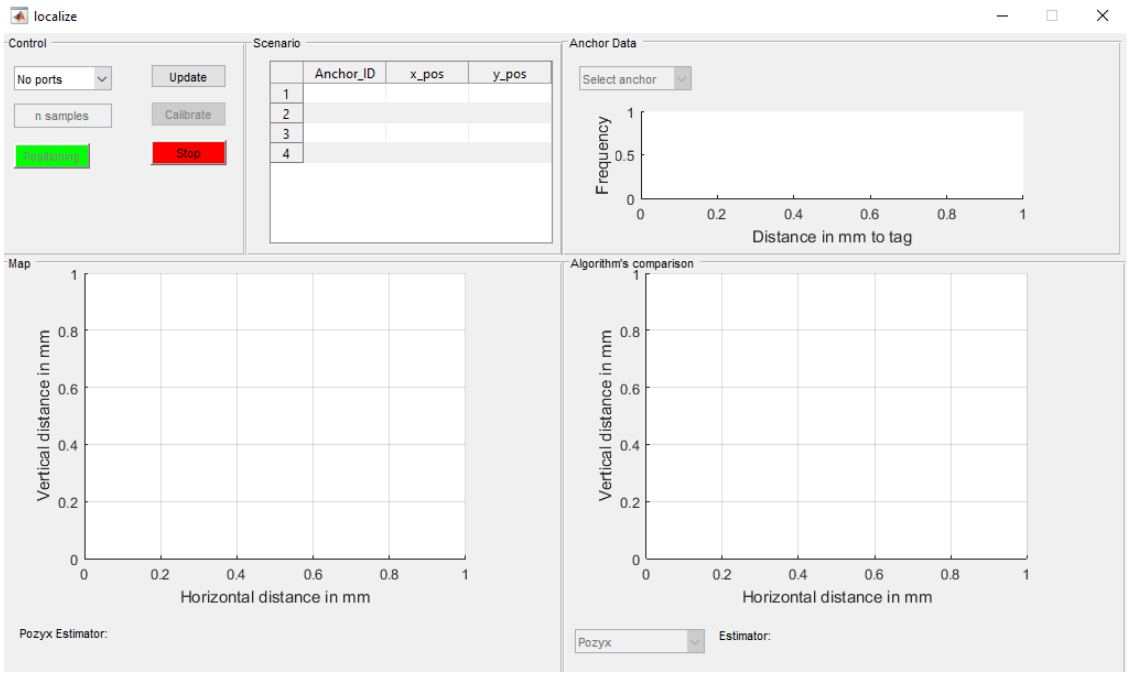

Figure 3.14: Pozyx positioning, interface running on Windows 10.

The interface provides a list of ports, which indicates the currently connected USB ports. In case nothing is connected, the execution of the calibration and positioning functions are not permitted. An update button, as its name indicates, updates the available ports and enable the execution of the functions if at least one of the ports is connected. The number of samples can also be adjusted to the number of samples that will be used for the positioning algorithms. For the case of the internal Pozyx device algorithm, this number will set the times that the Pozyx device is localized. The result will be obtained as the mean x and y coordinates. For the case of Nonparametric Belief Propagation (NBP), this number will set the number of messages that the agent will exchange with each anchor. The default value is 20. If nothing is changed in the edit text box, the agent will determine 20 ranging estimates to each anchor. These estimates will compose the input data for the NBP algorithm. It has to be considered that larger values require longer calculation times. The **calibrate** button (see Appendix A.2.2) sends to the agent a command to automatically (by default) calibrate the position of the fixed anchors. Once the calibration is completed, the positions are plotted whit each anchor identification. The position of the agent during the specified number of samples is obtained according to the internal Pozyx device algorithm. The current progress is shown in the meantime. Gathered these positions as (x,y) coordinates in millimetres, its mean is displayed in the map. The scenario table is updated. Once this process is successfully completed, both outputs anchor data and algorithms comparison panels are available.

The **positioning** button has a very similar behaviour to the calibrate button, as the calibration of the anchors is previously done and plotted. However, it also shows a real time positioning of the agent during the number of samples specified. It will not provide input data (ranging estimates) for the algorithms. The **stop** button interrupts the execution of the code. For example, in the case where a very large number of samples was set and the process is wanted to be closed. Pressing this button will consider the calibration as unsuccessful and no comparison will be available. It is recommended pressing the update button afterwards to restart it because the interruption may produce some synchronization issues between Matlab and the Arduino.

At this point the Pozyx device obtains its estimated position, and the scenario is calibrated in order to compare distinct positioning algorithms.

3.3.3 Output

Once the calibration is successfully complete, the Pozyx device internal positioning algorithm is performed. The objective of the experiments is to compare the different position estimates among the two possible positioning algorithms. An example of the execution is shown in the Figure 3.15.

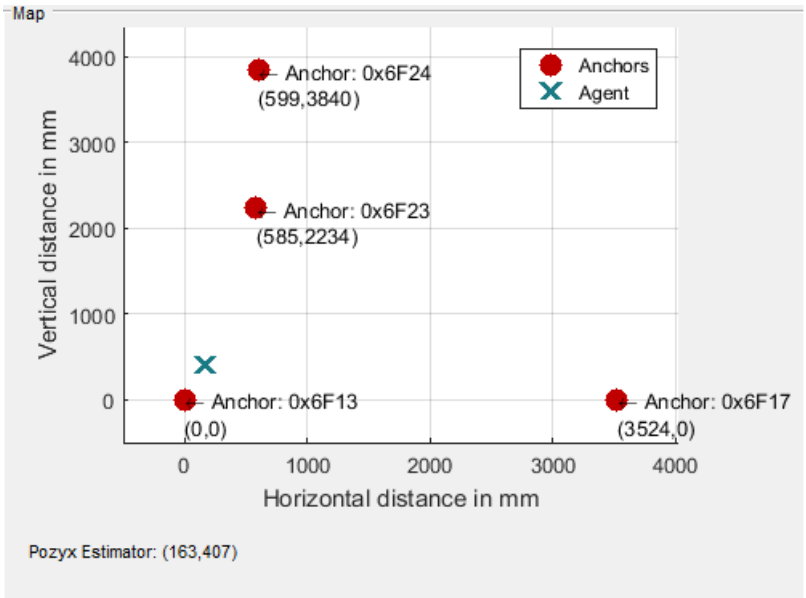

Figure 3.15: Pozyx device position estimate.

For the positioning function, a number of '**x**' marks equal to the number of samples will be plotted in real time in the map. In the anchor data pannel, each anchor can be selected with a pop-up. The anchor shows an histogram containing the ranging estimates to the agent, as well as its mean and variance values.

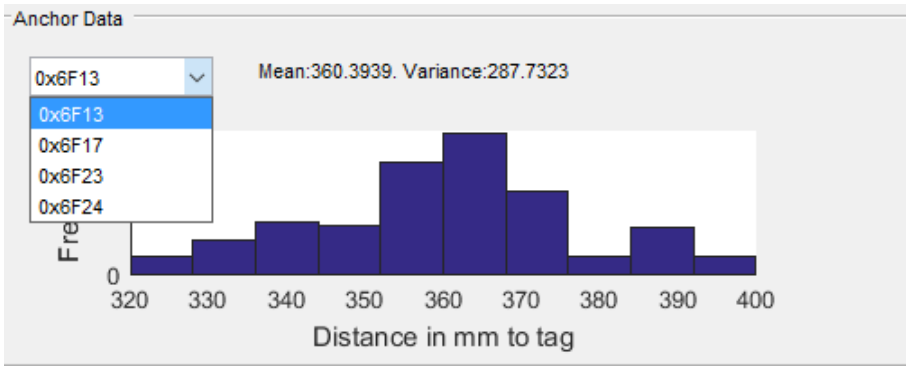

Figure 3.16: Pozyx positioning, ranging estimates between an anchor and an agent.

For the comparison, only two algorithms has been implemented, the internal black box Pozyx device algorithm; and the Nonparametric Belief Propagation (NBP) algorithm. Each algorithm can be selected with a pop-up menu. For the Pozyx device case (the default one), no calculation is required and it is the same than the agent position in the scenario map. For the NBP algorithm, a calculation is required (which can take some time depending on the number of samples). After finishing, the position estimate is plotted with an 'x' and displayed like in the Figure 3.17. This position estimate will be recalculated every time the algorithm is selected. The estimated position may change between each execution

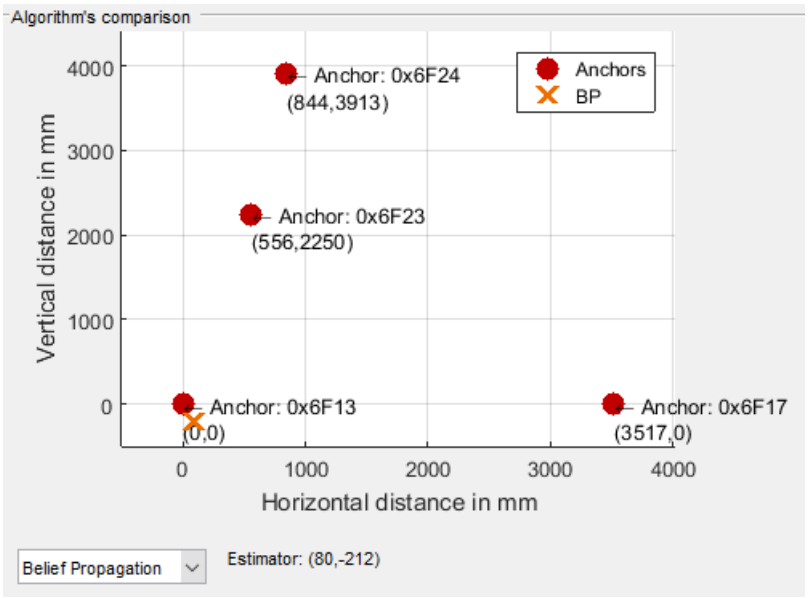

Figure 3.17: Pozyx positioning, NBP algorithm position estimate.

Note that the progress of this algorithm can be followed on the console. It will provide different results after each execution and some other parameters regarding these calculations may be set in the code, especially in the *presentBP.m* file:

```
% Input Data
[simData.NSAMPLES, ~] = size(simTopology.msg);
simData.SAMPLE_TECHNIQUE = 'Importance'; % 'Importance', '
    Mixture', 'Gibbs'
simData.K_MIXTURE_PARAMETER = 300; % Only in mixture: Max value
    is the number of devices + 1 is connected a device
simData.RESAMPLE_OPTION = 0; % Only in Importance: 0-yes, 1-no
simData.MULTIPLICATION_OPTION = 0; % 0-2on2 1-all together
simData.BMS_OPTION = 0; %Bandwidth Matrix Selector from 0 to 16
simData.SIMULATION_ITERATIONS = 1;
```

[ATIW05] explains in detail the Nonparametric Belief Propagation (NBP) positioning algorithm. The **positioning** experiments are performed in Experiment 4.3.

3.4 Environment

The experiments took place inside the building N of the TUHH, which is situated as the Figure 3.18 shows.

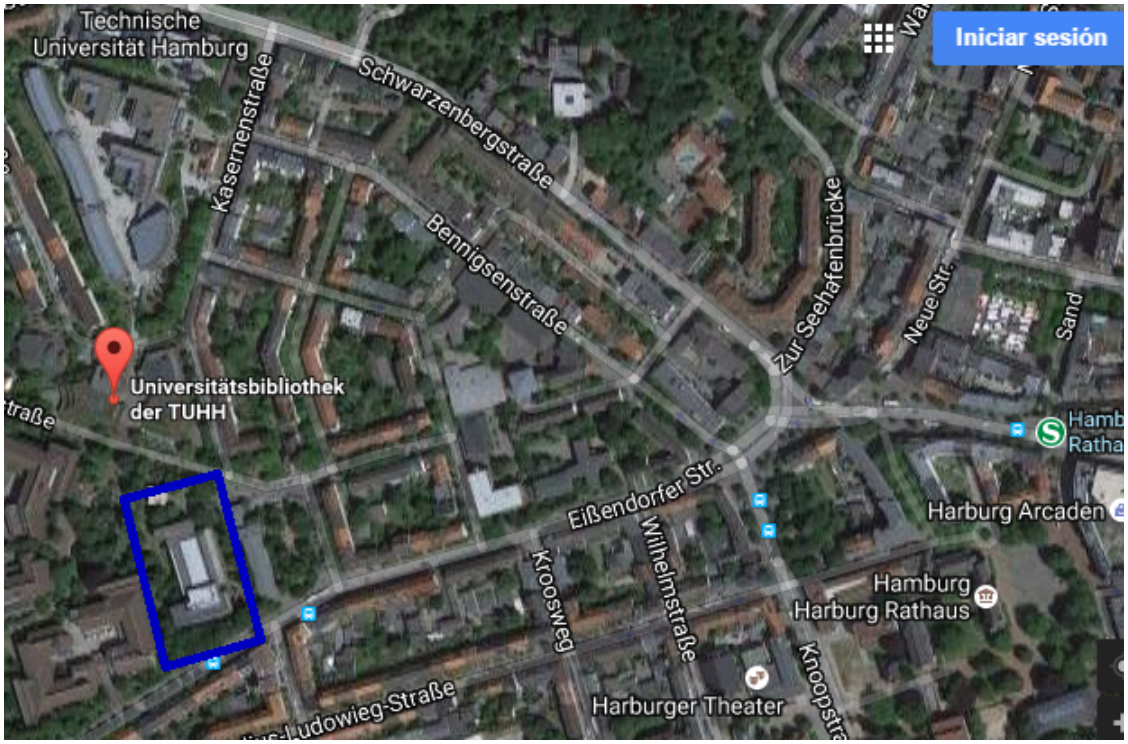

Figure 3.18: Localization of building N

3.4.1 Experiment 1

The experiment 1 (Section 4.1) took place in two rooms. These rooms are the N-2017 and N-1072. They are shown in the map of the building N in the Figure 3.19 below.

