



Norwegian University of
Science and Technology

An Experimental Study of a Redundant Position System for the Autonomous Ferry Milliampere Utilizing Ultra Wide- Band Radio Frequencies

Svein Olav Hegerland

Master of Science in Industrial Cybernetics

Submission date: June 2018

Supervisor: Edmund Førland Brekke, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Problem Formulation

The student should investigate the feasibility and compatibility of the Pozyx UWB system as part of the existing sensor network onboard the autonomous ferry Milliampere. The tasks are as follows:

- Algorithms must be developed to process the raw data from the Pozyx UWB system and to estimate position.
- Experiments should test the 2D and 3D precision of the pose estimates generated from the Pozyx UWB sensors.
- Experiments should test how the precision is affected when the sensor is close to the perimeter defined by the reference points.
- It should be discussed how the physical geometry of the Pozyx UWB system (as reference points) will affect the precision.
- The system should be discussed concerning compatibility with the existing sensor system onboard Milliampere, in addition to be evaluated regarding the practical working range of the sensors.

Summary

The interest in developing Autonomous Surface Vessels (ASVs) has increased significantly during the last decades, and cutting-edge systems are being developed. The maritime industry is focusing heavily on the research topic, and progress towards the next generation of smart vessels is made every day. Yet, there are still challenges associated with autonomous ships, one being the reliability of the sensors used for guidance.

The Global Navigation Satellite System (GNSS) is commonly used onboard ASVs for this purpose. Although the precision of GNSS is considered to be adequate, the performance is still unreliable if radio signals are blocked, or signal coverage is poor. It is therefore necessary to develop robust methods for redundant positioning, capable of bringing the ASVs safely to dock in case of failing GNSS signals.

Milliampere is an autonomous ferry intended to operate in the narrow channel between Brattøra and Ravnkloa in Trondheim. It is an ongoing project founded by the Norwegian University of Science and Technology (NTNU) and the Centre for Autonomous Marine Operations and Systems (AMOS), and is called the Autoferry Project. The primary navigation system onboard the ferry is utilizing GNSS, and a redundant method for positioning is sought.

This thesis is an experimental study of a redundant position system for the presented ferry. As opposed to GNSS which measures relative distances to satellites, the investigated sensors measure the distances to stationary reference points utilizing Ultra Wide-Band (UWB) radio frequency signals. Evaluating the candidate system concerned of experimental testing of hardware and post-processing of the collected data. Trilateration algorithms were investigated and evaluated concerning precision. The accuracy of the algorithms was also tested when the targeted receive was moving close to the perimeters defined by the reference points, and simulations were used to validate the experimental results. The sensor system was further evaluated concerning compatibility with the existing sensor network onboard Milliampere. A node was implemented in Robot Operating System on a laptop running the same distributions of software as Milliampere. The node published sensor data in real time. The sensor system was also investigated regarding practical working range, evaluating the feasibility of the suggested location system in the channel.

Sammendrag

Interessen for å utvikle autonome skip har økt betydelig i løpet av de siste tiårene, og nye banebrytende systemer blir stadig utviklet. Den maritime industrien fokuserer tungt på forskningsområdet, og fremskritt mot neste generasjons smarte fartøyer blir gjort hver dag. Likevel er det fortsatt utfordringer knyttet til autonome skip, en er påliteligheten til sensorer som brukes til posisjonering og navigering.

Global Navigation Satellite System (GNSS) brukes ofte ombord på autonome skip for dette formålet. Selv om presisjonen kan være svært god, så er fortsatt ytelsen til GNSS variabel når radiosignaler blir blokkert, eller signaldekningen er dårlig. Det er derfor nødvendig å utvikle robuste metoder for redundant posisjonering, som er i stand til å bringe fartøyene trygt i havn dersom GNSS svikter.

Milliampere er en autonom ferge som utvikles for å frakte personell over kanalen mellom Brattøra og Ravnkloa i Trondheim. Dette er et pågående prosjekt som finansieres av Norges Teknisk-Naturvitenskapelige Universitet og Centre for Autonomous Marine Operations and Systems, og omtales som "the Autoferry Project". Det primære navigasjonssystemet ombord på fergen benytter GNSS, og det søkes derfor en redundant metode for posisjonering.

Denne masteroppgaven er en eksperimentelt studie av et foreslått redundant posisjoneringssystem for den nevnte fergen. I motsetning til GNSS som måler relativ avstand til satellitter, måler de undersøkte sensorene avstand til stasjonære referansepunkter ved bruk av ultra wide-band radiosignaler. Evalueringen av det aktuelle sensorsystemet har omfattet eksperimentell testing og post-prosessering av innhentet data. Flere algoritmer har blitt implementert for å beregne posisjon, både innenfor og utenfor perimeterne definert av referansepunktene. Simuleringer ble også brukt til å validere de eksperimentelle resultatene. Sensorsystemet ble ytterligere vurdert med hensyn til kompatibilitet med eksisterende sensornettverk ombord på Milliampere. En sensornode ble implementert i Robot Operating System på en bærbar datamaskin som kjørte de samme software-distribusjonene som Milliampere. Noden publiserte sensordata i sanntid. Sensorsystemet ble også undersøkt angående praktisk rekkevidde, med hensikt å vurdere om sensorene kunne brukes over avstandene i den aktuelle kanalen.

Preface

This research work has been conducted under supervision of Associate Professor Edmund Førland Brekke (Department of Cybernetics) and co-supervisor Associate Professor Egil Eide (Department of Electronic Systems) at the Norwegian University of Science and Technology, during the spring semester of 2018. I will always be grateful for their guidance, and for responding to my questions throughout the semester.

The work in this thesis concerns testing of sensor hardware and post-processing of collected data, aiming to evaluate the feasibility of these sensors as part of a docking system onboard the autonomous ferry Milliampere.

Prior to writing this thesis, my specialization project was an analysis of strengths and weaknesses of different candidate sensors. Based on this analysis, I chose to focus on ultra-wideband radio as a promising candidate. In contrast to the specialization project, this thesis concerns practical implementation of the system, and significant amounts of time was therefore spent to become familiar with code, sensor systems and other hardware.

Mainly, two programming languages have been used during the project. Matlab has been used for post-processing and for implementing the various trilateration algorithms presented in later chapters. An Arduino has been used to collect the sensor data, and working with the Arduino IDE has thus been an important part of the project.

Last, but not least, two other persons must also be paid gratitude, making this thesis possible. First, I would like to extend a massive thank to Krzysztof Cisek for helping me with the sensors, and for sharing his knowledge with me. Some of the code used in this thesis is originating from Cisek et al. (2017), and will from now on be referred to as "the UAV-code". Also, thanks to Cisek for borrowing me his micro-USB, even though he used it to charge his phone. Second, I would like to thank Doctor Torleiv Håland Bryne for his help, and for the fast replies and expertise regarding RTK-GPS configurations. Also, thanks for sharing some example code with me, originating from Farrell (2008), which from now on will be referred to as "the ECEF to NED-code".

Trondheim, 2018-06-28

Bvein Olav Hegerland

Table of Contents

Problem Formulation	i
Summary	iii
Sammendrag	v
Preface	vii
Table of Contents	xi
Abbreviations	xiii
Symbols	xv
1 Introduction	1
1.1 The Autonomous Ferry Scenario	2
1.2 Motivation	3
1.3 Objective and Scope	3
1.4 Outline of the Thesis	4
2 A Brief Literature Review on Ultra Wide-Band and Trilateration Algorithms	5
3 Background Theory	7
3.1 Ultra Wide-Band Location Systems	7
3.2 Common Coordinate Systems Used to Describe Position on the Earth	9
3.3 Trilateration	11
3.4 Linear Least Squares Trilateration	13
3.5 Non Linear Least Squares Trilateration	14
3.6 An Algebraic Approach To the Trilateration Problem	15

4	Sensor System	17
4.1	Description of the Pozyx System	17
4.2	Real Time Clock	20
4.3	RTK-GPS	20
4.4	Robot Operating System	20
5	Implementation and Experiments	23
5.1	The Indoor Experiment	23
5.2	The Outdoor Experiment	25
5.3	Working With the Collected Data	29
5.3.1	Processing of the Collected Indoor Data	29
5.3.2	Processing of the Collected Outdoor Data	32
5.4	Implementation of the Trilateration Algorithms	37
5.4.1	Linear Least Squares Pose Estimation	37
5.4.2	Algebraic Solution to Pose Estimation	39
5.4.3	Non Linear Least Squares Pose Estimation	39
6	Experimental Results	41
6.1	Results From the Indoor Experiment	41
6.1.1	Linear Least Squares Solutions	42
6.1.2	Non Linear Least Squares Solutions	44
6.1.3	Algebraic Solutions	50
6.2	Validation of the Indoor Experimental Results	54
6.2.1	Monte Carlo Simulations	54
6.3	Results From the Outdoor Experiment	60
6.3.1	Walk NR. 1	61
6.3.2	Walk NR. 2	66
6.3.3	The Outdoor Range Test	71
6.4	Validation of the Outdoor Experimental Results	74
6.4.1	Monte Carlo Simulation	74
7	Implementing a Sensor Node in Robot Operating System	75
7.1	Software Environment	75
7.2	Publishing Data	75
8	Discussion	79
8.1	The Processing Algorithms	79
8.2	Measuring The Local Coordinate System	79
8.3	2D and 3D Precision	80
8.4	Tag Located Close to the Perimeter	81
8.5	Robustness to Multipath of Signals	82
8.6	Physical Geometry of the Location System	83
8.7	The Outdoor Range Test	83
8.8	Robot Operating System	84
8.9	The Sensors	84

9 Conclusion and Future Work	87
9.1 Conclusion	87
9.2 Future Work	88
Bibliography	89
Appendix A	93
Appendix B	95

Abbreviations

ASV	=	Autonomous Surface Vessel
ECEF	=	Earth Centered Earth Fixed
GNSS	=	Global Navigation Satellite System
GLONASS	=	Globalnaya Navigazionnaya Sputnikovaya Sistema
GPS	=	Global Positioning System
LIDAR	=	Light Detection and Ranging
LLS	=	Linear Least Squares
LOS	=	Line Of Sight
MCU	=	Micro Processor Unit
NED	=	North East Down
NLLS	=	Non Linear Least Squares
OD	=	Over Determined
ROS	=	Robot Operating System
RSS	=	Received Signal Strength
RTC	=	Real Time Clock
RTK	=	Real Time Kinematic
TDOA	=	Time Difference Of Arrival
TOF	=	Time Of Flight
TRR	=	Thrust-Region-Reflective
UAV	=	Unmanned Aerial Vehicle
UD	=	Under Determined
USB	=	Universal Serial Bus
UTC	=	Coordinated Universal Time
UWB RTLS	=	Ultra Wide-Band Real Time Location System

Symbols

Δt	Time difference
Δf	Bandwidth
x, y, z	Cartesian coordinates
x_e, y_e, z_e	Earth fixed coordinates
\vec{x}	Position vector
\vec{x}_p	Position vector, particular solution
\vec{x}_h	Position vector, homogeneous solution
N, E, D	North, East, Down
λ	Longitude
φ	Latitude
R	Rotation matrix
P	Calculated position
r_i	Range measurement
B_i	Reference point
J	Jacobian Matrix

Introduction

Research concerning autonomous vehicles has become increasingly popular during the last decades, largely due to the economic interest from the car industry. Stakeholders are seeing the possibility to reduce costs by relying on autonomous navigation, in addition to reduce the risk of human-based errors. This has led to great progress on the field of autonomous land-based robots, while the remaining branches of autonomous navigation have been lagging behind.

However, autonomous ships are more relevant than ever before, and many, including Jokioinen (2016), are of the opinion that the technologies to make them a reality exists, but the field is immature and challenges are still to combine them cost-effectively and reliably. The fact that human-based errors can be removed is, of course, a great advantage, but other kinds of risk will arise. By solving problems and challenges such as guidance, navigation and control, as discussed in Liu et al. (2016), ASVs have the potential to redefine the maritime industry.

GNSS is a common way for ASVs to navigate, and is often the main navigation system onboard these vessels. These systems are not entirely immune to environmental noise and accumulative errors, so some uncertainty in position is always present (USA Department of Defense, 2008). Coupled with this is that blocked GNSS signals are also more likely to occur when the ASV is navigating in close proximity of e.g. tall buildings and bridges and could therefore also add to the problem (Leedekerken et al., 2014). With this in mind, ASVs relying on GNSS for navigation would therefore need additional measurements to ensure redundancy in case of uncertain or failing GNSS signals.

Various passive and active sensor systems for navigation might be adequate for these mentioned environments; examples could be the vision-based system presented in Huntsberger et al. (2011) or the Light Detection and Ranging (LIDAR) system presented in Lee et al. (2017). However, these systems might still fail in a marine environment due to challenging environmental conditions. As discussed in the technical report by Hegerland (2018), failing sensor systems could be due to factors such as fog, rain and snow. To that end, choosing the most suitable system for positioning in a marine environment can be challenging, and there are pros and cons regarding each and every sensor system.

Consequences of poorly designed sensor solutions might then be uncertain pose estimates, prohibiting the vehicle from performing assigned tasks as expected. The vehicle is especially vulnerable when maneuvering close to shore, and docking is consequently a topic worth paying some extra attention. Docking can be defined as the act of maneuvering a vessel from pose A to pose B , right next to the dock, for then to perform a low-velocity collision with the dock, applying some thrust towards it for safe loading and offloading (Bitar, 2017). Associated with this, is the need of a method for ensuring robust and precise measurements, making sure to always estimate position with a high degree of certainty and precision.

1.1 The Autonomous Ferry Scenario

The autonomous passenger ferry Milliampere is founded by NTNU and AMOS, and is referred to as the Autoferry Project. It is intended to operate in the approximately 110 m wide channel between Brattøra and Ravnkloa in Trondheim as an "on demand" ferry for transporting personnel. Milliampere will mainly be relying on GNSS for state estimation, but will also utilize several other sensor systems, such as LIDARs, radars and cameras, to aid the primary navigation system. A picture of the ferry can be seen in Figure 1.1.



Figure 1.1: Milliampere approaching land. The picture is taken from gemini.no. Photo: Egil Lund, NTNU.

The ferry will most of the time travel the same distance (back and forth between the docks) on a daily basis, and systems utilizing UWB signals and stationary reference points alongside the channel, for redundant positioning, are therefore to be considered. The scenario is illustrated in Figure 1.2. This location system can possibly provide the ferry with precise and frequent pose estimates, as long as the ferry is within signal coverage of the given system.

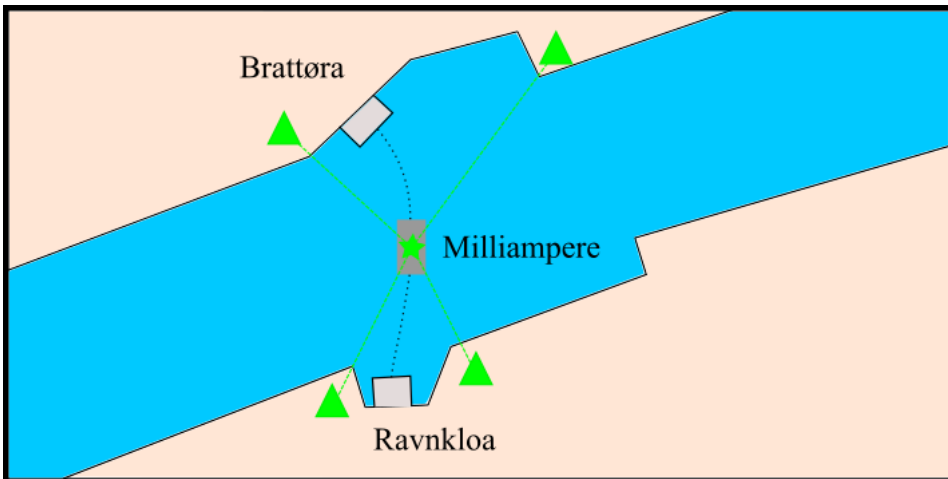


Figure 1.2: An illustration of Milliampere traveling between the docks. The stationary reference points are illustrated by green triangles, the mobile target (tag) is illustrated by a green star.

1.2 Motivation

This master thesis will investigate the possibility of integrating the Pozyx UWB location system on board the autonomous ferry Milliampere and the surrounding harbor. The UWB system is utilizing radio frequency signals for communication between the devices, much resembling the working principles of GNSS and satellite technology. However, where GNSS uses satellites for calculating position, the UWB system uses local, stationary reference points for the same task. These stationary reference points are later referred to as "anchors".

Motivating this research is the need of a robust, efficient and low-cost solution to the autonomous docking problem. The system will hopefully provide pose estimates of centimeter accuracy, and an update rate sufficient to support the main navigation system relying on GNSS. Furthermore, this system is, as far as it is known to the author, not implemented on an ASV before, and is therefore also of academic interest.

1.3 Objective and Scope

The work in this thesis will investigate the feasibility and compatibility of the Pozyx UWB system as part of the existing sensor network onboard the presented ferry. The location system will be evaluated concerning precision and reliability both inside and outside of the perimeters defined by the anchors, and a recommendation will be given to the Autoferry Project. See also the "Problem Formulation" for a complete description of objectives.

1.4 Outline of the Thesis

The thesis is structured in the following way:

- First, a short literature review concerning UWB systems and trilateration algorithms is presented in Chapter 2. Here, the articles with the most significant contributions to the project are presented and made acquaint for the reader.
- Next, the background theory on calculating distances, trilateration and coordinate systems are presented in Chapter 3.
- Chapter 4 introduces the sensors which have been used during the project work.
- Chapter 5 explains the experimental methods and how data was collected. The chapter does also explain how the trilateration algorithms were implemented.
- The experimental results are presented in Chapter 6. The chapter is divided into two parts: the indoor experiment and the outdoor experiment. Data from each of the trilateration algorithms are presented and compared to true earth coordinates. The experimental results are also validated by Monte Carlo simulations.
- Chapter 7 describes how the Pozyx location system was implemented as a sensor node in Robot Operating System (ROS).
- In Chapter 8 we analyze and evaluate the conducted experiments and the lessons learned when the sensor systems were used.
- Finally, a conclusion follows in Chapter 9. A recommendation is made for the Autoferry project.

A Brief Literature Review on Ultra Wide-Band and Trilateration Algorithms

This chapter will briefly present the articles which have contributed the most to this master thesis, shedding some light on the research fields of pose estimation utilizing UWB systems and possible applications for ASVs.

Ultra Wideband Indoor Positioning Technologies: Analysis and Recent Advances

Alarifi et al. (2016) presented a general review of recent advances on the field of UWB systems. Here, several indoor UWB positioning systems and algorithms were compared and evaluated. The paper was ended with a SWOT analysis concerning the UWB technology (see Appendix A for complete SWOT analysis):

- *Strengths*: Low power consumptions, high level of multipath resolution, does not interfere with most of the existing radio systems.
- *Weaknesses*: May affect Global Positioning Systems (GPS) and aircraft navigation radio equipment, potential interference to the existing systems which operates in the ultra-wide spectrum due to misconfiguration.
- *Opportunities*: Robot guidance, tracking systems, shipboard environment applications, applications for noisy environments.
- *Threats*: In some cases not totally immune to multipath effects.

Ultra-Wide Band Real Time Location Systems: Practical Implementation and UAV Performance Evaluation

Cisek et al. (2017) Implemented a UWB Real Time Location System (RTLS) onboard an Unmanned Aerial Vehicle (UAV) aiming to evaluate the sensor performance and the pose estimation capability. The research was conducted in order to test an alternative positioning solution to Real Time Kinematic (RTK) - GPS, due to problems such as Time-To-Fix and unavailability of GNSS signals for fast moving AUVs. The Pozyx system was one of the tested UWB modules, and distance-errors and positioning-errors were presented for both 2D and 3D positioning. An encountered challenge discussed in the paper was misalignment between the calculated UWB RTLS coordinates and the local North-East-Down (NED) frame, thus being a possible error source.

Determination of a Position in Three Dimensions Using Trilateration and Approximate Distances

Murphy and Hereman (1995) illustrated and discussed the mathematical solution to an ill-conditioned position estimation problem for the Thunder Basin Coal Company. A Linear Least Squares (LLS) algorithm was tested but did not satisfy the requested precision. Instead, a Non Linear Least Squares (NLLS) algorithm was recommended, fulfilling the initial goals for accuracy. They concluded that the precision was severely degraded when the target was located outside of the perimeter of the fixed positioned beacons (anchors).

An Algebraic Solution to the Multilateration Problem

Norrdine (2012) presented an efficient method for solving nonlinear multilateration problems both with and without over determination (more reference points than unknowns and an equal number of reference points and unknowns, respectively). The computational complexity was low, and the problem could be solved by means of linear algebra methods. Indoor experiments concluded that the algorithm performed well in terms of precision, but slow update rate of the UWB system prohibited trials with moving targets.

Background Theory

This chapter will present the background theory relevant to the project. It starts by explaining some UWB radio frequency signal applications and three different methods for measuring distance. Next is a short introduction to common coordinate systems used to represent position. After this, the chapter proceeds to algorithms capable of utilizing this range information and convert it to pose estimates.

3.1 Ultra Wide-Band Location Systems

Federal Communications Commission is defining UWB as a radio signal occupying a portion of the frequency spectrum that is greater than 20 % of the center carrier frequency, or has a bandwidth greater than 500 MHz (Alarifi et al., 2016). UWB location systems are designed for transmitting extremely short pulses and spreading the information across a wide portion of the frequency band. By dividing the information over many frequencies, very low spectral power density is needed for transmitting vast amounts of data. Furthermore, the significant bandwidth and short pulses are also diminishing the effects from multipath interference and is therefore well suited for positioning systems.

For a UWB location system to calculate distances and position with cm accuracy, the width of the pulses must be extremely narrow. These narrow pulses are generated by mixing multiple radio signal frequencies with the aim of creating distinct and narrow pulses, as illustrated in Figure 3.1. According to Heisenberg’s uncertainty principle (3.1), also discussed in The Pozyx Manual (accessed June 07, 2018), a UWB system with bandwidth, Δf , of 500 MHz can in theory generate a pulse width, Δt , of approximately 0.16 ns, which is quite short compared to the 4 ns wide pulses which can be generated by a regular 20 MHz bandwidth WI-FI network.

$$\Delta f \Delta t \geq \frac{1}{4\pi} \tag{3.1}$$

Various techniques are used for calculating distances utilizing these UWB systems. Three possible approaches are Time Of Arrival (TOA), Time Difference Of Arrival (TDOA)

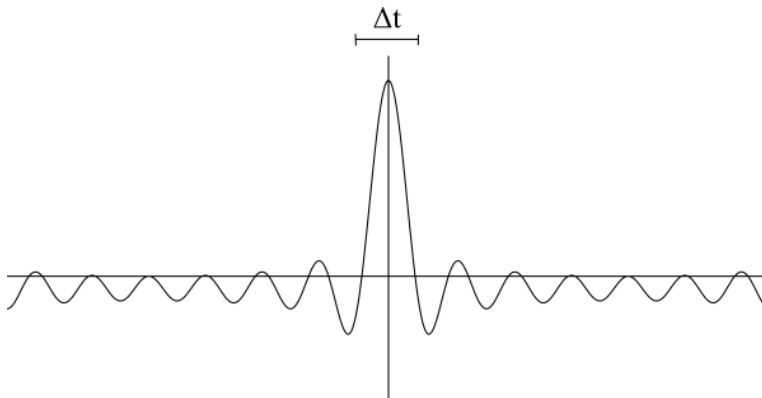


Figure 3.1: An illustrative pulse generated by adding sine waves of frequency 1 Hz, 2 Hz, 3 Hz, 4 Hz, 5 Hz, 6 Hz, 7 Hz and 8 Hz.

and Received Signal Strength (RSS).

