Politecnico di Torino

Master of Science Course in
Electronic Engineering

Academic Year 2022/2023

Graduation Session April 2023

Low-cost UWB system for mobile robotic facility ground truth measurements

A localization setup for extra-terrestrial outposts at Thales Alenia Space Italia - RoXY facility in Turin

Supervisors

Prof. Marcello Chiaberge

Ing. Patrick Roncagliolo

Ing. Ciro Napolitano

Candidate

Raffaele De Filippo

Student ID

287748

Let me say quite categorically that there is no such thing as a fuzzy concept... We do talk about fuzzy things but they are not scientific concepts. Some people in the past have discovered certain interesting things, formulated their findings in a non-fuzzy way, and therefore we have progressed in science.

Rudolf Emil Kalman

Abstract

Ultra-WideBand (UWB) technology has an acknowledged importance in the field of localization, particularly for what concerns Global Navigation Satellite Systems denied zones, and space and extraterrestrial environments make no exception. This thesis presents the design and development of a UWB localization system setup for a mobile rover in a $22 \times 26\,\mathrm{m}^2$ open-air facility that mimics the Martian environment. The system is based on commercially available UWB modules and custom hardware, consisting of 12 antenna systems (anchors) placed at three different heights on the perimeter walls of the facility and a mobile tag on the robot. The ranging information is processed by a ROS2 node implemented on the rover, which employs Kalman filters to estimate the pose of the robot using the trilateration algorithm, based on the known absolute positions of the anchors. The thesis includes the design of a custom PCB to adapt the 24V AC power supply of the facility to the 12V DC voltage required by the microcontroller, the 3D modeling of a detachable waterproof case for the antenna system, and the development of a 3D graphical simulator of the facility terrain and the rover using ROS2, to test filter algorithms and features and to emulate the real behavior with noisy measurements. Simulation tests show extremely precise tracking on the XY plane, with average errors of about 5 cm, while in experimental tests they stand on about 10 cm but still provide a reliable localization of the rover, also by virtue of an extensive activity of calibration and error mapping. Overall, this thesis presents a low-cost and high-performance UWB localization system for mobile robotics, providing ground truth measurements for facility tests but also making it suitable for a possible setup in the extraterrestrial outposts of tomorrow.

Contents

List of Figures

List of Tables

Introduction to the Thesis work

Ultra-WideBand (UWB) technology has been gaining significant attention in recent years as a viable alternative for localization in scenarios where Global Navigation Satellite System (GNSS) signals may not be accessible. The most notable advantages that UWB offers comprise high accuracy, low latency, and robustness against noise and multipath interference, making it suitable even for harsh environments. UWB technology has found several applications in fields like healthcare, automotive, and industrial automation, and it is also being explored for space and extraterrestrial environments, where a satellite-based system for localization is still not available.

This thesis presents the design and development of a low-cost UWB localization system for a mobile rover in a $22 \times 26\,\mathrm{m}^2$ open-air facility that simulates the Martian environment. The system employs commercially available UWB modules from Qorvo (Decawave) and comprises 12 antenna systems, called anchors, placed on the perimeter of the facility at three different heights and a movable one on the robot, the tag to be localized. In order to estimate the pose of the rover using the trilateration technique, the ranging data and the absolute positions of the anchors are fed to a ROS2 node that implements the Kalman Filter which performs the localization.

The first part is focused on UWB technology and its application in the localization field: in particular, in Chapter 1, a synopsis of the localization techniques and positioning systems that rely on satellite support is provided.
Chapter 2 instead deepens the UWB technology and presents an overview of its functioning, regulations, and the various techniques that can be applied to perform the ranging activity.
In Chapter 3 it is described how the trilateration algorithm works and how a Kalman Filter can implement it to deliver a reliable estimation of position. Furthermore, the chapter contains an introduction to the Kalman Filter math and base functioning, also presenting the two main typologies of filters that are commonly used when dealing with non-linear models, the Extended and Unscented Kalman Filters, which have been both implemented in this thesis for comparison purposes.

The second part is devoted to the presentation of the case study and the experimental activity that has been performed at the TAS-I RoXY facility: Chapter 4 collects some knowledge about the state of the art of the localization of robots in facilities and spatial environments, while in Chapter 5 the actual fieldwork is thoroughly described.

In particular, the facility setup and equipment are presented along with the hardware that has been used, which includes UWB antennas and STM32 microcontrollers and a custom PCB that has been designed to adapt the AC power supply of the facility to the DC voltage required by the microcontroller; the chapter also shows the development of a 3D graphical simulator of the facility terrain and the rover using ROS2, due to the necessity to extensively test and tune the algorithms accounting for noisy measurements, while the need for a reliable placing of the anchors that would not obstruct eventual reprogramming needs has led to the design of a 3D modeled detachable waterproof case.
Moreover, it is provided a concise explanation of the ROS2 framework and the characterization of the nodes that have been coded to perform the localization algorithms. Finally, both the simulation and field tests are described and analyzed, highlighting the differences between them and their causes. The thesis ends with a synopsis of the work that has been presented and predictions of what could be future improvements that can follow this activity.

Part I

UWB and Localization

Chapter 1

Localization techniques

Localization and positioning have become essential techniques in many areas of modern society: from navigation systems in automotive and transportation to tracking devices for wildlife, the ability to determine the precise location of objects has become increasingly important. This need is actually firmly bound to the necessity of mankind to explore the external environment and impose its control upon it. That led to the creation of the first instruments based on the observation of the sky and stars, used to let people get oriented during world exploration. With the advent of wireless technologies such as the employment of radio waves, the field of localization began to see its greatest development, with astonishing improvements in terms of reliability and variety of use cases.

There is now a multitude of positioning systems available for a wide range of applications. They present differences in terms of employed technology, field of application, accuracy, and so on. The most widespread and commonly used is surely the Global Positioning System (GPS), with falls in the bigger category of the Global Navigation Satellite Systems (GNSS), but it is usual to find positioning systems that rely on Wi-Fi and local wireless sensors to estimate the position of objects and people, such as the Bluetooth technology that nowadays can be found in every smartphone. The variety of techniques that have been implemented up to the present day can be classified to better delimiter the characteristics of each one as well as to group their similarities.

1.1 Classification of Positioning Systems

There exists more than one way to classify the various positioning systems and localization techniques. Here it has been chosen to follow two classifications: one based on the reference frame, as proposed by Zekavat and Buehrer [40], and the other which takes into account the signal measurement employed and the tracking method, as per Isaia and Michaelides [18].

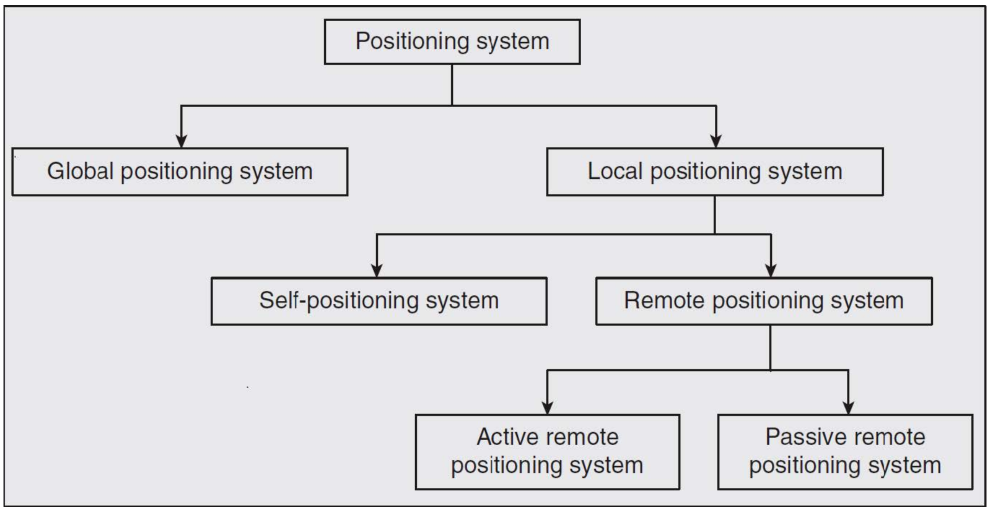

Positioning system
Global positioning system
Local positioning system
Self-positioning system
Remote positioning system
Active remote positioning system
Passive remote positioning system

Figure 1.1.1: General Classification on Positioning Systems [40]

The first mentioned classification focuses on the differences of reference frames: a **Global Positioning System** allows the localization of targets on the whole globe, using latitude and longitude information and providing an absolute measurement, while a **Local Positioning System** employs the information about other reference objects and their positions with respect to the actual target to deliver the estimated location, so constituting a form of relative localization.

LPSs can be furtherly differentiated considering again the reference frame: if the system always refers to the same static point at any given location and time, as happens for Inertial Navigation Systems, it is called a **Self-Positioning System**, otherwise, when the position is obtained through the referencing to other nodes located in the coverage area, it is defined as a **Remote Positioning System**.

An additional difference that can be pointed out regards the role of the reference nodes within the area of remote positioning: apart from the staticity or dynamicity of the nodes involved, they can actively cooperate in the localization of the target, as for example happens in the Radio Frequency Identification (RFID), or can be passive and not participating to the localization process, as in case of vision systems and radars. A system belonging to the first group is hence referred to as an **Active Remote Positioning System**, whereas a **Passive Remote Positioning System** reside in the second one.

Figure 1.1.1 summarizes this first categorization of positioning systems. Although it points out very clearly delineated boundaries, it makes no mention of the type of signal involved in the measurement process, neither does of the localization method itself. For this reason, it is now presented another type of classification that instead is based on the previously mentioned characteristics.

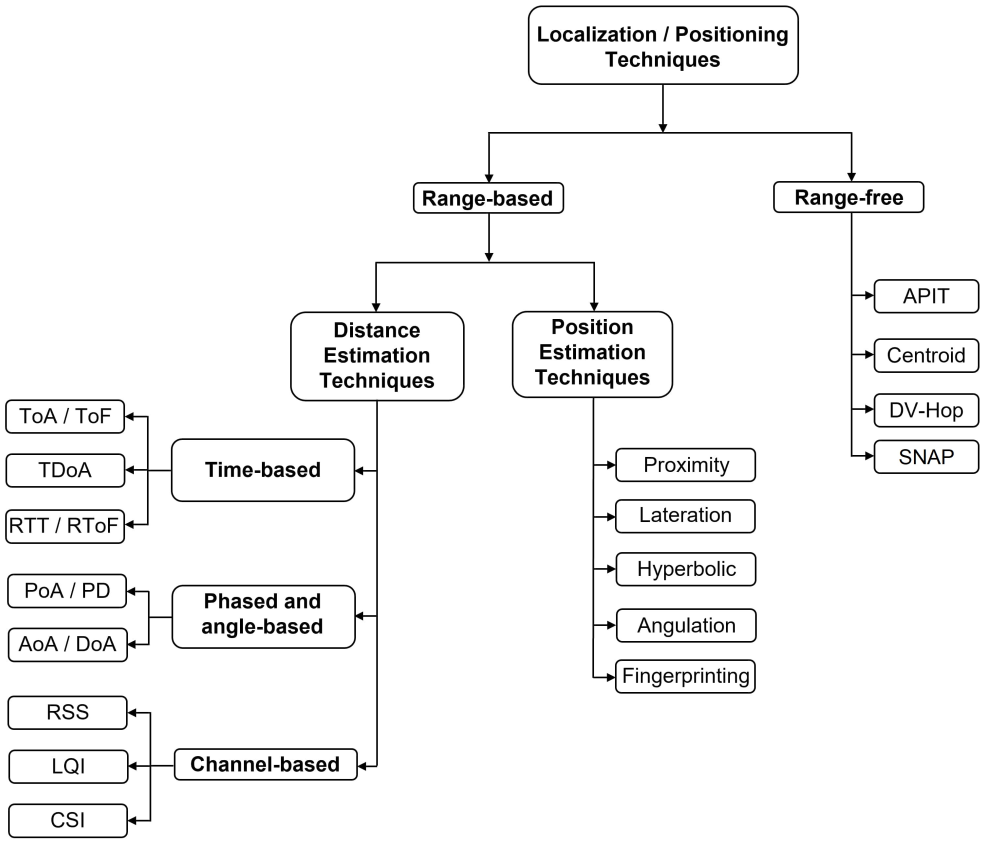


Figure 1.1.2: Localization/Positioning Techniques Classification [18]

Figure 1.1.2 shows a bigger variety of localization techniques, which are mainly distinguishable by the localization method (Range-based and Range-free) and the signal measurement employed (Time-based, Angle-based, and Channel-based).

1.1.1 Range-free localization

Range-free techniques directly estimate the position by using the topological data of the sensor nodes (i.e. the number of hops made in the propagation of the information). Despite providing limited accuracy because of the lack of correlation between absolute distance and signal strength, they don't need any additional hardware, which makes them more cost-effective and affordable than their counterpart.

Examples include the Centroid Method, Distance Vector-Hop, and Subtract on Negative, Add on Positive (SNAP) algorithm:

- The **Centroid** method estimates the receiving node's position as the centroid of all transmitting nodes' positions with minimum computations. The receiver node first determines whether it is situated close enough to the known sending nodes to communicate with them, then it pinpoints its location by utilizing the average coordinates of all known transmitting nodes that are situated inside the threshold zone. Relatively easy to implement, but its accuracy heavily depends on the environmental conditions and the density of anchor deployment.

- The **DV-Hop** technique is a distributed algorithm based on the distance vector routing protocol. All the network nodes are assumed to broadcast their localization information and estimate their distance from the other nodes, so that all nodes acquire the last hop count required to reach each other; then, the average hop count is computed and broadcasted by all nodes, and the unknown node utilizes it to calculate its distance from the other nodes. The assumption that the pathways are straight or that the nodes are dispersed uniformly is the major flaw in the DV-hop localization technique.