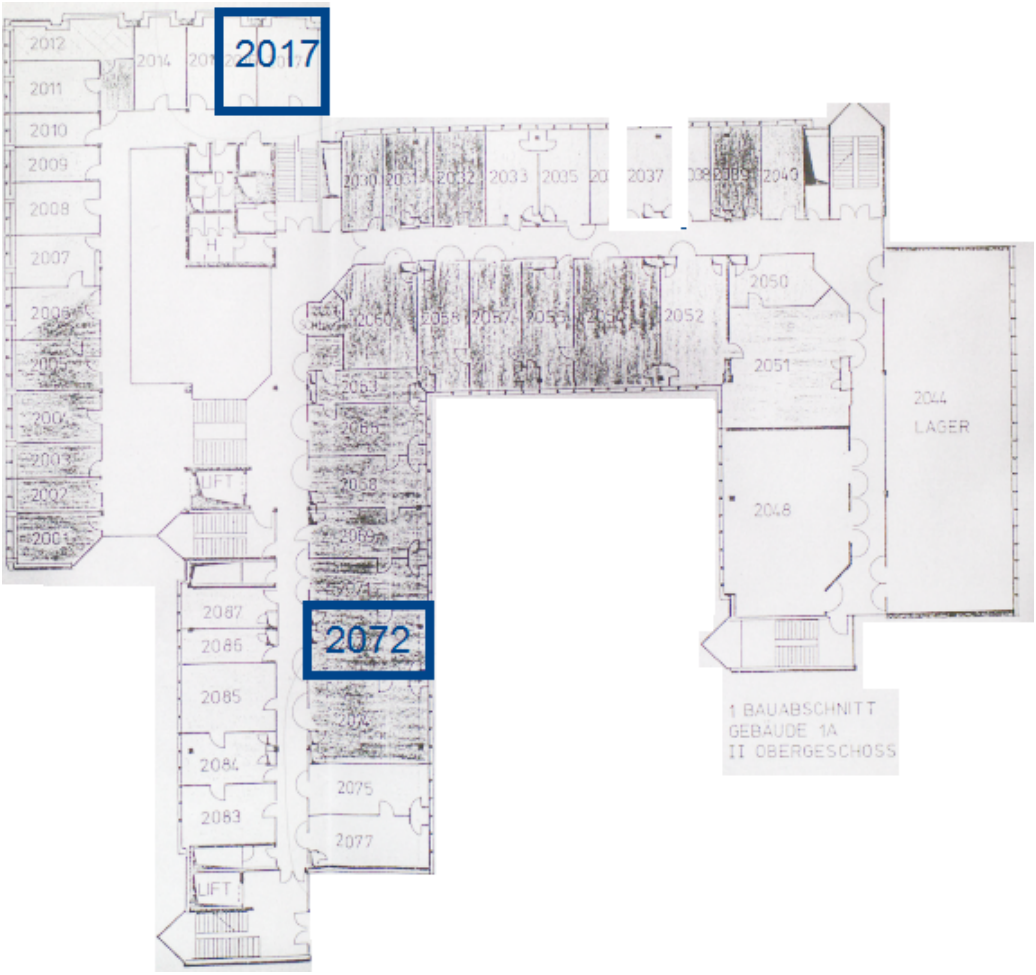

Figure 3.19: Map of the second floor of building N

The detailed setup for each experiment will be explained in further sections. For the first room, the experiments were actually developed in the first floor. A difference between a LOS and NLOS conditions are observed in Figure 3.21 and Figure 3.22

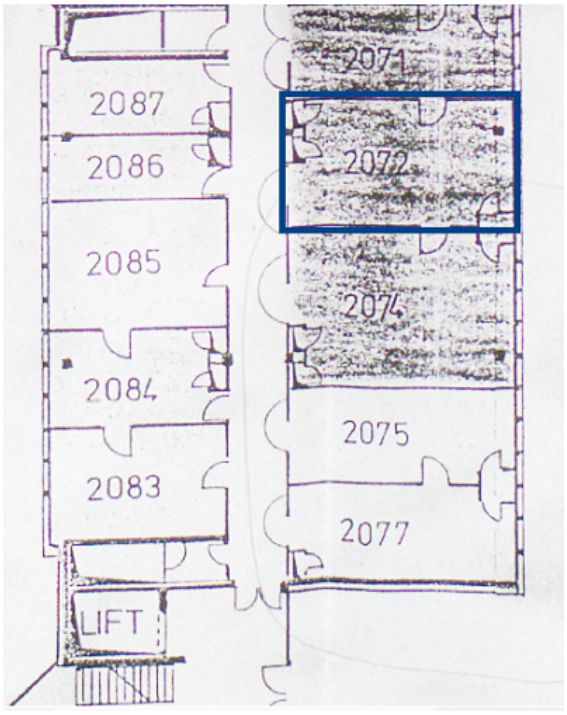

Figure 3.20: Map of the second floor of building N. Room 2072.

Figure 3.21: Room N-1072. LOS condition.

Figure 3.22: Room N-1072. NLOS condition.

In the room N-2017, there is a wall as an obstacle. The ranging estimates were calculated with some nodes inside the adjacent room N-2016.

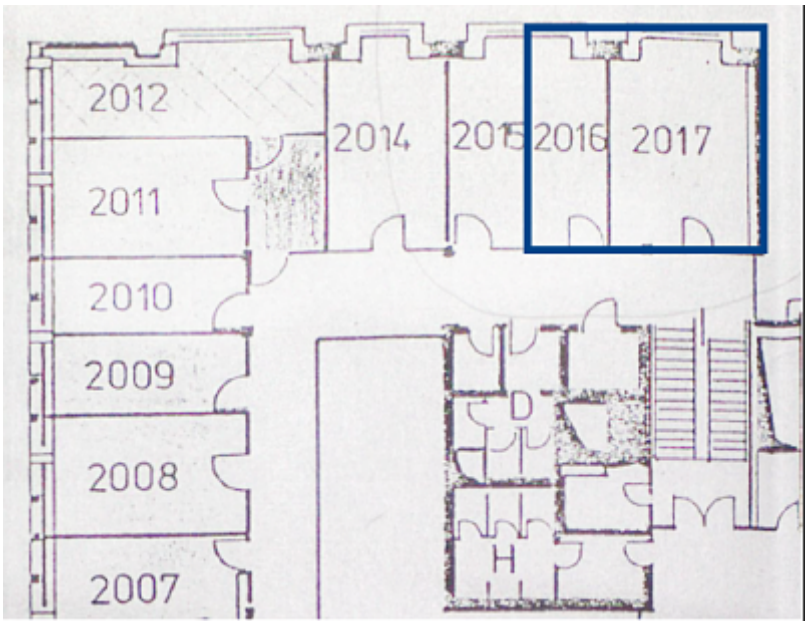

Figure 3.23: Map of the second floor of building N. Room 2017

3.4.2 Experiment 2

This experiment was developed in an anechoic chamber in order to perform ranging measurements in a context without any interferences and noise. This chamber is located in the campus of the TUHH and some pictures are displayed below:

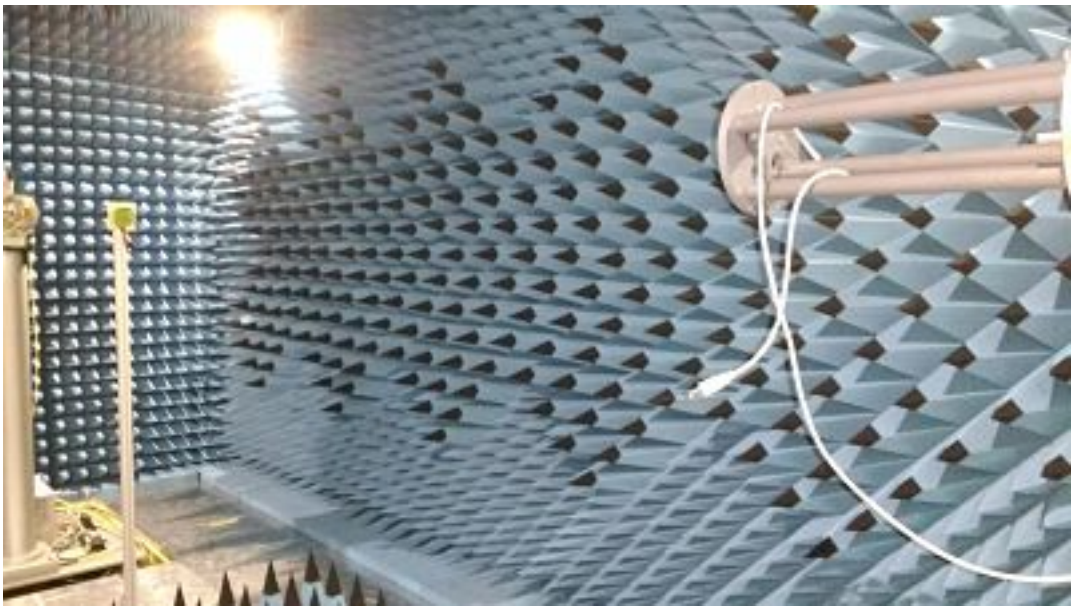

Figure 3.24: Anechoic chamber.

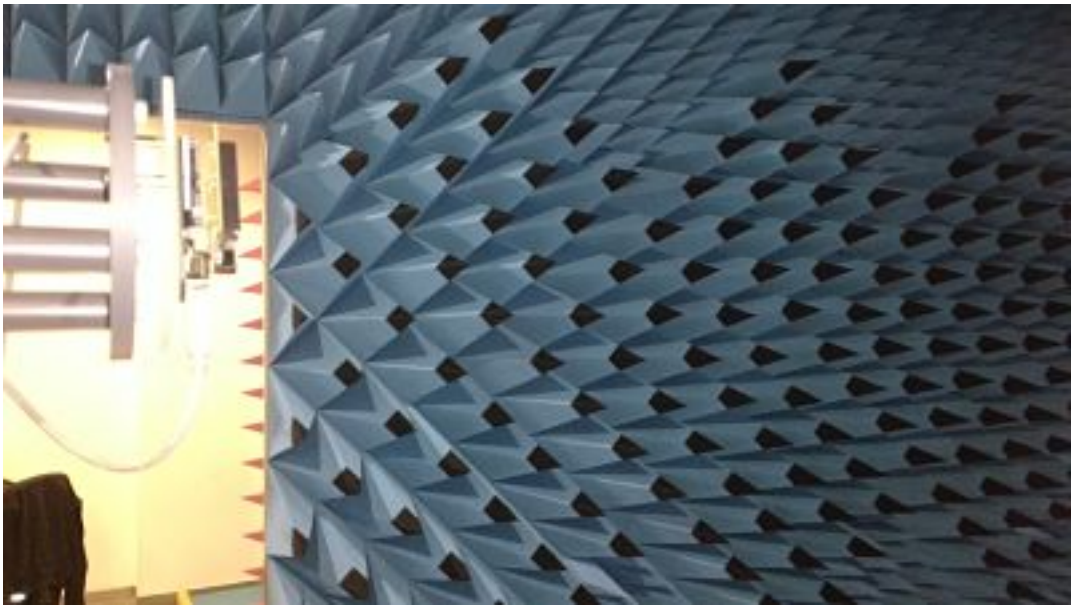

Figure 3.25: Anechoic chamber.

3.4.3 Experiment 3

As the Experiment 1, it was developed in room N-1072. This section is referred to see a detailed explanation (see Figure 3.19).

Chapter 4

Performing the Experiments

This chapter deals with the experiments based on the functions introduced in the chapter 3. This is, experiments related to ranging and positioning. A brief description is given for each experiment and their objectives are defined. Then, the setup is described. When the experiment is conducted, the main results are provided. Consequently, these results are analyzed. Finally, the conclusions are inferred.

The following experiments were conducted:

- Experiment 1: related to the characterization of the **ranging** errors between two devices.
- Experiment 2: related to the characterization of the **ranging** errors between two devices inside an anechoic chamber.
- Experiment 3: related to the **positioning** algorithms comparison, including internal Pozyx device and Nonparametric Belief Propagation (NBP) algorithms.

4.1 Experiment 1

4.1.1 Description

This experiment pretends to collect ranging estimates and characterize its error using two Pozyx devices. The ranging is repeated meanwhile several conditions of an scenario are changed. The next circumstances will be considered: changing the LOS situation, the source and destination node, the frequency channel, the actual distance and the orientation of the UWB antenna. To analyze the error, the next steps are followed (as explained in Section 3.2): first, the actual distance between two devices is measured using a laser meter. Secondly, the code is uploaded onto the Arduino board. Then, a Pozyx agent estimates the distance to another remote node. The interface of the ranging program displays the estimated distance. Every test is limited to 60 seconds. Finally, the measurements are gathered, exported and analysed using and Excel sheet.

A test is defined for each different situation. The Table B.1 in the Appendix B.1 summarizes all the tests whit their setup depending on the LOS condition, frequency, source (always an agent) and destination node, actual distance and orientation of the UWB antenna.

4.1.2 Objectives

The objective of this experiment is to characterize the ranging errors produced between two Pozyx devices depending on several different circumstances. The evolution over time of the estimated distance and the error density function is presented and analyzed. For that purpose, the actual distance, which is measured with the laser, is defined as d, whereas the estimated distance is defined as $\hat{d}$. Hence, the error can be defined as follows:

$$e(t) = \hat{d}(t) - d(t) \tag{4.1}$$

Ideally, the shape of $e(t)$ is expected to have a zero value over the time, if no errors have occurred. Nevertheless, and due to some non-ideal situations, it actually has a noise-like behaviour. As a result, the ranging errors are not always a zero. Moreover, it may have a mean distinct to zero or bias. The bias influences in the error as it has the main disturbing effect. This equation will result either in a negative error if the distance is underestimated or in a positive error iif the distance is overestimated. This is translated into a negative or positive bias respectively. The noise is characterized making use of the density function $f(e)$. In this case the density function is expected to be a Gaussian-like behaviour if no other distortions happened during the development of each test, apart from white Gaussian noise. This supposition will be evaluated during the experiment. The histogram will be centred in the mean of this error $\mathbf{E[e(t)]}$ (or bias) value. The standard deviation σ will define the dispersion of the ranging errors. Both mean and standard deviation may diverge from one test to another.

The idea of the experiment is to investigate the influence that several factors may have in the estimation of $\hat{d}(t)$. For example, it can be anticipated that the variance of the Gaussian will be greater in NLOS situations rather than in the LOS situations. In addition, a larger bias is expected in NLOS situations (as explained in Section 2.4. The frequency channel can also affect the estimates. Finally, the orientation of the antenna will affect the ranging errors if the antenna does not omni-directionally radiates. An analysis and quantification of these factors will be studied systematically.

4.1.3 Setup

The set up of the experiment is described in this section. The software has to be uploaded onto the source agent. The destination address of the node and the frequency channel must be set according to the number of test. Then every single ranging test estimate is exported. Several scenarios were considered as explained above. Test 1-48 and 67-86 were localized inside the room N-1072 (see Section 3.4).

In this room, the LOS or NLOS situation (for Test 1-48, odd or even test respectively) was changed with a blackboard between the Pozyx devices. The blackboard is assumed to be hollow. This results in a **direct path excess delay** of the signal transmitted. The frequency channel is changed with the code (every 2 tests). Identically, the source and destination addresses of the nodes are changed using the code. The antenna orientation was assumed horizontal for the Tests 1-48, as the nodes were parallel to the tables. The orientation is changed pointing upward or downward using some racks for Tests 67-86.