TOA algorithms are based on the propagation time of signals (in this case radio frequency signals) from multiple transmitters (also referred to as "base stations") to the targeted receiver. By measuring the relative distance, i.e. the speed of light \times measured time, one can find the radius on where the receiver is located. By obtaining range information from three or more base stations, the intersection of these measured radii can be used to describe the transmitter's location in space.

TDOA algorithms are based on measuring the time difference on a received signal between pairs of base stations. An early application of this was locating cell phones for emergency calls based on information supplied from base stations (Drane et al., 1998). When the base stations measured the time difference between received signals (from the targeted cell phone), they could calculate the hyperbolic locus on which the phone had to be located, as illustrated in Figure 3.2. By comparing time differences between three or more base stations, they could use the intersection between two or more hyperbolic loci as a position estimate. The scenario can also be flipped, for instance, if the cell phone wants to know its own location. Then the phone would act as a receiver, measuring the time difference of incoming signals from the base stations. This would demand a very precise time synchronization between the base stations.

RSS algorithms are based on the signal strength measured at the targeted receiver. However, several factors are affecting the propagating signal, for instance geometrical loss and dissipation (Mohus and Holand, 1983). The measured signal at the receiver is, therefore, not proportional to the traveled signal distance, and is as a consequence not considered to be a good indicator of distances (Alarifi et al., 2016).

According to Alarifi et al. (2016) TOA and TDOA are the most accurate choices due

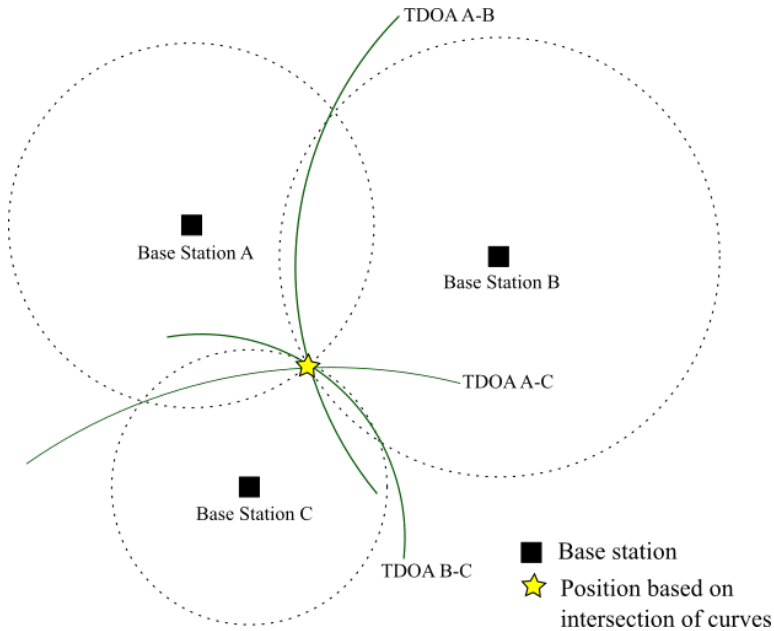


Figure 3.2: Base stations using TDOA in order to locate a target.

to the high time resolution of UWB systems and should be favored for UWB location applications. However, it should be noted that not all UWB systems are FCC certified, and precautions must be taken before use. Although indoor use of UWB is mostly not regulated, outdoor applications are more restricted (but temporary use can be allowed). This is due to interference of signals and for limiting the pollution of radio signal frequency bands such as e.g. WI-FI and other industrial networks.

Nevertheless, It is worth mentioning that the terms "outdoor" and "temporary" can be viewed as loosely defined, and much is left to interpretation. This could make it legal for UWB systems to operate in the outdoors if they are mounted outside on buildings and windows, or installed not so permanently (not "cemented") to the ground.

3.2 Common Coordinate Systems Used to Describe Position on the Earth

The geodetic coordinate system is a common way for GNSS systems to express position on the Earth's surface, using position information relative to the prime meridian, the equatorial plane and altitude above sea level, to inform the user where he or she is. It is not a regular Cartesian coordinate system, but describes a point on the earth in terms of longitude, latitude, ranging from -180° to 180° and -90° to 90° , respectively, and height.

However, other coordinate systems can be used to describe a point on the Earth's surface as well. The Earth Centered Earth Fixed (ECEF) coordinate system describes a point in space through Cartesian coordinates with respect to the origin defined to be located in

the center of the Earth (O_e) (Cai et al., 2011). The z-axis (Z_e) is defined to point towards the north pole, through the Earth's spin axis, and the x-axis (X_e) is pointing towards the Earth's surface at 0° latitude and 0° longitude. As expected, the y-axis (Y_e) is defined according to the right hand-rule, as shown in Figure 3.3.

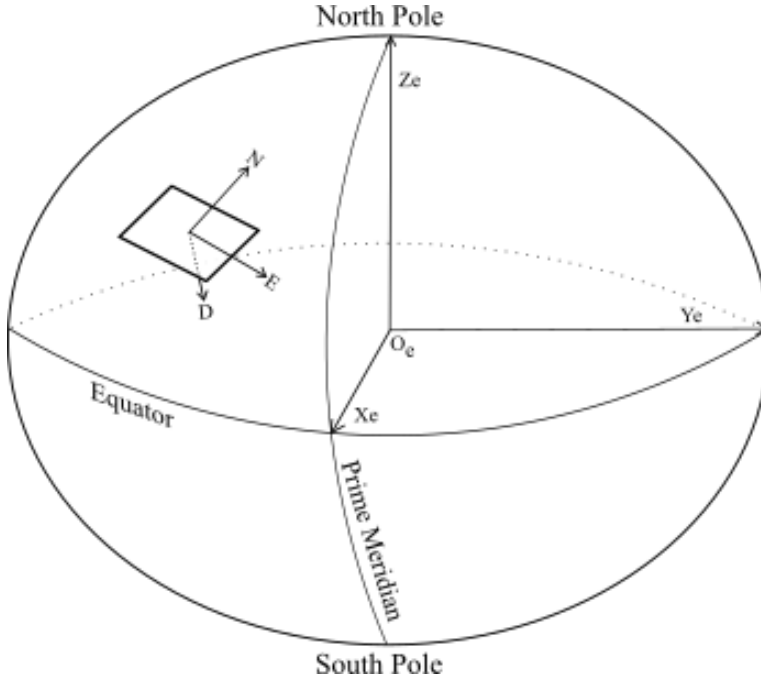


Figure 3.3: The ECEF coordinate system with the origin located in the center of the Earth. The NED frame has positive x-axis pointing towards the north pole and positive y-axis pointing towards the east.

As one can imagine, coordinates in terms of ECEF can be quite "big" (because of the radius of the Earth), and difficult to interpret intuitively. For this reason, when GNSS systems use ECEF coordinates to express position, it is often a good idea to translate these coordinates to a local coordinate system on the Earth's surface. A common way to do this through the NED coordinate system, with x-axis pointing north, y-axis pointing east and, by definition, z-axis pointing down, also shown in Figure 3.3. The translation from the ECEF frame (center of the Earth) to the local NED frame, also shown by Cai et al. (2011), can be expressed by the formula

$$P_{\text{NED}} = R_{\text{NED/ECEF}}(P_{\text{ECEF}} - P_{\text{ECEF}_{\text{REF}}}), \quad (3.2)$$

where P_{ECEF} is the ECEF position, $P_{\text{ECEF}_{\text{REF}}}$ is the the position of the origin of the local NED frame, $R_{\text{NED/ECEF}}$ is the rotation matrix

$$R_{\text{NED}/\text{ECEF}} = \begin{bmatrix} -\sin \varphi_{\text{ref}} \cos \lambda_{\text{ref}} & -\sin \varphi_{\text{ref}} \sin \lambda_{\text{ref}} & \cos \varphi_{\text{ref}} \\ -\sin \lambda_{\text{ref}} & \cos \lambda_{\text{ref}} & 0 \\ -\cos \varphi_{\text{ref}} \cos \lambda_{\text{ref}} & -\cos \varphi_{\text{ref}} \sin \lambda_{\text{ref}} & -\sin \varphi_{\text{ref}} \end{bmatrix}, \quad (3.3)$$

and λ_{ref} and φ_{ref} are longitude and latitude corresponding to $P_{\text{ECEF}_{\text{REF}}}$. A method for calculating geodetic coordinates, i.e. λ_{ref} and φ_{ref} , from ECEF coordinates is presented by Vermeille (2004), and is also used later in this thesis.

3.3 Trilateration

Calculating position can be approached in many different ways, but common to them all are the two bullet points listed below:

- Reference points with known coordinates.
- Measurements measuring the relative distance from the reference points.

Trilateration is the act of using distance measurements in order to calculate the 2D- or 3D-coordinates of unknown positions (Murphy and Hereman, 1995). By knowing the exact distance to a single point in space, one can only be sure that the position is on the surface of a sphere of the corresponding radius. Consequently, if exact distances to two different points in space are given, the position is defined by the intersection of these two spheres.

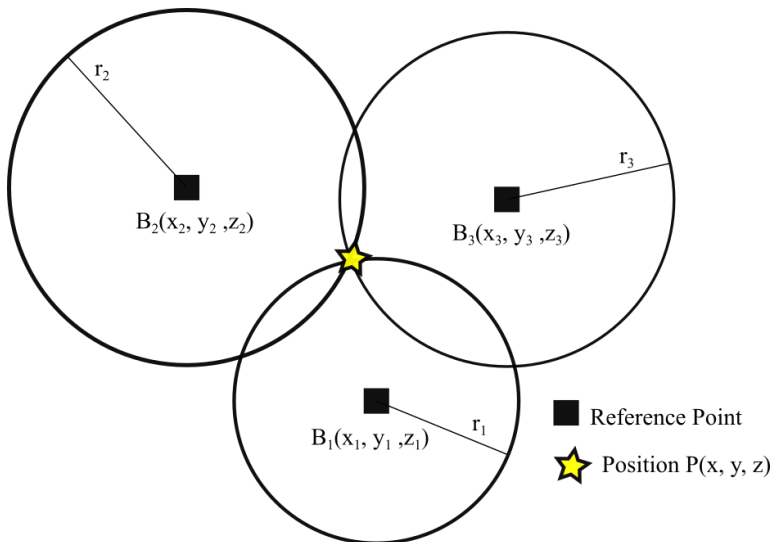


Figure 3.4: Three reference points B_1 , B_2 and B_3 and range measurements used to calculate the unknown position $P(x, y, z)$.

The obvious approach would then be to find the intersection between three or more spheres to locate the unknown position, that is $P(x, y, z)$ in Figure 3.4. The equation for

a sphere is shown in (3.4), note that this is a quadratic equation. Solving for x, y and z with $i = 1, 2, 3, \dots, n$ where n is the number of spheres, gives a nonlinear equation of high degree.

$$(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2 = r_i^2 \quad (3.4)$$

This is not a feasible solution because of the high degree of the resulting equation and because the sign would have to be considered on several occasions.

A more feasible approach is to linearize the system of equations, also explained by Murphy and Hereman (1995), thus finding the intersection of planes instead. In order to linearize the system, an additional constraint is added to (3.4) as a linearization tool. This is the j^{th} constraint and can be seen in 3.5.

$$(x - x_j + x_j - x_i)^2 + (y - y_j + y_j - y_i)^2 + (z - z_j + z_j - z_i)^2 = r_i^2, \quad (3.5)$$

where $(i = 1, 2, \dots, j - 1, j + 1, \dots, n)$. Some rearranging of the terms leads to

$$(x - x_j)(x_i - x_j) + (y - y_j)(y_i - y_j) + (z - z_j)(z_i - z_j) = \frac{1}{2}[r_j^2 - r_i^2 + d_{ij}^2] = b_{ij}, \quad (3.6)$$

and

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}, \quad (3.7)$$

where d_{ij} is the distance between reference point B_i and B_j , and is thus a calculated constant. As Murphy and Hereman discussed, it is arbitrarily which constraint is used as a linearization tool. By choosing $j = 1$ (which is the same as choosing the first reference point), we have $i = 2, 3, \dots, n$ and a $n - 1$ system of equations, as seen in (3.8) - (3.10).

$$(x - x_1)(x_2 - x_1) + (y - y_1)(y_2 - y_1) + (z - z_1)(z_2 - z_1) = \frac{1}{2}[r_1^2 - r_2^2 + d_{21}^2] = b_{21} \quad (3.8)$$

$$(x - x_1)(x_3 - x_1) + (y - y_1)(y_3 - y_1) + (z - z_1)(z_3 - z_1) = \frac{1}{2}[r_1^2 - r_3^2 + d_{31}^2] = b_{31} \quad (3.9)$$

⋮

$$(x - x_1)(x_n - x_1) + (y - y_1)(y_n - y_1) + (z - z_1)(z_n - z_1) = \frac{1}{2}[r_1^2 - r_n^2 + d_{n1}^2] = b_{n1} \quad (3.10)$$

This linearized system can now be written in matrix form:

$$A\vec{x} = \vec{b}. \quad (3.11)$$

This system has $n - 1$ equations compared to the number of reference points, as opposed to the fully non-linear approach, thus the smallest feasible number of reference points is *four* for this method of 3D-positioning. Note that in Figure 3.4 there are only three reference points, and it would therefore not be feasible to calculate the 3D-position, $P(x, y, z)$, when a linearized system of equations is used. The solution would then be to add an additional reference point, B_4 , to the system, for then to proceed as discussed above.

3.4 Linear Least Squares Trilateration

Because no distance measurement can be completely perfect, the radii of the spheres are not intersecting in one specific point. This situation is more likely to occur in a real-world system and is best illustrated in 2D-space with three reference points, as seen in Figure 3.5. As earlier, the system is linearized before solving for x and y . The same principle yields that three reference points are needed for solving the $n - 1$ system of equations.

Because the spheres are not intersecting at the same point, the position of $P_1(x, y)$ has to be estimated. This is equivalent to say (3.11) does not have a solution in a real-world

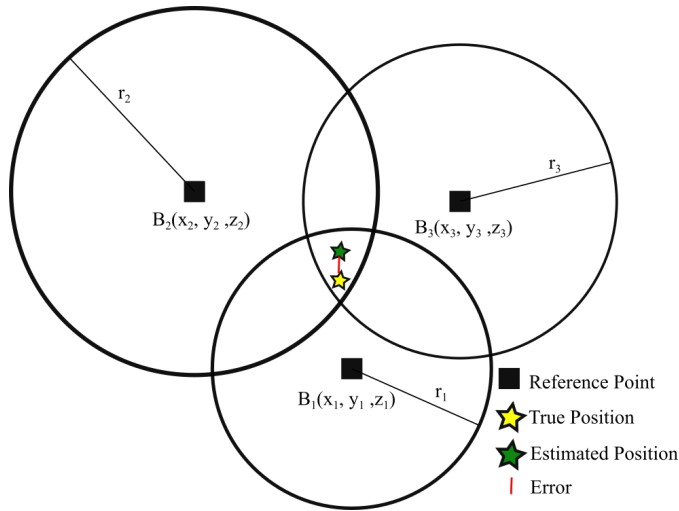


Figure 3.5: The position of the green star is estimated based on the the distances r_1 , r_2 and r_3 and the reference points B_1 , B_2 and B_3 . The red line is the error norm between the estimated position and the true position (yellow star).

system (because r_i is only approximate), and it is necessary to estimate \vec{x} in order to find the best approximate position. In other words, the task is to find the best \vec{x} which makes $A\vec{x} - \vec{b}$ in equation (3.11) as close to zero as possible. This can be done by minimizing the sum of squares

$$\vec{r}^T \vec{r} = (A\vec{x} - \vec{b})^T (A\vec{x} - \vec{b}), \quad (3.12)$$

which leads to the normal equation (Noble, 1988):

$$A^T A \vec{x} = A^T \vec{b}. \quad (3.13)$$

It is now easy to rearrange (3.13) in order to find the best fit \vec{x} based on the approximate measurements supplied to the linearized system, also referred to as the pseudo inverse:

$$\vec{x} = (A^T A)^{-1} A^T \vec{b}. \quad (3.14)$$

The pseudo inverse, $(A^T A)^{-1} A^T$, calculates the closest linear fit compared to using the inverse matrix directly. The method concerning minimizing the sum of squares of a linearized system by taking advantage of the pseudo-inverse is called Linear Least Squares (LLS).

3.5 Non Linear Least Squares Trilateration

NLLS is known for more accurate pose estimates compared to LLS, as long as approximate distances are used (Murphy and Hereman, 1995). This is by nature an iterative approach, and appropriate software must be used for the method to converge towards a solution (Befus, 2014). As for LLS, the familiar equation

$$(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2 = r_i^2, \quad (3.15)$$

is used for describing the intersection of spheres. However, instead of linearizing and taking advantage of the pseudo inverse as done in (3.14), the system of equations is formulated as an optimization problem. Recall that r_i is the approximate distance, and by defining \hat{r}_i as the exact distance, the problem can be formulated

$$F = \sum_{i=1}^n (\hat{r}_i - r_i)^2 = \sum_{i=1}^n f_i(x, y, z)^2, \quad (3.16)$$

where

$$f_i(x, y, z) = \hat{r}_i - r_i = \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2} - r_i, \quad (3.17)$$

and $\hat{r}_i - r_i$ is the measurement error. The optimization method would then iterate until the minimum value of the cost function was found.

To illustrate this, imagine a scenario with four reference points. If the number of reference points is four, then F is the sum of four squared errors. The partial derivatives of F then become

$$\frac{\delta F}{\delta x} = 2 \sum_{i=1}^4 f_i \frac{\delta f_i}{\delta x} \quad (3.18)$$

$$\frac{\delta F}{\delta y} = 2 \sum_{i=1}^4 f_i \frac{\delta f_i}{\delta y} \quad (3.19)$$

$$\frac{\delta F}{\delta z} = 2 \sum_{i=1}^4 f_i \frac{\delta f_i}{\delta z}, \quad (3.20)$$

which can then be rewritten to $\vec{g} = 2J^T \vec{f}$ where

$$J = \begin{bmatrix} \frac{\delta f_1}{\delta x} & \frac{\delta f_1}{\delta y} & \frac{\delta f_1}{\delta z} \\ \frac{\delta f_2}{\delta x} & \frac{\delta f_2}{\delta y} & \frac{\delta f_2}{\delta z} \\ \frac{\delta f_3}{\delta x} & \frac{\delta f_3}{\delta y} & \frac{\delta f_3}{\delta z} \\ \frac{\delta f_4}{\delta x} & \frac{\delta f_4}{\delta y} & \frac{\delta f_4}{\delta z} \end{bmatrix}, \quad \vec{f} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix}, \quad \vec{g} = \begin{bmatrix} \frac{\delta F}{\delta x} \\ \frac{\delta F}{\delta y} \\ \frac{\delta F}{\delta z} \end{bmatrix}.$$

By introducing $\vec{R} = [x, y, z]^T$, the Newton iterate gives (Murphy and Hereman, 1995):

$$R_{k+1}^{\vec{}} = R_k^{\vec{}} - (J_k^T J_k)^{-1} J_k^T \vec{f}_k. \quad (3.21)$$

$R_{k+1}^{\vec{}}$ would then be the new solution to (3.21) based on the old solution $R_k^{\vec{}}$ and the Jacobian. The algorithm would normally iterate a finite number of times, or until the error was sufficiently small.

The iterative algorithm recommended by Matlab for solving NLLS problems is the *Trust-Region-Reflective* (TRR) algorithm. This is a trust region optimization algorithm, i.e. "small" trust regions are investigated for each iteration. The goal is to move from one point, let us say (x, y, z) , within the trust region to another point with a lower function value of $F(x, y, z)$ (Nocedal and J. Wright, 2006). This is a built-in function in Matlab, for further documentation see Matlab (accessed June 18, 2018).

3.6 An Algebraic Approach To the Trilateration Problem

Norrdine (2012) presented an efficient method for solving the trilateration problem with and without Over Determination (OD). This section will explain the developed method for solving Under Determined (UD) systems, and is later referred to as "the Algebraic algorithm".

By treating the nonlinear properties of the system as additional unknowns, and at the same time as constraints, the resulting equation system can be solved by the mean of linear algebra methods.

$$(x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2 = r_1^2 \quad (3.22)$$

$$(x - x_2)^2 + (y - y_2)^2 + (z - z_2)^2 = r_2^2 \quad (3.23)$$

$$(x - x_3)^2 + (y - y_3)^2 + (z - z_3)^2 = r_3^2 \quad (3.24)$$

The solution of the familiar equation set (3.22) - (3.24) is equivalent to finding the solution of (3.25) (see appendix B for proof)

$$\begin{bmatrix} 1 - 2x_1 - 2y_1 - 2z_1 \\ 1 - 2x_2 - 2y_2 - 2z_2 \\ 1 - 2x_3 - 2y_3 - 2z_3 \end{bmatrix} \begin{bmatrix} x^2 + y^2 + z^2 \\ x \\ y \\ z \end{bmatrix} = \begin{bmatrix} r_1^2 - x_1^2 - y_1^2 - z_1^2 \\ r_2^2 - x_2^2 - y_2^2 - z_2^2 \\ r_3^2 - x_3^2 - y_3^2 - z_3^2 \end{bmatrix}, \quad (3.25)$$

and can be presented, as before

$$A\vec{x} = \vec{b}, \quad (3.26)$$

where $\vec{x} = [x_0, x_1, x_2, x_3]^T$ and $\vec{x} \in E$ (note that $x_0 = [x^2 + y^2 + z^2]$). Assuming that the reference points, $B(x_i, y_i, z_i)$ where $i = 1, 2, 3$, are not laying in a straight line, the solution of (3.26) is

$$\vec{x} = \vec{x}_p + t \cdot \vec{x}_h, \quad (3.27)$$

with the real parameter t . Here, x_p is the particular solution and x_h is the homogeneous solution ($A\vec{x} = 0$) of (3.27). Now, the vectors \vec{x}_p and \vec{x}_h can be calculated using simple Gaussian elimination.

The next step is to determine the parameter t . To explain this, equation (3.27) is first written explicit

$$x_0 = x_{p0} + t \cdot x_{h0} \quad (3.28)$$

$$x_1 = x_{p1} + t \cdot x_{h1} \quad (3.29)$$

$$x_2 = x_{p2} + t \cdot x_{h2} \quad (3.30)$$

$$x_3 = x_{p3} + t \cdot x_{h3} \quad (3.31)$$

and by using that $x_0 = [x_1^2 + x_2^2 + x_3^2]$, the expression

$$x_{p0} + t \cdot x_{h0} = (x_{p1} + t \cdot x_{h1})^2 + (x_{p2} + t \cdot x_{h2})^2 + (x_{p3} + t \cdot x_{h3})^2, \quad (3.32)$$

can be formulated. This can be rearranged to a more familiar expression of a second order equation

$$t^2 \cdot (x_{h1}^2 + x_{h2}^2 + x_{h3}^2) + t \cdot (2 \cdot x_{p1}x_{h1} + 2 \cdot x_{p2}x_{h2} + 2 \cdot x_{p3}x_{h3} - x_{h0}) + x_{p1}^2 + x_{p2}^2 + x_{p3}^2 - x_{p0} = 0, \quad (3.33)$$

that is $at^2 + bt + c = 0$. The solutions, t_1 and t_2 , are then calculated from the *abc - formula*

$$t_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}. \quad (3.34)$$

The solution of (3.26) is then:

$$\vec{x}_1 = \vec{x}_p + t_1 \cdot \vec{x}_h \quad (3.35)$$

$$\vec{x}_2 = \vec{x}_p + t_2 \cdot \vec{x}_h. \quad (3.36)$$

If the trilateration problem can not be solved due to too short range measurements, the real part of t_1 and t_2 can be used as an approximation.