- The **SNAP** algorithm presupposes the usage of binary sensors and maximum likelihood estimation. In order to create the likelihood matrix, the sensor nodes' (+1) and (-1) contributions are used, depending on whether they are in an alarm state. While the non-alarmed sensors add a negative one, each alarmed sensor adds a positive one to the likelihood matrix elements corresponding to the cells inside its region of coverage (ROC). The region of influence (ROI) is identified by a probability of at least 0.5 for the sensor to be alarmed. An excellent explanatory picture is reported below:

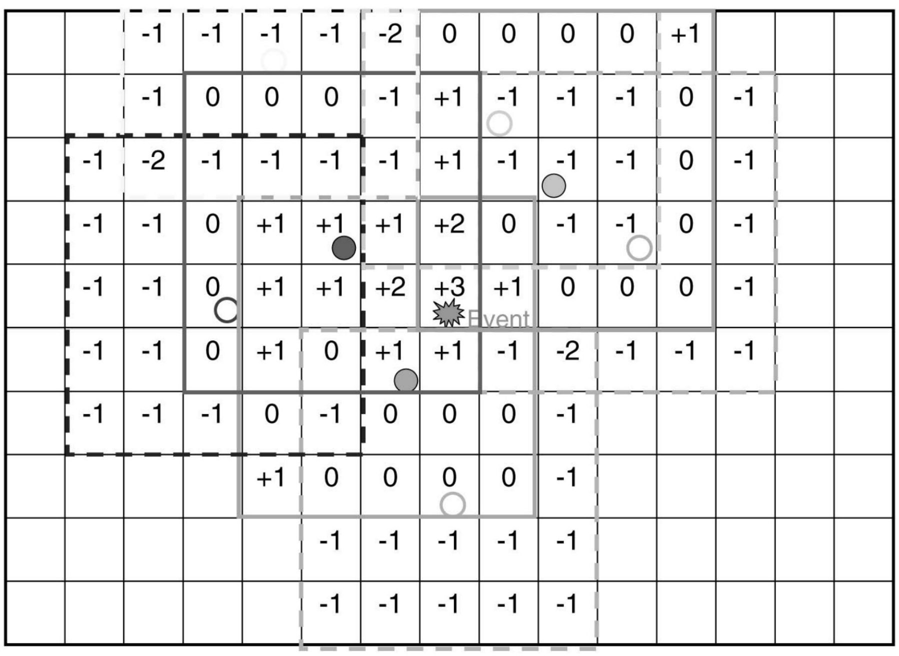


Figure 1.1.3: SNAP Likelihood matrix with eight sensor nodes, three of which are alarmed and shown in whole line. The event is correctly localized in the grid cell with the maximum value of +3 [23]

1.1.2 Range-based localization

Range-based localization techniques first utilize the signal measurements for estimating the distance from various reference nodes, relying on time-based, angle-based, and channel-based signals. Then, the distance measurements are combined using suitable algorithms for evaluating the nodes' unknown positions.

1.1.2.1 Distance Estimation Techniques

Time-based measurements estimate the time difference between the transmitting and receiving nodes based on the signal propagation delay, and that makes them particularly susceptible to environmental bias. Among them, there is worth mentioning Time of Arrival (ToA), Time Difference of Arrival (TDoA), and Round Trip Time (RTT):

- **Time of Arrival (ToA)** is a one-way time measurement of the signal propagation time and is employed to determine how far apart the transmitting and receiving nodes are from one another. The timestamps of the received packets are used by the receiving node to calculate its distance d from the transmitting node through the following equation:

  $$d = v \cdot (t_1 - t_0)$$

  where v is the velocity of the signal ($c = 3 \times 10^8$ m/s for electromagnetic waves) and t_0 and t_1 are the transmitting and receiving time, making their difference the time of flight (ToF). A graphical representation is shown in Figure 1.1.4a. Both precise time synchronization between the nodes' clocks and timestamp data to be included in the various packets transferred inside the network are necessary.

- **Time Difference of Arrival (TDoA)** is a variant of the method just presented, which is precisely employed to overcome its limitation concerning the synchronization of the nodes. The TDoA measurements estimate the time lapse between the receiving node and the two signals received from two different transmitting nodes, as shown in Figure 1.1.4b. The propagation distance difference $\Delta d_{i,j}$ at the receiving node side can be estimated as follows:

  $$\Delta d_{i,j} = d_i - d_j = v \cdot (t_i - t_0) - v \cdot (t_j - t_0) = v \cdot (t_i - t_j) = v \cdot \Delta t_{i,j}$$

  where i and j are the different transmitting nodes and v is the velocity of the signal. Typically, compared to ToA, the TDoA can offer improved positioning precision and simply needs temporal synchronization between the transmitting nodes.

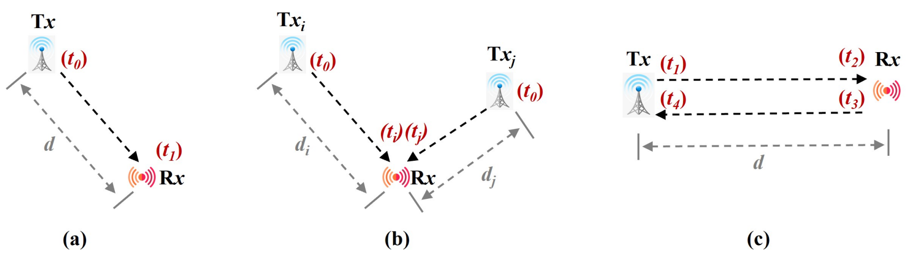


Figure 1.1.4: Time-based estimations: **(a)** Time of Arrival (ToA), **(b)** Time Difference of Arrival (TDoA) and **(c)** Round Trip Time (RTT) [18]

- **Round Trip Time (RTT)** calls for two-way communication between the nodes and the inclusion of the timestamp in the messages sent. It has been utilized in some Wi-Fi amendments, like the IEEE 802.11mc with Wi-Fi fine timing measurement (FTM). As it can be seen from Figure 1.1.4c, the distance measurement is obtained by:

  $$d = v \cdot \frac{(t_4 - t_1) - (t_3 - t_2)}{2}$$

  where t_1 and t_4 are the timestamps of the signal sent and received at the transmitting node, t_2 and t_3 are the timestamps of the signal received and sent at the receiving node, and v is the velocity of the signal.

Angle-based measurements include both phase measurements to compare the arriving signal phase at the target node (incident signal) with the phase of the signal transmitted and measurements of the angle between the nodes and the intersection of multiple directions of the wireless signal. It is so straightforward to discuss Phase of Arrival (PoA) and Angle of Arrival (AoA).

- **Phase of Arrival (PoA)** makes use of the phase difference of the carrier signal for estimating the distance between the transmitting and receiving nodes. The incident signals arrive with a phase difference at different antennas, as shown in Figure 1.1.5a. This quantity is then employed in the following formula that returns the distance d:

  $$d = \lambda \cdot \left(\frac{\phi_{Rx}}{2\pi} + k \right)$$

  where k is a positive integer, λ is the wavelength, and ϕ_{Rx} is the phase of the received signal. This technique suffers from poor accuracy, so it is often used in combination with other methods.

- **Angle of Arrival (AoA)** exploits the information about the angle between the sending and receiving nodes as well as the intersection of multiple wireless signal directions, as can be seen in Figure 1.1.5b. While ToA and TDoA require temporal synchronization between nodes, AoA does not need it and its accuracy is proportional to the density of the nodes. The disadvantage of this method is that additional hardware, like an antenna array, is needed. Many algorithms are available to compute the direction of the incoming signals, such as Multiple Signal Identification and Classification (MUSIC) and Estimation of Signal Parameters via Rotational Invariant Techniques (ESPRIT), the former providing more accuracy and more resolution the latter [29].

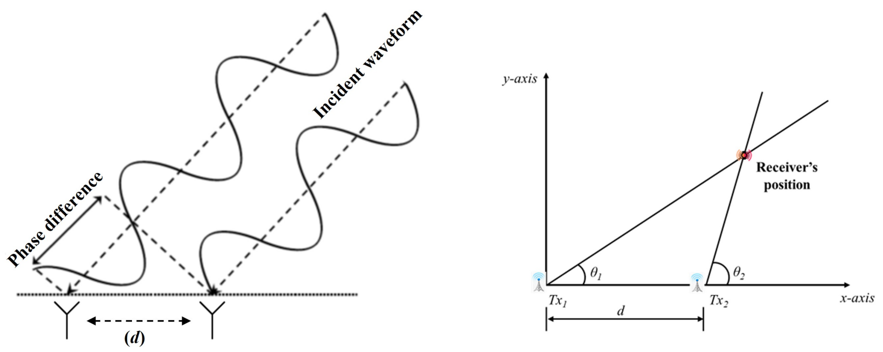


Figure 1.1.5: Angle-based estimations: **(a)** Phase of Arrival (PoA) [39] and **(b)** Angle of Arrival (AoA) [5]

Finally, if a technique employs information extracted from the analysis of the carrier wave itself, it falls back into the category of *Channel-based* measurements. Among the various ones, they include Received Signal Strength (RSS) and Channel State Information (CSI).

- **Received Signal Strength (RSS)** estimates the distance between the transmitting and receiving nodes by means of the signal strength attenuation, calculating the signal path loss, often expressed in decibel milliwatts (dBm). The latter is the reduction in power density of an electromagnetic wave as it propagates through space, caused by effects such as refraction, reflection, and multipath, as well as the environment and propagation medium. The signal strength is inversely proportional to the distance d and can be estimated by the following formula:

$$P(d) = P(d_0) - n \cdot 10 \log_{10}\left(\frac{d}{d_0}\right)$$

  where $P(d_0)$ is the signal power at a reference distance d_0, measured in dBm, n is the path loss attenuation factor and d is the distance between the transmitter and the receiver. The parameters are often evaluated statistically, performing measurements at known distances. Despite being one of the simplest and most widely used techniques, the accuracy of RSS is strongly affected by the presence of obstacles and multipath propagation, as well as environmental conditions such as rain, humidity, and temperature [15].

- **Channel State Information (CSI)** is a sophisticated ranging method employed to deliver precise received signal parameters throughout the whole signal bandwidth. It is exclusively employed in orthogonal frequency division multiplexing (OFDM) systems, in which data is simultaneously modulated on a number of subcarriers operating at various frequencies. The OFDM separates the data stream into several sub-streams and broadcasts them concurrently across several orthogonal frequency sub-channels, thus avoiding multipath effects and reducing the noise. The amplitude and phase frequency characteristics of the signal can be used to extract the CSI information in the frequency domain as a channel frequency response (CFR), which represents the multipath propagation of the signal. This frequency diversity provides a unique fingerprint, which can be used for constructing a radio map. As reported in [37], the combination of RSS and CSI can provide more accurate distance information, as compared to only using an existing RSS attenuation model. Moreover, the CSI was found to be more stable in the indoor environment.

Table 1.1 summarizes the distance estimation techniques that have been presented, highlighting their main features and limitations.

1.1.2.2 Position Estimation Techniques

After the distance measurements have been acquired, they have to be properly elaborated to get the actual information about the target position. Some of the most effective methods that can be used surely are (Multi)Lateration, Hyperbolic technique, and Fingerprinting.

- **(Multi)Lateration** works by intersecting the various acquired distances knowing all the nodes' positions in advance. The potential locations of the receiver are first indicated by a theoretical circle constructed in accordance with the measured range from each known transmitting node. The receiving node can be positioned anywhere on the perimeter of the circle for a single sending node, as shown in Figure 1.1.6a. The receiving node can be placed on either of the two intersection locations created by adding another transmitting node, as shown in Figure 1.1.6b. Finally, the ambiguity is reduced to a single point: the intersection of the three circles by adding a third transmitting node, as illustrated in Figure 1.1.6c.

  In 2D, the minimum number of anchors required is three (trilateration of circles), while for the localization in a 3D space, at least four sensors are required (multilateration of spheres), as visible in Figure 1.1.7. These theoretical circles or spheres cross at a single point in an error-free environment, revealing the location of the receiving node in space; however, due to the ever-present noise, the circles or spheres usually do not meet at a single point in real-world applications; as a result, a potential region is produced, from which a single point is chosen employing further methods such as the least squared method, the centroid method or using Kalman filters.