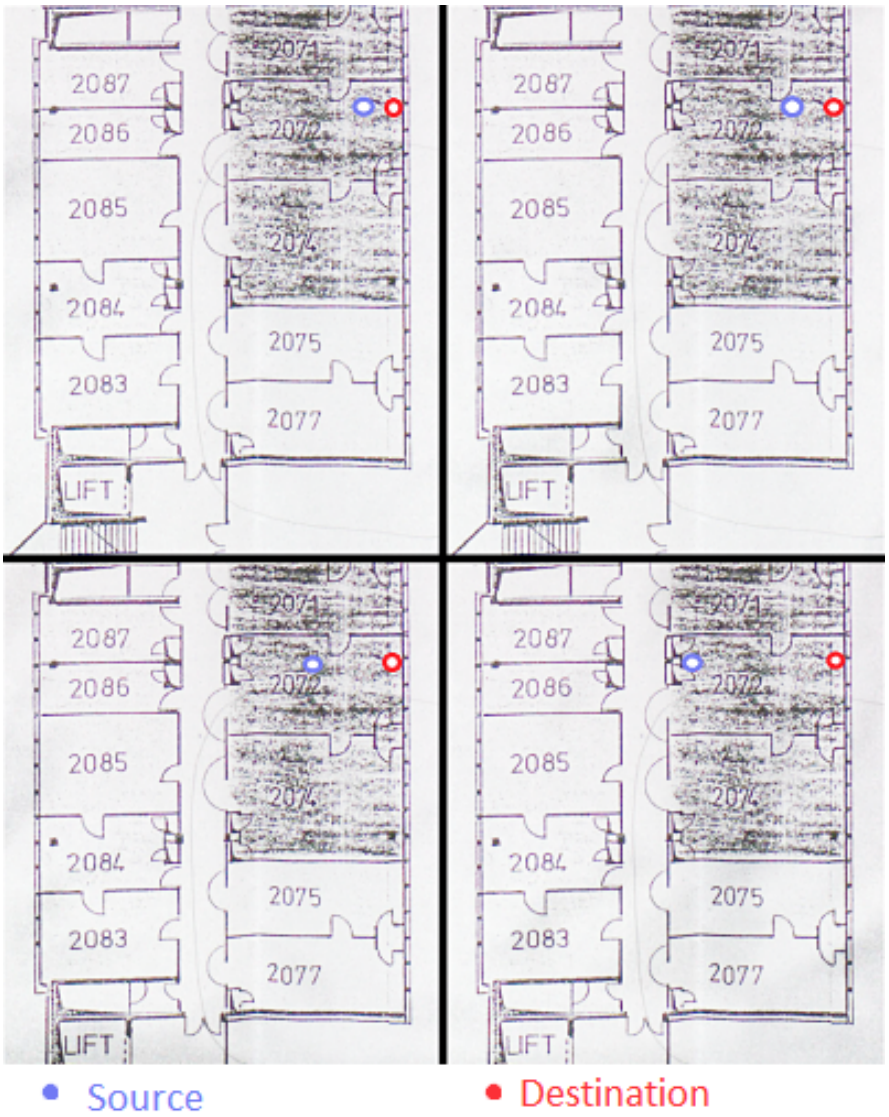

Figure 4.1: Experiment 1. Map of the position of the nodes in the room N-1072. From top to button and left to right, the red dot represents the situation of the destination (fixed) node, whereas the blue dot represents the source agent. It is situated at 1 m, 2 m, 3 m and 5 m respectively. *Note: the map shows the room N-2072 but the second floor distribution is identically the same to the first floor.*

On the other hand, Tests 49-66 were located in rooms N-2016 and N-2017. This scenario consist of two rooms and a wall between them. This means that there is a **direct path blockage** of the signal transmitted. Moreover, this also causes a complication to measure the actual distance. The laser meter cannot directly aim to the destination node. In order to measure the true distance d, the imaginary line joining the nodes were projected as perpendicular as possible to the wall. The Pozyx devices were put equidistantly to the wall as well. Thus, for a distance of 1 m, the source were put close to the window at a distance of a 0.5 m to the wall in the room N-2016, whereas the destination node were put *similarly* in the room N-2017. The word *similarly* is emphasized as it can be amused some human error. The laser meter does provide a distance to the wall rather than a distance between the nodes. Therefore, the thickness of the wall add a distance to be considered. This distance is 0.1 m and will be subtracted in the calculation of the ranging errors. In addition, it reduces the speed of the signal, which may increase the estimated distance (see Section 2.4).

Figure 4.2: Experiment 1. Map of the nodes in the room N-2017. From left to the right, the red dot represents the situation of the destination node, whereas the blue dot represents the source agent situated at 1 m, 3 m and 5 m respectively. An error can be amused for the actual distance due to the non-perpendicular line joining the devices to the wall. The thickness of the wall has to be considered.

The steps to conduct all the tests considered are the following:

1. The code is uploaded onto the Arduino Board.

2. Secondly, two nodes are put at a beforehand known distance using the laser meter. Figure 3.8 is refereed to see this procedure. For the tests in room N-1072, this can be directly done as the Figure 3.21 presented. The agent is connected to the computer to upload the code. The node can be moved with a mobile table. The right node is the destination and is always fixed in the room N-1072.

3. After measuring during 60 seconds, the steps 1 and 2 are repeated changing the frequency channel, which have the values 1, 2, 3, 4, 5 and 7.

4. The steps 1, 2 and 3 are repeated, but with an obstacle between the nodes. This procedure for the room N-1072 was shown in the Figure 3.22.

5. The actual distance is changed and the steps 1, 2, and 3 are repeated again. This is shown in the Figure 4.1. The distance is changed to a value of 1 m, 3 m and 5 m for the tests in room N-1072.

6. Up to here, tests 1-48 are obtained. Now there are two possibilities.

   - In the room N-1072: the steps 1 and 2 are repeated, but changing the orientation of the antenna either to upward or downward for the source and destination device respectively. The antenna orientation is changed using a rack (see Figure 4.3). A real device is considered pointing upward or downward according to its UWB antenna (Test 67-86).
   - In the room N-2017: the actual distance is changed to a value of 0.5 m, 1.5 m and 2.5 m for the room N-2017 (Figure 4.2), which corresponds to the distance to the wall (half actual distance of the total). The steps 1, 2 and 3 are repeated for each case (Test 49-66).

7. All the data is exported and analyzed, extracting the conclusions and most relevant information.

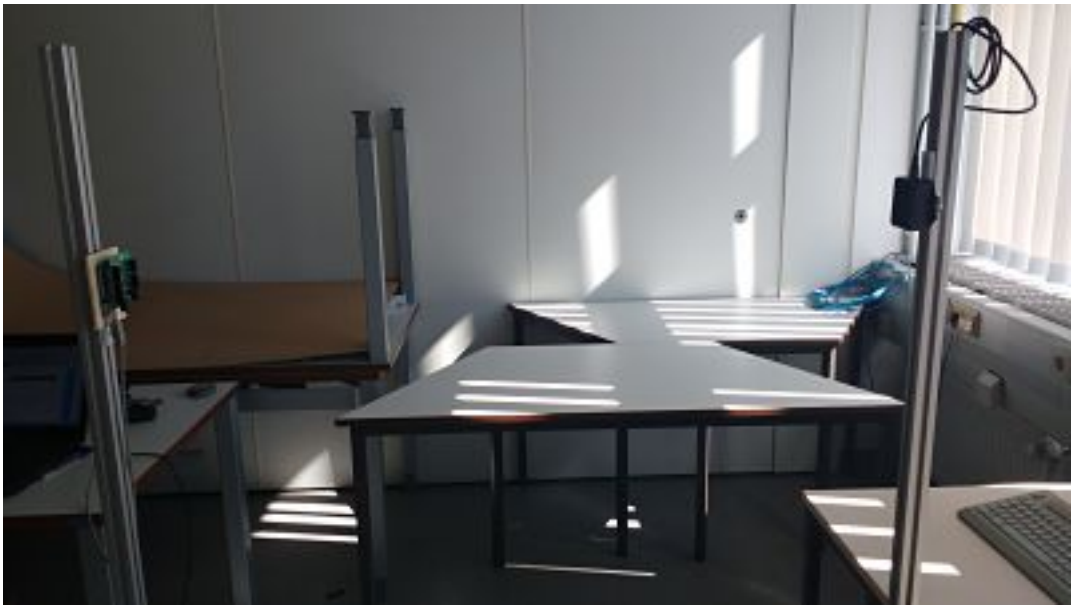

Figure 4.3: Experiment 1. Racks to change the antenna orientation. Antennas pointing upward (source) and downward (destination).

4.1.4 Results

Once all the tests have been conducted, the data gathered with Matlab is exported to an Excel file *(Experiment1.xls)*. There is a sheet per test. In each test sheet, the conditions of the test, as well as the results of the evolution of the estimated distance over the time $\hat{d}(t)$ is given. Then, some calculations are performed, like the maximum, minimum, mean and variance of the ranging errors values. This is shown in the Table 4.1.

Test 1	**SumUp**	**t**	$\hat{\mathbf{d}}(\mathbf{t})$	$\mathbf{e}(\mathbf{t})$	**e_int**	**times_e**	**p(e)**
d (mm)	1000	1876ms	964	-36	-88	0	0
LOS	Yes	1942ms	992	-8	-87	1	0.0012
Channel	ch1	2005ms	974	-26	-86	0	0
Source	0x6670	2068ms	988	-12	-85	0	0
Destination	0x6674	2131ms	974	-26	-84	0	0
Room	1072	2194ms	974	-26	-83	1	0.0012
Resolution	1	2257ms	992	-8	-82	0	0
Mean d	970.46	2320ms	969	-31	-81	0	0
Sigma d	16.48	2383ms	1006	6	-80	0	0
Max d	1030	2447ms	969	-31	-79	0	0
Min d	913	2511ms	1002	2	-78	1	0.0012
Mean e	-29.53	2574ms	969	-31	-77	0	0
Max e	30	2637ms	1006	6	-76	0	0
Min e	-87	2701ms	969	-31	-75	0	0
Samples	860	2764ms	992	-8	-74	0	0
1st cell e	$G2$	2829ms	959	-41	-73	2	0.0023
Last cell e	$G861$	2892ms	1020	20	-72	0	0
Source orientation	horizontal	2955ms	969	-31	-71	0	0
Destination orientation	horizontal	3019ms	978	-22	-70	0	0
		...	...	...	...	...	...

Table 4.1: Experiment 1. Test 1. Sum up and results. Distances in mm.

Later the error $e(t)$ is calculated. There is a parameter adjustable in each sheet, which is called resolution. This parameter will change the graph of the density function $f(e)$. A value of 1 will accumulate the number of times that each ranging error millimetre appears. For example, in Test 1 an error of -86 mm is counted once, -85 zero times, -84 zero again, etc. Setting a value of 10 mm will count ten by ten: -89 to -80 values appear two times, -79 to -70 three times, and so on. This parameter changes the number of the accumulated step for the counted errors in $f(e)$. The higher the resolution, the more probability accumulated, but the less accuracy. Values too low provide more accuracy, but can result in not enough resolution. This would be announced inside the Excel file.

The estimated distance $\hat{d}(t)$ (Figure 4.4), as well as the density function $f(e)$ for a resolution of 1 mm (Figure 4.5) and 10 mm resolution (Figure 4.6) respectively are shown below:

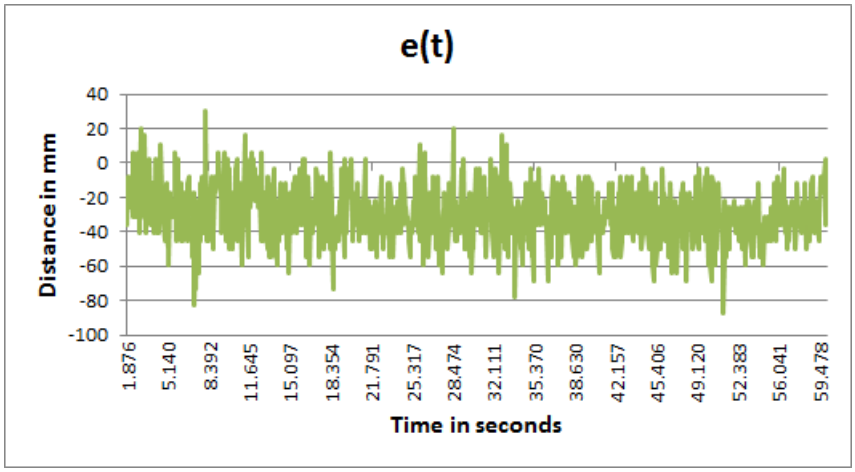

Figure 4.4: Experiment 1. Test 1. Evolution of error over the time $e(t)$.

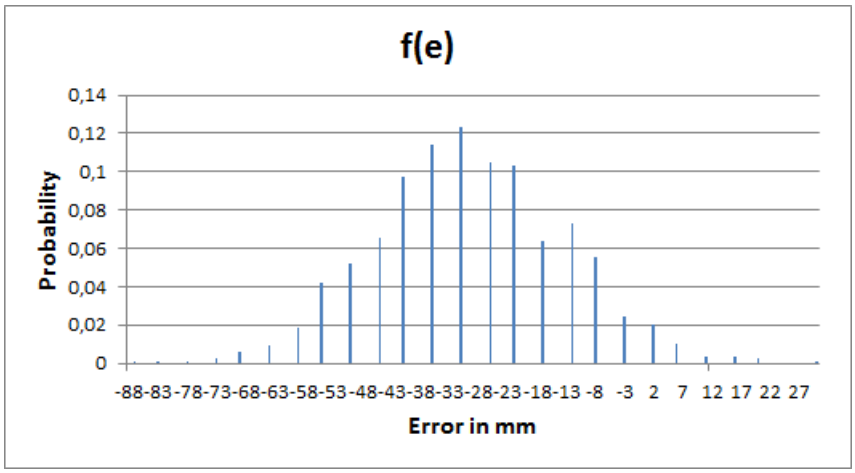

Figure 4.5: Experiment 1. Test 1. $f(e)$, 1 mm resolution.

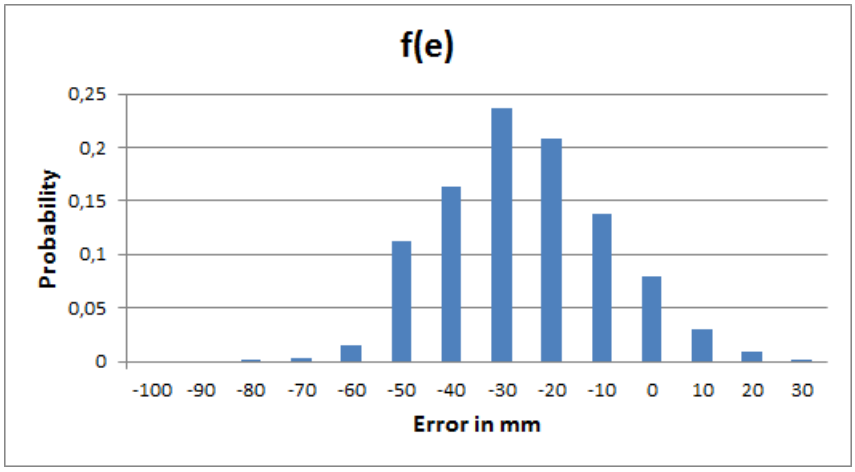

Figure 4.6: Experiment 1. Test 1. $f(e)$, 10 mm resolution. *Note: for a correct visualization, the input data must be selected accordingly, as the graph will always consider the total number of samples as generalization case.*

More sheets are added in order to put all the data together and summarize into a sheet more than 70,000 samples, corresponding to all the tests. This is projected and plotted into a pivot graphic (see Figure 3.13), which are shown in the pivots sheets (distance and error). Using this graph, all the density functions (for the errors and distances) are plotted together and can be compared easily using the data segmentation.

It is important to remark that due to the number of samples vary from each test to another, the probability density function it is actually not represented but the number of times that each millimetre appears (denoted previously as $f(e)$). Dividing the number of appearances over the total number of samples, the probability density function $p(e)$ is obtained. This is, $p(e)$ is a normalization of $f(e)$. As the number of samples is almost the same in every test, this does not corrupt the results in excess.