Sensor System

This chapter describes the sensors which have been used during the project work. The chapter ends with an explanation of how the Pozyx UWB system can be implemented to run in real time in ROS.

4.1 Description of the Pozyx System

Pozyx Labs is a small company founded in 2015 in Ghent, Belgium, which has specialized in indoor position systems utilizing UWB radio frequencies. Pozyx delivers their own devices ("shields") compatible with Arduino, containing both Python and Arduino libraries for tracking and 3D positioning. Each device can be configured to be either "tag" or "anchor", which means that they are either mobile targets for tracking or stationary reference points, respectively. For devices configured to tag mode, an Arduino board can be used to supply processing power, as seen in Figure 4.1. The device uses the standard Universal Serial Bus (USB) 2.0 A to B cable for powering and communication. On the other hand, devices configured to anchor mode do not need an Arduino to operate, the only requirement is a power source. Because every device comes with an onboard 3.3 V regulator one can choose between either battery or USB, the latter is shown in Figure 4.2.

Every Pozyx device contains a Micro Processor Unit (MCU) for communication between the onboard sensors, ranging, pose estimation and calibration algorithms, as shown in Figure 4.3. The UWB chip next to the MCU requires a dedicated clock source and provides the wireless information exchange capability of the device. The centroid of the onboard UWB antenna is used as reference point to the device's local coordinate-system, which must be considered when several devices are used for trilateration purposes. Pozyx uses two-way ranging as default, which is a variant of TOA. The tag is sending out messages to the anchors, which measure the distances and returns the information to the tag. Pozyx does also support TDOA. Here, the tag is sending out a one-way message, whereas the anchors (which must be time synchronized) measure the time differences of received signals.

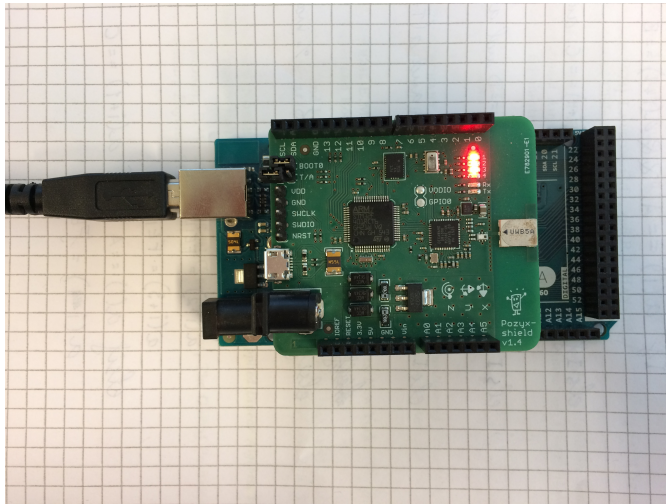


Figure 4.1: A Pozyx device (also referred to as "shield") configured to function as a tag installed on top of an Arduino Mega 2560, see Arduino (accessed May 12, 2018) for documentation.



Figure 4.2: A Pozyx device configured to anchor mode.

The 0x1 firmware version of the Pozyx devices handles multiple anchors in ranging mode and can measure range up to approximately 75 m. The update rate is very high, close to 140 Hz, which is adequate for location systems relying on frequently updated pose estimates. Pozyx uses the frequency band between 3.5-6.5 GHz.

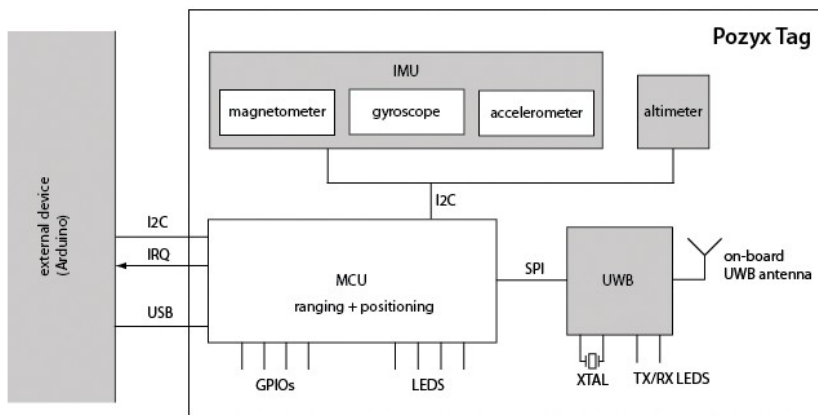


Figure 4.3: System description of the Pozyx tag. The onboard UWB antenna is defined as the local origin of the device (Pozyx, accessed March 5, 2018).

4.2 Real Time Clock

A DS3231 AT24C32 module (Maxim Integrated, accessed May 10, 2018) also called a "Real Time Clock" (RTC), was used to add timestamps to collected range data from the Pozyx system. The clock time can be synchronized to local Unix time and the RTC module has a precision of ± 2 ppm from 0°C to $+40^{\circ}\text{C}$.

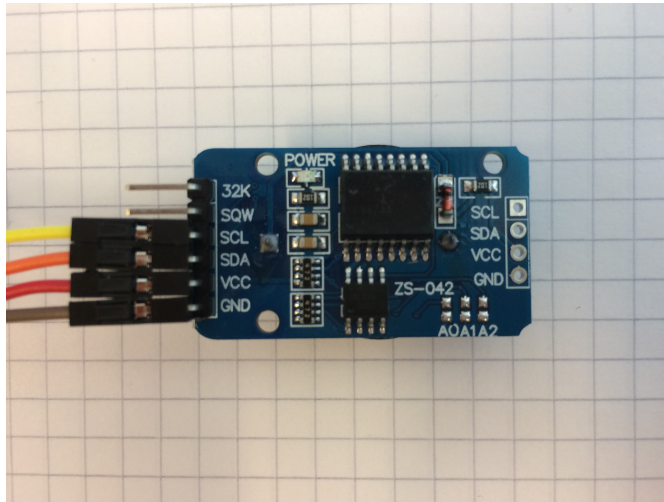


Figure 4.4: The DS3231 AT24C32 module used to add timestamps to data collected from the Pozyx system.

4.3 RTK-GPS

An RTK system was used as ground truth for the outdoor experimental data. The RTK system consisted of an U-Blox NEO-M8T receiver and breakout board and was configured to receive concurrent Global Positioning System (GPS) and GLONASS signals at 2 Hz (U-Blox, accessed May 12, 2018).

A TW4421 Wideband GPS antenna was used to receive GNSS signals, both devices can be seen in Figure 4.5 (Tallysman, accessed May 12, 2018).

Rtklib ver.2.4.3 was used for post-processing of the GNSS data. Base data (also referred to as correction data) was downloaded from Sonel (accessed May 27, 2018), and was used to improve the GNSS measurements.

4.4 Robot Operating System

Robot Operating System (ROS) is an open source software framework for robot applications.

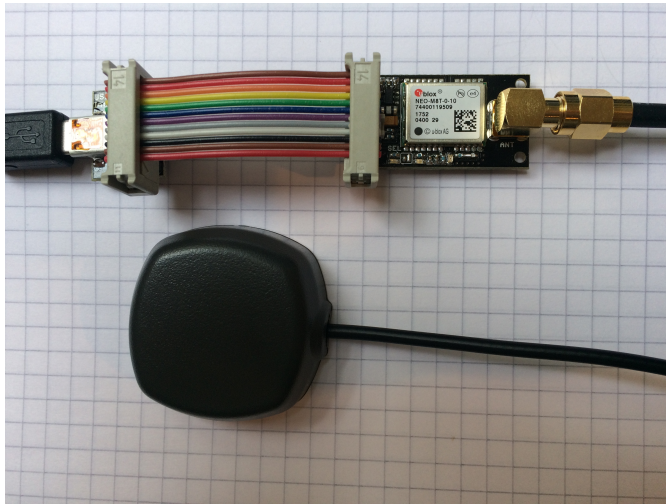


Figure 4.5: The NEO-M8T chip and breakoutboard accompanied with the TW4421 active antenna.

The software comes with libraries, visualization tools, methods for message-passing and can be used to create a very modular system. ROS uses nodes which can publish or subscribe to information (topics) and is a common way to distribute information in a robot's control loop. These nodes are executable files and are supporting several programming languages, and new are constantly being added.

rosserial_arduino is a client library of the *rosserial* package. This library provides the necessary protocols for integrating Arduinos to the ROS environment, illustrated in Figure 4.6.

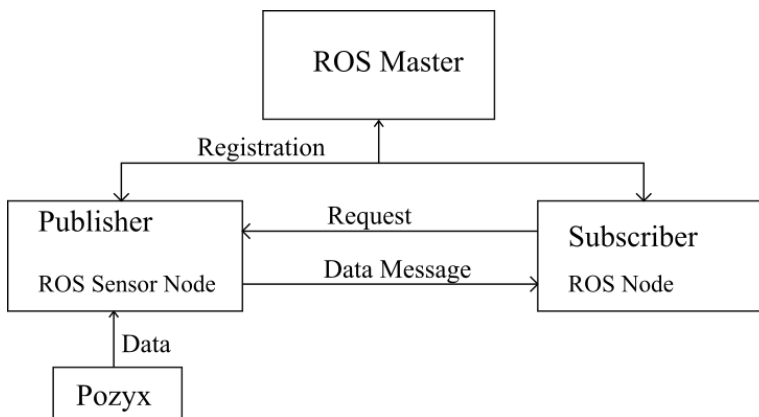


Figure 4.6: Illustration of ROS nodes publishing and subscribing to information.

Implementation and Experiments

Two experiments were conducted to test the accuracy of the Pozyx UWB location system. The first was indoor with small distances between the devices, the second was outdoor with greater distances between the devices. This chapter begins by describing the experimental set-up of both of the experiments. Then, the chapter proceeds to explain how the measured data had to be processed before it could be further utilized for pose estimating purposes. Last in this chapter is a description of how the trilateration algorithms were implemented for post-processing of the collected data.

5.1 The Indoor Experiment

The indoor experiment was conducted in an approximately $6 \times 6 \times 10$ m room at NTNU, Gløshaugen, with four anchors mounted at different heights on tripods. A corner of the room was chosen to be the origin of the local coordinate-system, and a point 407 mm above the floor was chosen as a stationary reference point (where the Pozyx tag was located), as shown in Figure 5.1. Notice that the tag was located close to the diagonals defined by the anchors. These diagonals would then be the new perimeters when one of the anchors was removed. A selection of tools was used to measure the local coordinate system, and can be seen in Figure 5.2.

The Pozyx tag was programmed to address the anchors in the following order:

- First: 0x600C
- Second: 0x6057
- Third: 0x6036
- Forth: 0x603C

The tag would then restart the process when the last anchor had been addressed.

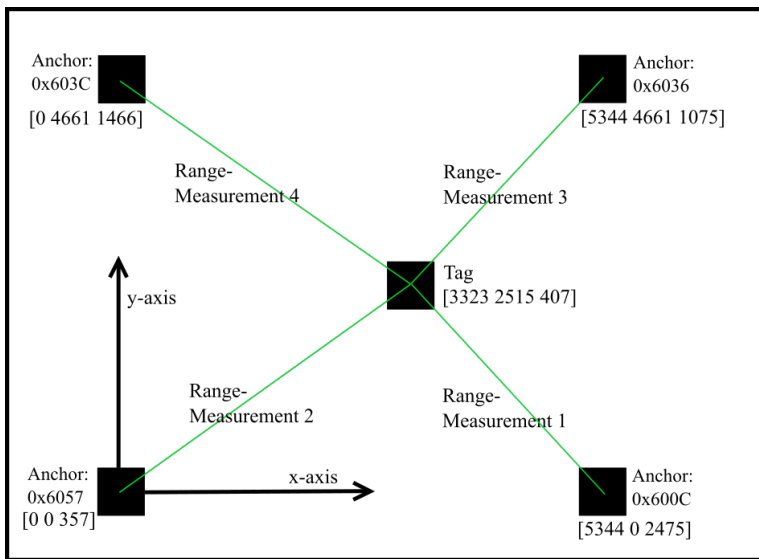


Figure 5.1: A blueprint of the experimental set-up seen from above. The four anchors were mounted on tripods with heights ranging from 357 mm to 2475 mm.



Figure 5.2: Tools used to measure the local coordinate system.

The indoor experiment resulted in 2343 range measurements with corresponding network addresses from the anchors. The system was connected to a laptop through USB, and data was recorded for post-processing using the UAV-code.

5.2 The Outdoor Experiment

The outdoor experiment was conducted on the roof at one of the campus buildings. The experimental set-up was scaled up compared to the indoor experiment, with distances up to approximately 19 m between the anchors. The following two sections will describe how the locations of the anchors were measured, and how the experiment was conducted.

Measuring the Local Coordinate System:

The presented RTK-GPS was used to measure the locations of the anchors in ECEF coordinates, generating at least 1200 measurements for each of the anchors. The RTK-GPS was favored for this task because the tools used for the indoor experiment were not applicable for the distances in the outdoor experiment.

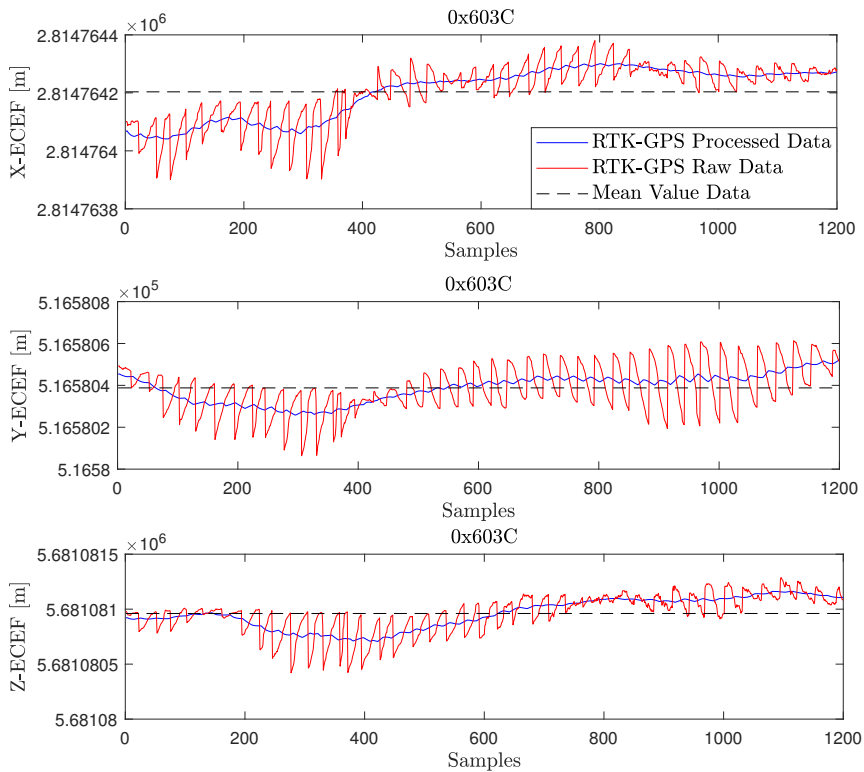


Figure 5.3: Raw data from the RTK-GPS smoothed with the *Movmean* Matlab Function and the calculated mean value. The presented data is from measuring the position of anchor 0x603C. The raw data was varying with approximately 20 cm (on the vertical axis).

The measured anchor positions were then smoothed with the *movemean* Matlab function with a sliding window of $n = 100$ samples. This is a function that calculates the average for each sample based on the n nearest samples. The mean position values were then calculated for each of the anchors, as shown in Figure 5.3. The resulting ECEF coordinates of the anchors can be seen in table 5.1.

Table 5.1: Anchor positions in ECEF coordinates.

Anchor	X-ECEF [m]	Y-ECEF [m]	Z-ECEF [m]
0x603C (origin)	2814764,20352080	516580,387936750	5681080,96062066
0x6057	2814760,49063547	516588,105008901	5681084,21497048
0x600C	2814750,40146171	516572,568802287	5681087,56603793
0x6036	2814748,63305697	516579,630645302	5681091,83957546

Then, one of the anchors was chosen to be the origin of the local NED frame, and the remaining anchors were translated to this coordinate system using a modified version of the ECEF to NED - code. The resulting anchor coordinates in the NED frame can be seen in table 5.2. Notice that the measured anchors' positions were still inaccurate in the z-direction.

Table 5.2: Anchor positions in NED coordinates.

Anchor	X-NED [m]	Y-NED [m]	Z-NED [m]
0x603C (origin)	0	0	0
0x6057	3,47634980336571	8,26051908310568	-1,89956030529141
0x600C	16,3583735957830	-5,19927310136043	0,799004757658777
0x6036	18,6861815741631	2,06578105808305	-2,81489939227559

It was therefore necessary to manually measure the height of each of the anchors, for then to adjust the height values of the NED coordinates. Also, because the RTK-GPS and the anchors did not share the same antenna, it was not possible to measure the exact location of the anchors. Thus, it was necessary to compensate for this possible error source as well. This was done by measuring the anchors' positions with the RTK-GPS antenna 5 cm north of the anchors' antennas. Then the anchor coordinates could be adjusted as shown below:

$$p_{Anchor}^n = p_{rtk}^n + R_b^n(0)r_b^b, \quad (5.1)$$

where p_{rtk}^n is the RTK-GPS position (translated to the NED frame), $r_b^b = [-5\text{cm}, 0, 0]^T$ is the distance between the antennas and

$$R_b^n(0) = \begin{bmatrix} \cos 0 & -\sin 0 & 0 \\ \sin 0 & \cos 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (5.2)$$

is the rotation matrix around the vertical axis. The resulting anchor coordinates in the NED frame are shown in table 5.3.

Table 5.3: Adjusted anchor positions in NED coordinates.

Anchor	X-NED [m]	Y-NED [m]	Z-NED [m]
0x603C (origin)	0	0	0
0x6057	3,42634980336571	8,26051908310568	-0,394
0x600C	16,3083735957830	-5,19927310136043	0,603
0x6036	18,6361815741631	2,06578105808305	-1,198

Conducting the Experiment:

The same RTK-GPS system was used as ground truth to validate the outdoor experimental data with a moving Pozyx tag. The two systems were mounted together on a mobile sensor platform with the receiving GNSS and UWB antennas 10,5 cm apart from each other, as seen in Figure 5.4. The presented RTC module was installed on the Pozyx tag to add timestamps to the collected data. This way, the Pozyx data (stamped with Unix time) could be compared to the RTK-GPS data stamped with GPS time.

The experiment was conducted by walking with the sensor platform at approximately 2 m height connected to a laptop through USB. Data was recorded at locations where it was assumed to be poor geometry for trilateration, in order to investigate how the geometry of the anchors versus the tag would influence the precision of the trilateration algorithms (Murphy and Hereman, 1995). The collected data was stored for post-processing using a modified version of the UAV-code.



Figure 5.4: The sensor platform used to carry the Pozyx system next to the RTK-GPS. The antennas were 10,5 cm apart from each other.

5.3 Working With the Collected Data

Several steps had to be completed before the raw data from the Pozyx system could be evaluated and compared to true earth coordinates, this was true for both the indoor and the outdoor experiments. The following sections will describe how the collected data from the experiments were processed.

5.3.1 Processing of the Collected Indoor Data

The indoor experiment resulted in 2343 measurements from the Pozyx system. Because there were four anchors, four range measurements should be used to produce one pose estimate, as illustrated in Figure 5.5 (the algorithms for pose estimating are described in later sections). Thus, the total number of measurements should result in a whole number when divided by four. Because $\frac{2343}{4} = 585.75$ it was clear that some samples were missing in the indoor data set. To solve this problem, it was decided to discard ranging cycles

Pose estimate 1				Pose estimate 2				Pose estimate 3			
0x600c	0x6057	0x6036	0x603c	0x600c	0x6057	0x6036	0x603c	0x600c	0x6057	0x6036	0x603c
Range-Measurement 1	Range-Measurement 2	Range-Measurement 3	Range-Measurement 4	Range-Measurement 1	Range-Measurement 2	Range-Measurement 3	Range-Measurement 4	Range-Measurement 1	Range-Measurement 2	Range-Measurement 3	Range-Measurement 4

} Pozyx System

Figure 5.5: The first row is for anchor network addresses, the second row is for corresponding range measurements.

where measurements were missing. The range measurements and the network addresses were, therefore, stored in two vectors of size 2343×1 . Then algorithm 1 iterated through the vector containing network address and removed the failed ranging cycles. Every time a ranging cycle was deleted from the vector, the corresponding indexes were deleted from the vector containing range measurements. Then, the algorithm iterated through the resulting vectors and saved the range information and network addresses as illustrated in Figure 5.6. Notice that each column of the matrix contained range data from one specific anchor. This matrix was used as the input to the trilateration algorithms in later sections, and is referred to as "the Pozyx matrix".

The matrix in Figure 5.6 was also manipulated to investigate how the precision of the trilateration algorithms was affected when losing signal coverage from the anchors. This was done by deleting one of the columns containing range information from the Pozyx system at a time. These manipulated matrices were used as input to the trilateration algorithms estimating position based on information from three anchors.

Algorithm 1: Delete bad ranging cycles and create Pozyx matrix containing sorted range measurements.

Input : Network address vector ($n \times 1$) and range vector ($n \times 1$)

Output: Pozyx matrix containing sorted range measurements

```
1 for  $i=1:n$  do
2   if order is correct then
3     | do nothing
4   else
5     | delete indexes from the network address vector;
6     | delete indexes from the range vector;
7   end
8 end
9  $m =$  length of network address vector;
10 for  $i=1:m$  do
11 | save range vector to the Pozyx matrix ( $\frac{m}{4} \times 4$ )
12 end
```

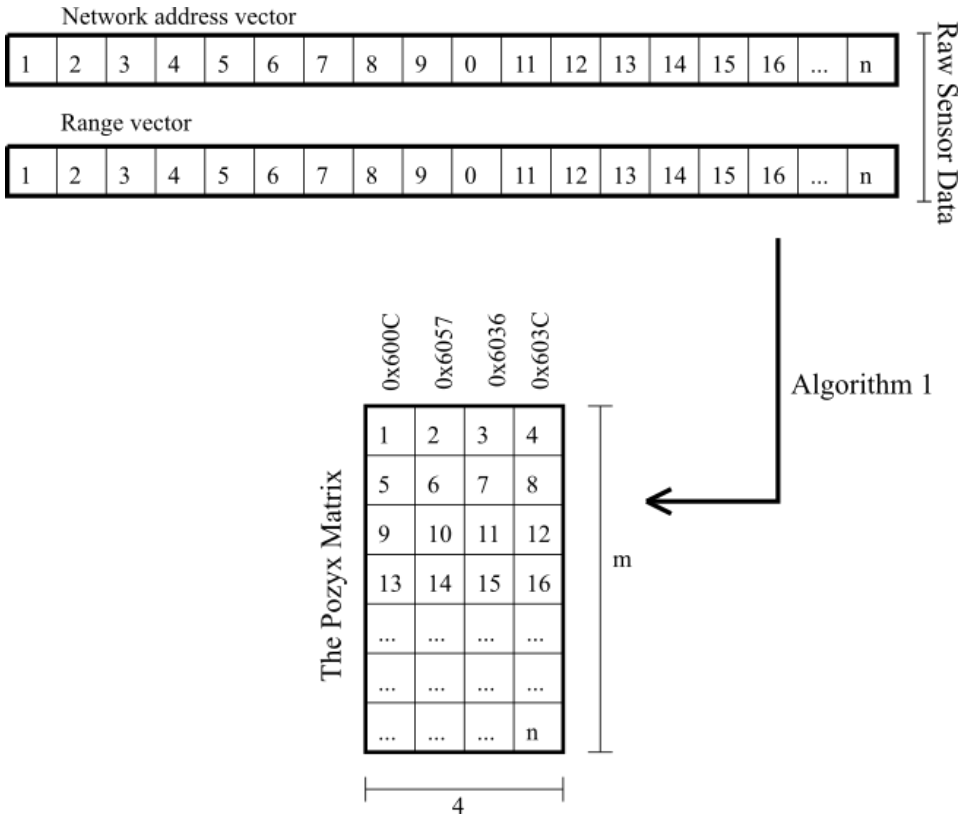


Figure 5.6: An illustration of how the raw data was stored in the Pozyx matrix of dimension $m \times 4$. Each column contains range data from one specific anchor, and each row is corresponding to one correct ranging cycle.