Distance Estimation Techniques	**Features**	**Limitations**
Time of Arrival (ToA)	More accurate results compared to RSSI measurements	• Precise time synchronization between nodes is required • Highly biased from the environment • Additional hardware may be required
Time Difference of Arrival (TDoA)	• Contrary to ToA, TDoA only requires precise time synchronization between the transmitting nodes • Provides higher accuracy than ToA • It can be used with different communication technologies, i.e. ultrasound and RF	• Time synchronization between transmitting nodes is required • Performance is degraded in the Non-Line of Sight signal propagation • Higher cost due to nodes deployment density required
Round Trip Time (RTT)	• It eliminates the need of utilizing different clocks and time synchronization	• It requires bidirectional communication capabilities • Accuracy is biased from the different changes in the environment • Latencies may occur for fast-moving devices
Phase of Arrival (PoA)	• It can be combined with other measurements for higher accuracy	• The phase cannot be distinguished, so it can only determine in which fraction of the wavelength a signal is received • It requires Line of Sight • Latencies may occur for fast-moving devices
Angle of Arrival (AoA)	• At least two transmitting nodes are required • No time synchronization between the nodes is required	• Accuracy is proportional to the nodes' density • It requires Line of Sight • Additional hardware is required (e.g., antenna arrays)
Received Signal Strength (RSS)	• Easily scaled to larger areas and multiple users • Minimum infrastructure required • Cost-effective computation	• Accuracy is affected by the existence of obstacles and multipath propagation • Affected by weather conditions, such as rain • Affected by Non-Line of Sight conditions
Channel State Information (CSI)	• It can be used as a unique fingerprint • More accurate estimates, compared to the RSS estimations	• It can only be used with technologies supporting OFDM • Special hardware is required (increased cost and complexity) • It is affected by the dynamic changes of the transmitting nodes' power

Table 1.1: Comparison of range-based distance estimation techniques

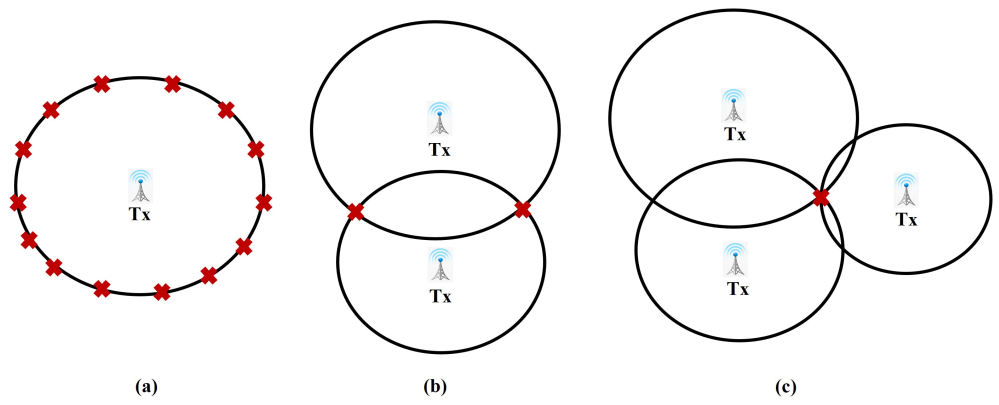


Figure 1.1.6: Trilateration localization technique: **(a)** the receiving node can be located anywhere on the circle circumference; **(b)** the receiving node can be located on both the two intersection points; **(c)** the receiving node can only be located on the intersection of the three circles [18]

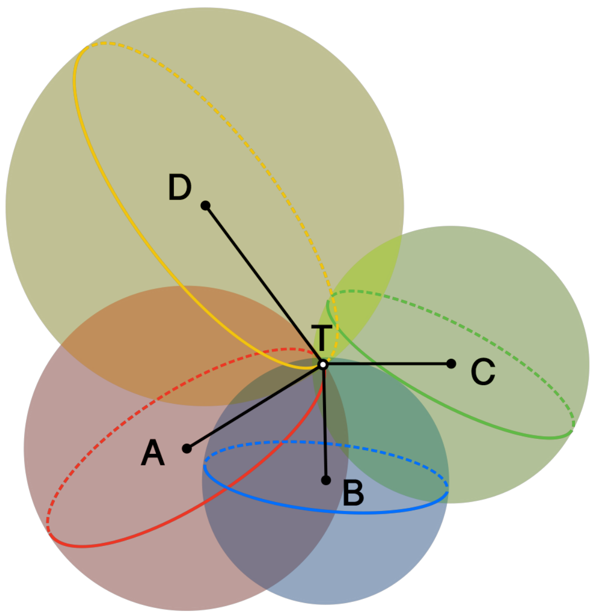


Figure 1.1.7: Multilateration localization technique: the fourth sphere identifies a single point between the two intersections of the other three spheres

- **Hyperbolic** localization makes use of TDoA measurements. Figure 1.1.8 illustrates how each TDoA offers a hyperbolic curve in the localization space on which the location of the unknown receiving node resides, and how their intersection identifies a specific location. Particularly, the numerous hyperbolic curves created indicate the various transmission times, while the known transmitting nodes represent the foci of the hyperbolae. The known anchor nodes 1 and 2 lay in the hyperbola's foci, and the focal length is equal to their difference in distance, $\Delta d_{1,2}$. The known anchor nodes 1 and 3 are the foci of the new hyperbola created by the addition of a third transmitting node, and their distance difference, $\Delta d_{1,3}$, serves as the focal length; clearly, the third node automatically generates another hyperbola by its relation with the second node, and this serves as a further reduction of the uncertainty.

At the intersection of all the hyperbolae, the unknown receiving node position is calculated as:

$$\Delta d_{i,1} = \sqrt{(x-x_i)^2+(y-y_i)^2+(z-z_i)^2} - \sqrt{(x-x_1)^2+(y-y_1)^2+(z-z_1)^2}$$

where (x_i, y_i, z_i) are the known i-th transmitting node's coordinates.

Many techniques involving a Taylor series expansion and transformation into a linear set of equations are frequently employed to solve these nonlinear hyperbolic equations. Chan's, Foy's, Fang's, and Friedlander's approaches all are valid alternatives: in [33], it was determined that Chan's method is the best option for solving the hyperbolic equations, even requiring an approximate a priori knowledge of all the distances.

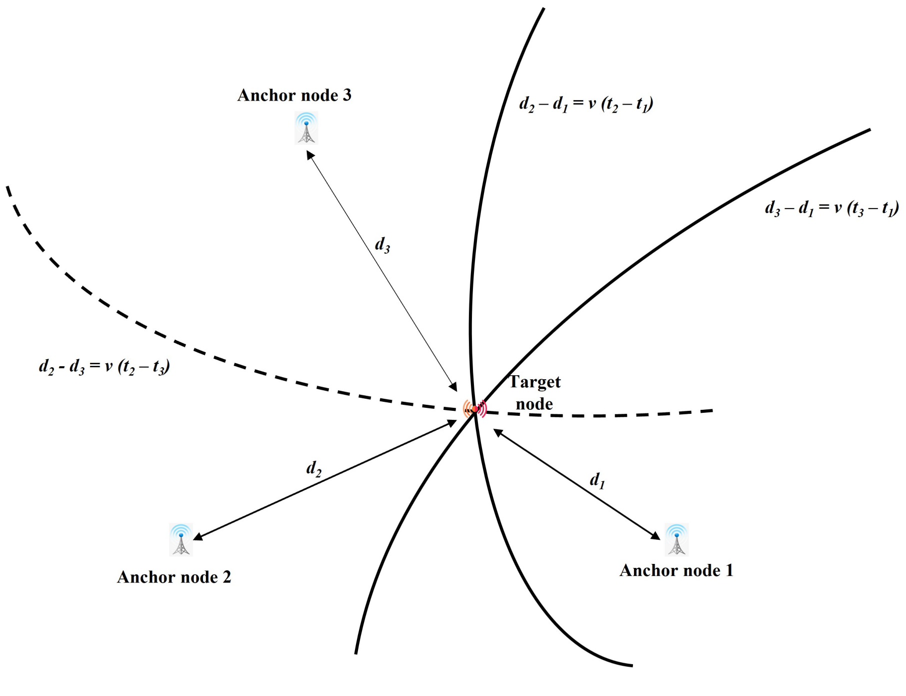

Anchor node 3
$d_2 - d_1 = v\,(t_2 - t_1)$
$d_3 - d_1 = v\,(t_3 - t_1)$
d_3
$d_2 - d_3 = v\,(t_2 - t_3)$
Target node
d_2
d_1
Anchor node 2
Anchor node 1

Figure 1.1.8: Hyperbolic localization technique [39]

- **Fingerprinting** is one of the most commonly used localization techniques that make use of Machine Learning. It consists of an offline data-collecting phase and an online matching phase and is based mostly on the analysis of particular measures or properties of the signal received by the unknown receiving node. A site survey is conducted during the offline phase using the specific location fingerprints, i.e., signal characteristics, coordinates of sample locations, and signal strength. It can be classified as deterministic or probabilistic [6]: the former involves the storage of various fingerprints, while the latter computes some statistics of the data first, such as taking a simple average of the RSS measurements. Then, during the online phase, real-time measurements are gathered and contrasted with those that have previously been recorded during the offline phase. The matching phase includes a comparison that determines the best match between the stored and real-time fingerprints, and this is frequently based on a matching score computed by means of the k-Nearest Neighbors ML algorithm. The main drawback of this approach is the requirement for large training datasets and high computational complexity, which can be challenging if the available devices only have low computational power.

1.2 Global Navigation Satellite Systems (GNSS)

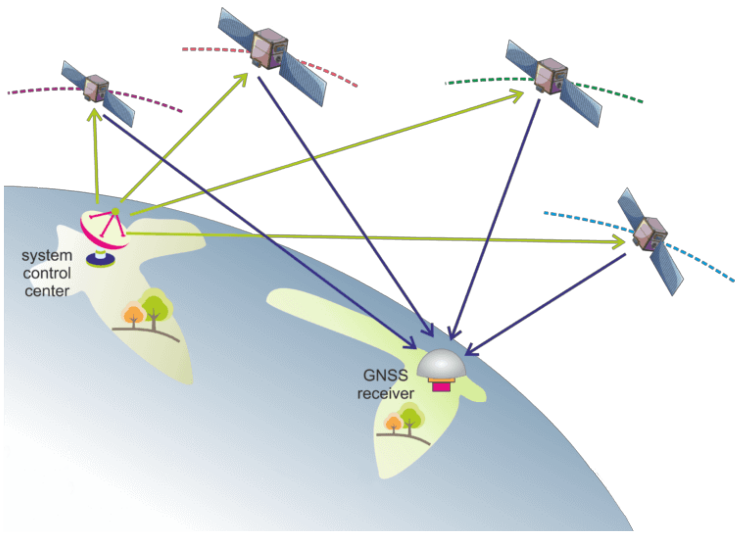


Figure 1.2.1: GNSS functioning. Source: Global Geodetic Observing System (*ggos.org*)

A GNSS provides a geo-spatial positioning service with global coverage that enables small, dedicated electronic receivers to determine their geographic coordinates in terms of longitude, latitude, and altitude on any point on the earth's surface or atmosphere with an error of a few meters by processing radio-frequency signals transmitted in line-of-sight from such satellites. To calculate the distances between satellites and receivers, the travel times of radio waves continuously emitted by satellites are measured in a very accurate way, since the timestamps are produced by the satellites' atomic clocks.

The distance between the receiver's unknown position and the satellite's known position is given by the difference between the signal's arrival and emission times, which is then multiplied by the speed of light. The initial distances determined from the signal travel times are known as pseudo-ranges since the satellite and receiver's clocks are not in sync. Clocks in satellites and receivers must be synchronized at the level of a nanosecond to tens of picoseconds in order to enable a correct determination of the distance between satellite and receiver: the system operators synchronize the atomic clocks in the satellites from the control center, while the receiver manages to get the information about its clock bias with respect to the true system time exploiting a fourth satellite, even if theoretically three of them would be enough.

Every GNSS estimate is affected by the variations in weather conditions, the availability and position of satellites relative to the receiver, the quality and type of receiver, radio signal propagation effects in the ionosphere and troposphere (e.g., the refraction), and the effects of relativity; for this reason, there have been developed several aiding systems, known as GNSS Augmentation Systems, that employ additional measurements to improve the performance of standard GNSSs: some examples are the Wide Area Augmentation System, the European Geostationary Navigation Overlay Service, and the Differential GPS.

1.2.1 Operating satellite-based positioning systems

At the present time, there are a few main GNSSs that are publicly available, which are the United States' most famous GPS, the European Galileo, and the Russian GLONASS; there exist other ones which are restricted to regional use, as the Chinese BeiDou 2, the Indian NavIC (formerly IRNSS) and the Japanese QZSS. Now follows a description of the above with their main features and limitations:

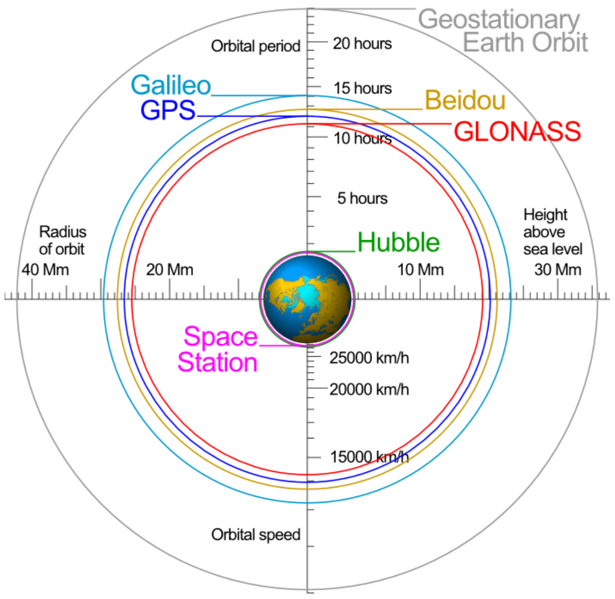


Figure 1.2.2: GNSSs orbits. Source: *Wikipedia*

- **GPS** is the abbreviation of NavSTAR GPS, which is, in turn, the acronym of Navigation Satellite Timing And Ranging Global Positioning System. It comprehends a total of 31 satellites arranged into six equally-spaced orbital planes surrounding the Earth, of which only 27 are actually considered part of the core constellation after an expansion made by the Air Force in 2011

[24]. Each GPS satellite flies in Medium Earth Orbit (MEO) at an altitude of approximately 20 200 km with a revolution period of half a day. In 1991, when the service first became available, the United States delivered two types of GPS services, the Standard Positioning Service (SPS) dedicated to civil use, while they reserved the Precise Positioning System (PPS) for the US military forces, which had an accuracy of $10 - 20$ m: in principle, the SPS was supposed to introduce errors in the satellite signal, reducing the accuracy to $900 - 950$ m, but in May 2000 this degradation was disabled by a decree of US President Bill Clinton. Each satellite broadcasts on two channels: L1, the only one available to SPS service, and L2 for exclusive use for PPS service and the carrier frequencies are 1575.42 MHz and 1227.6 MHz, respectively. As an example of the GPS accuracy, on April 20, 2021, as stated in [24], the global average User Range Error (URE) across all satellites was ≤ 0.643 m, 95% of the time.