Using the data segmentation tools, each characteristic can be easily clicked to compare the ranging results. For instance, if frequency channel 2 is selected, source 0x6670 and destination 0x6674 will be selected automatically (because there are no more tests with different source and destination in frequency channel 2). $f(e)$ for Test 3-4, 15-16, 27-28, 39-40, 50, 56 and 62 (as the Table B.1 showed) are then plotted simultaneously. If it is clicked LOS, it will be marked just the odd tests (excepting 49-66 and 75-86 Tests). And each combination wanted can easily be chosen.

The total count of the ranging errors was shown in the Figure 3.13. It is difficult to observe in detail any interesting data, as the number of samples and tests is too high to have enough resolution. However, it can be inferred the maximum and minimum error performed during the Experiment: 671 mm and -262 mm respectively. This can be considered as low accuracy if it is taken into account that Poxyz devicez provide cm accuracy. Therefore, most of the samples are centred around the zero or less, indicating a negative bias. *The tests with negative bias will usually been discarded, as many speculations or errors can be inferred from these results. As the calculation of the ranging estimates with Pozyx devices are unknown, these tests are not considered in general*. Finally, most of the tests provide a Gaussian-like behaviour.

The idea in the following sections is to show the ranging errors with one of the conditions fixed while the others change. Thus, an easier analysis is achieve reducing the amount of tests studied simultaneously. The behaviour and regularity of each factor is studied under the following situations: LOS or not condition, frequency channel used, actual distance, device, and orientation of the antenna. For every section, one condition will be fixed and there will be a point for every condition that is changed, emphasized in italic.

LOS Analysis

This sections analyzes the effect of a LOS situation against an NLOS condition in the ranging errors.

1. The LOS situation provides in general errors closer to zero than the NLOS as expected because the nodes have a direct communication link between each other. The ranging is better done in terms of ranging accuracy. The NLOS condition provides larger bias because of the thickness of the wall and its distinct electrical permeability. In effect, the experiments developed in the room N-2017 provide a larger bias as expected for an actual distance of 1 m. This is shown in the Figure 4.7. This is repeated for every frequency channel.

   (a) LOS and room N-1072 provide **LOS** situations (as those considered in the RX_1 communication link in Figure 2.5),

   (b) NLOS and room N-1072 provide **direct path excess delay** situations (as those considered in RX_2 communication link in Figure 2.5),

   (c) and NLOS and room N-2017 provide **direct path blockage** situations (as those considered in RX_3 communication link in Figure 2.5).

2. Changing *the actual distance* provoke that point one no longer happens. As the distance increases, the bias is lower. A first approach is the especulation of some human errors. Larger distances are more sensible to incorrect measurements. Thus, d may be overestimated. The bias has less dependency of errors if the actual distance is lower. This behaviour is inferred from the Table 4.2. Therefore, the results with LOS provides better results than NLOS in terms of standard deviation. The situation is repeated independently from the channel.

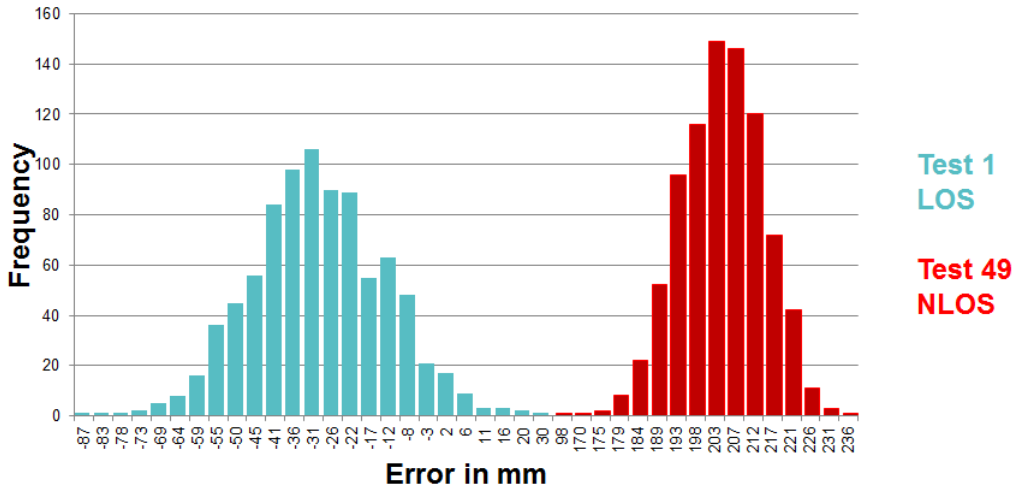

Figure 4.7: Experiment 1. $f(e)$ for Test 1 and 49. Channel 1, distance of 1 m, source 0x6770 and destination 0x6774 selected. NLOS conditions result in a greater bias because of the thickness of the wall. The situation is also repeated for every frequency channel.

Test	**E[$e(t)$]**	$\sigma[e(t)]$	**LOS**	**Distance**
Test 61	-170.21	22.55	Wall	5000
Test 55	-156.81	16.58	Wall	3000
Test 37	-84.50	19.35	Yes	5000
Test 2	-59.23	18.85	No	1000
Test 38	-46.27	27.78	No	5000
Test 26	-37.28	21.53	No	3000
Test 1	-29.53	16.48	Yes	1000
Test 25	96.77	28.36	Yes	3000
Test 49	203.98	10.12	Wall	1000
Total	-31.65	112.59		

Table 4.2: Experiment 1. Showing LOS behaviour depending on distance in mm. Tests in ascending order of the bias in mm. Channel 1, source 0x6770 and destination 0x6774 selected. The wall decrease the bias for very long distances. For larger distances, d may be overestimated.

Frequency Channel Analysis

In this section all the density functions for the different frequency channels are studied. Actually, only channels 2 and 5 are allowed to be used.

1. The frequency channel presents a regular behaviour both in *LOS conditions* and *with the wall* for these frequency channels. Indeed, even the standard deviation remains always the same, as the Figure 4.8 shows. In this case the results are almost identical and the Gaussian behaviour is clearly observed.

2. The effect of *the actual distance* have been already explained. Larger distances provoke the actual distance to be overestimated. In any case, the regular frequency channel behaviour remains. For a distance of 2 m, and 3 m the Figure 4.9 shows how channels 2 and 5 behave. They provide lower bias than in the previous case. The standard devation is also larger. However, no differences depending on channel frequency are observed.

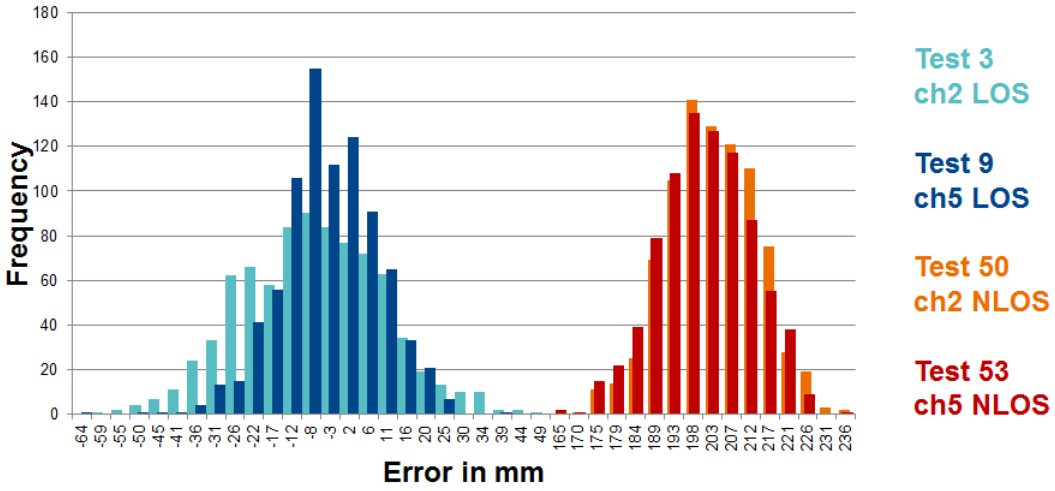

Figure 4.8: Experiment 1. Showing frequency behaviour $f(e)$ depending on the LOS situation. Distance of 1 m, channel 2 and 5, source 0x6670 and destination 0x6674 selected. Tests with NLOS show a increment of the bias in mm, but no changes with the frequency.

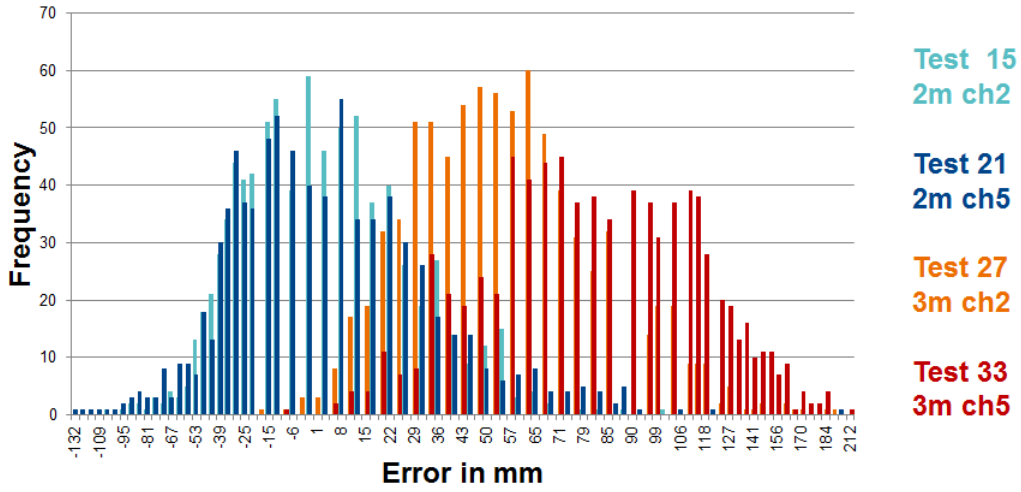

Figure 4.9: Experiment 1. Showing frequency behaviour $f(e)$ depending on distance. LOS condition, distance of 2 m and 3 m, channel 2 and 5, source 0x6670 and destination 0x6674 selected. Tests with a larger actual distance show a increment of the bias in mm, but no changes with the frequency.

Distance Analysis

The LOS and frequency channel analysis revealed more accurate ranging estimates when the actual distance is lower. As explained, this can be caused due to the assumption that larger distances may be overestimated. It is shown one more example to prove this:

- The actual distance has an irregular behaviour for the bias. As it could be observed in the Table 4.2, the bias is the lowest for a distance of 5 m and greater for a distance of 3 m; and finally the greatest for a distance of 1 m. The previous results showed that the *LOS situations* present a lower bias and better ranging estimates, with lower standard deviations. It can be concluded that the longer the distance the lower the bias.

Antenna Orientation Analysis

This section presents the errors for the different position of the antennas. Up to now, both devices have been considered to be pointing directly, with their antenna parallel to an horizontal plane. In this section the antennas will be considered to point upward or downward. The position of the antenna is changed using racks. After discussing the previous results with the manufacturer, there is a high influence of the ranging errors depending on this situation.

The Test 75-86 deal with this matter. Therefore, the previous analysis revealed that the minimum bias were conducted in the experiments of 1 m. The rest of the distances previously considered will be discarded. The four possible configurations provide the following results

1. When both source and destination are pointing downward, it the ranging presents the most negative bias, although very low variance. If the antenna of the anchor is shifted 180 degrees (down-up), the bias is greater. If the source is pointing upward and the anchor (destination) downward (up to down), a higher bias value is obtained. When both antennas are pointing upward, the estimates have the greatest bias. This can be observed in Figure 4.10.

2. The Table 4.3 provide low standard deviations for any case. This can be interpreted as good fidelity results. However, the mean of the ranging error vary among the tests. The most accurate ranging estimates are obtained when both antennas of the devices are pointing downward

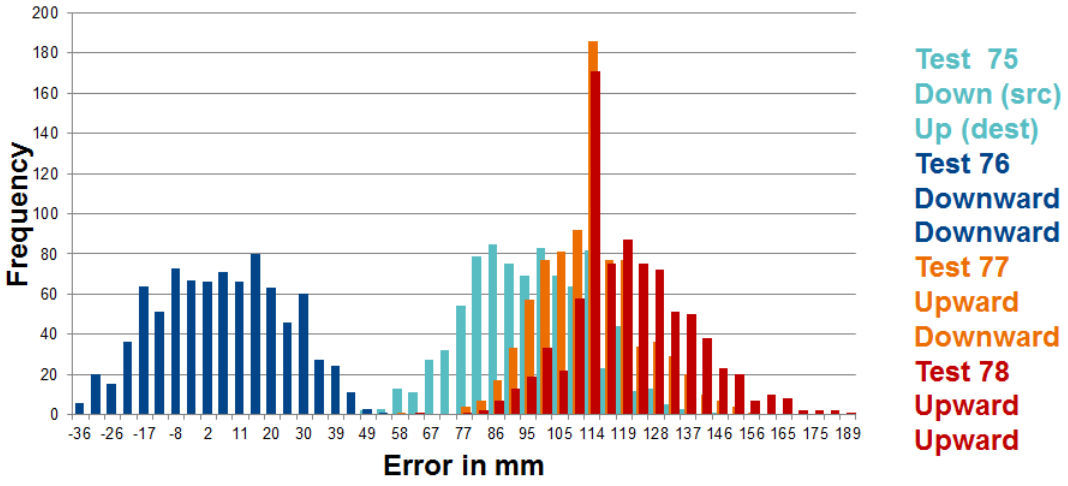

Figure 4.10: Experiment 1. Antenna orientation analysis. LOS condition, distance of 1 m, channel 1 and 5, source 0x664E and destination 0x6F0F selected. The bias seems to be affected in any case.

Test	E[$e(t)$]	$\sigma[e(t)]$	Src. orientation	Dest. orientation
Test 76	5.65	18.69	down	down
Test 75	111.94	12.94	down	up
Test 77	121.92	15.78	up	down
Test 78	121.92	15.78	up	up

Table 4.3: Experiment 1. Antenna orientation analysis. Error in mm.

4.2 Experiment 2

4.2.1 Description

The conclusion of the Experiment 1 has shown that, among the considered conditions, the **orientation of the antenna is really relevant** and the most important one. As a result, most of the tests in that section should have taken into account this situation. The Experiment 2 was created in order to analyze in more detail this effect. This Experiment pretends to measure and estimate the error while ranging between two Pozyx devices considering several positions for their UWB antenna. Experiment 1 showed that the ranging campaign had a great influence by the orientation of the antenna. The next situations will be considered to study this phenomena: different angles positions for the roll, different angles positions for the azimuth and different attachment to the rack of the Pozyx devices. Moreover, the transmitter gain will be augmented to observe if the ranging accuracy is improved. This analysis will take place in a controlled environment. This is achieved with an anechoic chamber. To analyze the error, the next steps are followed (as explained in Section 3.2): first, the actual distance between two devices is measured using a laser meter. Secondly, the code is uploaded onto the Arduino board. A Pozyx agent estimates then the distance to another remote node. The nodes perform the ranging, limiting the number of samples to 1000. Finally, the measurements are gathered, exported and analysed using and Excel sheet.