5.3.2 Processing of the Collected Outdoor Data

As for the indoor experiment, the collected range measurements from the Pozyx system had to be sorted before the trilateration algorithms could utilize them. In addition, because the tag was mobile, as opposed to the indoor experiment, the recorded data had also attached Unix timestamps. This is illustrated in Figure 5.7. Because both GNSS data from the RTK system and the position data from the Pozyx system had timestamps, it was possible to compare corresponding position data from the two systems, as illustrated in Figure 5.8. Note that the RTK-GPS was two hours behind UTC+2, this was compensated by comparing corresponding data with two hours difference, but will for now be illustrated as the same times. The next section will explain how data with corresponding timestamps were compared.

Pose estimate 1				Pose estimate 2				Pose estimate 3			
0x600c	0x6057	0x6036	0x603c	0x600c	0x6057	0x6036	0x603c	0x600c	0x6057	0x6036	0x603c
Range-Measurement 1	Range-Measurement 2	Range-Measurement 3	Range-Measurement 4	Range-Measurement 1	Range-Measurement 2	Range-Measurement 3	Range-Measurement 4	Range-Measurement 1	Range-Measurement 2	Range-Measurement 3	Range-Measurement 4
Timestamp 1				Timestamp 2				Timestamp 3			

} Pozyx System
} RTC

Figure 5.7: The first row is for anchor network addresses, the second row is for corresponding measurements, and the third row is for Unix timestamps.

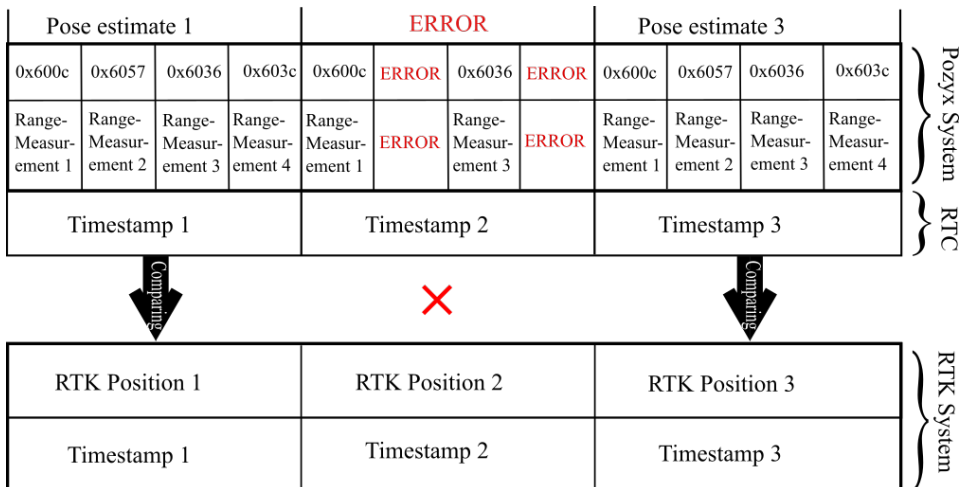


Figure 5.8: The figure is illustrating how data recorded by the Pozyx system were compared to data recorded by the RTK-GPS. Note that only data with corresponding timestamps were compared.

Comparing Pozyx data to RTK-GPS data

The two systems sampled at different frequencies, which lead to different dimensions (number of samples) of the two matrices containing the recorded data, this is illustrated in Figure 5.9. The Pozyx matrix consisted of rows with position data, that is range measurements and corresponding network addresses, and timestamps. The RTK-GPS matrix consisted of rows with position measurements and timestamps. Note that the start time, i.e. the time on the day, and the stop time were the same for both of the matrices.

Pozyx		RTK-GPS	
Position Data 1	12:20:10	Position 1	12:20:10
Position Data 2	12:20:11	Position 2	12:20:10
Position Data 3	12:20:12	Position 3	12:20:11
Position Data 4	12:20:14	Position 4	12:20:12
Position Data 5	12:20:15	Position 5	12:20:14
Position Data 6	12:20:16	Position 6	12:20:14
Position Data 7	12:20:18	Position 7	12:20:14
Position Data 8	12:20:19	Position 8	12:20:15
Position Data 9	12:20:20	Position 9	12:20:15
Position Data 10	12:20:21	Position 10	12:20:15
Position Data 11	12:20:23	Position 11	12:20:16
Position Data 12	12:20:24	Position 12	12:20:16
		Position 13	12:20:16
		Position 14	12:20:18
		Position 15	12:20:19
		Position 16	12:20:22
		Position 17	12:20:23
		Position 18	12:20:24
		Position 19	12:20:24

Figure 5.9: The two matrices containing position and time information. Note that the lengths of the matrices are different, but the start times (green) and stop times (red) are the same.

Another issue was that both of the systems were suffering from slightly irregular sampling frequencies. The Pozyx system was configured to sample at 1 Hz, but occasionally went down to roughly 0.5 Hz. The RTK-GPS was configured to sample at 2 Hz, but could go down to approximately 0.5 Hz as well.

The first part of comparing the data was then to remove double timestamps from the

RTK-GPS data (for instance the two first rows in the RTK-GPS matrix). An algorithm was designed to iterate through the samples and remove the undesired data, shown in algorithm 2.

Algorithm 2: Delete double timestamps from RTK-GPS data.

Input : Raw RTK-GPS data
Output: RTK-GPS data with no double timestamps.

```
1 for Elements in RTK-GPS measurements do
2   | if current second == next second then
3   |   | delete current position data and time;
4   | else
5   |   end
6 end
```

The next job was to correct the dimensions of the two matrices, so that data with corresponding timestamps could be compared. Because the recorded data from both systems had the same start and end time, it was known how many seconds which were between those two times. So an algorithm was designed to iterate through the time vectors and fill in the missing seconds (For instance between row 4 and 5 in the RTK-GPS matrix). Every time a missing second was replaced, the corresponding position (which was *not* estimated or measured) replaced by a NaN value. This was done for both of the matrices containing position and time data. The procedure is roughly outlined in algorithm 3.

After this, the matrices with measured data were of the same lengths, as illustrated in Figure 5.10. However, not all of the rows in the Pozyx matrix consisted of three or four range measurements (thus only one or two), and would therefore not be feasible for estimating position. These rows were therefore also replaced by NaN values.

After the last rows had been replaced by NaN values, the two matrices (Figure 5.10) were compared to each other. For every row there was a NaN value, the corresponding row was deleted in the opposite matrix. After this, the two matrices consisted of position data with corresponding timestamps between the two systems.

Algorithm 3: Replacing missing values in position and time matrix with NaN values.

Input : Position and time matrix with missing samples.

Output: Position and time matrix with missing samples replaced by NaN.

```
1 for Length of position and time matrix do
2   if current second - next second == -1 or 59 then
3     | do nothing;
4   else
5     end
6   if current second - next second is less than -1 then
7     | find out how many seconds that are missing;
8     | replace the number of missing seconds;
9     | replace corresponding positions with NaN;
10  else
11  | find out how many seconds that are missing % new minute;
12  | replace the number of missing seconds;
13  | replace corresponding positions with NaN;
14  end
15 end
```

Pozyx			RTK-GPS	
Position Data 1	12:20:10		Position 2	12:20:10
Position Data 2	12:20:11		Position 3	12:20:11
Position Data 3	12:20:12		Position 4	12:20:12
NaN	NaN		NaN	NaN
Position Data 4	12:20:14		Position 7	12:20:14
Position Data 5	12:20:15		Position 10	12:20:15
Position Data 6	12:20:16		Position 13	12:20:16
NaN	NaN		NaN	NaN
Position Data 7	12:20:18		Position 14	12:20:18
Position Data 8	12:20:19		Position 15	12:20:19
Position Data 9	12:20:20		NaN	NaN
Position Data 10	12:20:21		NaN	NaN
NaN	NaN		Position 16	12:20:22
Position Data 11	12:20:23		Position 17	12:20:23
Position Data 12	12:20:24		Position 19	12:20:24

Figure 5.10: The two matrices containing position and time information. Note that the lengths of the matrices were the same and the start times (green) and stop times (red) were corresponding. Only measurements with corresponding (transparent green) timestamps were compared.

5.4 Implementation of the Trilateration Algorithms

Three different algorithms were tested with the experimental data. Both a linearization approach, an entirely non-linear approach and one variant of an algebraic approach were used.

5.4.1 Linear Least Squares Pose Estimation

The presented LLS algorithm was based on the work of Murphy and Hereman (1995) and was applicable when information from at least four anchors was available. The local coordinate system was assumed to be known and stationary, and the user had to manually provide this information to the algorithm before initiation. The Algorithms described in Section 5.3 were used to process the raw data, making it available for the LLS algorithm.

For the indoor dataset, the LLS algorithm would then iterate through the Pozyx matrix presented in Figure 5.6 from top to bottom, generating one pose estimate for each row in the matrix, this is illustrated in Figure 5.11.

For the outdoor dataset, remember Figure 5.10, the LLS algorithm would iterate through the Pozyx matrix, but would only estimate poses from the rows containing *four* measurements from the Pozyx system. The rows containing *three* range measurements were therefore discarded.

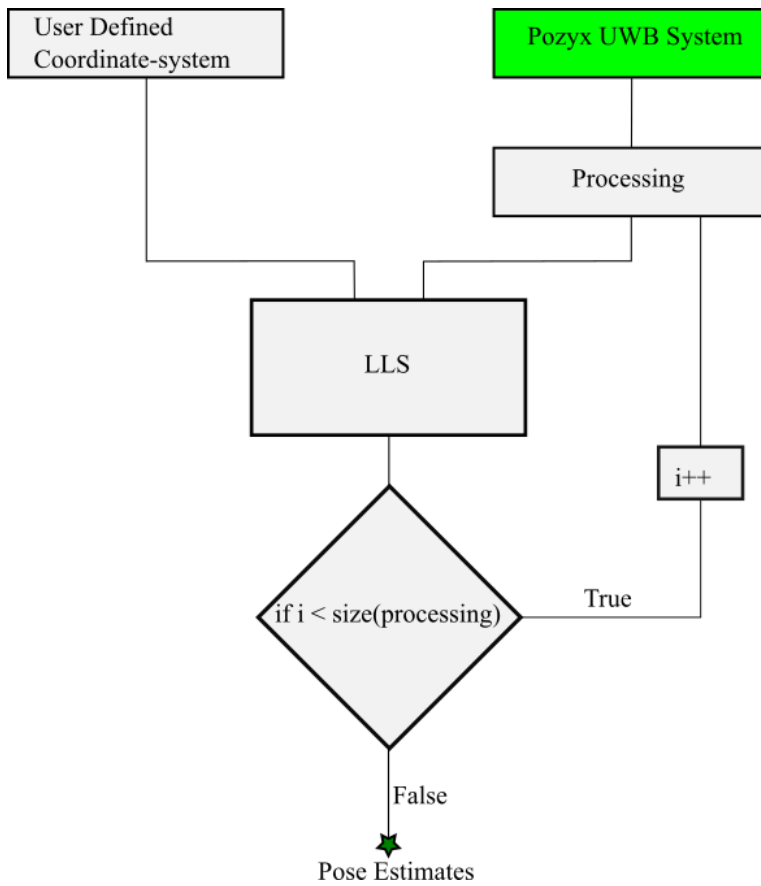


Figure 5.11: The structure of the LLS algorithm. The "Size" of "Processing" is referring to the number of rows in the Pozyx matrix, and "i" was incremented for each pose estimate.

5.4.2 Algebraic Solution to Pose Estimation

The Algebraic algorithm was based on the work of Norrdine (2012) and was used when only three range measurements were available, and thus locating the tag close to the perimeter defined by the remaining anchors (remember the diagonals in Figure 5.1).

For the indoor experiment, this was done by deleting one of the columns of the Pozyx matrix in Figure 5.6 at a time. Thus, *four* scenarios were tested for when deleting and replacing each of the columns containing range information:

- Scenario 1: Remove anchor 0x603C.
- Scenario 2: Remove anchor 0x6036.
- Scenario 3: Remove anchor 0x6057.
- Scenario 4: Remove anchor 0x600C.

The rest of the structure of the algorithm was the same as for the LLS algorithm, presented in Figure 5.11, the only difference was that the LLS block was replaced with the Algebraic algorithm instead.

5.4.3 Non Linear Least Squares Pose Estimation

The TRR algorithm used the LLS or the Algebraic algorithm, depending on the number of available measurements, as a good initial guess for each iteration. The initial guess made the TRR algorithm converge towards a solution using fewer iterations compared to a random initial starting point.

For the indoor dataset, the algorithm iterated through the rows of the Pozyx matrix from top to bottom, as explained before. For each iteration, the LLS algorithm made an initial pose estimate which was used by the TRR algorithm as an initial starting point, as illustrated in Figure 5.12. The TRR algorithm was also used to test how the precision was affected when the columns of the Pozyx matrix were deleted, each at a time, as for the Algebraic algorithm. For this, the TRR algorithm used the Algebraic algorithm as an initial starting point.

For the outdoor data set, remember Figure 5.10, both the Algebraic algorithm and the LLS algorithm were used (however not at the same time) to generate initial starting points. This was because some of the rows containing Pozyx data were not consisting of more than *three* range measurements, while others were consisting of *four* range measurements. Thus, rows with three range measurements used the algebraic algorithm to generate initial starting points, while rows with four range measurements used the LLS algorithm.

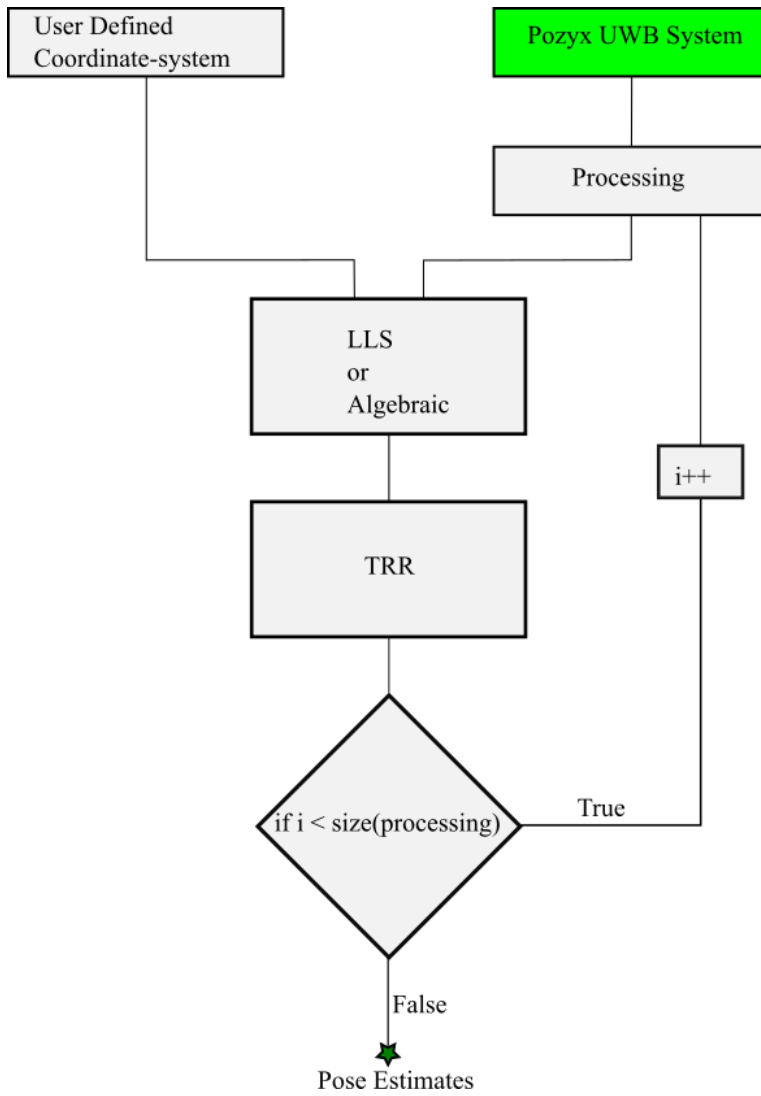


Figure 5.12: The structure of the NLLS algorithm.

Experimental Results

6.1 Results From the Indoor Experiment

The collected data from the indoor experiment was used to test the precision of the LLS algorithm and the TRR algorithm with four anchors signal coverage for the presented experimental set-up.

The dataset was also used to test how the Algebraic algorithm and the TRR algorithm performed when the tag was located close to the perimeters defined by the anchors. This was done by removing one anchor at a time (from the Pozyx matrix), and thus trilaterate with three anchors signal coverage. The scenarios were presented in Section 5.4.2.

All the results were compared and evaluated the following way: The standard deviation of each algorithm was calculated relative to the mean position value. The mean values of the 2D and 3D position errors were also calculated.

Table 6.1: Mean values, standard deviation and calculated error from the LLS algorithm.

Axis:	x [mm]	y [mm]	z [mm]
Mean Position:	3298	2387	818
Standard deviation:	14	14	51

Mean 2D Position Error	131 mm
Mean 3D Position Error	432 mm

6.1.1 Linear Least Squares Solutions

A total of 553 pose estimates could be calculated from the indoor dataset using four reference points and the LLS algorithm, as seen in Figure 6.1. The green triangles are representing the anchors, and the text-arrows are pointing out the specific anchor network address. The blue dots are the poses estimated by the LLS algorithms. A summary of standard deviations and 2D and 3D errors is presented in table 6.1.

Standard Deviation:

The standard deviation was 14 mm in the x-direction, 14 mm in the y-direction, but increased to 432 mm in the z-direction.

2D and 3D Position Error:

The LLS algorithm had the smallest 2D and 3D position errors from the trials with four anchor signal coverage. The resulting error plots are presented together with the TRR algorithm in Section 6.1.2 due to practicality.

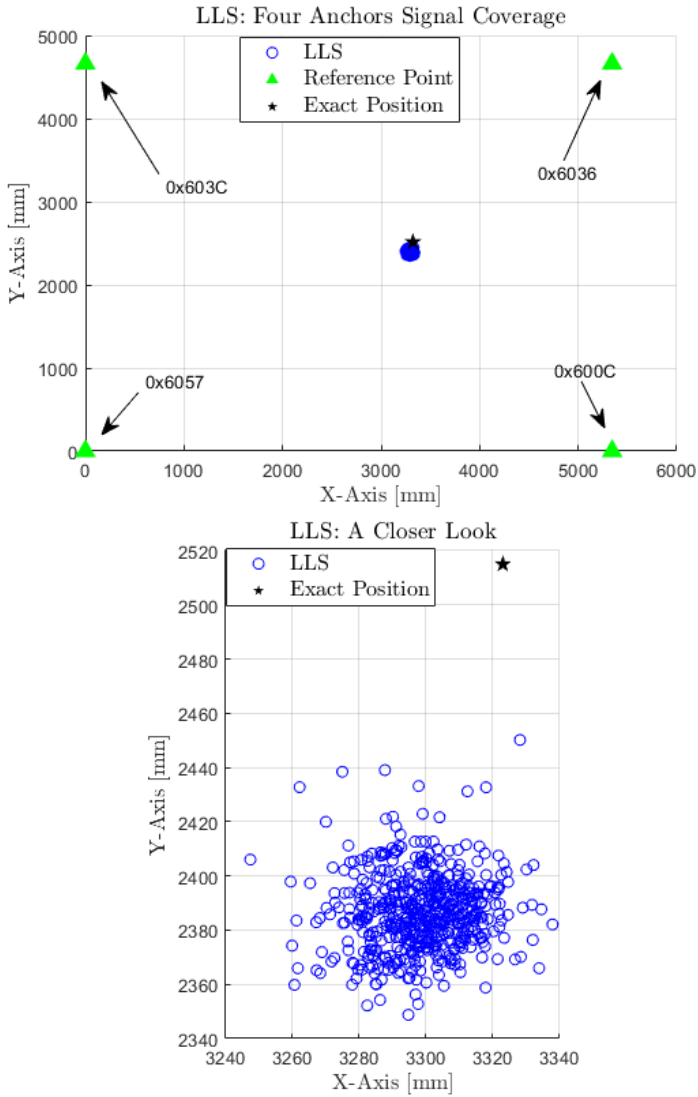


Figure 6.1: The upper plot is presenting all the poses estimated (blue) by the LLS algorithm together with the (green) anchors. The plot beneath is a closer look at the estimated poses from the same algorithm.

6.1.2 Non Linear Least Squares Solutions

The TRR algorithm was tested for both four and three anchors signal coverage. As earlier, this resulted in 553 pose estimates for both of the cases.

Four scenarios were investigated when testing the algorithm with three anchors signal coverage and the tag located close to the perimeter, as explained earlier.

Four Anchors Signal Coverage

The poses estimated by the TRR algorithm can be seen in Figure 6.2. As seen in the figure, the estimated poses were clustered tightly around one point. A summary of standard deviation and 2D and 3D position errors is presented in table 6.2.

Standard Deviation:

The smallest standard deviation was achieved using the TRR algorithm with four anchors signal coverage, only 11 mm in the x-direction, 14 mm in the y-direction and 32 mm in the z-direction. This resulted in pose estimates more clustered around one single point, compared to the other algorithms.

2D and 3D Position Error:

The mean 2D error of the pose estimates was 135 mm but the error increased to 528 mm for 3D precision. The error plots of the LLS algorithm and the TRR algorithm are compared in Figure 6.3. Notice that the 2D errors of the algorithms were quite similar, but the LLS algorithm had a smaller mean 3D error.

Table 6.2: Mean values, standard deviation and calculated error from the TRR algorithm with four anchors signal coverage.

Axis:	x [mm]	y [mm]	z [mm]
Mean Position:	3276 mm	2389	917
Standard deviation:	11	14	32

Mean 2D Position Error	135 mm
Mean 3D Position Error	528 mm

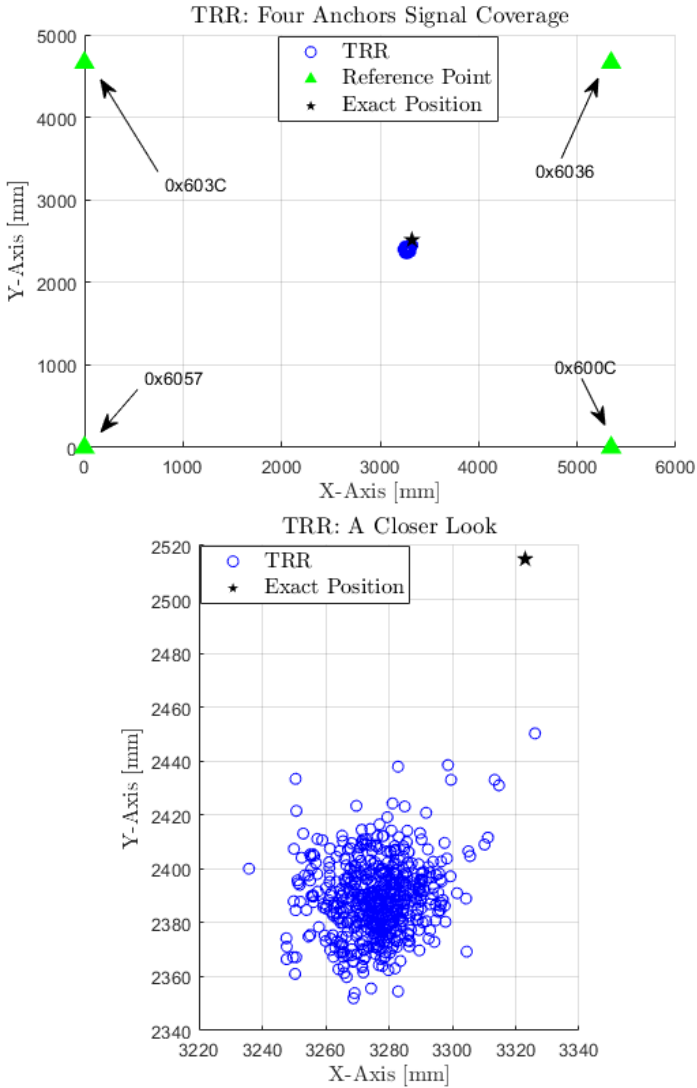


Figure 6.2: The upper plot is presenting all the poses estimated (blue) by the TRR algorithm together with the (green) anchors. The plot beneath is a closer look at the estimated poses from the same algorithm.