- **Galileo** is the European GNSS co-developed by the European Commission (EC) and the European Spatial Agency (ESA), with an important contribution from the Italian Spatial Agency (ASI). Its commissioning, initially scheduled for the end of 2019, has been brought forward to December 15, 2016. Entirely designed for civilian use, Galileo is composed of 30 orbiting satellites (24 operating and 6 as backup) laying on 3 planes inclined to the Earth's equatorial plane by about 56° at an altitude of about 23 925 km (MEO), an optimal geometry constellation that allows better coverage and accuracy all around the globe if compared to GPS. The launch program, using Soyuz and Ariane rockets, began on October 21, 2011, with the departure of the first two satellites and continued with the launch of the second pair, IOV3 and IOV4, in October 2012. In August 2013, the trial phase of PRS (Public Regulated Service), a high-precision service designed to provide positioning data for the development of sensitive applications, began for users expressly authorized by national governments [38]. The other services that Galileo provides consist of an Open Service (OS), which delivers free positioning, timing, and navigation services for all users, with approximately 4 m accuracy; a Commercial Service (CS), a paid encrypted service that allows sub-meter accuracy, up to 10 cm; a Safety of Life (SoL) service, specifically designed for security operators (police and military) of the EU and member states and strategic applications for aviation, maritime and rail transport security, with the main purpose of delivering robustness against disturbances and reliable detection of problems within 10 seconds; a Search and Rescue (SaR) service that provides assistance to the COSPAS-SARSAT system for managing alerts and locating endangered users in order to assist rescue operations. Thales Alenia Space is responsible for industrial support activities related to system design, performance, integration, and validation. [19]

- **GLONASS** stands for GLObal NAvigation Satellite System, transliterated from Russian, and constitutes the Russian alternative to GPS and Galileo. Differently from GPS and Galileo, given the historical closeness of the Russian government and its distrust of abroad allies, GLONASS control stations are only located inside the former Soviet Union territory, making that a weakness that causes a reduction of precision and efficiency with respect to the other GNSSs. At the current state, the constellation comprises 24 operating satellites plus 4 back-ups, arranged on three orbit planes inclined at 64.8° to the equator and spaced by 120°, with satellites on the same orbit separated by 45°; GLONASS satellites are placed in roughly circular orbits with the nominal orbit altitude 19 100 km and an orbital period of 11 hours, 15 minutes, 44 seconds. Due to the period value, it became possible to create a sustainable orbital system that unlike GPS does not require supporting correcting pulses during its active lifetime [31].

- **BeiDou 2**, also known as *Compass Navigation Satellite System* (CNSS), is China's second-generation satellite navigation system capable of providing continuous, real-time passive 3D geo-spatial positioning and speed measurement. The first generation, 北斗 (běidǒu), was named after the Chinese name of the asterism of the Ursa Major constellation and was meant to provide navigation and timing for both military and civil use only to China and its surroundings, through its 4 satellites in Geostationary Earth Orbit (GEO); with the advent of Beidou 2, which at its final development will comprise 35 satellites, of which 5 in GEO (approximately at an altitude of 36 000 km) to maintain compatibility with BeiDou 1, 27 in MEO (about 21 000 km)and 3 in Inclined Geo-Synchronous Orbit (IGSO) and will ensure global coverage [9]. CNSS will provide two types of services: a free service for civilian users with positioning accuracy of within 10 m, velocity accuracy of within 0.2 m/s and timing accuracy of within 50 ns, and a licensed service with higher accuracy (up to 10 cm for authorized and military users only [32].

- **NavIC**, the acronym of NAVigation with Indian Constellation, is the operational name of the *Indian Regional Navigation Satellite System* (IRNSS). It covers India and a region extending 1500 km around it and an extended service area lies between the primary service area and a rectangle area of $1500 - 6000$ km km enclosed by the 30th parallel south to the 50th parallel north and the 30th meridian east to the 130th meridian east [35]. The system currently consists of a constellation of 7 satellites: three of them are located in GEO at longitudes 32.5° E, 83° E, and 131.5° E, approximately 36 000 km above Earth's surface, while the remaining four are in IGSO. NavIC provides two levels of service, the Standard Positioning Service (SPS), which is meant for civilian use, and a Restricted Service (RS), an encrypted service available only to authorized users. The system is reported to provide position accuracy better than 20 m during all weather conditions thanks to its signal which travels at dual frequencies in L5 and S bands [35].

- **QZSS**, also known as *Michibiki*, is the Japanese Quasi-Zenith Satellite System and it was initially meant to provide video and audio communication services and to serve as an augmentation for the GPS on the Japanese region [7]. The initial design called for the constellation to consist of three satellites, arranged in a geosynchronous, highly inclined, and slightly elliptical orbit (Quasi-Zenith Orbit QZO) at around 32 000 km, and separated from each other by 120°; thus, they would have to travel, relatively to a ground observer, on a figure-of-eight trajectory (called an analemma and visible in Figure 1.2.3) that should have ensured coverage of Japanese territory by at least one satellite at all times. Subsequently, though, the Japanese government decided to increase the number of satellites to four and signed an agreement with the European Spatial Agency (ESA) to integrate QZSS into the Galileo system in the near future.

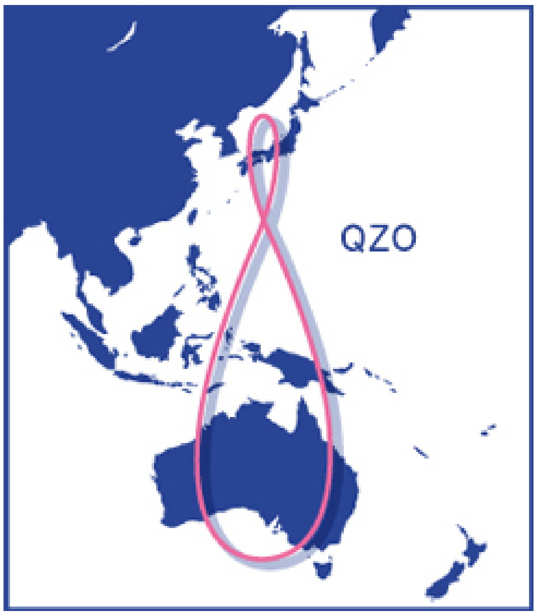


Figure 1.2.3: QZSS analemma path. Source: *qzss.go.jp*

1.3 Local Positioning Systems

Local Positioning Systems are a crucial technology for a wide range of applications, particularly in situations where the use of GNSSs is not feasible or reliable: for example, in indoor environments, such as warehouses or factories, GNSS signals may not be able to penetrate walls or other obstacles, or may do it at expenses of the signal strength, making it difficult to obtain accurate location information; furthermore, another undesired effect is the multipath propagation, which happens when the signal that reaches a receiver in Not Line of Sight (NLoS) comes from one or more reflections on the nearby obstacles and walls: this would cause the estimated range to be greater than it actually is, leading to even significant errors in the position estimation.

In these situations, LPS can provide accurate and reliable location data, allowing companies to track inventory, monitor equipment, and manage their operations more efficiently, with accuracies that can vary from centimeters to a few meters or more, depending on the system characteristics and the necessities of the case. These systems use a range of technologies, including Radio Frequency Identification (RFID), Bluetooth, Wi-Fi, and Ultra-WideBand (UWB) signals, to determine the location of objects or people within a specific area.

- **Radio Frequency Identification** is actually a not-so-recent technology, since its first prototypes date back to the mid-90s, with the first working device patented in 1973 [8]; it is a wireless communication technology that enables the automatic identification of objects, animals, or people by means of electromagnetic fields. It operates by using low-power radio waves to transmit information between a tag and a reader: the tag typically contains an integrated circuit and an antenna, while the reader consists of an antenna and a transceiver that reads and writes data to the tag. When RFID tags are affixed to objects or people, they can be tracked within a designated area, typically within a few meters or less. One of the benefits of RFID technology is its ability to function in harsh environments, including areas with high levels of dust, dirt, or moisture. RFID tags are also durable and can withstand exposure to extreme temperatures and physical impacts; furthermore, RFID technology does not require line-of-sight communication, meaning that the tags can be read through walls or other barriers.
  RFID tags are distinct into active and passive ones: active RFID tags rely on a power source, typically a battery, which enables them to transmit their signal over longer distances, up to hundreds of meters, and it is the one that actually sends interrogation signals and receives authentication replies; passive tags, on the other hand, depend on the energy of the reader's signal to power the tag's response and are typically employed because of cost or size requirements, being less expensive than active ones but having a shorter range, typically within a few meters. RFID technology has a wide range of applications, including inventory management, asset tracking, access control, and personnel tracking: for example, in a warehouse environment, RFID tags can be affixed to items, allowing the warehouse manager to track the location of inventory with the aid of Real-Time Locating System (RLTS). RFID technology can also be used to track personnel movements in hospitals or manufacturing plants, ensuring that workers are in the right location at the right time.

- **Bluetooth** is a well-known wireless communication protocol that enables data transfer between devices over short distances using radio waves. Bluetooth technology has evolved over the years, and the latest version of Bluetooth, Bluetooth Low Energy (BLE), has become increasingly popular for use in local positioning systems. Named after the 10th-century Danish king, Bluetooth operates at frequencies between 2.402 and 2.480 GHz, or 2.400 and 2.483 GHz, including guard bands 2 MHz wide at the bottom end and 3.5 MHz wide at the top. Bluetooth beacons are small, low-power devices that broadcast a signal at regular intervals, which can be picked up by Bluetooth-enabled devices, such as smartphones or tablets, allowing the devices to determine their proximity to the beacon; they typically have a range of up to 70 meters, although the

actual range can vary depending on environmental factors, such as the presence of obstacles.
BLE beacons are often placed at strategic locations throughout the environment, and when a Bluetooth-enabled device comes within range of a beacon, the device can determine its location based on the beacon's signal strength. There are several advantages to using Bluetooth technology as a local positioning system: first of all, Bluetooth technology is ubiquitous, and most smartphones and tablets are equipped with Bluetooth capabilities, meaning that implementing a Bluetooth-based system does not require the installation of additional hardware; moreover, BLE beacons are small and easy to install, making them an ideal solution for indoor application, where they can be placed inconspicuously throughout the environment, with the not secondary feature of low power consumption, that can allow them to run for months or even years on a single battery; in addition, Bluetooth technology is highly accurate when it comes to determining proximity, providing location information with an accuracy of up to a few tens of centimeters, making it possible to track the movement of individuals or objects in real-time.

- **Wi-Fi** stands for Wireless Fidelity and is a wireless communication protocol that enables data transfer between devices over short and long distances using radio waves. In recent years, Wi-Fi has become increasingly popular for use as a local positioning system due to its omnipresent nature and the ability to leverage existing Wi-Fi infrastructure. Wi-Fi positioning systems (WPS) can employ different position estimation techniques, such as AoA, Multilateration, or Fingerprinting, using a measurement of the strength of Wi-Fi signals from multiple access points. One of the key advantages of using Wi-Fi technology as a local positioning system is the non-invasivity, since it can exploit the already present access points often placed throughout the environment to provide full coverage, and the technology can determine the position of a device with the accuracy of up to a few meters; another benefit of Wi-Fi technology is its ability to integrate with other location-based technologies, such as BLE beacons or RFID tags: this integration enables location-based services to provide more accurate and comprehensive information about a user's position, allowing for more effective tracking and management of resources.
  This type of technology is also employed in a different way in public location databases, such as the Combain Positioning Service and Mozilla Location Service, whose aim is to provide both global indoor and outdoor positioning [10] by exploiting the connectivity of online devices and their Wi-Fi data information. There are, however, some limitations to using WiFi technology as a positioning system: Wi-Fi signals can be disrupted by physical barriers, such as walls or other obstructions, and the technology can be affected by interference from other electronic devices. In addition, WiFi signals can be affected by variations in the environment, such as changes in temperature or humidity.

- **Ultra-WideBand** is a wireless technology that enables high-precision positioning and tracking of devices within a designated area. UWB technology

works by emitting low-power pulses of energy across a wide frequency band, typically in the range of 3.1 to 10.6 GHz. UWB-based positioning systems work by using these pulses to measure the time of flight (ToF) between a transmitting device and multiple receiving devices, such as anchors or tags. By measuring the ToF of the signal between the transmitting device and each anchor, the device's position can be calculated with high accuracy, up to the centimeter level, making it ideal for use in applications where high precision is required, such as in industrial or manufacturing environments. Another advantage of UWB technology is its resistance to interference from other wireless devices since its signals are designed to operate in crowded environments and can coexist with other wireless technologies, being suitable for indoor location tracking applications, also thanks to its ability to penetrate walls and other physical barriers.
One limitation of UWB technology is its range: UWB signals have a limited range, typically not superior to 100 meters, compared to other wireless technologies, depending on the power of the signal and the environmental conditions; this makes it less suitable for outdoor applications where a longer range is required. However, the UWB topic is addressed much more deeply in the next chapter.