A test is defined for each different situation. The Table B.2 in the Appendix B.2 summarizes all the tests whit their setup depending on orientation of the UWB antenna and the transmitter gain. The naming of the tests is a continuation of the Experiment 1. Unlike Experiment 1, in the Experiment 2 all the tests are assumed to have LOS conditions, same frequency channel (1) and same location (the anechoic chamber). These circumstances are no longer studied, focusing on the antenna orientation. The actual distance is only changed as a result of the displacement of the position of the antenna when the azimuth is changed.

4.2.2 Objectives

The objective of this Experiment is to estimate the ranging errors produced between two Pozyx nodes under different positions for the UWB antenna. Equally to Experiment 1, the evolution over time of the estimated distance and the error density function is presented and analyzed. The estimated distance was defined in the previous section ($e(t) = \hat{d}(t) - d(t)$), as well as the expected results.

4.2.3 Setup

The setup of the experiment is described in this section. The software has to be uploaded onto the source device. The destination address of the remote node and the transmitter gain must be set according to the number of test. Then every single test ranging measurement is saved and exported. Several scenarios were considered as explained

above. Test 87-156 were located in the anechoic chamber of the TUHH (see Section 3.4).

An anechoic chamber is a closed and controlled room where all the electromagnetic or sound waves are absorbed. In this chamber, the external source of noises are isolated. The ranging tests performed in this room are under LOS conditions. The multipath interferences are eliminated.

The steps to acquire all the tests considered are the following:

1. The first set of tests (Test 87-109) deal with roll rotation with 0 dB transmitter power gain.
   - angular spacing of 20 degrees
   - source: hull – destination: hull
   - start at 0 degrees and move in 20 degrees decrements
2. The second set of tests (Test 110-113) deal with the azimuth rotation with 0 dB transmitter power gain.
   - steps of 45 degrees
   - from 90 to -90 degrees
   - 0 was not evaluated because that was already done in the first set
   - source: hull – destination: hull
3. The third set of tests (Test 114-116) deal with roll rotation with 33.5 dB transmitter power gain.
   - from 90 to -90 degrees in 45 degrees increment
   - source: hull – destination: hull
4. The fourth set of tests (Test 117-134) deal with roll rotation with 0 dB transmitter power gain
   - angular spacing of 20 degrees
   - source: hull – destination: PCB
   - start at 0 degrees and move in 20 degrees decrements
5. The fifth set of tests (Test 135-138) deal with azimuth rotation with with 0 dB transmitter power gain.
   - steps of 45 degrees
   - from 90 to -90 degrees
   - 0 was not evaluated because that was already done in the first set
   - source: hull – destination: PCB
6. The sixth set of tests (Test 139-156) deal with roll rotation with 0 dB transmitter power gain.

- angular spacing of 20 degrees
- source: PCB – destination: PCB
- start at 0 degrees and move in 20 degrees decrements

The following considerations have to be taken into account for every test:

- Two nodes are put and connected at a beforehand known distance using the laser meter. Figure 3.8 is refereed to see the procedure; in that case, it showed an actual distance of 5 m.
- The nodes are attached to a rack to change the orientation antenna. The chamber is provided with a tool which accurately changes the orientation of the source device antenna. The attachment indicates where the node is pointing at. When source and destination nodes are marked as PCB, this means that both of them are pointing each other.
- 0 roll degrees means that the transmitter antenna is pointing downward.
- The destination antenna is always pointing downward.
- The location of the tests is inside the anechoic chamber.
- There is always a LOS condition and the channel frequency is set to the first one.

4.2.4 Results

Once all the tests have been conducted, the data gathered is exported to an Excel file *(Experiment2.xls*). There is a sheet per test. In each test sheet, the conditions of the test, as well as the results of the evolution of the estimated distance over the time $\hat{d}(t)$ is computed. Then, some calculations are performed, like the maximum, minimum, mean and variance of the error values. The procedure to analyze the data was shown in Section 4.1. Like in the Experiment 1, there is a resolution parameter, and plots for $\hat{d}(t)$ and $p(e)$. In this case the number of samples is the same for every test. As a result, $p(e)$ is gotten instead of $f(e)$. The segmentation tools and pivot tables enable to fix the conditions of the antennas of the devices. The next section analyzes every combination according to the considered condition.

Roll Rotation Analysis

The Roll Rotation revealed a high influence in the bias. This can be observed in Figure 4.11. The bias for the first set of tests showed a high range of values for the error. These values move between -74 and 339 mm. As a result, around 0.4 m of difference in the estimated distance can occur when the roll rotation angle is modified.

The Table 4.4 shows the standard deviation. The standard deviation presents low values. This is expected due to the controlled environment. It can be assumed a good fidelity measurements during the Experiment.

This Table also presents a regularity for the roll rotation. To study this aspect, a polar plot is used. It reflects the mean error for $e(t)$, which is used as a module, depending on the roll, which is used as an angle. For mathematical purposes (non-existing negative modules) a normalization must be done. Indeed, a regular behaviour is reflected in the Figure 4.12. It can be inferred in a great influence of the roll position of the antenna while ranging. The most accurate results considering the lowest mean of $e(t)$ occur when the antenna is pointing downward. This was also obtained in the Experiment 1. Therefore, this also happens in another attachments, as the Figure 4.13 shows.

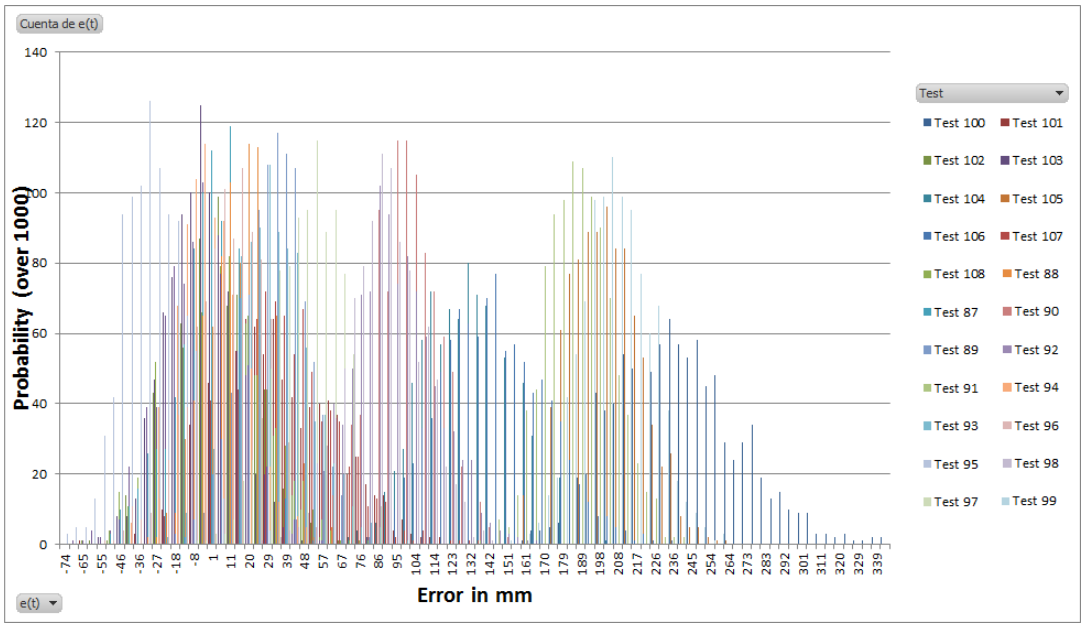

Figure 4.11: Experiment 2. $f(e)$ depending on roll rotation in degrees. Source 0x6770 and destination 0x666D selected. 0 degrees for the azimuth angle. Hull attachment for the source and destination. 0 dB transmitter gain. These results are repeated for other attachments.

Test	E[$e(t)$]	$\sigma[e(t)]$	Roll
Test 104	-3.915	17.19	-350
Test 109	3.37	23.97	-340
Test 103	2.401	20.43	-330
Test 106	32.974	27.95	-320
Test 102	29.834	30.91	-310
Test 107	144.447	26.94	-300
Test 101	237.627	31.31	-290
Test 106	199.792	19.45	-280
Test 100	205.191	17.30	-270
Test 105	134.122	23.46	-260
Test 99	87.636	18.55	-240
Test 98	50.201	17.33	-220
Test 97	9.178	19.17	-200
Test 96	-29.287	15.80	-180
Test 95	-1.105	18.42	-160
Test 94	30.036	19.93	-140
Test 93	92.264	21.84	-120
Test 92	187.238	16.78	-100
Test 91	99.331	17.39	-80
Test 90	32.835	16.48	-60
Test 89	17.345	17.24	-40
Test 88	2.011	18.46	-20
Test 87	-3.37	19.74	0
Total	67.83	80.94	

Table 4.4: Experiment 2. $p(e)$ depending on roll rotation in degrees. Tests 87-109, source 0x6770 and destination 0x666D selected. 0 degrees for the azimuth angle. Hull attachment for the source and destination. 0 dB transmitter gain. Tests in ascending order for the roll.

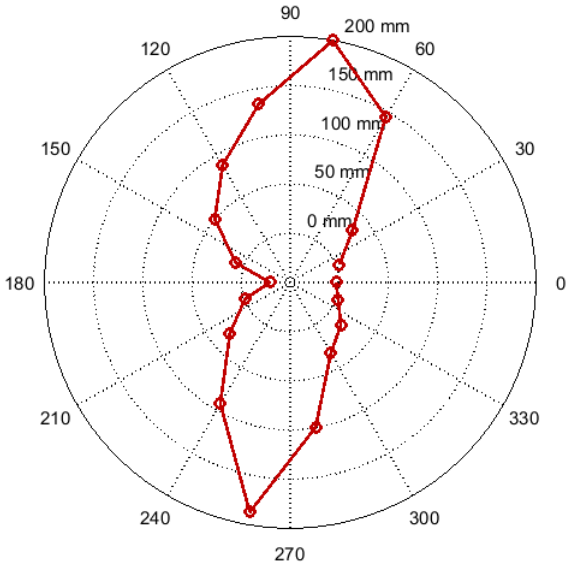

Figure 4.12: Experiment 2. Mean of $e(t)$ depending on roll rotation in degrees. Polar plot. Source 0x6770 and destination 0x666D selected. 0 degrees for the azimuth angle. Hull attachment for the source and destination. 0 dB transmitter gain. These results are repeated for other attachments.

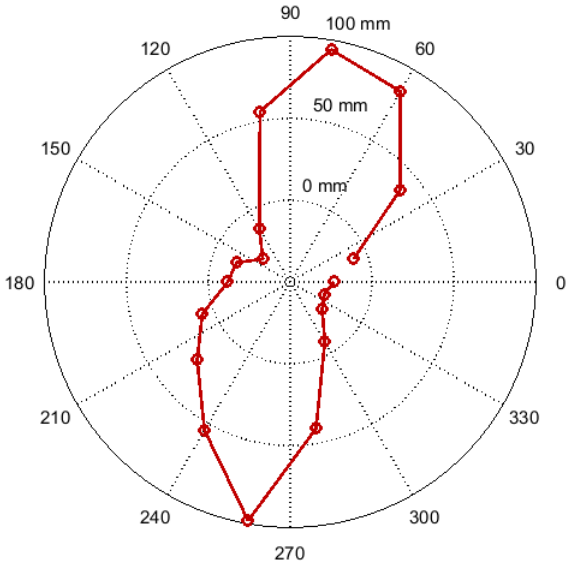

Figure 4.13: Experiment 2. Mean of $e(t)$ depending on roll rotation in degrees. Polar plot. Source 0x6770 and destination 0x666D selected. 0 degrees for the azimuth angle. PCB attachment for the source and destination. 0 dB transmitter gain.

Azimuth Rotation Analysis

This section analyzes the effect of the azimuth position of the source node in the ranging errors. As the roll rotation, this effect resulted in high influence in the bias. This can be observed in Figure 4.14. The bias for the second set of tests showed a high range of ranging error values. These values move between -158 and 24 mm. As a result, a negative bias can be observed. The distance is underestimated.

The Table 4.5 shows the standard deviation presenting low values. This is expected due to the controlled environment. It can be assumed a good fidelity results during the experiment.

It can be concluded that the azimuth roll has a great influence in the results. The most accurate results considering the lowest mean of $e(t)$ occur when the azimuth angles is set at -45 degrees when the roll angle is 0 degrees.

Test	E[$e(t)$]	$\sigma[e(t)]$	Azimuth
Test 113	-107.09	16.87	-90
Test 112	-76.353	16.95	-45
Test 111	-103.685	16.22	45
Test 110	-97.672	16.16	90
Total	-96.20	20.41	

Table 4.5: Experiment 2. $p(e)$ depending on azimuth rotation in degrees. Tests 110-113, source 0x6770 and destination 0x666D selected. 0 degrees for roll angle. Hull attachment for the source and destination. 0 dB transmitter gain. Tests in ascending order for the azimuth.

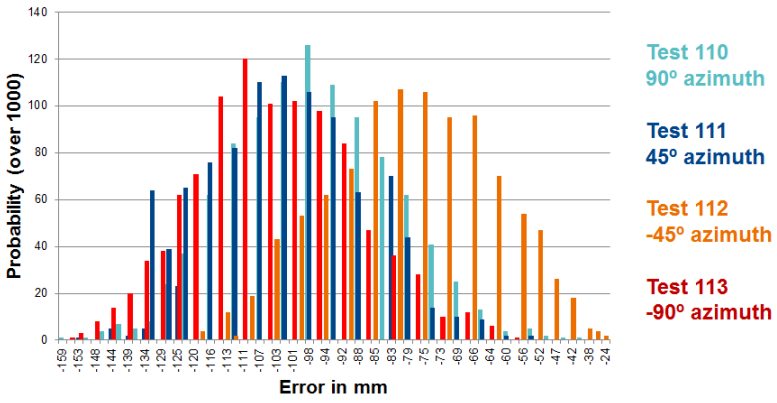

Figure 4.14: Experiment 2. $f(e)$ depending on azimuth rotation in degrees. Source 0x6770 and destination 0x666D selected. 0 degrees for the roll angle. Hull attachment for the source and destination. 0 dB transmitter gain. These results are repeated for other attachments.

4.3 Experiment 3

4.3.1 Description

The objective of this experiment is to employ a Matlab interface to calibrate a real scenario. The GUI acquire the two dimensional coordinates of the anchors either automatically or manually. An agent will be localized in real time. This node will obtain a collection of ranging estimates from several anchors. These estimates will be the input arguments for the NBP positioning algorithms. Hence, this experiment compute information required for the algorithms, which are finally implemented.

4.3.2 Objectives

The objectives of this experiment are:

- To calibrate a real case scenario and determine the coordinates of the anchors.
- To show the real time position of a determined agent.
- To generate the input data for the positioning algorithms and show the localization results.

4.3.3 Setup

The set up of the experiment is described in this section. The software has to be uploaded once onto the source device. The address of the anchors and its amount must be set.

This experiment took place in the room N-1072 (see Figure 4.1). Section 3 is refereed to remember the considerations about the scenario that have to be taken into account to perform this experiment.