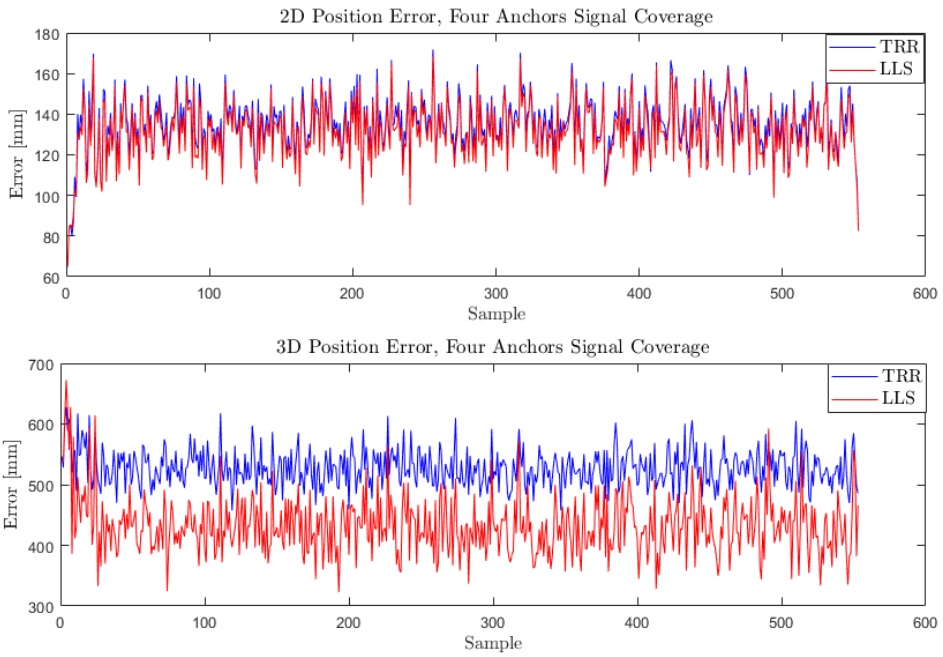


Figure 6.3: The calculated 2D and 3D position errors of the LLS and the TRR algorithms with four anchors signal coverage.

Three Anchors Signal Coverage

The poses estimated by the TRR algorithm with three anchors signal coverage and the tag located close to the perimeter can be seen in Figure 6.4. Note that four scenarios from three anchors signal coverage were investigated, only one is plotted here. A summary of standard deviations and 2D and 3D position errors for each of the scenarios are presented in table 6.3.

Standard Deviation:

The standard deviation increased notably when only three reference points were used. As explained, there were four possible scenarios, and differences in performance were noted. Removing anchor 0x600C had the most negative impact on the standard deviation, note that this was also the anchor that contributed the most to the height difference between the anchors. The remaining scenarios were performing approximately equally.

2D and 3D Position Error:

The smallest 2D position error was achieved when anchor 0x6057 was removed. This resulted in a 2D position error of 91 mm and a 3D position error of 254 mm. The biggest 2D and 3D position errors were measured when anchor 0x603C was removed, that is 254 mm in 2D position error and 728 mm in 3D position error. The resulting error plots are presented in the Section 6.1.3 together with the Algebraic algorithm due to practicality.

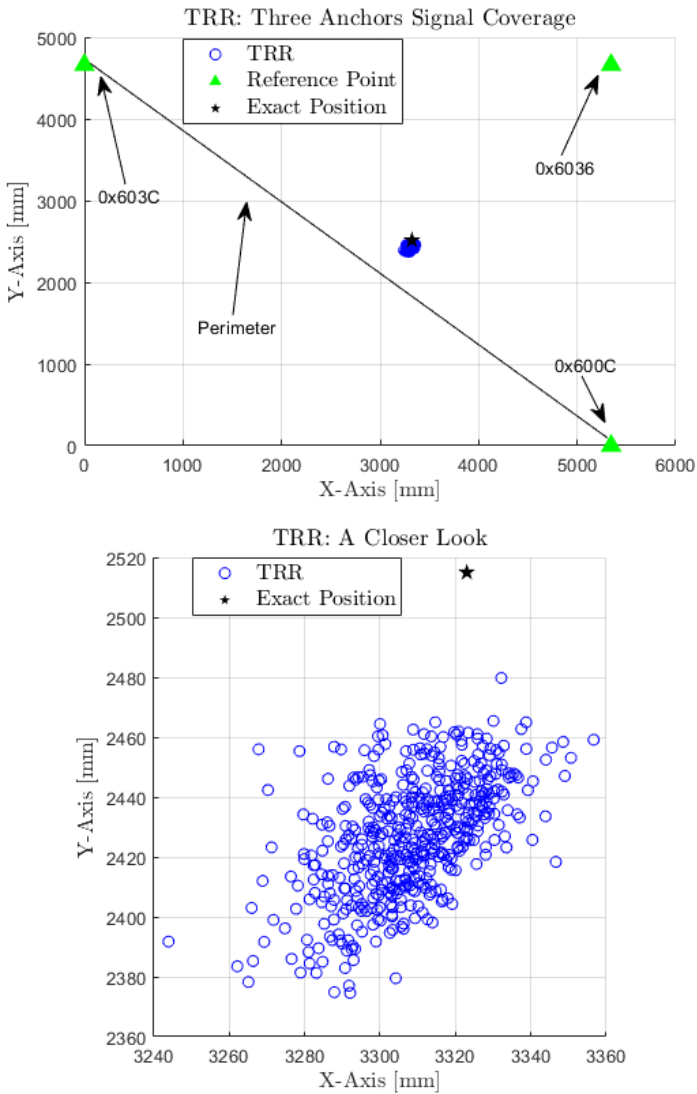


Figure 6.4: The upper plot is presenting all the poses estimated (blue dots) by the TRR algorithm together with the (green) anchors. The plot beneath is a closer look at the estimated poses from the same algorithm.

Table 6.3: A summary of calculated mean positions, standard deviations and 2D and 3D position errors for the TRR algorithm.

		Axis:		
		X [mm]	Y [mm]	Z [mm]
Scenario 1: removed 0x603C	Mean Position:	3259	2417	918
	Standard Deviation:	22	20	27
	Mean 2D Position Error	254 mm		
	Mean 3D Position Error	728 mm		
Scenario 2: removed 0x6036	Mean Position:	3259	2363	918
	Standard Deviation:	14	17	35
	Mean 2D Position Error	166 mm		
	Mean 3D Position Error	538 mm		
Scenario 3: removed 0x6057	Mean Position	3307	2425	948
	Standard Deviation	15	18	36
	Mean 2D Position Error	91 mm		
	Mean 3D Position Error	553 mm		
Scenario 4: removed 0x600C	Mean Position	3289	2419	684
	Standard Deviation	17	27	47
	Mean 2D Position Error	161 mm		
	Mean 3D Position Error	578 mm		

6.1.3 Algebraic Solutions

As for the other algorithms, 553 pose estimates were calculated by the Algebraic algorithm. The resulting pose estimates can be seen in Figure 6.5. Remember that the algorithm was tested for four scenarios, as the TRR algorithm, but only one of the plots is presented here. A summary of the resulting mean values, standard deviations and 2D and 3D precision errors for each of the scenarios can be seen in table 6.4.

Standard Deviation:

The standard deviations for the four different scenarios were similar to the measured standard deviations of the TRR algorithm with three anchor signal coverage.

2D and 3D position Error:

The 2D and 3D precision were also similar to the calculated values of the TRR algorithm. However, the Algebraic algorithm's pose estimates were slightly closer to the exact position for three of the scenarios. The position error of the fourth scenario, when anchor 0x6057 was removed, was on the other hand a little bigger compared to the TRR algorithm.

The position errors of the Algebraic and the TRR algorithms with three anchor signal coverage are compared in Figure 6.6. The four scenarios are plotted in the same figure in order to compare them to each other. Note how the 2D and the 3D position errors were varying accordingly to anchors being removed and replaced.

Table 6.4: A summary of calculated mean positions, standard deviations and 2D and 3D position errors for the Algebraic algorithm.

		Axis:		
		X [mm]	Y [mm]	Z [mm]
Scenario 1: removed 0x603C	Mean Position:	3258	2409	920
	Standard Deviation:	24	21	32
	Mean 2D Position Error	247 mm		
	Mean 3D Position Error	713 mm		
Scenario 2: removed 0x6036	Mean Position:	3262	2359	981
	Standard Deviation:	14	17	35
	Mean 2D Position Error	168 mm		
	Mean 3D Position Error	538 mm		
Scenario 3: removed 0x6057	Mean Position	3301	2419	930
	Standard Deviation	16	19	37
	Mean 2D Position Error	99 mm		
	Mean 3D Position Error	533 mm		
Scenario 4: removed 0x600C	Mean Position	3289	2413	678
	Standard Deviation	17	27	49
	Mean 2D Position Error	181 mm		
	Mean 3D Position Error	627 mm		

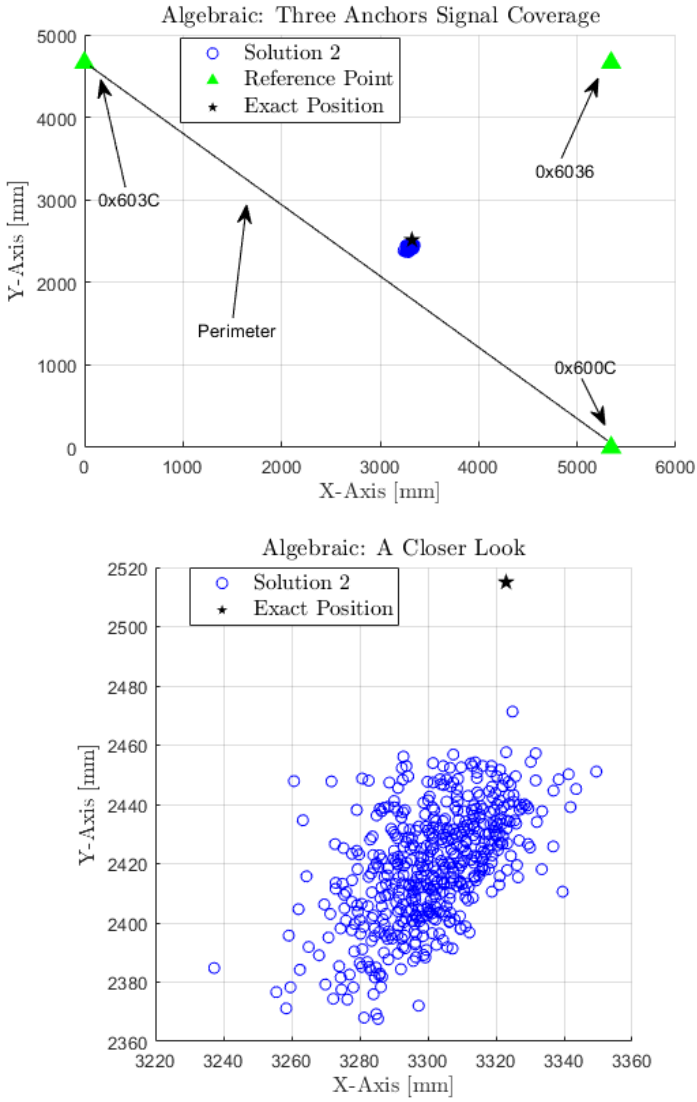


Figure 6.5: The upper plot is presenting all the poses estimated (blue) by the Algebraic algorithm together with the (green) anchors. The plot beneath is a closer look at the estimated poses from the same algorithm.

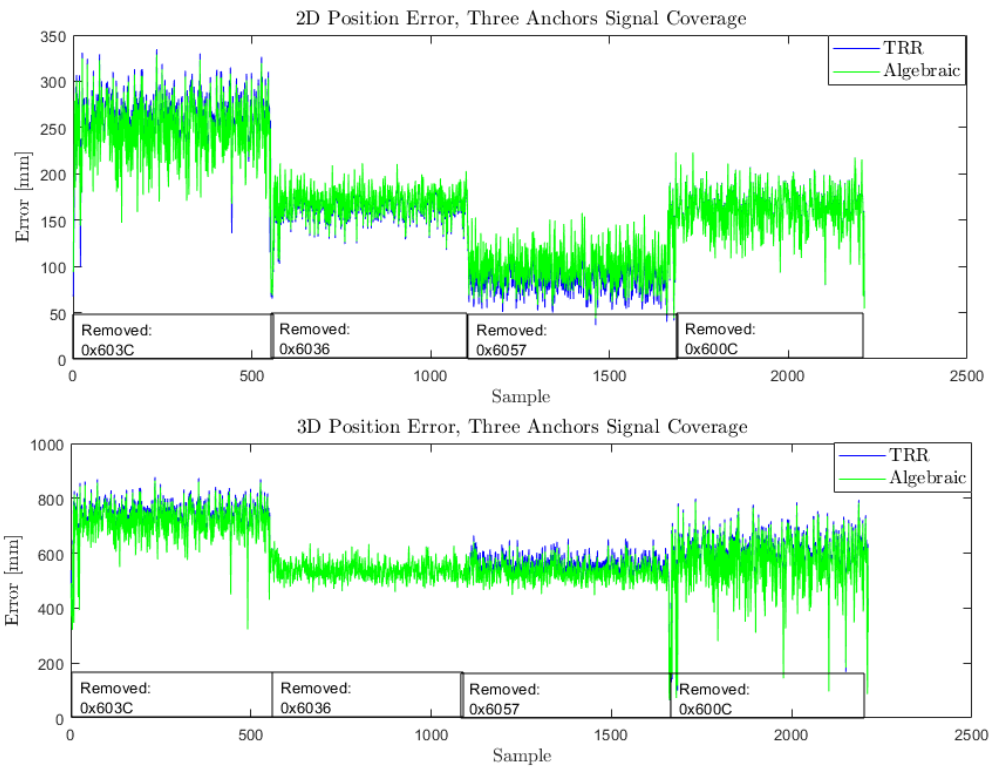


Figure 6.6: The 2D and 3D position errors of the TRR and the Algebraic algorithms with three anchors signal coverage and the tag located close to the perimeters. The four scenarios are plotted together in order to compare them to each other.

6.2 Validation of the Indoor Experimental Results

The experimental results were validated by conducting Monte Carlo simulations replicating the experimental set-up (Raychaudhuri, 2008).

6.2.1 Monte Carlo Simulations

Four Monte Carlo simulations were conducted to validate how the tag's location versus the perimeters was impacting the 2D position error.

Four Anchors Signal Coverage:

The tag was first simulated to be elevated from 0 mm to 2000 mm height, while being locked at $x = 3325$ mm and $y = 2515$ mm. The range measurements were assumed to be equally affected by noise and had a standard deviation of 16,5 mm. This value was assumed based on the collected data set. The simulation is outlined in algorithm 4 and Figure 6.7 shows the simulated set-up.

The resulting error plot can be seen in Figure 6.8. Note that the error was smallest when the tag was located approximately at 1200 mm height.

Algorithm 4: Pseudo-code illustrating the Monte Carlo simulation.

```

1 nMC = 3000 % number of iterations;
2 nEst = 1000 % number of estimates for each iteration;
3 t_array = zeros(nEst,3,nMC);
4 t_position = [nEst × 3] % Exact tag positions from height 0 mm to 2000 mm ;
5 for iMC = 1:nMC do
6   for h=0:2000 do
7      $\vec{x}_1, \vec{x}_2, \vec{x}_3, \vec{x}_4$  % anchor coordinates ;
8      $z_1 = ||t\_position(h, :) - \vec{x}_1|| + w$  % range measurements with noise ;
9      $z_2 = ||t\_position(h, :) - \vec{x}_2|| + w$  ;
10     $z_3 = ||t\_position(h, :) - \vec{x}_3|| + w$  ;
11     $z_4 = ||t\_position(h, :) - \vec{x}_4|| + w$  ;
12    t_estimate = LLS( $\vec{z}$ );
13    t_array(:, :, iMC) = t_estimate
14  end
15 end
16 for iMC = 1:nMC do
17   error_array(:, :, iMC) = t_array(:, :, iMC) - t_position % create error arrays ;
18 end
19 tPlane = error_array(:, 1:2, :) % extract x and y error;
20 errorSq = tPlane(:, 1, :).^2 + tPlane(:, 2, :).^2;
21 errorPlane = sqrt(mean(errorSq, 3)) % this was plotted ;

```

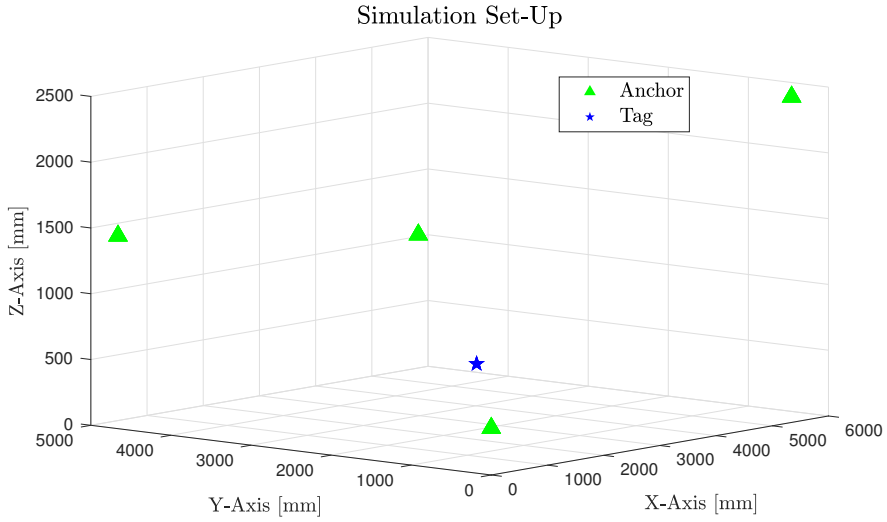


Figure 6.7: The set-up for the first and the second Monte Carlo simulations with four anchors signal coverage.

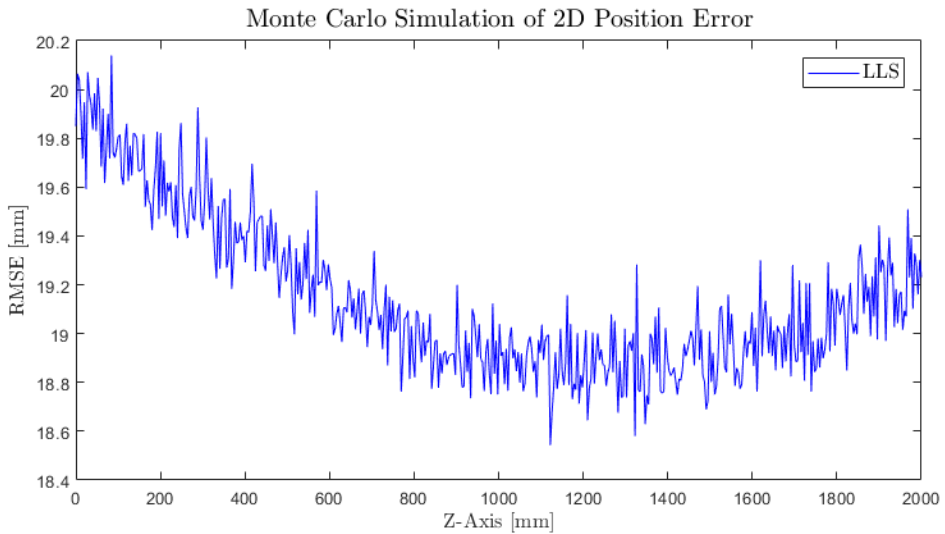


Figure 6.8: A simulation of elevating the tag from 0 mm to 2000 mm while being locked at $x = 3325$ mm and $y = 2515$ mm.

For the second simulation with four anchors signal coverage, the tag was simulated to move from -100 mm to 5000 mm in the y-direction (in the coordinate system), while being locked at $x = 3323$ mm and $z = 407$ mm. The result is shown in Figure 6.9. As it can be seen in the figure, the calculated error was smallest when the tag was located at approximately $y = 3500$ mm.

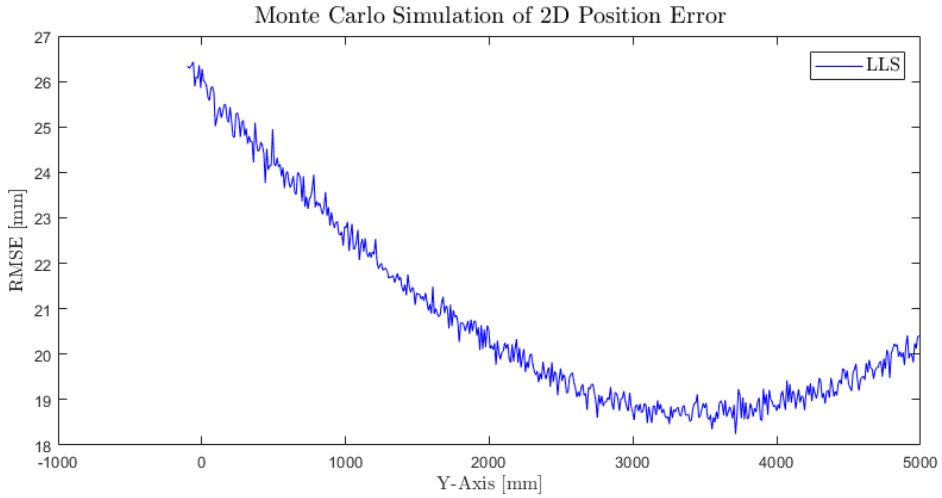


Figure 6.9: A simulation of moving the tag from $y = -100$ mm to $y = 5000$ mm while being locked at $x = 3323$ mm and $z = 407$ mm

Three Anchors Signal Coverage:

The third simulation did not include anchor 0x6036, and the Algebraic algorithm was used to trilaterate with three anchors signal coverage. The tag was simulated to move from 0 mm to 5000 mm in the y-direction, while being locked at $x = 3323$ mm and $z = 407$ mm. The simulated set-up can be seen in Figure 6.10 and the resulting error plot can be seen in Figure 6.11. Notice that the error was smallest when the tag was located at approximately $y = 3000$ mm.