Chapter 2

Ultra-Wideband and Localization

Ultra-Wideband (UWB) is a wireless communication technology that uses a broad range of frequencies to transmit data over short distances with high precision and accuracy. Unlike other wireless protocols such as Wi-Fi or Bluetooth, UWB does not use a carrier frequency to transmit data, sending instead very short and low-energy pulses across a wide frequency band. This allows for faster and more reliable communication, as well as better location accuracy, making it ideal for a variety of applications, including indoor positioning, tracking, and radar imaging. In recent years, UWB technology has gained popularity and it has been integrated into a growing number of consumer electronic devices, such as smartphones, and smart home devices, as well as industrial and automotive applications, but its history actually started in the XIX century.

2.1 UWB technology overview

2.1.1 History and Developments

UWB technology is often perceived as a recent innovation compared to narrow-band radio transmissions, because of the huge increase in interest and popularity over the past few years; however, it is in fact an old technology: Guglielmo Marconi's first radio communication in the late 19th century was based on spark gap transmitters, developed after the previous works of Hertz, which produced brief pulses within a large bandwidth, which undoubtfully fall back in the UWB category. However, the spark gap transmitter was not very energy-efficient, and interferences were common due to the lack of means of synchronization. As a result, the development of vacuum tubes and transistors technologies led to the abandonment of UWB technology until the late 1960s when the US military was in need of a technology capable to avoid interceptions of wireless communication. Between 1969 and 1984, H. F. Harmuth published papers and books on UWB, placing in the public domain the basic design for UWB transmitters and receivers, while from 1972 to 1987, Ross and Robbins patented devices for using UWB in communications, radar, and sensing applications [2].

In 1989, the U.S. Department of Defense referred to this technology as "Ultra-Wideband" for the first time and kept the technology restricted to military use for the following few years. In the 1990s, T. E. McEwan developed the first low-power application of UWB, the "Micropower Impulse Radar", which also was small and affordable despite being able to function with just a few microwatts of battery depletion. The rush for regulation and standardization of UWB technology followed, and the Federal Communication Commission (FCC) approved the use of UWB signals for commercial use in 2002, followed by the European Commission in 2006, with a little stricter regulation. In 2019, Apple announced the first consumer electronic products embedding UWB technology, precisely the U1 chip.

2.1.2 Definition and Regulation

Various countries have released regulations regarding the use of Ultra-Wideband devices to ensure compliance with applications, allocated frequency ranges, maximum emissions levels, and techniques for reducing interferences. Referring to the previously mentioned approvals of the FCC in 2002 and the ECC in 2006, the definitions of UWB and the limits on frequency bandwidth and power density spectral emission that they state are slightly different.

The US FCC allows the unlicensed use of UWB signals within a frequency range of 3.1 to 10.6 GHz and that occupy a 10 dB bandwidth B equal or greater of 500 MHz, or in terms of fractional bandwidth B_f, superior to the 20%:

$$B \geq 500\,\mathrm{MHz} \quad or \quad B_f = \frac{2(f_H - f_L)}{f_H + f_L} \geq 20\% \tag{2.1}$$

where f_H and f_L are the upper and the lower frequency at which the signal has a power spectral density 10 dB lower than its maximum.

In March 2006, Europe released its regulations with two frequency ranges of 4.2-4.8 GHz and 6-8.5 GHz for indoor applications, with potential extension up to 9 GHz, and fixed emission levels, as can be seen in Figure 2.1.1 where the FCC and ECC spectral mask are compared. All countries have strict power emission levels for UWB devices, with FCC's mask allowing for a power spectral density of -41.3 dB/MHz for the whole mask, resulting in total transmitted power of 0.56 mW, and also precise bans about the applications of UWB, as for example the prohibitions on toys, aircraft, ships or satellites [4].

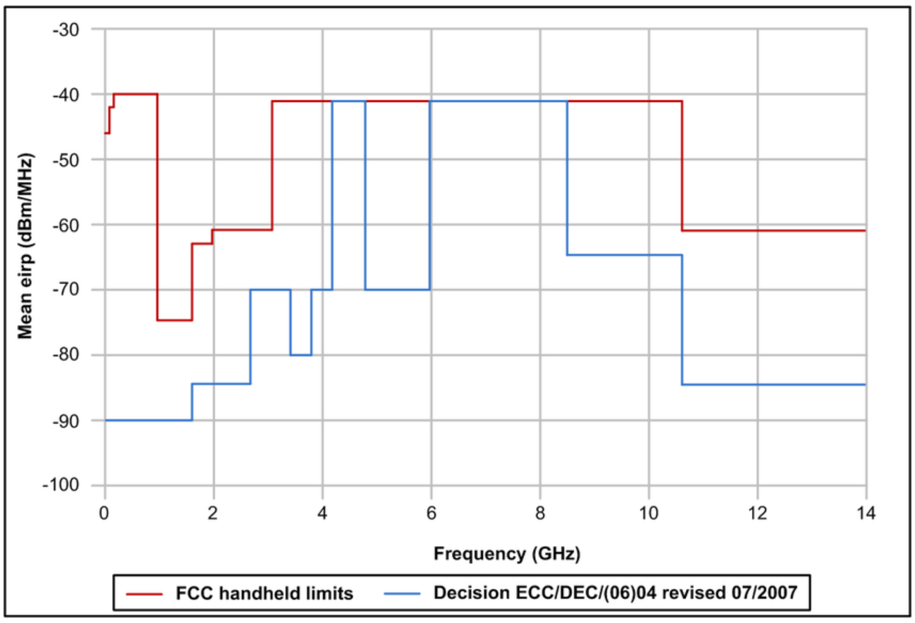


Figure 2.1.1: UWB spectral masks from FCC and ECC regulations [1]

2.1.3 UWB Theoretical Knowledge

UWB has several advantages over other technologies due to its unique characteristics. The distinctive trait of UWB is its wider bandwidth, which provides better results in terms of accuracy and resolution. This wider bandwidth, generated by the extremely short-duration pulses, allows for higher data rates over short ranges thanks to the high frequencies, or high penetration capabilities at lower data rates, because of its low-frequency content.

Another key advantage of UWB is its low power consumption, which makes it suitable for battery-powered devices, but also its low transmitted power, which instead has the benefit to cause minor interferences with other devices. In addition, UWB has a high multipath resolution, which means that it can distinguish between signals that have traveled along different paths, and also a high level of time accuracy, which is essential for applications that require precise synchronization. UWB has the ability to measure the time of flight of a signal to within a few nanoseconds, making it ideal for applications such as distance measurement and precise synchronization of wireless devices.

2.1.3.1 Channel model

The other main component of a UWB system, apart from the transmitter and the receiver, certainly is the wireless channel the electromagnetic waves travel through. The properties of this channel may alter the signal content output by the transmitter, thus it is crucial to know how to model them as precisely as possible, in order to apply the required correction at the receiver. There are typically two ways of modeling a channel: if the characteristics of the area that surrounds the transmitter are known from an electromagnetic point of view, it is possible to solve Maxwell's equations in a closed form; in doing so, what is obtained is a reliable prediction of the Channel

Input Response (CIR) of the environment, and it is called a deterministic model. Otherwise, if the model takes into account "canonical" characteristics of the channel, without specific reference to the actual location, the properties are known with a certain probability, and the model is said to be stochastic [13]. The latter is the most employed approach in modeling a UWB channel, and what follows are two examples of a stochastic model:

One of the simplest models of the CIR of a UWB channel is given by the following equation:

$$h(t) = \sum_{n=1}^{N} a_n \cdot \delta(t - \tau_n) \cdot e^{j\theta_n} \tag{2.2}$$

where $h(t)$ is indeed the channel input response, δ is the Dirac delta function, N is the number of signal components, a_n, τ_n and θ_n are respectively the amplitude, delay and phase of the n-th component.

The CIR is a useful form to express a channel model since the received signal $y(t)$ is simply given by the convolution of it with the input signal $x(t)$. Looking at Equation 2.2 it is trivial to recognize that the input signal is transformed by the CIR into a summation of N versions of it shifted in time, weighted by the respective a_n parameter, thanks to the Dirac delta function shifting property:

$$\begin{aligned} y(t) = h(t) * x(t) &:= \int_{-\infty}^{+\infty} h(\tau) \cdot x(t-\tau)\, d\tau \\ &= \int_{-\infty}^{+\infty} \sum_{n=1}^{N} a_n \cdot \delta(\tau - \tau_n) \cdot x(t-\tau) \\ &= \sum_{n=1}^{N} a_n \cdot x(\tau - \tau_n) \end{aligned} \tag{2.3}$$

where in this case a baseband signal is taken into account, thus omitting the θ parameter. Equation 2.3 shows in the clearest form why UWB works so well in a multipath environment: thanks to its extremely short pulses, the overlaps of the different components are minimum and the direct path signal is recognized correctly.

A quite more complex model is also presented, since it and its modified versions are the most used models to describe UWB channels. The Saleh and Valenzuela (S-V) model takes into consideration the characteristic of multipath components to cluster and so a double Poisson process is used [17]:

$$h(t) = \sum_{m=0}^{+\infty} \sum_{n=0}^{+\infty} a_{m,n} \cdot \delta(t - T_m - \tau_{m,n}) \cdot e^{j\theta_{m,n}} \tag{2.4}$$

where $a_{m,n}$ and $\theta_{m,n}$ indicates the same multipath amplitude and phase as the previous model, T_m is the arrival time of the first path of m-th cluster, and $\tau_{m,n})$ is the delay of the n-th ray of the m-th cluster. Figure 2.1.2 represent a typical power delay profile of the S-V model:

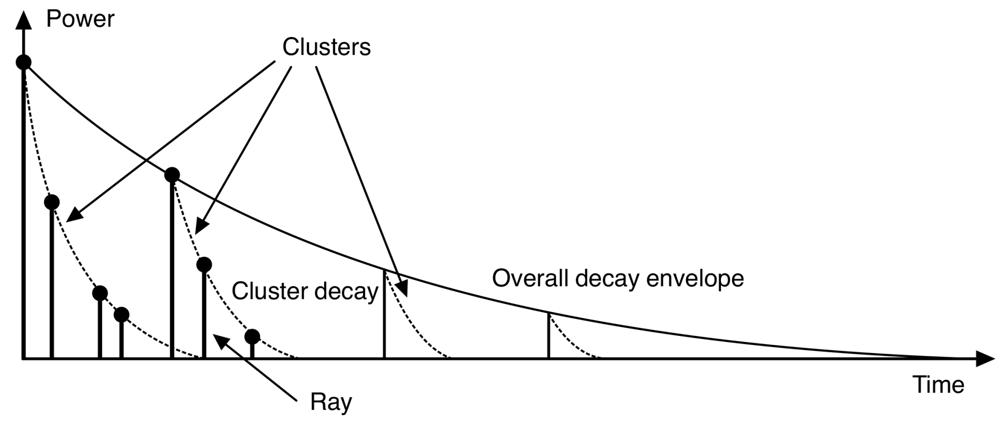


Figure 2.1.2: A typical power delay profile of the Saleh-Valenzuela model [12]

2.1.3.2 Signal Modulations

There is more than one way to exploit the UWB frequency spectrum: the simplest one is employing signals of the type described up to this point, so pulses of extremely narrow time duration, in the orders of nanoseconds; in this way, no carrier is needed and the efficiency is kept high despite simple architectural antennas. This method goes under the name of Impulse Radio (IR), but its simplicity leads to the major drawback of a not fully exploited band.

To overcome this limitation, different kinds of modulations can be used. Hereafter some of the most employed ones are presented:

- *Orthogonal Frequency-Division Multiplexing (OFDM)*: this technique makes use of multi-carrier frequencies, each one transmitting a specific portion of the band of the signal with its own modulation. By generating the sub-carriers such that they are all orthogonal to each other, undesired phenomena such as crosstalk can be avoided and small guard bands can be used, as shown in Figure 2.1.3; moreover, OFDM also delivers high-quality transmissions even in poor channel conditions, in addition to permitting complete exploitation of the characteristic wide-band. Clearly, this comes with a cost, specifically in terms of energy and hardware complexity, given the necessity to provide accurate frequency synchronization between the antennas.

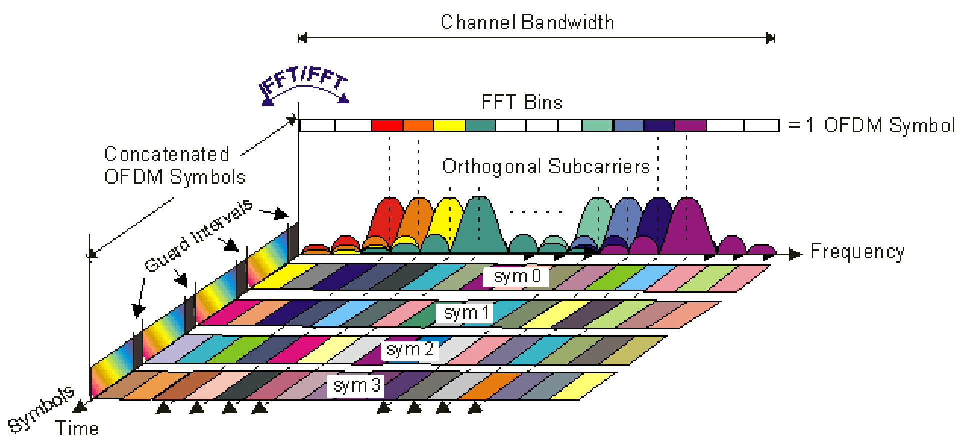


Figure 2.1.3: Frequency-Time representation of an OFDM signal. Source: *rfmw.em.keysight.com*

- *On-Off Keying (OOK)*: in order to transmit zeros and ones, one of the simplest modulation techniques is to encode the '1' with the transmission of a pulse and the '0' with no transmission at all. As can be seen in Figure 2.1.4, the bits' time slots have to be known precisely from the receiver to correctly identify the sequence.