All of these objectives will be achieved using a Matlab GUI hat enables to communicate a computer with Matlab to the Pozyx agent. The first objective is a requirement for the real time positioning and implementation of the localization algorithms. As a result, only two tests are defined (one for the two remaining objectives);

- Test 157: consist of a demonstration of the positioning functionality meanwhile a node is moving.
- Test 158: comparison between the internal positioning algorithm of the Pozyx device and NBP in a custom scenario with 4 anchors and an agent.

4.3.4 Results

Positioning in Real Time

Test 157 deals with the positioning in real time. The real time positioning is shown in Figure 4.15. It reflects in general a movement.

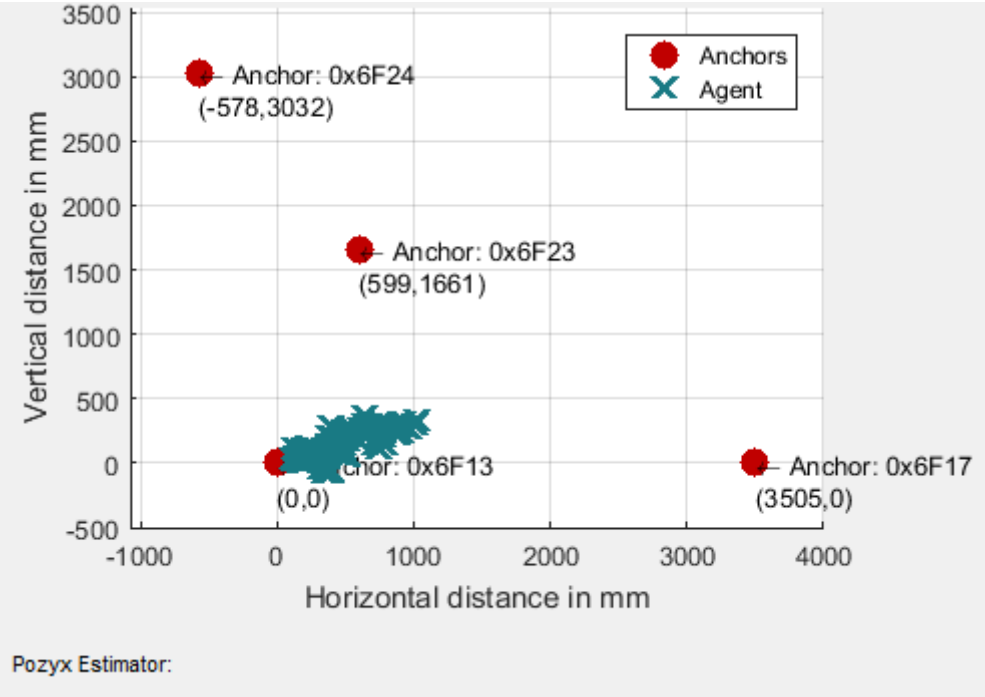

Figure 4.15: Experiment 3. Test 157. First execution. Pozyx device moving.

Pozyx and NBP Localization Algorithms Comparison

A custom scenario is considered. This is conducted in the Test 158. During the 100 samples calibration, the Pozyx device internal algorithm localizes an agent in the position (1050, 1567), as the Figure 4.16 shows:

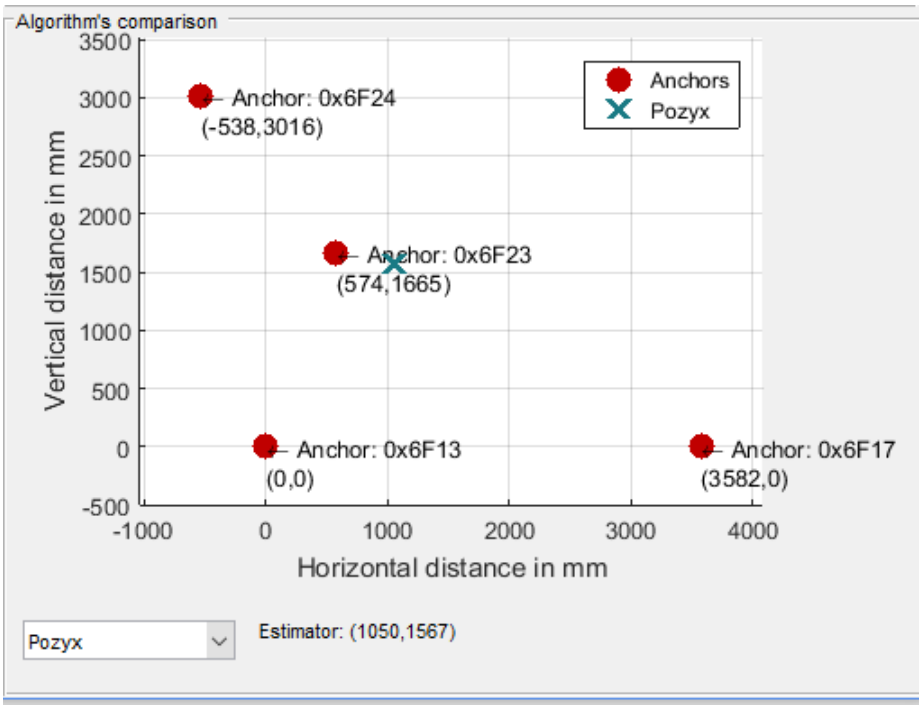

Figure 4.16: Experiment 3. Test 158. Pozyx algorithm position solution.

The Nonparametric Belief Propagation (NBP) is now selected with the next configuration.

```
% Input Data
simData.SAMPLE_TECHNIQUE = 'Importance'; % 'Importance', '
    Mixture', 'Gibbs'
simData.K_MIXTURE_PARAMETER = 300; % Only in mixture: Max value
    is the number of devices + 1 is connected a device
simData.RESAMPLE_OPTION = 0; % Only in Importance: 0-yes, 1-no
simData.MULTIPLICATION_OPTION = 0; % 0-2on2 1-all together
simData.BMS_OPTION = 0; %Bandwidth Matrix Selector from 0 to 16
simData.SIMULATION_ITERATIONS = 1;
```

It displays the results about the processing time to create the network, to interchange the messages between the agent and the anchors and to localize the agent.

```
% Input Data
Creating messages between anchors and devices.
        Done in 0.003638 seconds

Interchange of messages between anchors and devices.
        Network initialized in 2.679809 seconds
Simulation finished,
        total time: 2.691172 seconds
Localization error:
   61.3396
Localization time:
    2.6912
Position
   1050.0635  1566.6970
```

As a result, the agent is localized in position (1133, 1605) with this localization algorithm. This is a bit different from the estimated position by the Pozyx device, as the Figure 4.17 shows:

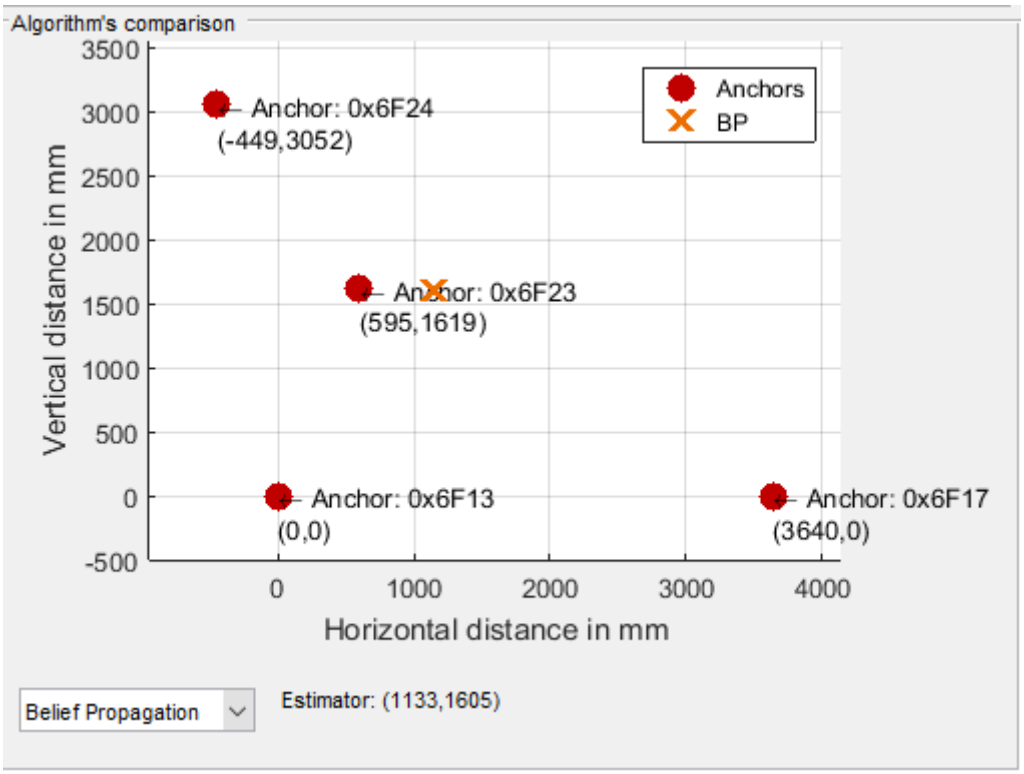

Figure 4.17: Experiment 3. Test 158. NBP position solution.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

During the work, the main functionalities of the Pozyx devices have been explained. The most relevant issues of the ranging estimates have been defined according to the accuracy results. The methodology has been explained in detail and the programmes have been documented.

This project has demanded a communication with the manufacturer of the Pozyx device. The ranging campaign has revealed the difficulties of getting really cm accuracy distance estimates. In effect, the student has experienced these problems while he was performing this campaign, which should have been considered firstly. Thus, the regression was essential and the isolated environment resulted in really relevant results.

The student has understood the major functionalities of the Pozyx nodes. Before developing the software, he familiarized with the code and software required to manipulate these devices. He had to understand the general behaviour of these devices. To get familiar with them, he read the adequate documentation. This fact was not new for the student, who already had work in a project with similar characteristics like his Bachelor's Thesis.

Related to the tools and methodology employed, it is remarkable that some of them were quite novel for the student, especially those related to programming in Arduino. The documentation was provided to learn it.

As an engineer working on a research project, the student confronted some problems more or less unknown for him in an efficient way.

Regarding the results, the orientation antenna resulted in the most relevant influence for the ranging campaigns. The student will purpose more studies to analyze this effect. Also, more localization algorithms are recommended to be implemented in the future.

Lastly, it can be concluded that the initial objectives purposed have been accomplished. However, several issues with accuracy ranging should be discussed in the future. Nevertheless, this is also part of the learning. It is very possible that during the future

work of the student, these kinds of problems occur. It is important to adapt to the circumstances and act like and engineer is expected to. Finally, the student developed this Thesis as an abroad student. He had to get used to the way of working of other cultures and adapt his language to make himself understood. That experience was new for him and he found it really good.

5.2 Future Work

In this section the student gives some guidelines to orientate a future researcher of this topic.

5.2.1 Ranging

Once the Ranging program has been presented, it has been observed that many functionalities can be implemented with the interface.

It would be interesting to provide a tool to manually set the destination device id. However, it was difficult to do it as the transmitted data is hexadecimal throughout a serial bus, interpreting the Arduino sketch the wrong id.

Also, some synchronization issues have been observed, as the data is read faster in some computers. If there is no data to be read, the code waits a time until it is again available, defining a buffer. However, it has been observed some unsuitability with these waiting times. Improve this buffer to work correctly in every scenario and computer is an important issue to be solved.

It is highly recommended to develop again a ranging campaign. For that case, it should be considered the orientation of the antenna and to finally quantify its effect.

Pozyx is also expected to develop a new firmware for the board. Inside its new functionalities, some algorithms to consider NLOS conditions will be implemented. It should be considered to work with this functionalities and others.

5.2.2 Positioning

Once the Positioning program has been presented, it has been observed that many functionalities can be implemented with the interface.

First thing would be to add more algorithms. As two of them have already been implemented, it seems easy to provide more of them if the code is already programmed. The only matter is to adapt it in order to present it onto the handles structure.

Also, it would be interesting to provide a tool to manually set the coordinates of the anchors, either using the table or by clicking the map.

The option of multiple agents can be considered too, but omitting the fact the agent provides its coordinates. Nevertheless, the library only supports a maximum of 6 anchors so not really complex scenarios would be considered though. Dealing with devices that are not in coverage between them presents a difficult challenge, as the devices should be at a physical distance of more than 100 m.

Also, some synchronization issues have been observed, as the data is read faster in some computers. If there is no data to be read, the code waits a time until it is again available, defining a buffer. However, it has been observed some unsuitability with these waiting times. Improve this buffer to work correctly in every scenario and computer is an important issue to be solved.

It is suggested to do experiments using this interface, studying the behaviour of the algorithms under different circumstances. The next algorithm that should be implemented is Least Squares (LS), which is very efficient in terms of complexity.

Finally, the use of the third dimensional axis presents a challenge. Pozyx provide these kind of localization. Processing and presenting it correctly could provide interesting functionalities in a field whose possibilities are not really exploited.