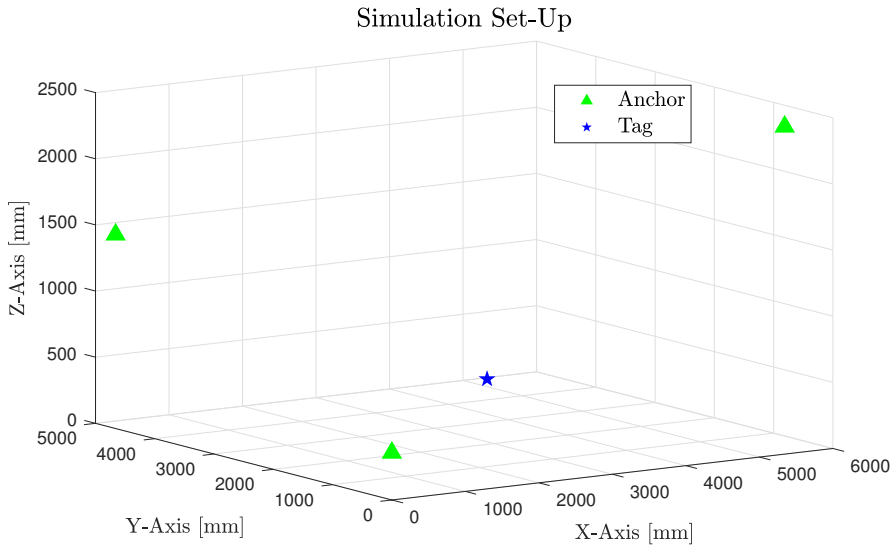


Figure 6.10: The setup for the third Monte Carlo simulation with three anchors signal coverage.

Six Anchors Signal Coverage:

The last simulation added two new anchors to the original experimental set-up. The new anchors locations were arbitrarily chosen to be $[0\text{mm}, 0\text{mm}, 2000\text{mm}]$ and $[4661\text{mm}, 3000\text{mm}, 2000\text{mm}]$. The anchors from the two first simulations were located at their old locations, this is shown in Figure 6.12. The tag was simulated to move from -100 mm to 5000 mm in the y-direction, while being locked at $x = 3323$ mm and $z = 407$ mm, thus the same simulated path as for the second simulation.

The resulting error plot can be seen in Figure 6.13. Note that the calculated error in this simulation was smaller compared to the second simulation (Figure 6.9).

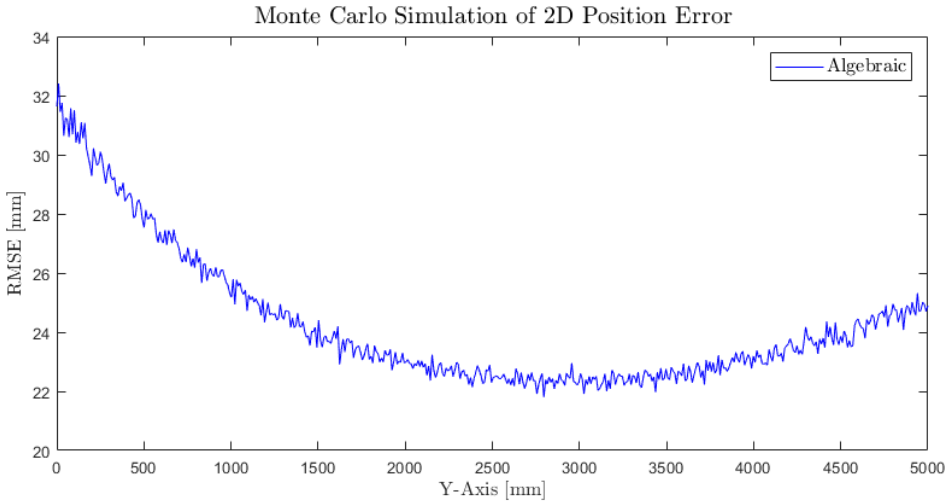


Figure 6.11: A simulation of moving the tag from $y = 0$ mm to $y = 5000$ mm while being locked at $x = 3323$ mm and $z = 407$ mm.

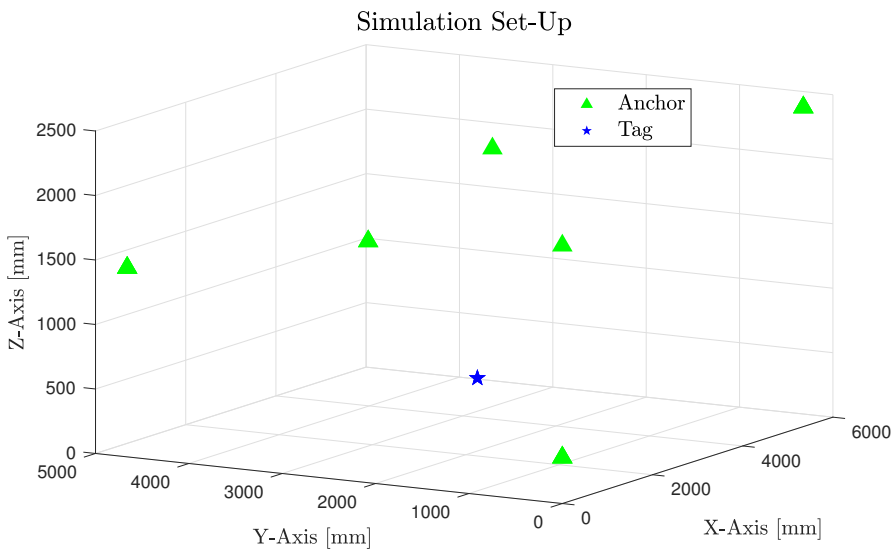


Figure 6.12: The set-up for the fourth Monte Carlo simulation with six anchors signal coverage.

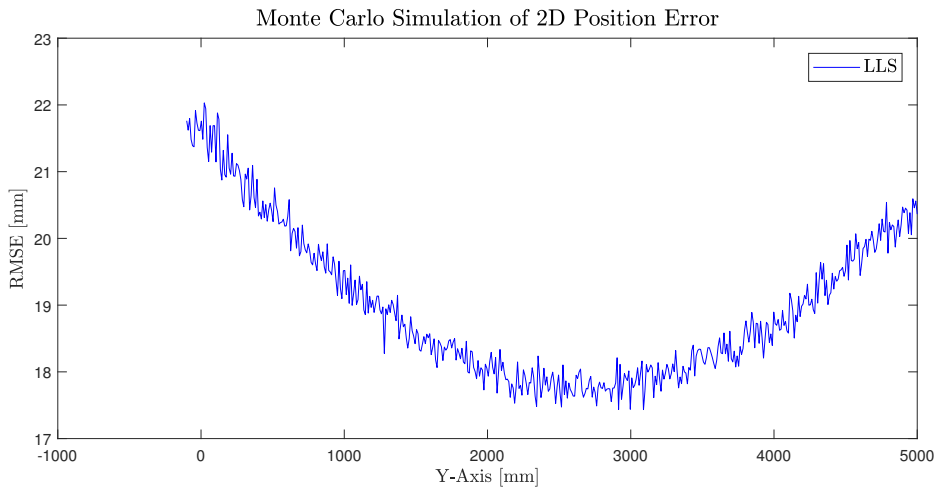


Figure 6.13: A simulation of moving the tag from $y = -100$ mm to $y = 5000$ mm while being locked at $x = 3323$ mm and $z = 407$ mm

6.3 Results From the Outdoor Experiment

The outdoor experiment was divided into three parts, later referred to as "Walk NR. 1", "Walk NR. 2" and "The outdoor range test". Figure 6.14 is illustrating the experimental location for Walk NR. 1. The RTK-GPS was used to validate the position estimates generated by the Pozyx system and the presented algorithms, this will from now on be illustrated by a black dotted line. As for the indoor experiment, the mean position errors of the algorithms were calculated for both 2D and 3D space. The Algebraic algorithm was only used as a starting point for the TRR algorithms for this experiment.



Figure 6.14: An illustration of the 2D path from Walk NR. 1. The path is roughly outlined with red and blue lines on top of a picture from google maps.

6.3.1 Walk NR. 1

Walk NR.1 generated approximately 65 pose estimates from the LLS algorithm. Remember how the LLS algorithm was described to iterate through the Pozyx matrix for the outdoor data set, described in Section 5.4.1. The resulting mean 2D and 3D position errors can be seen in table 6.5 marked with *LLS (OD)*. The calculated errors for the corresponding poses estimated by the TRR algorithm are also presented in the table, marked with *TRR (OD)*. The algorithms were thus compared based on the same data set.

Figure 6.15 is representing the related 2D path on which the position error values were calculated from. The figure shows the TRR and LLS solutions compared to RTK-GPS when the sensor platform was carried counterclockwise in the grid. Note how the dotted line from the RTK-GPS was almost aligned with two of the anchors (green triangles) on the right-hand side in the figure. Also, notice how the LLS and the TRR algorithms were deviating from ground truth during the same interval. The corresponding error plots for the two algorithms are presented in Figure 6.16. As it can be seen in the figure, the TRR algorithm was more accurate compared to the LLS algorithm.

Figure 6.17 is also representing the 2D path of Walk NR. 1. This figure shows the estimated poses of the TRR algorithm when the rows in the Pozyx matrix with *three* range measurements were included, and resulted in approximately 120 pose estimates. The resulting position errors can be seen in table 6.5, marked with *TRR*. The related error plots to the 2D path can be seen in Figure 6.18, and note that the error suddenly increased to approximately 2 m (2D) and 7m (3D) at sample number 60. By studying the related 2D path, this can be explained by poor RTK-GPS performance. As it can be seen, in quadrant 15 m north and -1 m east, the RTK-GPS was most likely affected by reflections from the environment. The error plot is thus showing a discrepancy between the two systems at sample number 60.

Table 6.5: The 2D and 3D position errors from walk NR.1.

Algorithm:	TRR (OD)	LLS (OD)	TRR
Mean 2D Position Error	0,637 m	0,6726 m	0,582 m
Mean 3D Position Error	1,659 m	2,521 m	1,558 m

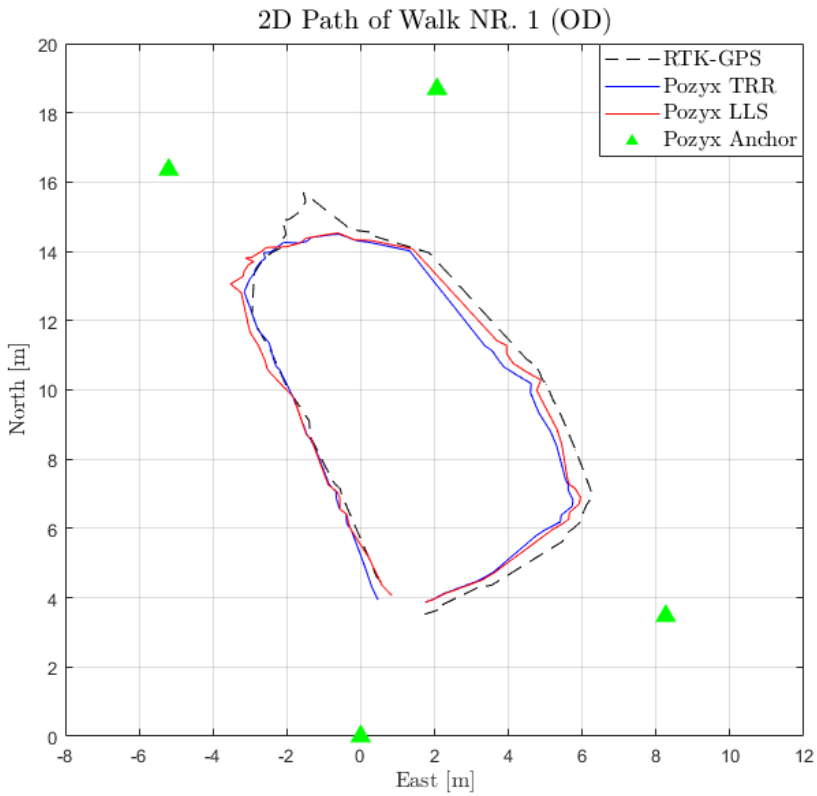


Figure 6.15: The 2D path from Walk NR. 1. Corresponding LLS and the TRR solutions are presented in the figure.

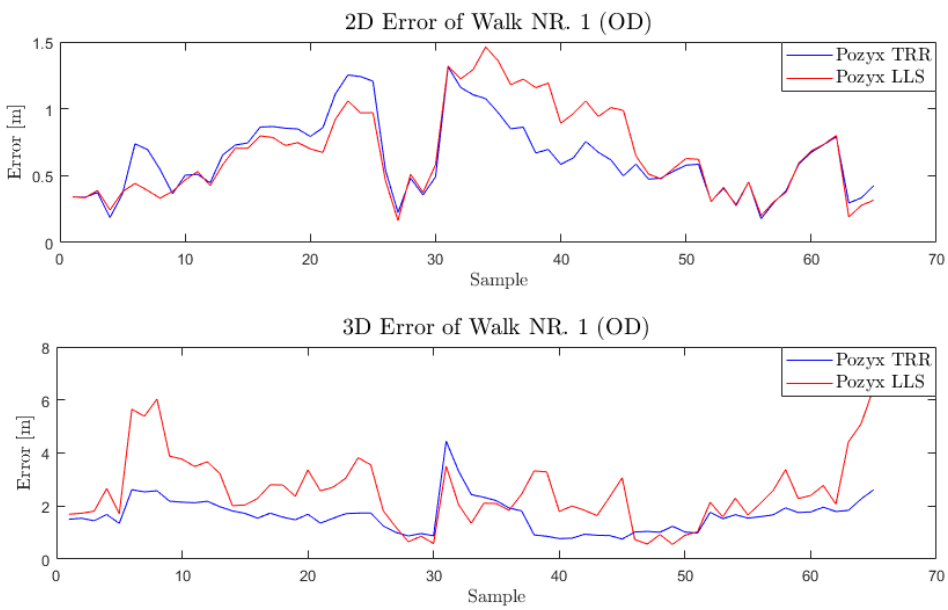


Figure 6.16: The 2D and 3D error plots from Walk NR. 1. Corresponding position errors from the LLS and the TRR algorithms are included in the figure.

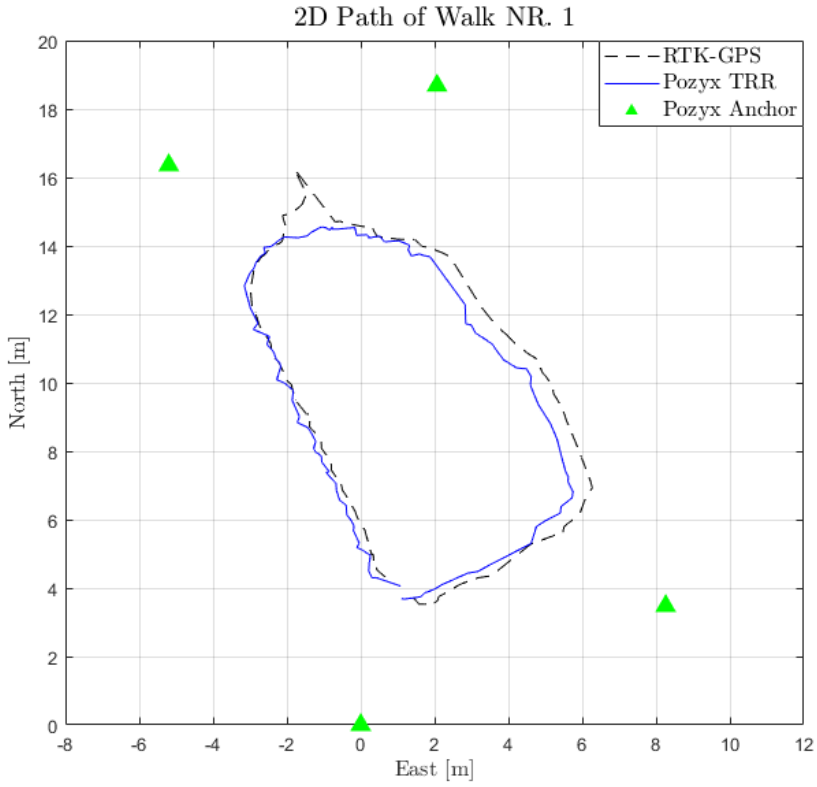


Figure 6.17: The 2D path from Walk NR. 1. All the TRR algorithm solutions are presented in the figure.

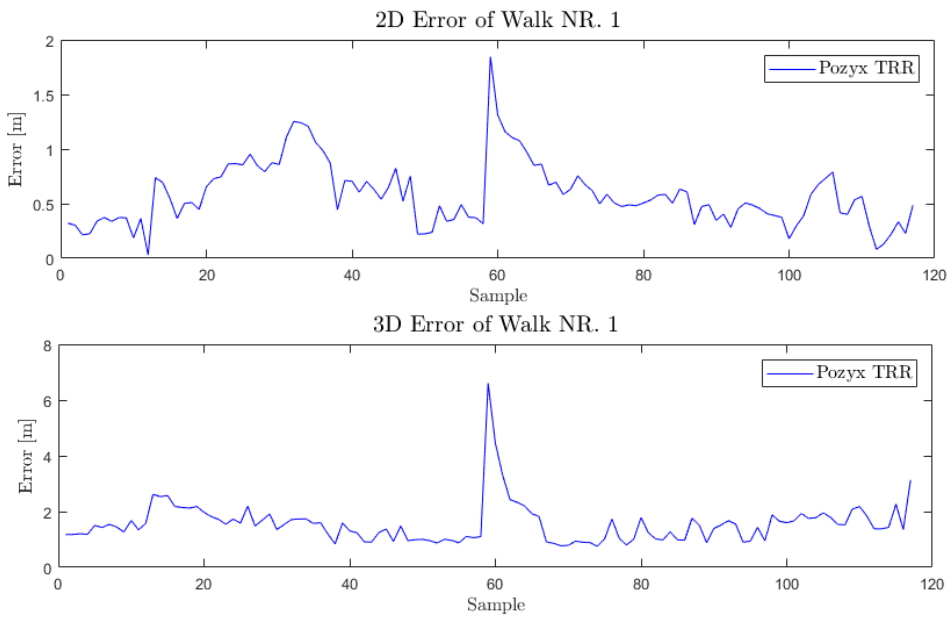


Figure 6.18: The 2D and 3D error plots from Walk NR. 1. All the solutions from the TRR algorithm are included in the figure.

6.3.2 Walk NR. 2

Walk NR.2 generated approximately 110 pose estimates from the LLS algorithm. The resulting mean 2D and 3D position errors can be seen in table 6.6 marked with *LLS (OD)*. The calculated errors for the same data set, only estimated by the TRR algorithm are also presented in the table, marked with *TRR (OD)*.

Figure 6.19 is representing the related 2D path on which the position error values were calculated from. The figure shows the TRR and the LLS solutions compared to RTK-GPS when the sensor platform was carried from the bottom-right corner to the upper-left corner of the grid. Note how the dotted line from the RTK-GPS was almost aligned with two of the anchors (green triangles) on the left-hand side in the figure. Also, notice how the LLS and the TRR algorithms were deviating from ground truth during the same interval. The corresponding error plots for the two algorithms are presented in Figure 6.20. As it can be seen in the plots, the TRR algorithm was also here more accurate compared to the LLS algorithm.

Figure 6.21 is also representing the 2D path of Walk NR. 2. This figure shows the estimated poses of the TRR algorithm when the rows in the Pozyx matrix with *three* range measurements were included, and resulted in approximately 190 pose estimates. The resulting position errors can be seen in table 6.6, marked with *TRR*. The related error plots to the 2D path can be seen in Figure 6.18, and notice the sudden change in precision at approximately sample number 170. By studying the 2D path, this can, as earlier, be explained by poor RTK-GPS performance. To that end, the sensor platform was put to rest on the ground before the walk was ended. This was done at approximately 14 m north and -3 m east in the 2D path. As it can be seen in the figure, the measured RTK-GPS position deviated with roughly 4 m while being located on the ground. Despite this, the poses estimated by the Pozyx system deviated less compared to the RTK-GPS measurements. Thus, the sudden change in the error plots at sample number 170 was also a discrepancy between the two systems.

Table 6.6: The 2D and 3D position errors from walk NR.2.

Algorithm:	TRR (OD)	LLS (OD)	TRR
Mean 2D Position Error	0,809 m	0,811 m	0,896 m
Mean 3D Position Error	1,361 m	2,361 m	1,359 m

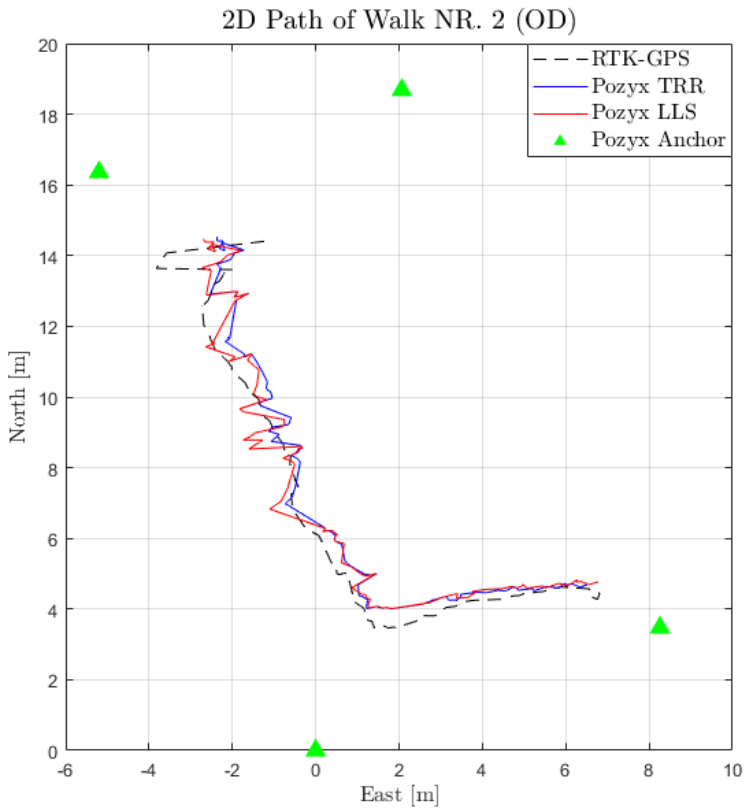


Figure 6.19: The 2D path from Walk NR. 2. Corresponding LLS and TRR solutions are presented in the figure.

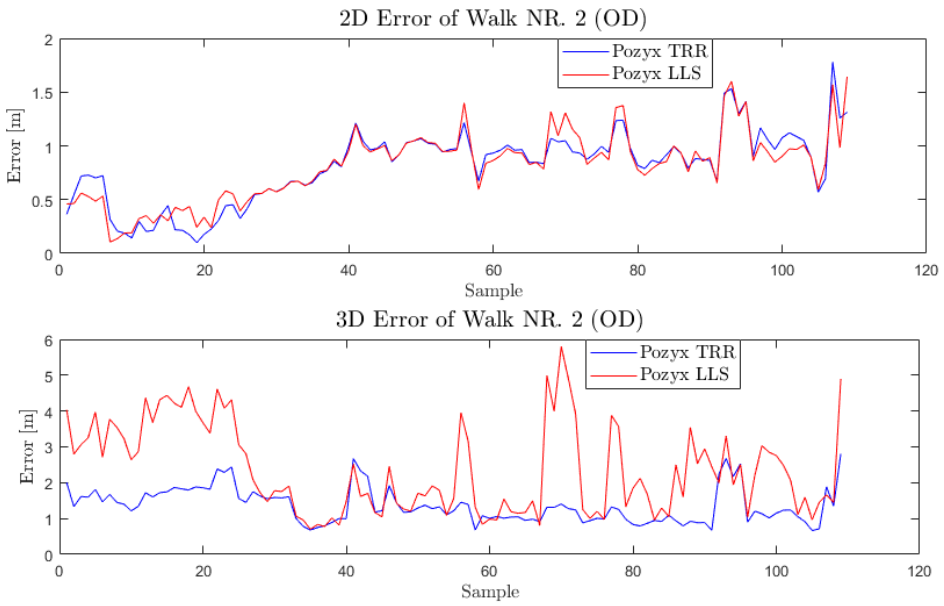


Figure 6.20: The 2D and 3D error plots from Walk NR. 2. Corresponding position errors from the LLS and the TRR algorithms are presented in the figure.