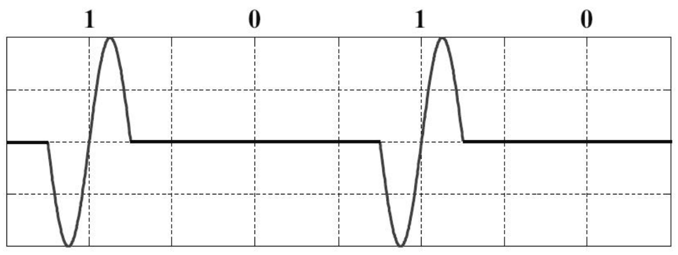


Figure 2.1.4: On-Off Keying bit pulses (Rayleigh monocycles [13])

- *Pulse Amplitude Modulation (PAM)*: this method can be seen as a generalization of the OOK, since in this case also the zeros are transmitted but encoded with a low amplitude pulse, while the ones' pulses get higher amplitude (Figure 2.1.5).

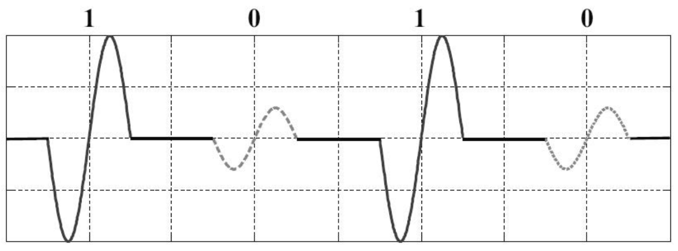


Figure 2.1.5: Pulse Amplitude Modulation bit pulses (Rayleigh monocycles [13])

- *Phase Shifting Keying (PSK)*: dissimilarly from the two previously mentioned modulations, PSK involves the polarity of the pulses to differentiate zeros and ones; if only two phases are employed (e.g. 0° and 180° as in Figure 2.1.6), the technique is referred to as Bi-Phase Shifting Keying (BPSK) and has the advantage of generating a smoother spectrum with respect to the spikes present in PAM and OOK modulations.

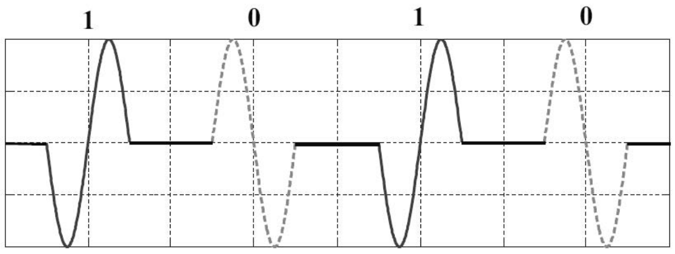


Figure 2.1.6: Bi-Phase Shifting Modulation bit pulses (Rayleigh monocycles [13])

2.2 UWB ranging techniques

In subsection 1.1.2, the main distance estimation techniques have been discussed and analyzed. Among them, the most used one when it comes to UWB antennas certainly is the Time of Arrival (TOA); it is usually preferred to other techniques such as AoA or RSS because of its hardware simplicity and higher accuracy. In general, time-based ranging techniques provide the best result in terms of accuracy when used in UWB systems, and it can be proved by means of a mathematical formulation of the distance estimation accuracy:

$$\sigma(d) \geq \frac{c}{2\sqrt{2}\pi\sqrt{SNR}\beta} \tag{2.5}$$

where $\sigma(d)$ is the standard deviation of the measured distance d, which is equal to the square root of the distance variance and provides the information about the measurement accuracy, c is the speed of light, SNR is the Signal to Noise Ratio and β is the effective signal bandwidth. Equation 2.5 is referred to as the Cramer-Rao lower bound for Additive White Gaussian Noise (AWGN) channels [16], and it is evident how the characteristic large band of UWB systems allows the distance estimation to be extremely precise.

As previously seen, the main drawback of the ToA method is the requirement for extremely precise clock synchronization between transmitters and receivers in order to get a reliable ToA estimation. This is usually not easy to ensure, especially if the UWB has to be designed to be low-cost; to overcome this limitation, the most commonly adopted solution is to apply to so-called Two-Way Ranging (TWR) protocol.

2.2.1 Two-Way Ranging

2.2.1.1 Single-Sided Two-Way Ranging

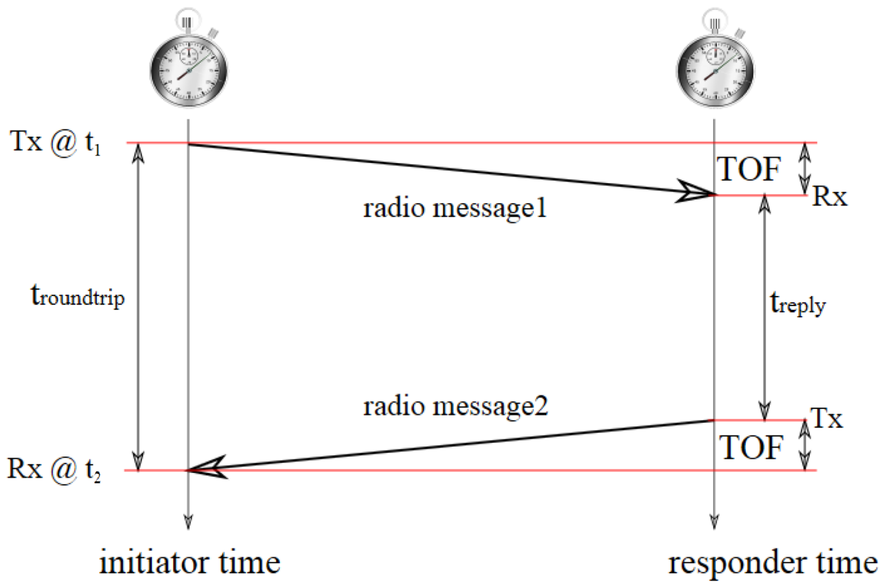


Figure 2.2.1: Single-Sided Two-Way Ranging (SS-TWR). Source: *Implementation of TWR, Qorvo*

In Figure 2.2.1 is depicted the Single-Sided Two-Way Ranging (SS-TWR) protocol: the initiator (on the left, also referred to as Tag) transmits a radio message to the responder (on the right, also referred to as Anchor) and records its time of transmission (transmission timestamp) t_1; the responder receives the message and transmits a response back to the initiator after a particular delay t_{reply}; the initiator then receives this response and records a reception timestamp t_2. Using the timestamps t_1 and t_2, the initiator can calculate the round trip time $t_{roundtrip}$ and knowing the reply time t_{reply}, the TOF can be determined by:

$$TOF = \frac{t_{roundtrip} - t_{reply}}{2} = \frac{t_2 - t_1 - t_{reply}}{2} \tag{2.6}$$

and the distance between the two nodes can be easily obtained by multiplying the TOF by the speed of light c, assuming that the speed of radio waves through the environment is the same as the speed of light.

As already said, the main source of error is known to be the varying clock speed between tag and anchor. To measure the impact of it on the measurement, Nierynck et al. proposed a fairly simple model [25]:

$$\hat{T} = \frac{1}{2} \cdot (\hat{t}_{roundtrip} - \hat{t}_{reply}) \tag{2.7}$$

$$\hat{t}_{roundtrip} := t_{roundtrip}(1 + e_{init})$$
$$\hat{t}_{reply} := t_{reply}(1 + e_{resp})$$

where T stands for TOF, X and $\hat{X}$ represent the nominal and actual values for each quantity involved, and e_{init} and e_{resp} model the deviation from the nominal frequency of, respectively, the initiator and the responder, and is typically expressed in parts per million (ppm).
From Equation 2.6 and Equation 2.7, the error on the Time of Flight $\hat{T} - T$ equals:

$$\begin{aligned}
\hat{T} - T &= \frac{1}{2}(\hat{t}_{roundtrip} - \hat{t}_{reply}) - \frac{1}{2}(t_{roundtrip} - t_{reply}) \\
&= \frac{1}{2}(\hat{t}_{roundtrip} - \hat{t}_{reply} - t_{roundtrip} + t_{reply}) \\
&= \frac{1}{2}(t_{roundtrip}(1 + e_{init}) - t_{reply}(1 + e_{resp}) - t_{roundtrip} + t_{reply}) \\
&= \frac{1}{2}(t_{roundtrip} + t_{roundtrip}e_{init} - t_{reply} - t_{reply}e_{resp} - t_{roundtrip} + t_{reply}) \\
&= \frac{1}{2}(t_{roundtrip}e_{init} - t_{reply}e_{resp})
\end{aligned} \tag{2.8}$$

Recalling that $t_{roundtrip}$ strictly depends on t_{reply}, given the structure of the protocol, it is clear that t_{reply} is the dominant factor determining the size of the ToF error. Typically, the ToF is in the order of nanoseconds, while t_{reply} is in the order of milliseconds and it is determined by the length of the packets to be transmitted, the processing speed of the measurement circuit, and its control logic. In subsection 5.2.1, it will be analyzed how the SS-TWR can be corrected in a particular way in post-processing to greatly increase its performance.

2.2.1.2 Double-Sided Two-Way Ranging

As just seen, SS-TWR is subject to a number of sources of error due to clock drift and frequency drift, so a variation of this method can be used to obtain better results.

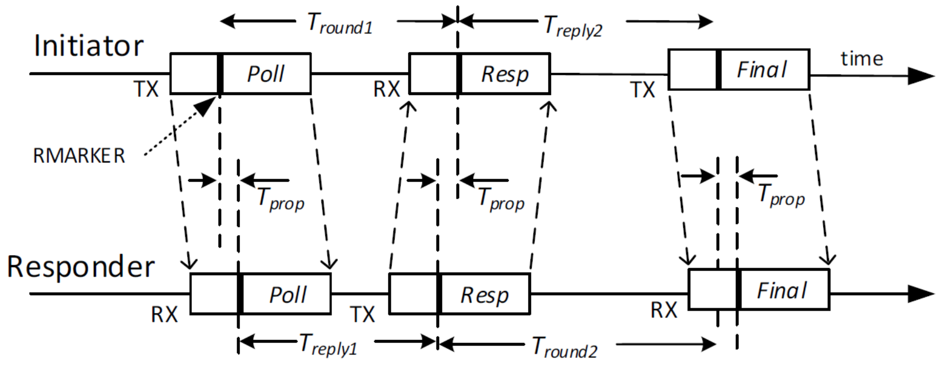


Figure 2.2.2: Double-Sided Two-Way Ranging (DS-TWR). Source: *Implementation of TWR, Qorvo*

The Double-Sided Two-Way Ranging (DS-TWR) is a particular implementation of TWR that makes use of three transmissions to overcome the potential errors that could be generated by the lack or loss of clocks' synchronization. In DS-TWR, a "poll" frame is sent by the initiator (recording the TX time-stamp of the poll) and then it waits for a "response" message expected from the responder. When the response is received, its RX timestamp is recorded and the initiator transmits a "final" message to complete the exchange. The final message contains all the timestamps recorded by it, including the poll TX, the response RX and the calculated/predicted TX timestamp for the final message itself; with this data and the local timestamps of poll RX, response TX, and final RX, the responder works out a value for the ToF over-the-air (T_{prop}) and, thus, the estimated distance between the two devices using the formula:

$$\begin{aligned} T_{round1} &= 2T_{prop} + T_{reply1} && (2.9a)\\ T_{round2} &= 2T_{prop} + T_{reply2} && (2.9b)\\ \implies T_{prop} &= \frac{1}{4}(T_{round1} - T_{reply1} + T_{round2} - T_{reply2}) && (2.9c) \end{aligned}$$

Again, the error that this variation of TWR produces can be evaluated with a model analogous to the previous one [25]:

$$\begin{aligned} \hat{T}_{prop} &= \frac{1}{4}(\hat{T}_{round1} - \hat{T}_{reply1} + \hat{T}_{round2} - \hat{T}_{reply2}) \qquad (2.10)\\ \hat{T}_{round1} &:= T_{round1}(1 + e_{init})\\ \hat{T}_{reply1} &:= T_{reply1}(1 + e_{resp})\\ \hat{T}_{round2} &:= T_{round2}(1 + e_{resp})\\ \hat{T}_{reply2} &:= T_{reply2}(1 + e_{init}) \end{aligned}$$

Therefore, the error on the Time of Flight $\hat{T}_{prop} - T_{prop}$ is derived as before:

$$\begin{aligned} \hat{T}_{prop} - T_{prop} &= \frac{1}{4}(\hat{T}_{round1} - \hat{T}_{reply1} + \hat{T}_{round2} - \hat{T}_{reply2}) +\\ &\quad - \frac{1}{4}(T_{round1} - T_{reply1} + T_{round2} - T_{reply2})\\ &= \frac{1}{4}(T_{round1}(1 + e_{init}) - T_{reply1}(1 + e_{resp}) +\\ &\quad + T_{round2}(1 + e_{resp}) - T_{reply2}(1 + e_{init})) +\\ &\quad - \frac{1}{4}(T_{round1} - T_{reply1} + T_{round2} - T_{reply2})\\ &= \frac{1}{4}(e_{init}(T_{round1} - T_{reply2}) + e_{resp}(T_{round2} - T_{reply1})) \end{aligned} \tag{2.11}$$

which, recalling the definitions of T_{round1} and T_{round2} from Equation 2.9a and Equation 2.9b, becomes:

$$\hat{T}_{prop} - T_{prop} = \frac{1}{2}T_{prop}(e_{init} + e_{resp}) + \frac{1}{4}(e_{init} - e_{resp})(T_{reply1} - T_{reply2}) \tag{2.12}$$

It is easy to recognize that if T_{reply1} and T_{reply2} were designed to be equal (Symmetrical DS-TWR), the second addendum would cancel out, and the error would be minimized, depending only on T_{prop} which has been said to be in the order of nanoseconds and would lead to a time error smaller than a picosecond. In practice, the two reply times cannot really be perfectly equal, but, referring to the IEEE 802.15.4a standard and considering the worst case $(e_{init} + e_{resp})$ of $\pm$ 40 ppm and the worst case $T_{reply1} - T_{reply2}$ in the order of nanoseconds, the second half of the error term would be in the order of femtoseconds, leading to a distance measurement error of tens or hundreds of micrometers [25].