Appendix A

Code

A.1 Ranging code

A.1.1 Arduino Part

The channel and the destination device must be modified accordingly. Part of the code containing some lines of *ready_to_range_arduino.ino* is shown below:

```
#include <Pozyx.h>
#include <Pozyx_definitions.h>
#include <Wire.h>

//////////////////////////////////////////////////
////////////////// PARAMETERS ////////////////////
//////////////////////////////////////////////////

uint16_t destination_id = 0x6670;     // the network id of the
   other pozyx device: fill in the network id of the other
   device
signed int range_step_mm = 1000;      // every 1000mm in range,
   one LED less will be giving light.
int uwb_channel = 1;                  // set the channel
int nSamples;
//int distances[50];
// int times[50];
int ite = 1;
boolean syn;



//////////////////////////////////////////////////

void setup() {
  Serial.begin(115200);
```

```
  if (Pozyx.begin() == POZYX_FAILURE) {
    Serial.println("ERROR: Unable to connect to POZYX shield");
    Serial.println("Reset required");
    Pozyx.setUWBChannel(uwb_channel); // setting the UWB channel
    Pozyx.setUWBChannel(uwb_channel, destination_id); // setting
        the UWB channel destination
    delay(100);
    abort();
  }


    Serial.println("------------POZYX RANGING V1.0------------")
       ;
    Serial.println("NOTES:");
    Serial.println("- Change the parameters:\n\tdestination_id (
       target device)\n\trange_step (mm)\n\t");
    Serial.println();
    Serial.println("- Approach target device to see range and\n
       led control");
    Serial.println("------------POZYX RANGING V1.0------------")
       ;
    Serial.println("");
    Serial.println("READY");


  // make sure the pozyx system has no control over the LEDs, we
     're the boss
  uint8_t configuration_leds = 0x0;
  Pozyx.regWrite(POZYX_CONFIG_LEDS, &configuration_leds, 1);

  // do the same with the remote device
  Pozyx.remoteRegWrite(destination_id, POZYX_CONFIG_LEDS, &
     configuration_leds, 1);

  syn = false;
}

void loop() {
  // if there is data to read
  if (Serial.available() > 0 && syn == false) {
    nSamples = Serial.read(); // read data
  }

  // Wait a bit an syn first lecture
  if (ite == 1 && syn == false) {
    delay(1000);
    syn = true;
  }
```

```
  int status = 1;
  device_range_t range;

  // let's do ranging with the destination
  status &= Pozyx.doRanging(destination_id, &range);

  //
  if (status == POZYX_SUCCESS && syn == true) {
    Serial.println(ite);
    // Serial.print("sample \t");
    Serial.println(range.timestamp);
    // Serial.print("ms \t");
    Serial.println(range.distance);
    // Serial.print("mm \t");

    //times[ite] = int(range.timestamp);
    // distances[ite] = int(range.distance);
    // Serial.println(times[ite]);
    //Serial.print(ite);
    //Serial.println(" Sample  \t");
    ite++;
  }


  // now control some LEDs; the closer the two devices are, the
     more LEDs will be lit
  if (ledControl(range.distance) == POZYX_FAILURE) {
    //Serial.println("ERROR: setting (remote) leds");
  }

  else {
    // Serial.println("ERROR: ranging");
  }

}

int ledControl(uint32_t range) {
  int status = 1;

  // set the LEDs of this pozyx device
  status &= Pozyx.setLed(4, (range < range_step_mm));
  status &= Pozyx.setLed(3, (range < 2 * range_step_mm));
  status &= Pozyx.setLed(2, (range < 3 * range_step_mm));
  status &= Pozyx.setLed(1, (range < 4 * range_step_mm));

  // set the LEDs of the remote pozyx device
  status &= Pozyx.setLed(4, (range < range_step_mm),
     destination_id);
```

```
  status &= Pozyx.setLed(3, (range < 2 * range_step_mm),
     destination_id);
  status &= Pozyx.setLed(2, (range < 3 * range_step_mm),
     destination_id);
  status &= Pozyx.setLed(1, (range < 4 * range_step_mm),
     destination_id);

  // status will be zero if setting the LEDs failed somewhere
     along the way
  return status;
}
```

A.1.2 Matlab Part

The code regarding the Range button is shown below:

```
% --- Executes on button press in calculate_button.
function calculate_button_Callback(hObject, eventdata, handles)
% hObject    handle to calculate_button (see GCBO)
% eventdata  reserved - to be defined in a future version of
   MATLAB
% handles    structure with handles and user data (see GUIDATA)



% Clear the maps
cla(handles.dt);
cla(handles.pe);


% Initialize the arduino
global port;
global arduino;
arduino = initArduino(port, handles);
% if (port == '0')
%     close;
% end
fopen(arduino);

% Disable rest of buttons
changeVisibility(handles, 'off');
set(handles.update_button, 'Enable', 'off');

% Variables to stop the code if:
% - is limited by samples
% - is limited by time
% - is pressed the stop button (isDone)
global isLimitedBySamples;
global nSamples;
global isLimitedByTime;
global nTime;
global isStopped;
isStopped = 0;

% More input data
global actual_distance;
global nData;

% Output data, READ matrix:
% - row: number of sample, initialize it to the first one
% - nData column: nSample, time (ms) and distance (mm)
clear READ;
```

```
global READ;
READ = [];
row = 1;

% Wait until Arduino outputs READY: that means next output is
   nSample = 1
    % see .ino code for more info
% and can store into READ the first value
isReadyToRange = 0;
while(not(isReadyToRange))
    if(arduino.BytesAvailable > 0)
        if (strcmp('READY',fscanf(arduino, '%s')))
            isReadyToRange = 1;
        end
    end
end

% Write the output data into READ while is not finished and it
   is ready
Finished = 0;
try
    while(not(Finished) && isReadyToRange)
        % no data to transmit => wait 0.5 s
        if(arduino.BytesAvailable <= 0)
            pause(0.5);
        end

        % wait until Arduino outputs data
        if (arduino.BytesAvailable >= nData)

            % write a row matrix (nData columns)
            for column = 1:nData
                READ(row, column)= fscanf(arduino, '%i');
            end

            % Get sub columns
            samples_vector =  READ(:,1);
            times_vector =  READ(:,2)/1000;
            distances_vector =  READ(:,3);

            % Plot d(t) and p(e)
            hold(handles.dt, 'on');
            plot (handles.dt, times_vector, distances_vector,'g'
               );
            % pause(0.01);

            ERROR = distances_vector - actual_distance;
            hist(handles.pe, ERROR);
            grid(handles.pe, 'on');
```

```
xlabel('Error in mm');
ylabel('Probability');
% pause(0.01);

% Update the text info
set(handles.result_samples, 'String', samples_vector
    (end));
set(handles.result_time, 'String', times_vector(end)
    );
set(handles.result_max_error, 'String', max(ERROR));
set(handles.result_min_error, 'String', min(ERROR));
set(handles.result_mean, 'String', mean(ERROR));
set(handles.result_variance, 'String', var(ERROR));
progress = 'Calculating';
set(handles.calculate_button, 'String', progress);
if (isLimitedBySamples && not(isLimitedByTime))
    progress =  strcat(num2str((round(row/nSamples *
        100,2))),{' %'});
    set(handles.calculate_button, 'String', progress
        );
end
if (not(isLimitedBySamples) && (isLimitedByTime))
    progress =  strcat(num2str((round(times_vector(
        row)/nTime * 100,2))),{' %'});
    set(handles.calculate_button, 'String', progress
        );
end
if ((isLimitedBySamples) && (isLimitedByTime))
    progress1 =  (round(row/nSamples * 100,2));
    progress2 = (round(times_vector(row)/nTime *
        100,2));
    if (progress1 >= progress2)
        progress_num = progress1;
    else
        progress_num = progress2;
    end
    progress =  strcat(num2str(progress_num),{' %'})
        ;
    set(handles.calculate_button, 'String', progress
        );
end


% Prepare next row
row = row+1;

% Finally, check if it has finished (wheter nSamples
    , nTime or
```

```
            % isStopped)
            Finished = checkIfFinished(isLimitedBySamples,
                nSamples, isLimitedByTime, nTime, isStopped, READ
                , times_vector);
            pause(0.01);
            % Remove zero rows (just in case)
            % READ = READ(any(READ,2),:)

        end
    end
    set(handles.calculate_button, 'String', 'Range');
catch
    display('Stop');
    % Closes the arduino connection
    % fclose(arduino);

    % Active rest of buttons
    changeVisibility(handles,'on');
    set(handles.update_button, 'Enable', 'on');
end
% Closes the arduino connection
% fclose(arduino);

% Active rest of buttons
changeVisibility(handles,'on');
set(handles.update_button, 'Enable', 'on');
```

A.2 Positioning code

A.2.1 Arduino Part

```
// Please read the ready-to-localize tuturial together with this
    example.
// https://www.pozyx.io/Documentation/Tutorials/
   ready_to_localize

#include <Pozyx.h>
#include <Pozyx_definitions.h>
#include <Wire.h>

//////////////////////////////////////////////////
////////////////// PARAMETERS ////////////////////
//////////////////////////////////////////////////

uint8_t num_anchors = 3;                                    //
   the number of anchors, at least 3
uint16_t anchors[4] = {0x6F13, 0x6F17, 0x6F23};     // the
   network id of the anchors: change these to the network ids of
    your anchors.
int32_t heights[4] = {0, 0, 0};                             //
   anchor z-coordinates in mm

// only required for manual anchor calibration. Please change
   this to the coordinates measured for the anchors

/*
  int32_t anchors_x[4] = {0, 0, 0, 0};              // anchor x-
     coorindates in mm
  int32_t anchors_y[4] = {0, 0, 0, 0};                //
     anchor y-coordinates in mm
*/

//////////////////////////////////////////////////

void setup() {
  Serial.begin(115200);

  int status = Pozyx.doAnchorCalibration(POZYX_2_5D, 10,
     num_anchors, anchors, heights);


  // if the automatic anchor calibration is unsuccessful, try
     manually setting the anchor coordinates.
  // fot this, you must update the arrays anchors_x, anchors_y
     and heights above
  // SetAnchorsManual();
```

```
  printCalibrationResult();

  Serial.println(F("START"));
}

void loop() {

  coordinates_t position;
  int status = Pozyx.doPositioning(&position, POZYX_2_5D, 1000);

  if (status == POZYX_SUCCESS)
  {
        int statusR = 1;
        device_range_t range;
        uint16_t destination_id = anchors[index]; //change
        statusR &= Pozyx.doRanging(destination_id, &range);
        if (status == POZYX_SUCCESS) {
          // Serial.print("Distance to anchor 0x");
          Serial.println(destination_id, HEX);
          // Serial.print(": \t");
          Serial.println(range.timestamp);
          // Serial.print("ms \t");
          Serial.println(range.distance);
          // Serial.println("mm \t");
        }
        index = index + 1;
      } else {
      printCoordinatesProcessing(position);
    }
}
```

A.2.2 Matlab Part

The code regarding the calibration button is shown below:

```
% --- Executes on button press in calibrate_button.
function calibrate_button_Callback(hObject, eventdata, handles)
% hObject    handle to calibrate_button (see GCBO)
% eventdata  reserved - to be defined in a future version of
   MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Change buttons visibility
changeVisibility(handles, 'off');
set(handles.list_of_anchors_pop,'Enable','off');
set(handles.list_of_algos_pop,'Enable','off');

% Clear the maps
cla(handles.map);
cla(handles.axes_pdf);
cla(handles.axes_algo);

% Initialize the arduino, open connection
global arduino;
global port;
arduino = initArduino(port);
fopen(arduino);

% Handle stop execution
global isStoped;
isStoped = 0;

global plotedPozyx;
plotedPozyx = 0;
global plotedBP;
plotedBP = 0;

%% Calibration
while(not(isStoped))
    try

    % Print calibration result, 1 inidcates success (see ino
       code)
        if ((fscanf(arduino, '%s')))
            % Get the number of anchors found, split the read
               data by :
            number_of_devices= fscanf(arduino, '%s');
            subIndex = strfind (number_of_devices, ':') + 1;
            number_of_devices = str2double(number_of_devices(
               subIndex:end));
```

```
        % Calibration results
        fscanf(arduino, '%s');
        % Anchors found
        fscanf(arduino, '%s');

        % Define a cell with number_of_devices rows and 5
           columns
        % (name, id, x_pos, y_pos, z_pos)
        global list_of_devices;
        list_of_devices = cell(number_of_devices, 5);

        % For each row
        for row = 1:number_of_devices
            % Read the device data
            list_of_devices_aux = strsplit(fscanf(arduino, '
               %s'),',');
            % Introduce in each column de data separated by
               commas
            list_of_devices(row,:) = list_of_devices_aux
               (1,:);
        end

        % And plot the anchors coordinates
        displayAnchors(handles.map, list_of_devices);
        displayAnchors(handles.axes_algo, list_of_devices);
    %% Arduino algorithm
    % Will be ploted at the beginning
        % Now the dara is read according this sequence:
        % One time positioning: x_pos, y_pos, z_pos
        % Number of anchors times ranging: id, time,
           distance
        % And this is repeated nSamples
        % We define a 3 columns matrix to storage all the
           data
        % and nSamples + number_of_devices*nSamples rows
        global nSamples;
        rows = nSamples + number_of_devices*nSamples;
        nData = 3;
        READ = cell(rows, nData);
        % Storage the data row by rows
        row = 1;
        % If everything went fine, go ahead
        if (not(fscanf(arduino, '%s') == 'START'))
            changeVisibility(handles, 'on')
            fclose(arduino);
            exit();
        end

        % Read the data and storage it
```

```
% Define a buffer, wait if no data to transmit
while (row <= rows)
    % no data to transmit => wait 0.5 s
    if(arduino.BytesAvailable <= 0)
        pause(0.5);
    end
    % Data to trasmit
    if (arduino.BytesAvailable)
        % write a row matrix
        for column = 1:nData
            READ{row, column}= fscanf(arduino, '%s')
                ;
        end
        % Update the progress bar (current row /
            total rows)
        progress =  strcat((num2str(round(row/rows *
            100,2))),{'␣%'});
        set(handles.calibrate_button, 'String',
            progress);
        row = row+1;
    end
end

% Split the READ data and present the info into
    three matrixes
% tag: nSamples rows x and y of the tag measured
    during nSamples
% positions: number_of_devices rows x 3 columns (id,
    x, y)
% list_of_distances: number_of_devices*nSamples rows
    x 3 columns
global tag;
global positions;
global list_of_distances;
[tag, positions, list_of_distances] = presentInfo(
    list_of_devices, READ, handles);

% Set the algortihm's map
presentPozyx(handles, tag);

plotedPozyx = 1;

% Update the table scenario and anchor data
set(handles.table_scenario, 'Data', positions);

% And anchor data
setAnchors( handles, positions, list_of_distances,
    nSamples );
```

```
            fclose(arduino);
            changeVisibility(handles, 'on');
            isStoped=1;
        else
            fclose(arduino);
            changeVisibility(handles, 'on');
            fprintf('exit');
        end
    catch
        fprintf('Stop');
        updatePorts(handles);
    end
end
```

Appendix B

Tests

B.1 Experiment 1

Test	LOS	Channel	Src.	Dest.	d	Room	Src. or.	Dest. or.
1	Yes	ch1	0x6670	0x6674	1000	N-1072	hor.	hor.
2	No	ch1	0x6670	0x6674	1000	N-1072	hor.	hor.
3	Yes	ch2	0x6670	0x6674	1000	N-1072	hor.	hor.
4	No	ch2	0x6670	0x6674	1000	N-1072	hor.	hor.
5	Yes	ch3	0x6670	0x6674	1000	N-1072	hor.	hor.
6	No	ch3	0x6670	0x6674	1000	N-1072	hor.	hor.
7	Yes	ch4	0x6670	0x6674	1000	N-1072	hor.	hor.
8	No	ch4	0x6670	0x6674	1000	N-1072	hor.	hor.
9	Yes	ch5	0x6670	0x6674	1000	N-1072	hor.	hor.
10	No	ch5	0x6670	0x6674	1000	N-1072	hor.	hor.
11	Yes	ch7	0x6670	0x6674	1000	N-1072	hor.	hor.
12	No	ch7	0x6670	0x6674	1000	N-1072	hor.	hor.
13	Yes	ch1	0x6670	0x6674	2000	N-1072	hor.	hor.
14	No	ch1	0x6670	0x6674	2000	N-1072	hor.	hor.
15	Yes	ch2	0x6670	0x6674	2000	N-1072	hor.	hor.
16	No	ch2	0x6670	0x6674	2000	N-1072	hor.	hor.
17	Yes	ch3	0x6670	0x6674	2000	N-1072	hor.	hor.
18	No	ch3	0x6670	0x6674	2000	N-1072	hor.	hor.
19	Yes	ch4	0x6670	0x6674	2000	N-1072	hor.	hor.
20	No	ch4	0x6670	0x6674	2000	N-1072	hor.	hor.