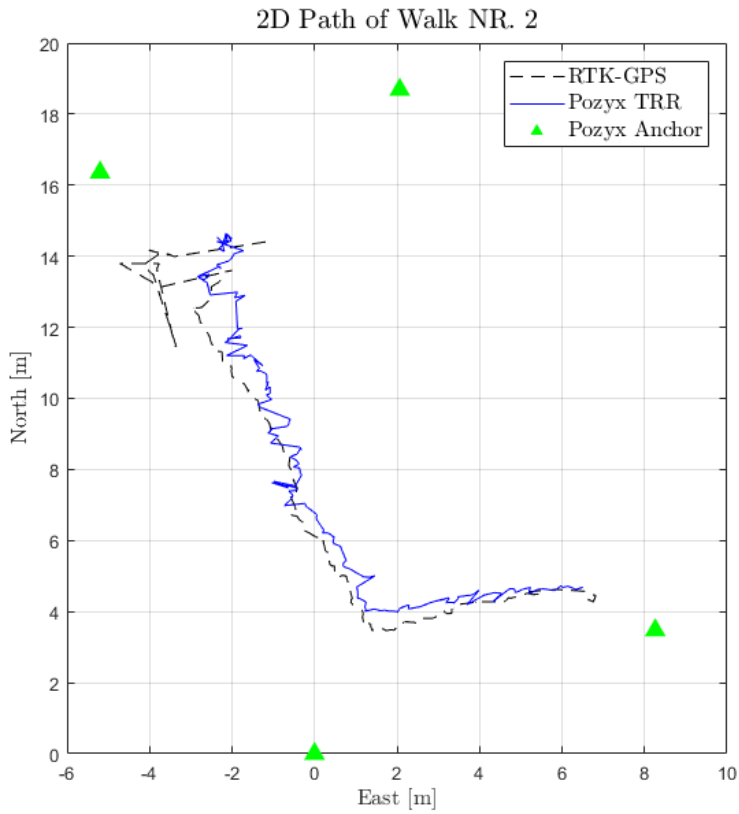


Figure 6.21: The 2D path from Walk NR. 2. All the TRR algorithm solutions are presented in the figure.

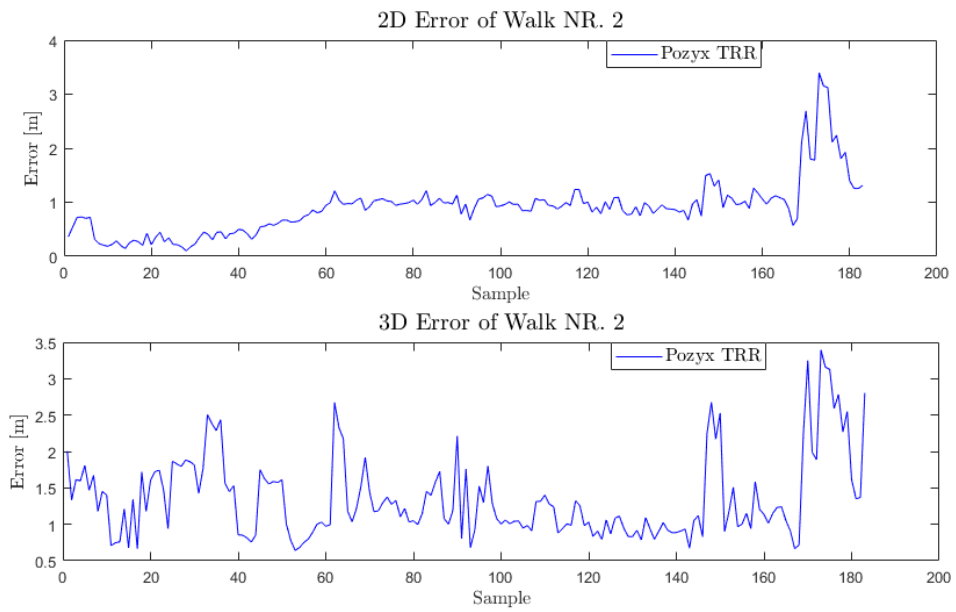


Figure 6.22: The 2D and 3D error plots of Walk NR. 2. All the measured position errors of the TRR algorithm are presented in the figure.

6.3.3 The Outdoor Range Test

The range experiment was conducted to measure the practical working range of the Pozyx location UWB system. The results were compared to the actual distances in the channel between Brattøra and Ravnkloa. Figure 6.23 is illustrating the scenario, note that it is only a suggestion for anchor placement.

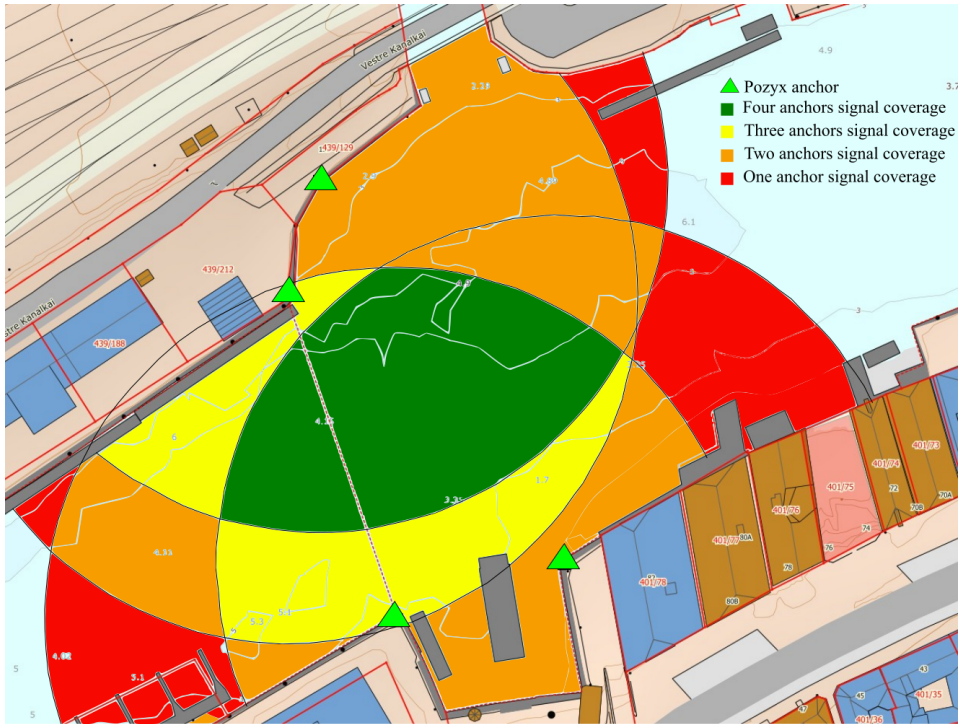


Figure 6.23: An illustration of the range (assumed to be approximately 75 m) of the Pozyx location system in the channel. The green area is where the tag is within four anchors signal coverage. Image from google maps.

Conducting the Experiment:

One anchor was placed on a tripod and the tag was carried from approximately 1 m distance in a straight line away from the anchor with continuously LOS between the devices.

The UWB antenna of the tag was pointing upwards in the vertical direction, as in Figure 4.1 (the flat side pointing upwards). This orientation was chosen to simulate how the tag would receive radio signals equally good from anchors located in 360 °around Milliampere. The range test was conducted at an open outdoor area to reduce the effects from multipath of signals. The measured range can be seen in Figure 6.24, note that the last measurement was recorded at 63.7 m when the RSS value was -101 dB.

One additional experiment was also conducted to test the best possible range of the Pozyx system. This was done in a similar manner as described above, but the tag's antenna

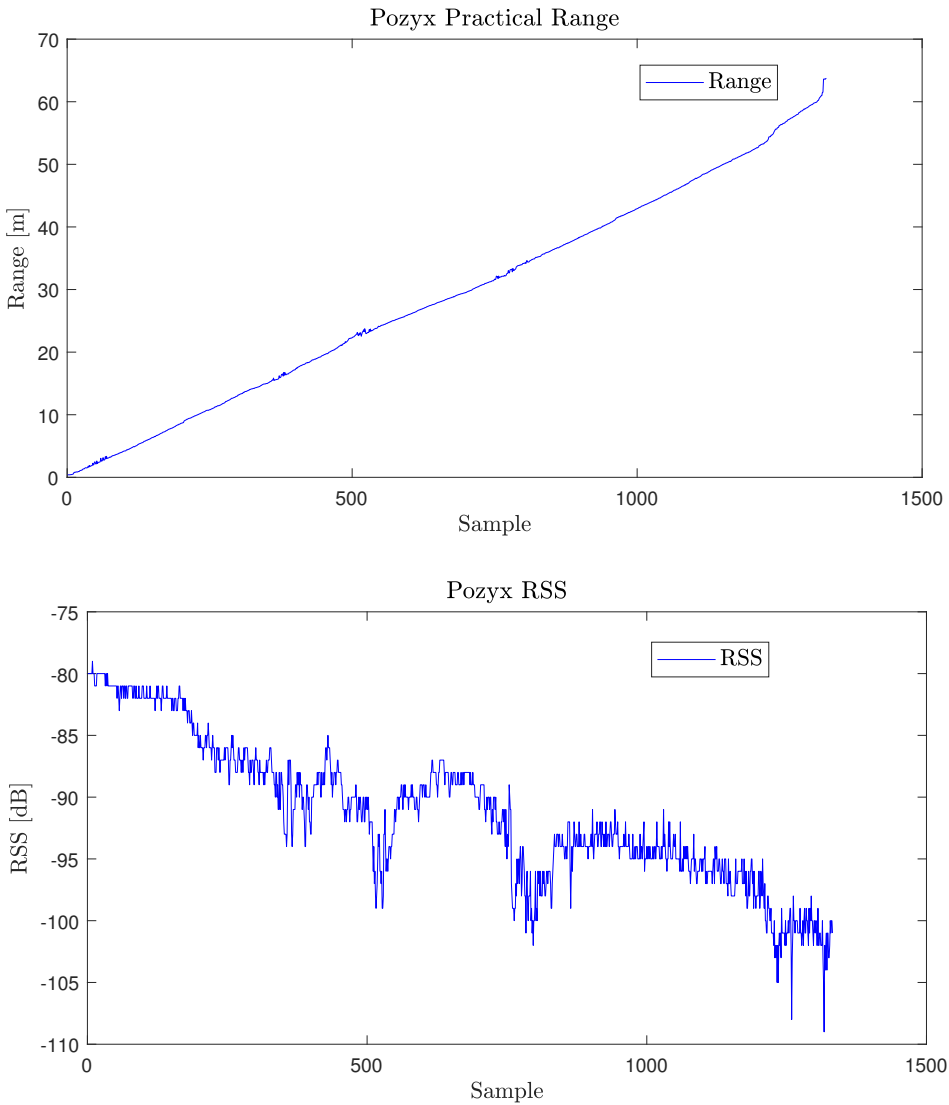


Figure 6.24: The range of the Pozyx UWB location system when the tag was carried in a horizontal orientation.

was (as opposed to earlier) constantly angled towards the anchor’s antenna. The measured range can be seen in Figure 6.25. Note that the last sample was recorded at 111,8 m when the RSS value was equal to -93 dB.

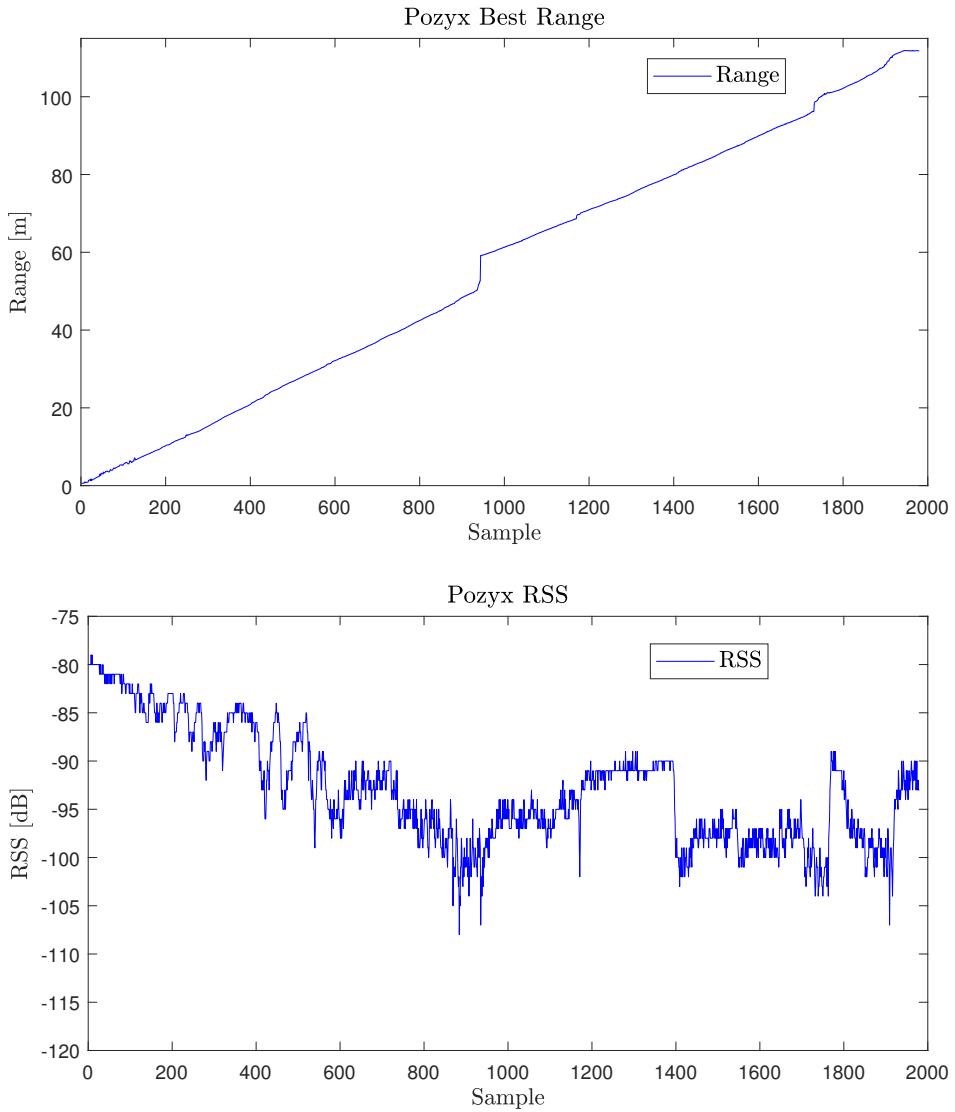


Figure 6.25: The range of the Pozyx UWB location system when the antenna was oriented towards the antenna of the anchor.

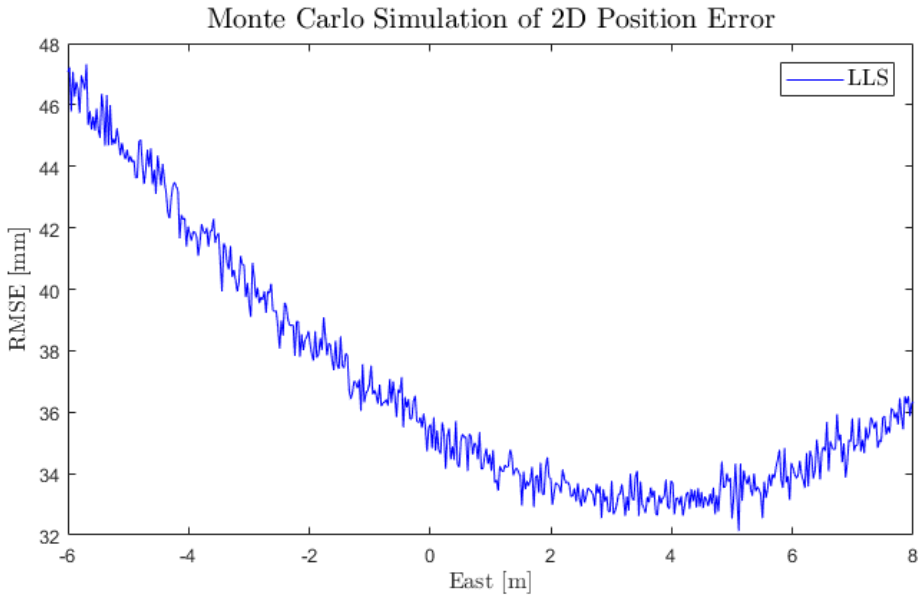


Figure 6.26: The simulation of the tag being carried at of 2 m height from -6 m east to 8 m east while being locked at 10 m north.

6.4 Validation of the Outdoor Experimental Results

A Monte Carlo simulation was used to evaluate the experimental results from Walk NR. 1 and Walk NR. 2. The simulation was based on the same assumptions which were presented in Section 6.2.1.

6.4.1 Monte Carlo Simulation

The tag was simulated to be carried at of 2 m height from -6 m east to 8 m east while being locked at 10 m north. The tag was then crossing the perimeters at both sides of the experimental set-up presented in Section 6.3 during the simulation. The LLS algorithm was used for the simulation, and the resulting error plot can be seen in Figure 6.26. Notice how the error gradually decreased as the tag moved towards the center of the grid, for then to increase as the tag continued towards the perimeter on the other side. Also, notice that the smallest calculated error from this simulation was roughly twice as big compared to the smallest calculated error from the indoor simulations (for instance Figure 6.8).

Implementing a Sensor Node in Robot Operating System

To investigate if the Pozyx system could be compatible with the existing ROS environment implemented on Milliampere, a ROS node was implemented so that data could be published and demonstrated in real time.

7.1 Software Environment

The following software was installed on the laptop used for testing the Pozyx system:

- Ubuntu 16.4 LTS
- ROS Kinetic 1.12.12

The node of the Arduino Mega was programmed in the Arduino's own C++ programming language by using a ROS library called *rosserial*. This was based on a modified version of the UAV-code.

7.2 Publishing Data

The Pozyx tag was connected to a laptop through USB running the Kinetic Kame version, which was also installed onboard Milliampere, as shown in Figure 7.1. The node published the following topics:

- `pozyx_range_anchors`
- `pozyx_network_addresses_anchors`
- `pozyx_time_anchors`

```
roscore http://svein-Aspire-V3-571:11311/
svein@svein-Aspire-V3-571:~$ roscore
... logging to /home/svein/.ros/log/c5981108-73f3-11e8-b5c9-6894235a7a5f/roslaun
ch-svein-Aspire-V3-571-6209.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://svein-Aspire-V3-571:34723/
ros_comm version 1.12.12

SUMMARY
=====

PARAMETERS
* /rostdistro: kinetic
* /rosversion: 1.12.12

NODES

auto-starting new master
process[master]: started with pid [6219]
ROS_MASTER_URI=http://svein-Aspire-V3-571:11311/

setting /run_id to c5981108-73f3-11e8-b5c9-6894235a7a5f
process[rosout-1]: started with pid [6232]
started core service [/rosout]
]
```

Figure 7.1: A roscore was started on the laptop. Note that the distribution used was Kinetic 1.12.12

The node published thus three topics with live data consisting of range measurements, the corresponding network addresses from the anchors, and the corresponding timestamps from the Pozyx tag. The publications can be seen in Figure 7.2.

The data published in the topics are shown in Figure 7.3. The first column contained anchor network addresses, the second column contained range measurements, and the third column contained timestamps. Note that the network addresses were represented as decimal numbers instead of hexadecimal numbers.

```
svein@svein-Aspire-V3-571: ~
svein@svein-Aspire-V3-571:~$ rosnode info /serial_node
-----
Node [/serial_node]
Publications:
* /diagnostics [diagnostic_msgs/DiagnosticArray]
* /pozyx_network_address_anchors [std_msgs/Int16]
* /pozyx_range_anchors [std_msgs/Int16]
* /pozyx_time_anchors [std_msgs/Int16]
* /rosout [rosgraph_msgs/Log]

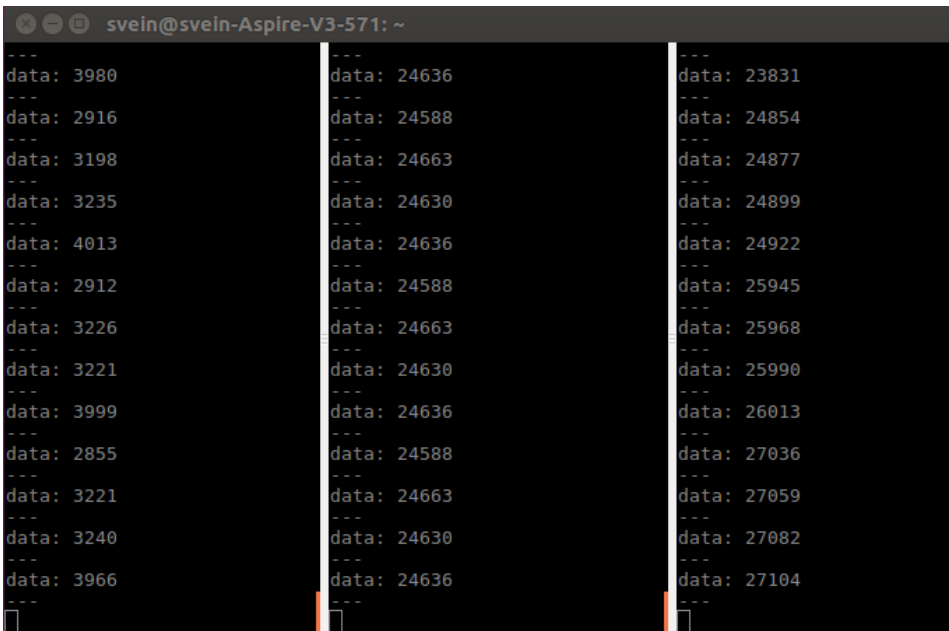
Subscriptions: None

Services:
* /serial_node/get_loggers
* /serial_node/set_logger_level

contacting node http://svein-Aspire-V3-571:43055/ ...
Pid: 6303
Connections:
* topic: /pozyx_range_anchors
  * to: /rostopic_6332_1529435216828
  * direction: outbound
  * transport: TCPROS
* topic: /pozyx_time_anchors
  * to: /rostopic_6376_1529435228559
  * direction: outbound
  * transport: TCPROS
* topic: /rosout
  * to: /rosout
  * direction: outbound
  * transport: TCPROS
* topic: /pozyx_network_address_anchors
  * to: /rostopic_6354_1529435222287
  * direction: outbound
  * transport: TCPROS

svein@svein-Aspire-V3-571:~$
```

Figure 7.2: The ROS-node information window.



```
svein@svein-Aspire-V3-571: ~  
---  
data: 3980  
---  
data: 2916  
---  
data: 3198  
---  
data: 3235  
---  
data: 4013  
---  
data: 2912  
---  
data: 3226  
---  
data: 3221  
---  
data: 3999  
---  
data: 2855  
---  
data: 3221  
---  
data: 3240  
---  
data: 3966  
---  
---  
data: 24636  
---  
data: 24588  
---  
data: 24663  
---  
data: 24630  
---  
data: 24636  
---  
data: 24588  
---  
data: 24663  
---  
data: 24630  
---  
data: 24636  
---  
data: 24588  
---  
data: 24663  
---  
data: 24630  
---  
data: 24636  
---  
---  
data: 23831  
---  
data: 24854  
---  
data: 24877  
---  
data: 24899  
---  
data: 24922  
---  
data: 25945  
---  
data: 25968  
---  
data: 25990  
---  
data: 26013  
---  
data: 27036  
---  
data: 27059  
---  
data: 27082  
---  
data: 27104  
---  
---
```

Figure 7.3: The live data from the published topics. The topics were from left to right: `pozyx_network_addresses_anchors`, `pozyx_range_anchors` and `pozyx_time_anchors`. Note that the network addresses are presented as hexadecimal values instead of decimal values.