There still are some disadvantages to the use of DS-TWR: it requires at least one packet transmission more than the SS-TWR, and only the anchor is able to compute the distance measurement, otherwise a further transmission has to be performed from the anchor to the tag, to inform it of all the quantities and let it do the calculations; this obviously results in a more time expensive communication, with the resulting increasing probability of faults for the computation of a single measurement, which would also lead to the repetition of the whole procedure from the start.

The desire to eliminate the requirement for equal reply delays led Nierynck et al. to propose alternative processing [25] that does not need a modification of the DS-TWR protocol, and is referred to as Asymmetrical DS-TWR: starting from Equation 2.9a and Equation 2.9b, rather than adding them as before, they are multiplied as follows:

$$T_{round1}T_{round2} = (2T_{prop} + T_{reply1})(2T_{prop} + T_{reply2}) \tag{2.13}$$

which lead to a final formulation of the ToF of:

$$T_{prop} = \frac{T_{round1}T_{round2} - T_{reply1}T_{reply2}}{T_{round1} + T_{round2} + T_{reply1} + T_{reply2}} \tag{2.14a}$$

$$= \frac{T_{round1}T_{round2} - T_{reply1}T_{reply2}}{2(T_{round1} + T_{reply1})} \tag{2.14b}$$

$$= \frac{T_{round1}T_{round2} - T_{reply1}T_{reply2}}{2(T_{round2} + T_{reply2})} \tag{2.14c}$$

In this way, the Time of Flight estimation $\hat{T}_{prop}$ can be written as:

$$\hat{T}_{prop} = \frac{\hat{T}_{round1}\hat{T}_{round2} - \hat{T}_{reply1}\hat{T}_{reply2}}{2(\hat{T}_{round1} + \hat{T}_{reply1})}$$

$$= \frac{(1+e_{init})(1+e_{resp})}{(1+e_{init})} \frac{T_{round1}T_{round2} - T_{reply1}T_{reply2}}{2(T_{round1} + T_{reply1})} = (1+e_{resp})T_{prop} \tag{2.15a}$$

$$= \frac{\hat{T}_{round1}\hat{T}_{round2} - \hat{T}_{reply1}\hat{T}_{reply2}}{2(\hat{T}_{round2} + \hat{T}_{reply2})}$$

$$= \frac{(1+e_{init})(1+e_{resp})}{(1+e_{resp})} \frac{T_{round1}T_{round2} - T_{reply1}T_{reply2}}{2(T_{round2} + T_{reply2})} = (1+e_{init})T_{prop} \tag{2.15b}$$

The error between the estimated and true values is therefore:

$$\hat{T}_{prop} - T_{prop} = (1+e_{init})T_{prop} - T_{prop} = e_{init}T_{prop} \tag{2.16a}$$

$$= (1+e_{resp})T_{prop} - T_{prop} = e_{resp}T_{prop} \tag{2.16b}$$

By comparing Equation 2.12 and Equation 2.16, it is evident that both of them have best-case errors of eT_{prop}, but the latter achieves this performance under all circumstances, despite different reply delays. In addition to that, this alternative processing makes the ToF error dependent only on one device, while the standard one averages the two errors, and that could be exploited if one device has a much better time reference than the other to yield superior performances.

Chapter 3

Position Estimation: Kalman Filters

With the distance measurements obtained by the chosen ranging technique, the next step to obtain the actual position estimation is to feed the measurements into a model that is able to relate each one of them to the others and the a priori known anchor's positions. Some of the techniques used to fulfill this purpose have been discussed in subsubsection 1.1.2.2; among them, Multilateration is probably the most used one, but the problem of dealing with ever-present errors in measurements and the consequent lack of a perfect estimation of a single point still remains.

The literature is plenty of algorithms designed to accomplish this goal and the most commonly employed ones surely rely on Linear Least Square (LLS) estimation, Gauss-Newton (G-N) algorithm, and Kalman Filters (KF).
The LLS method is a statistical algorithm used to find the parameters of a linear system that minimize the sum of the squared errors between the measurements and the predicted values, being computationally efficient and able to handle large datasets, but under the assumption of Gaussian noise and obviously, as the name suggests, of linear systems.
The G-N algorithm is an optimization method commonly used in nonlinear least squares problems, where precisely the former method fails. It iteratively updates the estimate of the unknown parameters by approximating the nonlinear equations with a linear model and solving for the parameters using the least squares method. The Gauss-Newton method can handle nonlinear systems, but it can converge to a local minimum and is sensitive to the initial estimate.

Compared to previously mentioned techniques, the Kalman Filter is more powerful and flexible, being able to handle nonlinear systems, non-Gaussian noise, and time-varying dynamics, making it a more versatile technique for state estimation. It consists of two main steps, the prediction step and the update step: in the prediction step, the filter predicts the state of the system at the next time step based on the current state and a model of the system's dynamics; in the update step, the filter adjusts the predicted state based on a new measurement, taking into account the uncertainty of the measurement and the predicted state.
The Kalman Filter is able to perform better than other techniques because it takes

into account the dynamics of the system and the uncertainty of the measurements, while also providing a means to update the estimate as new measurements become available: this allows the filter to be more accurate and robust, especially in situations where there is significant noise or uncertainty in the measurements. Additionally, the KF can be easily adapted to handle different types of systems and measurements, making it a widely used tool in a variety of applications, more than other types of filters such as the Particle filter because of its simpler structure and lower complexity.

In the next sections, the math of the basic Kalman filter is deepened and variations of it meant to deal with nonlinear problems are also discussed and compared [21].

3.1 Kalman Filter overview

The algorithm of a basic multivariate Kalman Filter is suited to deal with linear systems, and what follows is an explanation of how it works assuming to be in presence of white Gaussian noise and discretized process of interest, which is the case dealt with in the experimental part of this work.
Let us assume to be able to describe the process of interest by its state with a simple differential equation in matrix form:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \mathbf{w} \tag{3.1}$$

where $\dot{\mathbf{x}}$ is the first time derivative of the state $\mathbf{x}$, $\mathbf{A}$ is the system dynamic matrix, which indeed describes the system dynamics with respect to each component of $\mathbf{x}$, $\mathbf{B}$ is the control matrix, which models the effect of the control input $\mathbf{u}$ on the system state, and $\mathbf{w}$ is the process noise.

Since the quantity of interest is $\mathbf{x}$ rather than $\dot{\mathbf{x}}$, it can be characterized by means of this discrete-time equation obtained with the *state space method*:

$$\mathbf{x}_{k|k-1} = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{B}_k\mathbf{u}_k \tag{3.2}$$

where, ignoring the discrete-time noise $\mathbf{w}_k$ for a moment, $\mathbf{F}$ is the state transition matrix, also called *fundamental matrix*, and its purpose is to describe the transition to the state's value between discrete time steps, which are indicated with the $_k$ subscript as a convention for the quantity value at the k^{th} value of time t and with the $_{k|k-1}$ notation to indicate the quantity at the time step k given the evidence from the step $k-1$.

Then, to update the estimation of the state of the system, it is necessary to measure it. Typically, what the measurement returns is not in the state space, so to be able to compare the two quantities, one has to be transposed in the other's space; to accomplish this task many algorithms can be used depending on the specific nature of the process: in the case of positioning, the position estimation algorithms seen in the subsubsection 1.1.2.2 are fit for the purpose. In general, a measurement can be expressed as:

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k \tag{3.3}$$

where $\mathbf{z}$ is the measurement, $\mathbf{H}$ is the measurement function, which is just what is needed to yield a measurement starting from the information of the state $\mathbf{x}$, so it is chosen to implement one of the position estimation algorithm previously mentioned, and $\mathbf{v}$ is the measurement noise.

At this point, the various equation can be rewritten in the following way with the state space method:

$$\begin{aligned} \hat{\mathbf{x}}_{k|k-1} &= \mathbf{F}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_k \\ \mathbf{z}_k &= \mathbf{H}\mathbf{x}_k \end{aligned} \tag{3.4}$$

where $\hat{\mathbf{x}}$ denotes the fact that the state so obtained is a prior estimation, i.e. a first guess of the filter made only given the last state and the control input, and where in the measurement $\mathbf{z}$ expression the discrete-time noise $\mathbf{v}_k$ is left aside, as done before for $\mathbf{w} - k$. This is due to the fact that both of them are incorporated in two other matrices $\mathbf{R}_k$ and $\mathbf{Q}_k$, respectively the measurement noise covariance and the process noise covariance, which are used in the next steps.

The state estimation is not a sure thing, but clearly comes with a certain belief, a certain awareness of how much it is accurate. This awareness is mathematically rendered with the $\mathbf{P}$ matrix, the state covariance matrix, whose equation is:

$$\hat{\mathbf{P}}_{k|k-1} = \mathbf{F}\mathbf{P}_{k-1}\mathbf{F}^{\mathrm{T}} + \mathbf{Q}_k \tag{3.5}$$

where $\hat{\mathbf{P}}$ is again the prior estimation of the $\mathbf{P}$ matrix. The multiplication with $\mathbf{F}$ matrix and its transpose is common in linear algebra, where is called *congruence transformation*, and it can be viewed as *projecting* the state covariance by the state transition, thus obtaining an estimate of how the belief in the state estimation *transitions* over the time; the process noise $\mathbf{Q}_k$ is also used here to reflect a certain loss of knowledge due to the presence of the noise.

Equation 3.2 and Equation 3.5 are referred to as the Prediction Equations, as they are used to compute the prior next state of the system. Then, it follows the update phase of the algorithm, where the measurements are employed to correct the prior estimation.

As already mentioned, the measurement and the state estimation have to be compared to get a residual $\tilde{\mathbf{y}}$, i.e. the difference between the actual measurement and the hypothetical measurement that one would get if the state of the system were equal to the prior estimation. The residual is simply obtained with the equation:

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_{k|k-1} \tag{3.6}$$

Furthermore, to work in the measurement space, the Kalman filter has to project the state covariance into the measurement space: the method used is analogous to the one used to transition the matrix $\mathbf{P}_{k-1}$ in Equation 3.5:

$$\mathbf{S}_k = \mathbf{H}\hat{\mathbf{P}}_{k|k-1}\mathbf{H}^{\mathrm{T}} + \mathbf{R}_k \tag{3.7}$$

where $\mathbf{S}$ is called system uncertainty matrix. Again, the congruence transformation is used to pass from the state space to the measurement one, and therefore the $\mathbf{R}_k$ comes into play to add the uncertainty information due to the measurement noise, as was before with $\mathbf{Q}_k$.

Now a measurement and a prediction are available, and it is needed to select an estimate between the two. To perform this task, a matrix that constitutes the very core of the Kalman filter is employed, the Kalman gain $\mathbf{K}$; it is obtained through the formula:

$$\mathbf{K}_k = \hat{\mathbf{P}}_{k|k-1}\mathbf{H}^{\mathrm{T}}\mathbf{S}_k^{-1} \tag{3.8}$$

It can be seen as the quantity that models the correction to be applied to the prior state estimation given the measurement: if the inverse of a matrix can be thought to as the linear algebra's way to provide the reciprocal, then the Kalman gain can be seen as the "ratio" of the prediction uncertainty ($\hat{\mathbf{P}}_{k|k-1}$) over the measurement uncertainty ($\mathbf{S}_k$), which is then "converted" back to the state space through the term $\mathbf{H}^{\mathrm{T}}$.

At this point, the prior state estimation $\hat{\mathbf{x}}_{k|k-1}$ can be corrected by exploiting the information that comes from the residual, and it is done by scaling the latter by the Kalman gain:

$$\mathbf{x}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\tilde{\mathbf{y}}_k \tag{3.9}$$

The only thing left to update is the state covariance matrix $\mathbf{P}_{k|k-1}$ involving the same information used in the state update:

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\hat{\mathbf{P}}_{k|k-1} \tag{3.10}$$

where $\mathbf{I}$ stands for the identity matrix. This equation allows adjusting the size of the uncertainty on the state with respect to the prior estimation based on the Kalman gain, as done for the state update. One thing that can be pointed out is that the previous equation is numerically unstable, since the $(\mathbf{I} - \mathbf{K}_k\mathbf{H})$ subtraction can lead to a nonsymmetric matrix due to floating point errors when implemented on a real computation device, with a consequent divergence of the Kalman filter; for this reason, to this formulation, it is preferred the *Joseph* equation that can be proved to be equivalent and far less likely to make the filter diverge when facing real-world conditions [21]:

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\hat{\mathbf{P}}_{k|k-1}(\mathbf{I} - \mathbf{K}_k\mathbf{H})^{\mathrm{T}} + \mathbf{K}_k\mathbf{R}_k\mathbf{K}^{\mathrm{T}}{}_k \tag{3.11}$$

Table 3.1 summarizes the Prediction and Update equations discussed until this point and their role within the algorithm:

Predict Step	
$\hat{\mathbf{x}}_{k\|k-1} = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_k$	Prior state estimation at the next time step, given the system behavior and the control input
$\hat{\mathbf{P}}_{k\|k-1} = \mathbf{F}\mathbf{P}_{k-1}\mathbf{F}^{\mathrm{T}} + \mathbf{Q}_k$	Prior state covariance estimation, to adjust the belief in the prediction accounting for the uncertainty
Update Step	
$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_{k\|k-1}$	Residual between the prediction and the measurement
$\mathbf{S}_k = \mathbf{H}\hat{\mathbf{P}}_{k\|k-1}\mathbf{H}^{\mathrm{T}} + \mathbf{R}_k$	System uncertainty, given the belief in the accuracy of the measurement
$\mathbf{K}_k = \hat{\mathbf{P}}_{k\|k-1}\mathbf{H}^{\mathrm{T}}\mathbf{S}_k^{-1}$	Kalman gain, the scaling factor based on whether the prediction of the measurement is more accurate
$\mathbf{x}_{k\|k} = \hat{\mathbf{x}}_{k\|k-1} + \mathbf{K}_k\tilde{\mathbf{y}}_k$	Posterior state estimation, corrected based on the residual and the scaling factor
$\mathbf{P}_{k\|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\hat{\mathbf{P}}_{k\|k-1}$	Posterior state covariance, corrected according to the belief about the measurement correctness

Table 3.1: Multivariate Kalman filter equations

3.2 Extended Kalman Filter

As mentioned at the beginning of the previous section, the basic multivariate Kalman filter can only cope with linear problems as it is designed. This poses a huge limitation to its use, since the real world abounds in processes that can be modeled only as nonlinear, especially the ones for which a tool such as the Kalman filter is most useful, as the localization discussed in this work.