Test	LOS	Channel	Src.	Dest.	d	Room	Src. or.	Dest. or.
21	Yes	ch5	0x6670	0x6674	2000	N-1072	hor.	hor.
22	No	ch5	0x6670	0x6674	2000	N-1072	hor.	hor.
23	Yes	ch7	0x6670	0x6674	2000	N-1072	hor.	hor.
24	No	ch7	0x6670	0x6674	2000	N-1072	hor.	hor.
25	Yes	ch1	0x6670	0x6674	3000	N-1072	hor.	hor.
26	No	ch1	0x6670	0x6674	3000	N-1072	hor.	hor.
27	Yes	ch2	0x6670	0x6674	3000	N-1072	hor.	hor.
28	No	ch2	0x6670	0x6674	3000	N-1072	hor.	hor.
29	Yes	ch3	0x6670	0x6674	3000	N-1072	hor.	hor.
30	No	ch3	0x6670	0x6674	3000	N-1072	hor.	hor.
31	Yes	ch4	0x6670	0x6674	3000	N-1072	hor.	hor.
32	No	ch4	0x6670	0x6674	3000	N-1072	hor.	hor.
33	Yes	ch5	0x6670	0x6674	3000	N-1072	hor.	hor.
34	No	ch5	0x6670	0x6674	3000	N-1072	hor.	hor.
35	Yes	ch7	0x6670	0x6674	3000	N-1072	hor.	hor.
36	No	ch7	0x6670	0x6674	3000	N-1072	hor.	hor.
37	Yes	ch1	0x6670	0x6674	5000	N-1072	hor.	hor.
38	No	ch1	0x6670	0x6674	5000	N-1072	hor.	hor.
39	Yes	ch2	0x6670	0x6674	5000	N-1072	hor.	hor.
40	No	ch2	0x6670	0x6674	5000	N-1072	hor.	hor.
41	Yes	ch3	0x6670	0x6674	5000	N-1072	hor.	hor.
42	No	ch3	0x6670	0x6674	5000	N-1072	hor.	hor.
43	Yes	ch4	0x6670	0x6674	5000	N-1072	hor.	hor.
44	No	ch4	0x6670	0x6674	5000	N-1072	hor.	hor.
45	Yes	ch5	0x6670	0x6674	5000	N-1072	hor.	hor.
46	No	ch5	0x6670	0x6674	5000	N-1072	hor.	hor.
47	Yes	ch7	0x6670	0x6674	5000	N-1072	hor.	hor.
48	No	ch7	0x6670	0x6674	5000	N-1072	hor.	hor.
49	No	ch1	0x6670	0x6674	1000	N-2017	hor.	hor.
50	No	ch2	0x6670	0x6674	1000	N-2017	hor.	hor.
51	No	ch3	0x6670	0x6674	1000	N-2017	hor.	hor.

Test	LOS	Channel	Src.	Dest.	d	Room	Src. or.	Dest. or.
52	No	ch4	0x6670	0x6674	1000	N-2017	hor.	hor.
53	No	ch5	0x6670	0x6674	1000	N-2017	hor.	hor.
54	No	ch7	0x6670	0x6674	1000	N-2017	hor.	hor.
55	No	ch1	0x6670	0x6674	3000	N-2017	hor.	hor.
56	No	ch2	0x6670	0x6674	3000	N-2017	hor.	hor.
57	No	ch3	0x6670	0x6674	3000	N-2017	hor.	hor.
58	No	ch4	0x6670	0x6674	3000	N-2017	hor.	hor.
59	No	ch5	0x6670	0x6674	3000	N-2017	hor.	hor.
60	No	ch7	0x6670	0x6674	3000	N-2017	hor.	hor.
61	No	ch1	0x6670	0x6674	5000	N-2017	hor.	hor.
62	No	ch2	0x6670	0x6674	5000	N-2017	hor.	hor.
63	No	ch3	0x6670	0x6674	5000	N-2017	hor.	hor.
64	No	ch4	0x6670	0x6674	5000	N-2017	hor.	hor.
65	No	ch5	0x6670	0x6674	5000	N-2017	hor.	hor.
66	No	ch7	0x6670	0x6674	5000	N-2017	hor.	hor.
67	Yes	ch1	0x6674	0x6670	1000	N-1072	hor.	hor.
68	No	ch1	0x6674	0x6670	1000	N-1072	hor.	hor.
69	Yes	ch1	0x6670	0x6F7C	1000	N-1072	hor.	hor.
70	No	ch1	0x6670	0x6F7C	1000	N-1072	hor.	hor.
71	Yes	ch1	0x6670	0x6F5D	1000	N-1072	hor.	hor.
72	No	ch1	0x6670	0x6F5D	1000	N-1072	hor.	hor.
73	Yes	ch1	0x667C	0x6F5D	1000	N-1072	hor.	hor.
74	No	ch1	0x6670	0x6F5D	1000	N-1072	hor.	hor.
75	Yes	ch1	0x664E	0x6F0F	1000	N-1072	down	up
76	Yes	ch1	0x664E	0x6F0F	1000	N-1072	down	down
77	Yes	ch1	0x664E	0x6F0F	1000	N-1072	up	down
78	Yes	ch1	0x664E	0x6F0F	1000	N-1072	up	up
79	Yes	ch1	0x664E	0x6F0F	3000	N-1072	down	up
80	Yes	ch1	0x664E	0x6F0F	3000	N-1072	down	down
81	Yes	ch1	0x664E	0x6F0F	3000	N-1072	up	down
82	Yes	ch1	0x664E	0x6F0F	3000	N-1072	up	up

Test	LOS	Channel	Src.	Dest.	d	Room	Src. or.	Dest. or.
83	Yes	ch1	0x664E	0x6F0F	5000	N-1072	down	up
84	Yes	ch1	0x664E	0x6F0F	5000	N-1072	down	down
85	Yes	ch1	0x664E	0x6F0F	5000	N-1072	up	down
86	Yes	ch1	0x664E	0x6F0F	5000	N-1072	up	up

Table B.1: Experiment 1. List of tests. *d* in mm.

B.2 Experiment 2

Test	Src.	Dest.	d	Azimuth	Roll	Attachment	Gain
87	0x6670	0x666D	2180	0	0	S(Hull)-D(Hull)	0
88	0x6670	0x666D	2180	0	-20	S(Hull)-D(Hull)	0
89	0x6670	0x666D	2180	0	-40	S(Hull)-D(Hull)	0
90	0x6670	0x666D	2180	0	-60	S(Hull)-D(Hull)	0
91	0x6670	0x666D	2180	0	-80	S(Hull)-D(Hull)	0
92	0x6670	0x666D	2180	0	-100	S(Hull)-D(Hull)	0
93	0x6670	0x666D	2180	0	-120	S(Hull)-D(Hull)	0
94	0x6670	0x666D	2180	0	-140	S(Hull)-D(Hull)	0
95	0x6670	0x666D	2180	0	-160	S(Hull)-D(Hull)	0
96	0x6670	0x666D	2180	0	-180	S(Hull)-D(Hull)	0
97	0x6670	0x666D	2180	0	-200	S(Hull)-D(Hull)	0
98	0x6670	0x666D	2180	0	-220	S(Hull)-D(Hull)	0
99	0x6670	0x666D	2180	0	-240	S(Hull)-D(Hull)	0
100	0x6670	0x666D	2180	0	-270	S(Hull)-D(Hull)	0
101	0x6670	0x666D	2180	0	-290	S(Hull)-D(Hull)	0
102	0x6670	0x666D	2180	0	-310	S(Hull)-D(Hull)	0
103	0x6670	0x666D	2180	0	-330	S(Hull)-D(Hull)	0
104	0x6670	0x666D	2180	0	-350	S(Hull)-D(Hull)	0
105	0x6670	0x666D	2180	0	-260	S(Hull)-D(Hull)	0
106	0x6670	0x666D	2180	0	-280	S(Hull)-D(Hull)	0

Test	Src.	Dest.	d	Azimuth	Roll	Attachment	Gain
107	0x6670	0x666D	2180	0	-300	S(Hull)-D(Hull)	0
108	0x6670	0x666D	2180	0	-320	S(Hull)-D(Hull)	0
109	0x6670	0x666D	2180	0	-340	S(Hull)-D(Hull)	0
110	0x6670	0x666D	2260	90	0	S(Hull)-D(Hull)	0
111	0x6670	0x666D	2213	45	0	S(Hull)-D(Hull)	0
112	0x6670	0x666D	2200	-45	0	S(Hull)-D(Hull)	0
113	0x6670	0x666D	2259	-90	0	S(Hull)-D(Hull)	0
114	0x6670	0x666D	2180	0	0	S(Hull)-D(Hull)	33.5
115	0x6670	0x666D	2180	0	90	S(Hull)-D(Hull)	33.5
116	0x6670	0x666D	2180	0	-180	S(Hull)-D(Hull)	33.5
117	0x6670	0x6F59	2167	0	0	S(Hull)-D(PCB)	0
118	0x6670	0x6F59	2167	0	-20	S(Hull)-D(PCB)	0
119	0x6670	0x6F59	2167	0	-40	S(Hull)-D(PCB)	0
120	0x6670	0x6F59	2167	0	-60	S(Hull)-D(PCB)	0
121	0x6670	0x6F59	2167	0	-80	S(Hull)-D(PCB)	0
122	0x6670	0x6F59	2167	0	-100	S(Hull)-D(PCB)	0
123	0x6670	0x6F59	2167	0	-120	S(Hull)-D(PCB)	0
124	0x6670	0x6F59	2167	0	-140	S(Hull)-D(PCB)	0
125	0x6670	0x6F59	2167	0	-160	S(Hull)-D(PCB)	0
126	0x6670	0x6F59	2167	0	-180	S(Hull)-D(PCB)	0
127	0x6670	0x6F59	2167	0	-200	S(Hull)-D(PCB)	0
128	0x6670	0x6F59	2167	0	-220	S(Hull)-D(PCB)	0
129	0x6670	0x6F59	2167	0	-240	S(Hull)-D(PCB)	0
130	0x6670	0x6F59	2167	0	-260	S(Hull)-D(PCB)	0
131	0x6670	0x6F59	2167	0	-280	S(Hull)-D(PCB)	0
132	0x6670	0x6F59	2167	0	-300	S(Hull)-D(PCB)	0
133	0x6670	0x6F59	2167	0	-320	S(Hull)-D(PCB)	0
134	0x6670	0x6F59	2167	0	-340	S(Hull)-D(PCB)	0
135	0x6670	0x6F59	2260	90	0	S(Hull)-D(PCB)	0
136	0x6670	0x6F59	2213	45	0	S(Hull)-D(PCB)	0
137	0x6670	0x6F59	2167	-45	0	S(Hull)-D(PCB)	0

Test	Src.	Dest.	d	Azimuth	Roll	Attachment	Gain
138	0x6670	0x6F59	2259	-90	0	S(Hull)-D(PCB)	0
139	0x6670	0x6F59	2185	0	0	S(PCB)-D(PCB)	0
140	0x6670	0x6F59	2185	0	-20	S(PCB)-D(PCB)	0
141	0x6670	0x6F59	2185	0	-40	S(PCB)-D(PCB)	0
142	0x6670	0x6F59	2185	0	-60	S(PCB)-D(PCB)	0
143	0x6670	0x6F59	2185	0	-80	S(PCB)-D(PCB)	0
144	0x6670	0x6F59	2185	0	-100	S(PCB)-D(PCB)	0
145	0x6670	0x6F59	2185	0	-120	S(PCB)-D(PCB)	0
146	0x6670	0x6F59	2185	0	-140	S(PCB)-D(PCB)	0
147	0x6670	0x6F59	2185	0	-160	S(PCB)-D(PCB)	0
148	0x6670	0x6F59	2185	0	-180	S(PCB)-D(PCB)	0
149	0x6670	0x6F59	2185	0	-200	S(PCB)-D(PCB)	0
150	0x6670	0x6F59	2185	0	-220	S(PCB)-D(PCB)	0
151	0x6670	0x6F59	2185	0	-240	S(PCB)-D(PCB)	0
152	0x6670	0x6F59	2185	0	-260	S(PCB)-D(PCB)	0
153	0x6670	0x6F59	2185	0	-280	S(PCB)-D(PCB)	0
154	0x6670	0x6F59	2185	0	-300	S(PCB)-D(PCB)	0
155	0x6670	0x6F59	2185	0	-320	S(PCB)-D(PCB)	0
156	0x6670	0x6F59	2185	0	-340	S(PCB)-D(PCB)	0

Table B.2: Experiment 2. List of tests. *d* in mm, angles in degrees, gain in dB.

Bibliography

[ATIW05] I. R. L. M. A. T. Ihler, J. W. Fisher and A. S. Willsky, "Nonparametric belief propagation for self-localization of sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 23, pp. 809–819, April 2005.

[DDW09] U. F. A. G. Davide Dardari, Andrea Conti and M. Z. Win, "Ranging with ultrawide bandwidth signals in multipath environmentss," vol. 97, no. 2, pp. 404–407, February 2009.

[GG05] F. Gustafsson and F. Gunnarsson, "Mobile positioning using wireless networks," *IEEE Signal Processing Magazine*, vol. 22, no. 4, p. 41–53, July 2005.

[HB01] J. Hightower and G. Borriello, "Location systems for ubiquitous computing," vol. 34, pp. 57–66, August 2001.

[HWW09] J. L. Henk Wymeersch and M. Z. Win, "Cooperative localization in wireless networks," vol. 97, no. 2, p. 427–431, February 2009.

[Lie07] J. Lien, "A framework for cooperative localization in ultra-wideband wireless networks," pp. 15–25, May 2007.

[NPC05] S. K. A. O. H. I. R. L. M. N. Patwari, J. N. Ash and N. S. Correal, "Locating the nodes: Cooperative localization in wireless sensor networks," *IEEE Signal Processing Magazine*, vol. 22, pp. 54–69, July 2005.

[Poza] "Arduino: getting started," https://www.pozyx.io/Documentation/Tutorials/gettingStarted, accessed: 7/10/2016.

[Pozb] "Arduino support from matlab," https://de.mathworks.com/hardware-support/arduino-matlab.html, accessed: 7/10/2016.

[Pozc] "Arduino tutorial: ready to localize," https://www.pozyx.io/Documentation/Tutorials/ready_to_localize, accessed: 7/10/2016.

[Pozd] "Arduino tutorial: ready to range," https://www.pozyx.io/Documentation/Tutorials/ready_to_range, accessed: 9/10/2016.

[RMP03] D. K. R. Moses and R. Patterson, "A self localization method for wireless sensor networks," *EURASIP J. Appl. Signal Process*, March 2003.

[SGS05] G. B. G. H. K. A. F. M. H. V. P. S. Gezici, Z. Tian and Z. Sahinoglu, "Localization via ultra-wideband radios: a look at positioning aspects for future sensor networks," *IEEE Signal Processing Magazine*, vol. 22, no. 4, July 2005.

[SY04] F. Sivrikaya and B. Yener, "Time synchronization in sensor networks: A survey," vol. 18, no. 4, pp. 45–50, 2004.

[TPC06] M. M. D. D. T. Pavani, G. Costa and A. Conti, "Experimental results on indoor localization techniques through wireless sensors network," vol. 2, pp. 663–667, May 2006.

[TUH16a] *Handbook for Matlab GUI: Positioning Algorithms Comparison*, TUHH, October 2016.

[TUH16b] *Handbook for Matlab GUI: Ranging between two devices*, TUHH, October 2016.

[WS98] M. Z. Win and R. A. Scholtz, "On the energy capture of ultra-wide bandwidth signals in dense multipath environment," vol. 2, pp. 245–247, September 1998.