Discussion

Several sensors, software for data processing and trilateration, multiple methods for conducting experiments and results have been presented in earlier chapters. However, some aspects must be discussed more in depth, that being possible error sources and others, not already mentioned, factors which might have affected the results. This chapter will also discuss the lessons learned, the challenges faced and the assumptions made in which might also have affected the project.

8.1 The Processing Algorithms

The processing algorithm for the indoor experiment was designed so that failed ranging cycles were discarded. This resulted in some of the measurements being neglected, and thus decreasing the size of the data set. However, the number of discarded measurements was small, compared to the total amount of collected data, so it did not significantly affect the overall quality of the data set.

The processing algorithm developed for the outdoor experiment was conservative, meaning that only measurements with the same timestamps were compared relative to each other. As a consequence, some of the recorded data was discarded because the timestamps from the Pozyx system and the RTK-GPS system were not matching exactly. A solution to this could have been to design the algorithm so that samples with *approximately* the same timestamps were compared, and thus generate a bigger dataset.

8.2 Measuring The Local Coordinate System

Based on the results from the indoor and the outdoor experiments it was likely that the accuracy when locating the anchors in the local coordinate systems was affecting the precision of the estimated poses. As one can see in e.g. Figure 6.2, the estimated positions were not clustered around the assumed exact position, but were instead biased from the reference point. This was true for all the algorithms in the indoor experiment, and could

be caused by miss-alignments with the local coordinate system. However, the algorithms did still produce pose estimates relative close to the exact position.

That said, the outdoor experiment did not manage to reproduce the same level of accuracy. This could be caused by the method used to measure the local coordinate system with the RTK-GPS was not accurate enough. To that end, the biggest challenge was to measure the correct height of the anchors, as shown in table 5.2, and a method for compensating for this was developed. Manually measuring the anchors' heights resulted in the most accurate height values, while attempts at smoothing the measured height (from the RTK-GPS) positions did not improve the precision in the vertical direction. As a consequence of this, it was necessary to conduct the experiment on a leveled surface so that height could be measured manually. On the other hand, smoothing the measured positions in the northern and the eastern directions did improve the accuracy notably, and was therefore favored for locating the anchors in the horizontal plane.

Another possible error source was the translation from ECEF coordinates to the local NED frame. Because the ECEF-coordinates were inaccurate from the RTK-GPS, the translation was also affected (Cisek et al., 2017).

Thus, better methods for measuring the local coordinate system would have improved the accuracy of the experimental results.

8.3 2D and 3D Precision

The indoor experiment resulted in the most accurate pose estimates from the Pozyx UWB location system. The LLS algorithm obtained the best accuracy for 2D and 3D positioning, as seen in table 6.1. Here, the 2D error was only 131 mm, while the 3D error was 432 mm. Also, the Monte Carlo Simulations suggested that even better accuracy could have been obtained if the tag had been located differently in the experimental grid.

The TRR algorithm obtained the overall smallest standard deviations from the indoor experiment, as seen in table 6.2. This showed that the TRR algorithm was better at clustering the estimates from a stationary tag.

The performance of the Algebraic algorithm was heavily dependent on the anchors' geometry, as one could expect, and the 2D and 3D accuracy was, therefore, varying. This was also true for the TRR algorithm when dealing with three anchors signal coverage. However, using the Algebraic algorithm as an initial starting point for the TRR algorithm did not notably improve the accuracy of the estimated poses.

Bigger differences between the trilateration algorithms were noted from the outdoor experiment. As one can see in table 6.5 and 6.6, the TRR algorithm was better in both 2D and 3D precision compared to the LLS algorithm. Note that the TRR algorithm was also more accurate compared to the LLS algorithm even when all solutions were included (remember *TRR* in table 6.5 and 6.6).

To that end, based on the relatively small differences in precision in the indoor experiment compared to the outdoor experiment, the NLLS approach achieved a better overall accuracy.

8.4 Tag Located Close to the Perimeter

The Indoor Experiment

The *indoor experiment* was also used to evaluate how the algorithms were affected when the tag was located close to, or outside of, the perimeters defined by the anchors. This would be relevant for Milliampere, because docking requires that the vessel is moving to a position (the dock) with a relatively small distance of the anchors, as illustrated in Figure 6.23.

As seen in Figure 6.6, it was clear that both of the algorithms (TRR and Algebraic) were affected when the geometry was altered and the tag was close to the perimeter. However, small differences were noted between the two algorithms. Based on how the tag was placed relative to the perimeters, the 2D precision ranged from approximately 99 mm to 247 mm, while the 3D precision ranged from approximately 553 mm to 713 mm.

For 2D precision, the biggest position error was measured when anchor 0x603C was removed. For the given scenario, the tag was then in close proximity of the perimeter defined by the remaining anchors. Also, the 2D position error was the smallest when anchor 0x6057 was removed. Here, the tag was then located inside the perimeter defined by the remaining anchors.

For 3D precision, the biggest position error was also here obtained when anchor 0x603C was removed. This was consistent with the results from the 2D scenario because the tag was also here close to the perimeter defined by the remaining anchors.

The smallest 3D position error was obtained when anchor 0x6036 was removed. This was not expected because the tag was then clearly outside of the perimeter defined by the remaining anchors (Murphy and Hereman, 1995). However, the third Monte Carlo simulation (Figure 6.11), confirmed the observed phenomena, and it can be seen in the simulation that the position error was smallest when the tag was located outside of the perimeter.

However, the 3D position error was the second smallest when anchor 0x6057 was removed. This was more consistent with expectations with the tag located inside of the perimeter defined by the remaining anchors.

All things considered, due to some inconsistency from the results, it could not be determined exactly how the location of the tag inside the perimeter defined by three anchors affected the results.

On the other hand, the two first Monte Carlo simulations with four anchors signal coverage showed that the accuracy was best when the tag was located approximately centered in the grid. As it can be seen in Figure 6.8 and 6.9, the error increased exponentially as the tag was closing up to the perimeters.

The Outdoor Experiment

The outdoor experiment was also used to validate how the Pozyx location system performed when the tag was moving close to the perimeters defined by the anchors.

From Walk NR. 1, seen in Figure 6.15 and 6.17, one can see that the estimates from the Pozyx system and the measurements from the RTK-GPS were deviating from each other on the right-hand side of the walk. This was also where the tag was located closest to the perimeters. This can also be seen in the error plot in Figure 6.18 from approximately sample number 20 to 60, and shows that the accuracy was degraded when the tag was close to the perimeter. However, not that the calculated error in the same plot was increased at approximately sample number 60. By studying the 2D path (furthest north), this can be explained by a misreading from the RTK-system, and was thus a discrepancy between the two systems.

The same can be said about Walk NR. 2, shown in Figure 6.19. As seen in the 2D path, the tag's location was deviating increasingly from the RTK-GPS as the sensors were closing up to the perimeters. This was consistent with the results from Walk NR.1, and showed that the tag's location relative to the perimeters affected the precision.

Also, the Monte Carlo simulation which replicated the outdoor experiment revealed that the accuracy was degraded when the tag was moving in a straight line in the eastern direction, and thus crossing the perimeters on its way, seen in Figure 6.26. As seen from the simulation, the LLS algorithm achieved the best precision when the tag was located approximately centered between the anchors.

In light of this, it was clear the precision of the location system was degraded when the tag was moved close to the perimeters. It is therefore necessary for the Autoferry Project to consider how the anchors should be located alongside the channel. If possible, moving some of the anchors further back on shore relative to the dock would benefit the ferry with more accurate pose estimates when docking.

8.5 Robustness to Multipath of Signals

Before walk NR. 2 was ended, the sensor platform was put to rest on the ground while the RTK-GPS and the Pozyx system were still recording. As seen in the 2D path in Figure 6.21 at approximately 14 m North and -3 m East, the RTK-GPS position was seemingly affected by reflections. However, by studying the corresponding poses estimated from the trilateration algorithms, it can seem like the Pozyx system was not that much affected by multipath of signals. With this in mind, it should be noted that the error plots in Figure 6.22 from sample number 160 to 190, was affected by this. In other words, it was not the Pozyx system that failed the most, but rather the RTK-GPS, and was thus a discrepancy in the plots.

It can therefore be argued, however based on a small number of samples, that the Pozyx system was more robust to reflections compared to the RTK-GPS sensor.

8.6 Physical Geometry of the Location System

The indoor and the outdoor experiments were not consistent in terms of the achieved precision. Whereas the indoor experiment could show to 3D precision of accuracy down to 43 cm, the outdoor experiment did not manage to achieve better than 136 cm precision.

An explanation for this, in addition to that the measured ECEF-coordinates of the anchors were uncertain, could be that the geometry of the anchors was different, and thus affecting the precision of the system. For the outdoor experiment, the distances between the anchors were up to approximately 19 m. However, the height difference between the anchors was only approximately 1.8 m. For the indoor experiment, there were smaller distances between the anchors, but the height difference was approximately the same. This could imply that the outdoor experimental set-up caused the trilateration problem to be ill-conditioned, and the precision was degraded as a consequence. The last Monte Carlo simulation (Figure 6.26) did also reveal that the calculated error from the outdoor experimental set-up was greater compared to the simulations replicating the indoor experiment (roughly twice as big). This also showed that the geometry of the anchors affected the precision.

An approach to improve the anchors' geometry, and thus generate more accurate estimates, could be to add additional anchors to the grid. As seen from the forth Monte Carlo Simulation with two extra imaginary anchors, the precision was improved (Figure 6.13) compared to the simulations with fewer anchors (Figure 6.9). However, adding more anchors would decrease the update rate of the presented Pozyx system. As explained, the tag addressed the anchors one by one, and adding anchors would then decrease the update rate approximately linearly with the number of anchors. Still, this would not be a real problem because of the high update rate of the sensors, and should be sufficient for multiple anchors.

8.7 The Outdoor Range Test

The range test proved that the Pozyx system was capable of measuring distances up to approximately 64 m, seen in Figure 6.24, assuming that the Pozyx system's antenna was located in a neutral orientation, and not pointing directly towards the anchors.

As seen in Figure 6.23, this could lead to some trouble with the practical range of the system, if implemented onboard Milliamperere. The figure is only an approximate suggestion, but it was still clear that the system would face difficulties with anchors located around the docks and the measured range.

Also, in light of the degraded accuracy when the tag was in close proximity of the perimeters, locating anchors too close to the dock would also affect the precision of the estimated poses when docking. As a consequence, a trade-off between range and precision is therefore necessary.

It should therefore be developed methods to improve the practical range of the location system, providing range measurements up to 100 m, as demonstrated in Figure 6.25. This could include upgrading both software and hardware of the given Pozyx UWB location system.

8.8 Robot Operating System

The Pozyx node was implemented on a laptop running the same distributions of Ubuntu and ROS as Milliampere. This was done in order to test if the sensor system could be compatible with the existing sensor system onboard the ASV.

The Arduino Mega performed well, and three topics were published in real time. These topics would then be available for other nodes onboard the vessel. An alternative approach could have been to implement the trilateration algorithms directly onboard the Arduino Mega, and thus only publish the estimated poses. However, the computational complexity of the NLLS approach could have caused problems for the relatively limited processing capability onboard the Arduino. Thus, future work should aim to implement an additional ROS node onboard the ferry dedicated to estimate position from range measurements supplied from the Pozyx system.

8.9 The Sensors

This section will discuss the lessons learned while working with the sensors.

Pozyx

The Pozyx libraries also contain their own trilateration algorithms. These algorithms are processing the raw data prior to pose estimation, that being e.g. compensations for wall penetration and out-lier detection. That said, the results in this report would also benefit from better pre-processing of the collected data. This should be addressed for upcoming project aiming to implement the Pozyx location system in real-time.

RTK-GPS

The received signals were easily affected by surrounding buildings, in addition to reflecting obstacles, which several times resulted in experiments being discarded. Finding a place where the satellite coverage was adequate turned out to be very important in order to obtain RTK-GPS results which were close enough to ground truth (desired results). However, even though RTK-GPS can give pose estimates down to cm precision (theoretically), this was generally not the case during experiments. More significant deviations in the position were experienced, and it is believed that the antenna used for this project was not optimal, and could have been better. As a consequence, this added to the calculated errors between the RTK-GPS and the Pozyx location system.

RTC

The RTC module, used to add timestamps to Pozyx data, was synchronized to local Unix time for the outdoor experiment. This method was chosen because it was a reasonably simple way to keep track of time while collecting data. A problem was that the time from the PC and the Pozyx system did not synchronize the time perfectly to the RTC. Differences up to approximately two seconds were experienced. Because of this, it was

not possible to compare corresponding data from the two systems with more than roughly two seconds of certainty. This did also add to the calculated errors between the Pozyx location system and the RTK-GPS.

Conclusion and Future Work

9.1 Conclusion

The work in this thesis concerned of testing the Pozyx UWB location system and post-processing of collected data, aiming to evaluate the feasibility of the given sensors as part of a docking system onboard the autonomous ferry Milliampere.

The NLLS approach achieved the best overall accuracy, both concerning 2D and 3D precision. The indoor experiments showed that the location system was capable of estimating 2D poses with precision down to 13 cm, and would therefore adequate for redundant positioning onboard Milliampere. Although outdoor experiments did not manage to reproduce the same level of accuracy, this can be explained by several error sources.

The experiments and the simulations revealed that the precision of the location system was degraded when the Pozyx tag was in close proximity of the perimeters defined by the anchors. The anchors should therefore, considering the autonomous docking problem, be located some distance behind the dock, ensuring that the ASV will not cross the perimeters.

The range test showed that the practical working range of the location system was too short compared to the actual distance between the docks in the channel. The measured range of 64 m is not adequate for the Autoferry Project. A solution must be developed in order to improve the range of the investigated Pozyx UWB location system.

A ROS node was implemented and published Pozyx sensor data in real-time on a laptop running the same software distributions as Milliampere. Based on this, it is likely that the Pozyx location system will be compatible with the existing sensor systems onboard the vessel.

All things considered, it is recommended for the Autoferry Project to implement the Pozyx UWB location system onboard Milliampere, as long as the practical working range can be improved.

9.2 Future Work

Upcoming projects should aim to implement the location system onboard Milliampere, testing real-time performance and robustness to environmental factors. Further work must also be done to develop robust processing algorithms to improve the sensors' raw data.

The optimal geometry of the anchors' locations in the channel must be further investigated, aiming to optimize the accuracy of the location system. Also, the number of anchors which should be used for the autonomous docking scenario must be considered.

The practical working range of the given sensors must be improved, and research should address whether upgrading software and hardware can increase performance.

The legal aspect of the investigated UWB location system must be evaluated regarding outdoor use and the potential of interference of other systems operating in the radio frequency spectrum. The study should make sure that the Pozyx UWB location system will not affect GNSS signals.

Bibliography

- Alarifi, A., Al-Salman, A., Alsaleh, M., Alnafessah, A., Al-Hadhrami, S., Al-Ammar, M. A., Al-Khalifa, H. S., 2016. Ultra Wideband Indoor Positioning Technologies: Analysis and Recent Advances. *Sensors* 16 (5), 707.
- Arduino, accessed May 12, 2018. ARDUINO MEGA 2560 REV3.
URL <https://store.arduino.cc/arduino-mega-2560-rev3>
- Befus, K., 2014. Localization and System Identification of a Quadcopter UAV. Master's thesis, Western Michigan University.
- Bitar, G. I., 2017. Towards the Development of Autonomous Ferries. Master's thesis, NTNU.
- Cai, G., Chen, B. M., Lee, T. H., 2011. Unmanned Rotorcraft Systems. Springer Science & Business Media.
- Cisek, K., Zolich, A., Klausen, K., Johansen, T. A., 2017. Ultra-Wide Band Real Time Location Systems: Practical Implementation and UAV Performance Evaluation.
- Drane, C., Macnaughtan, M., Scott, C., 1998. Positioning GSM Telephones. *IEEE Communications magazine* 36 (4), 46–54.
- Farrell, J., 2008. Aided Navigation: GPS With High Rate Sensors. McGraw-Hill, Inc.
- Hegerland, S. O., 2018. A Survey of Sensor Technologies for Autonomous Ferry Docking. Tech. rep., Faculty of Information Technology and Electrical Engineering, Norwegian University of Science and Technology.
- Huntsberger, T., Aghazarian, H., Howard, A., Trotz, D. C., 2011. Stereo Vision-Based Navigation for Autonomous Surface Vessels. *Journal of Field Robotics* 28 (1), 3–18.
- Jokiainen, E., 2016. Remote and Autonomous Ships: The Next Steps. Advanced Autonomous Waterborne Application Partnership, Buckingham Gate, London.

-
- Lee, J., Woo, J., Kim, N., 2017. Vision and 2D LiDAR Based Autonomous Surface Vehicle Docking for Identify Symbols and Dock Task in 2016 Maritime Robotx Challenge. In: Underwater Technology (UT). IEEE, pp. 1–5.
- Leedekerken, J. C., Fallon, M. F., Leonard, J. J., 2014. Mapping Complex Marine Environments with Autonomous Surface Craft. In: Experimental Robotics. Springer, pp. 525–539.
- Liu, Z., Zhang, Y., Yu, X., Yuan, C., 2016. Unmanned Surface Vehicles: An Overview of Developments and Challenges. *Annual Reviews in Control* 41, 71–93.
- Matlab, accessed June 18, 2018. Trust-Region-Reflective Least Squares.
URL <https://se.mathworks.com/help/optim/ug/least-squares-model-fitting-algorithms.html#broz0i4>
- Maxim Integrated, accessed May 10, 2018. DS3231 Extremely Accurate I2C-Integrated RTC/TCXO/Crystal.
URL <https://datasheets.maximintegrated.com/en/ds/DS3231.pdf>
- Mohus, I., Holand, B., 1983. Fish Telemetry Manual. SINTEF.
- Murphy, W., Hereman, W., 1995. Determination of a Position in Three Dimensions Using Trilateration and Approximate Distances. Department of Mathematical and Computer Sciences, Colorado School of Mines, Golden, Colorado, MCS-95 7, 19.
- Noble, B., 1988. Applied Linear Algebra. Prentice-Hall.
- Nocedal, J., J. Wright, S., 2006. Numerical Optimization. Springer.
- Norrdine, A., 2012. An Algebraic Solution to the Multilateration Problem. In: Proceedings of the 15th International Conference on Indoor Positioning and Indoor Navigation, Sydney, Australia. Vol. 1315.
- Pozyx, accessed March 5, 2018. Pozyx System Description.
URL <https://www.pozyx.io/Documentation/Datasheet/SystemDescription>
- Raychaudhuri, S., 2008. Introduction to Monte Carlo Simulation. In: Simulation Conference, 2008. WSC 2008. Winter. IEEE, pp. 91–100.
- Sonel, accessed May 27, 2018. Base Data from Trondheim.
URL <http://www.sonel.org/spip.php?page=gps&idStation=860>
- Tallysman, accessed May 12, 2018. TW4421/TW4422 Wideband Dual Feed GPS/GLONASS Antenna.
URL http://www.tallysman.com/wp-content/uploads/TW4421_TW4422_Datasheet_Rev4.pdf
- The Pozyx Manual, accessed June 07, 2018. Pozyx.
URL https://www.pozyx.io/Documentation/how_does_uwb_work
-

U-Blox, accessed May 12, 2018. NEO/LEA-M8T Series.

URL https://www.u-blox.com/sites/default/files/products/documents/NEO-LEA-M8T_ProductSummary_%28UBX-16000801%29.pdf

USA Department of Defense, 2008. Global Positioning System Standard Positioning Service Performance Standard. Tech. rep., USA Department of Defense.

URL <https://www.gps.gov/technical/ps/2008-SPS-performance-standard.pdf>

Vermeille, H., 2004. Computing Geodetic Coordinates From Geocentric Coordinates. *Journal of Geodesy* 78 (1-2), 94–95.

Appendix A

Table 9.1: Internal Factors in SWOT analysis, courtesy of Alarifi et al. (2016).

<i>Strengths</i>	<i>Weaknesses</i>
Licence free	potential interference to the existing systems which operates in the ultra wide spectrum due to misconfiguration
Low power consumption	may affect GPS and aircraft navigation radio equipment
Does not interfere with most of the existing radio systems	Very short pulses in UWB may take a long time to synchronize
Large bandwidth	
High data rate communication	
High processing gain in communication systems	
Involves very short pulses	
Carrierless transformation proport offers the advantage of hardwars simplicity	
Works well with low SNR	
Low probability of interception and detection	
Resistance to jamming	
Can penetrate different kinds of materials	

Table 9.2: External Factors in SWOT analysis, courtesy of Alarifi et al. (2016).

<i>Opportunities</i>	<i>Threats</i>
Robot guidance	
Tracking systems	In some cases not totally immune to multipath effects
Medical procedures and surgeries that require sub-millimeters of accuracy	Design and implementation of UWB antennas can be more challenging
Indoor localization systems	
Short pulses which can be utilized for non-communication purposes	
Sensor, positioning, and identification network (SPIN)	
Industrial warehouse applications	
Shipboard environment applications	
Millitary applications	
Applications for noisy environments	

Appendix B

Mathematical proof of solving equation set (3.22) - (3.24) is equal to solving (3.25), courtesy of Norrdine (2012).

$$(x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2 = r_1^2 \quad (9.1)$$

$$(x - x_2)^2 + (y - y_2)^2 + (z - z_2)^2 = r_2^2 \quad (9.2)$$

$$(x - x_3)^2 + (y - y_3)^2 + (z - z_3)^2 = r_3^2 \quad (9.3)$$

$$x^2 - xx_1 + x_1^2 + y^2 - 2yy_1 + y_1^2 + z^2 - 2zz_1 + z_1^2 = r_1^2 \quad (9.4)$$

$$x^2 - xx_2 + x_2^2 + y^2 - 2yy_2 + y_2^2 + z^2 - 2zz_2 + z_2^2 = r_2^2 \quad (9.5)$$

$$x^2 - xx_3 + x_3^2 + y^2 - 2yy_3 + y_3^2 + z^2 - 2zz_3 + z_3^2 = r_3^2 \quad (9.6)$$

$$(x^2 + y^2 + z^2) - xx_1 - 2yy_1 - 2zz_1 = r_1^2 - x_1^2 - y_1^2 - z_1^2 \quad (9.7)$$

$$(x^2 + y^2 + z^2) - xx_2 - 2yy_2 - 2zz_2 = r_2^2 - x_2^2 - y_2^2 - z_2^2 \quad (9.8)$$

$$(x^2 + y^2 + z^2) - xx_3 - 2yy_3 - 2zz_3 = r_3^2 - x_3^2 - y_3^2 - z_3^2 \quad (9.9)$$

$$(9.10)$$

$$\begin{bmatrix} 1 - 2x_1 - 2y_1 - 2z_1 \\ 1 - 2x_2 - 2y_2 - 2z_2 \\ 1 - 2x_3 - 2y_3 - 2z_3 \end{bmatrix} \begin{bmatrix} x^2 + y^2 + z^2 \\ x \\ y \\ z \end{bmatrix} = \begin{bmatrix} r_1^2 - x_1^2 - y_1^2 - z_1^2 \\ r_2^2 - x_2^2 - y_2^2 - z_2^2 \\ r_3^2 - x_3^2 - y_3^2 - z_3^2 \end{bmatrix} \quad (9.11)$$

$$A\vec{x} = \vec{b} \quad (9.12)$$