In order to relax this constraint and be able to deal with nonlinear problems, many variations of the Kalman filter have been designed and proposed through the years, among which one of the very first techniques that still remains the most commonly employed is the Extended Kalman Filter (EKF). The EKF handles nonlinearity by linearizing the system at the point of the current estimate, and then the linear Kalman filter is used to filter this linearized problem, as is often done by many other algorithms that address nonlinear problems.

Recalling the previous equations for the process and measurement models and their corresponding discrete-time state space representation:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \mathbf{w} \\ \mathbf{z} &= \mathbf{H}\mathbf{x} + \mathbf{v} \end{aligned} \quad \Longrightarrow \quad \begin{aligned} \hat{\mathbf{x}}_{k|k-1} &= \mathbf{F}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_k \\ \mathbf{z}_k &= \mathbf{H}\mathbf{x}_k \end{aligned}$$

it is easy to substitute the linear expression $\mathbf{Fx} + \mathbf{Bu}$ with a nonlinear function $f(\mathbf{x}, \mathbf{u})$ and the linear expression $\mathbf{Hx}$ with a nonlinear function $h(\mathbf{x})$.

$$\hat{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) + \mathbf{w} \tag{3.12}$$

$$\mathbf{z} = h(\mathbf{x}) + \mathbf{v} \tag{3.13}$$

Finding a new set of Kalman filter equations that optimally solve these equations will not work in this case, since passing a Gaussian through a nonlinear function results in a probability distribution that is no longer Gaussian. In fact, what the EKF does is to *linearize* the nonlinear equations at the point of the current estimate and then directly employ the linear Kalman filter. Linearize differential equations as $f(\mathbf{x}, \mathbf{u})$ and $h(\mathbf{x})$ means taking partial derivatives of each one (obtaining the Jacobian matrix) to evaluate respectively $\mathbf{F}$ and $\mathbf{H}$ at the point $(\mathbf{x}_{t_{k-1}}, \mathbf{u}_{t_{k-1}})$.

$$\mathbf{F} = \left. \frac{\partial f(\mathbf{x}_t, \mathbf{u}_t)}{\partial \mathbf{x}} \right|_{(\mathbf{x}_{k-1}, \mathbf{u}_{k-1})} \tag{3.14}$$

$$\mathbf{H} = \left. \frac{\partial h(\hat{\mathbf{x}}_t)}{\partial \hat{\mathbf{x}}} \right|_{\mathbf{x}_{k-1}} \tag{3.15}$$

The other dissimilarities with respect to the linear Kalman filter are the different evaluations of the prior state estimation $\hat{\mathbf{x}}_{k|k-1}$ and the system uncertainty $\mathbf{S}$, and the computation of the residual: while in the linear version of the filter the $\mathbf{F}$ and $\mathbf{H}$ matrices are used respectively in the estimation of the state transition and the calculation of the residual, doing so in a linearized model would propagate the natural inaccuracies that always came along with the linearization; therefore, it is preferred to directly use the nonlinear equations $f(\mathbf{x}, \mathbf{u})$ and $h(\mathbf{x})$ to accomplish the task and get more accuracy, possibly exploiting a suitable numerical integration technique such as Euler or Runge-Kutta methods. In addition, two other Jacobian matrices are due: the Jacobian of f with respect to the input noise w, $\mathbf{W}$, and the Jacobian of h with respect to the measurement noise v, $\mathbf{V}$:

$$\mathbf{W} = \left.\frac{\partial f(\mathbf{x}_t, \mathbf{u}_t)}{\partial \mathbf{w}}\right|_{(\mathbf{x}_{k-1}, \mathbf{u}_{k-1})} \tag{3.16}$$

$$\mathbf{V} = \left.\frac{\partial h(\hat{\mathbf{x}}_t)}{\partial \mathbf{v}}\right|_{\mathbf{x}_{k-1}} \tag{3.17}$$

The EKF equations are thus reported in Table 3.2, together with direct comparison with their counterparts in the linear filter:

Linear Kalman Filter	Extended Kalman Filter	
	$\boxed{\mathbf{F} = \left.\frac{\partial f(\mathbf{x}_t, \mathbf{u}_t)}{\partial \mathbf{x}}\right\|_{(\mathbf{x}_{k-1}, \mathbf{u}_{k-1})}}$	$\boxed{\mathbf{W} = \left.\frac{\partial f(\mathbf{x}_t, \mathbf{u}_t)}{\partial \mathbf{w}}\right\|_{(\mathbf{x}_{k-1}, \mathbf{u}_{k-1})}}$
$\hat{\mathbf{x}}_{k\|k-1} = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_k$	$\boxed{\hat{\mathbf{x}}_{k\|k-1} = f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1})}$	
$\hat{\mathbf{P}}_{k\|k-1} = \mathbf{F}\mathbf{P}_{k-1}\mathbf{F}^{\mathrm{T}} + \mathbf{Q}_k$	$\hat{\mathbf{P}}_{k\|k-1} = \mathbf{F}\mathbf{P}_{k-1}\mathbf{F}^{\mathrm{T}} + \mathbf{W}\mathbf{Q}_k\mathbf{W}^{\mathrm{T}}$	
	$\boxed{\mathbf{H} = \left.\frac{\partial h(\mathbf{x}_t)}{\partial \mathbf{x}}\right\|_{\mathbf{x}_{k-1}}}$	$\boxed{\mathbf{V} = \left.\frac{\partial h(\hat{\mathbf{x}}_t)}{\partial \mathbf{v}}\right\|_{\mathbf{x}_{k-1}}}$
$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_{k\|k-1}$	$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \boxed{h(\hat{\mathbf{x}}_{k\|k-1})}$	
$\mathbf{S}_k = \mathbf{H}\hat{\mathbf{P}}_{k\|k-1}\mathbf{H}^{\mathrm{T}} + \mathbf{R}_k$	$\mathbf{S}_k = \mathbf{H}\hat{\mathbf{P}}_{k\|k-1}\mathbf{H}^{\mathrm{T}} + \mathbf{V}\mathbf{R}_k\mathbf{V}^{\mathrm{T}}$	
$\mathbf{K}_k = \hat{\mathbf{P}}_{k\|k-1}\mathbf{H}^{\mathrm{T}}\mathbf{S}_k^{-1}$	$\mathbf{K}_k = \hat{\mathbf{P}}_{k\|k-1}\mathbf{H}^{\mathrm{T}}\mathbf{S}_k^{-1}$	
$\mathbf{x}_{k\|k} = \hat{\mathbf{x}}_{k\|k-1} + \mathbf{K}_k\tilde{\mathbf{y}}_k$	$\mathbf{x}_{k\|k} = \hat{\mathbf{x}}_{k\|k-1} + \mathbf{K}_k\tilde{\mathbf{y}}_k$	
$\mathbf{P}_{k\|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\hat{\mathbf{P}}_{k\|k-1}$	$\mathbf{P}_{k\|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\hat{\mathbf{P}}_{k\|k-1}$	

Table 3.2: Extended Kalman filter equations and linear filter comparison

3.3 Unscented Kalman Filter

It has been seen how the need to deal with nonlinear processes has led to the formulation of a different approach to the modeling with respect to the basic Kalman filter, the one based on the linearization of the problem. That however gives rise to other complications and drawbacks that should be highlighted: the linearization is based on the computation of the Jacobian for the state and the measurement model, which is often non-trivial even with simple models. It is not rare to deal with problems that result in a Jacobian which is difficult or impossible to derive analytically, having to resort to numerical methods which are typically hard to implement and increase the whole complexity; in addition to that, even with all the matrix available, a linearized filter only takes in consideration the state and measurement equations in one point, which sometimes can be a too rough approximation.

To overcome these limitations, the idea of the Kalman filter has been readapted in many ways over the years, experimenting with different modeling approaches that could be compatible with the core math of the Kalman filter. An approach that turned out to be successful was the Unscented Kalman Filter (UKF), and the idea behind it is actually quite logical and straightforward: the EKF has limited accuracy in many cases due to the evaluation of the nonlinear equations in only one point, so what can be helpful is to increase the number of points considered in the estimation; for example, considering to a well-known technique as the Monte Carlo approach, the accuracy that a filter would get using very large amounts of random points to fed to the nonlinear equations and computing the mean and variance of the result would certainly be high, if the points are enough. Obviously, this kind of approach is terribly expensive from the computational burden point of view and exceptionally slow, so the core idea of the UKF is to deterministically choose the points in which to evaluate the functions, to keep the mean and covariance as similar as possible to the ones obtained with many more points.

The points, called *sigma points* $\boldsymbol{\mathcal{X}}$, can be chosen following different strategies, among which the most famous ones are the Julier sigma points algorithm proposed by Simon J. Julier, who was also one of the authors of the UKF along with Jeffrey Uhlmann, and the Van der Merwe Scaled sigma points algorithm, a reformulation of the Julier's algorithm published by Rudolph Van der Merwe in his 2004 Ph.D. dissertation [36], which is also the version research and industry have mostly settled on. Any technique anyway is based on the same core idea: they are chosen to sample a Gaussian distribution, which the state $\mathbf{x}$ and state covariance $\mathbf{P}$ both are, and typically they are in odds numbers so that there is always one sample point at the mean of the inputs and couples of other points that map the input in a symmetrical way. The points are then fed directly to the nonlinear state equation that returns a new distribution from which mean μ and covariance Σ can be extracted, constituting the state and state covariance prior estimations.

$$\boldsymbol{\mathcal{Y}} = f(\boldsymbol{\mathcal{X}}) \qquad \boldsymbol{\mu} = \sum_{i=0}^{2n} w_i^m \boldsymbol{\mathcal{Y}}_i \qquad \boldsymbol{\Sigma} = \sum_{i=0}^{2n} w_i^c (\boldsymbol{\mathcal{Y}}_i - \boldsymbol{\mu})(\boldsymbol{\mathcal{Y}}_i - \boldsymbol{\mu})^{\mathrm{T}} \tag{3.18}$$

where the subscript $_i$ denotes the sigma point and the subscripts ^{m} and ^{c} refer to the weights respectively for the mean and the covariance.

The problem could require weighing differently the sigma points, for instance putting much more belief in the center point than in the ones far from the mean. This applies both to the state and the covariance prior estimation, and the criteria to choose the weight is mostly arbitrary, and again the techniques present in the literature are multiple. In particular, the equations of the Van der Merwe's scaled algorithm for both the sigma points and the weights are presented, being also the ones employed in the implementation of UKF in the experimental work.

The formulation uses 3 parameters to control how the sigma points are distributed and weighed: α, β, and κ. Of the three, α has the more straightforward role: Figure 3.3.1 depicts the same distribution of which two covariance ellipses showing the first and second standard deviations are represented, comparing the effect of α on the points layout.

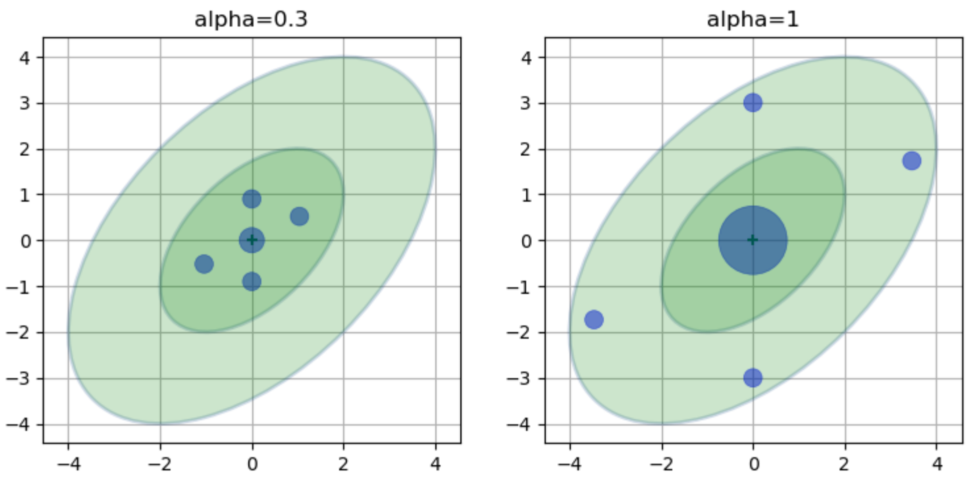


Figure 3.3.1: Effect of α on the sigma points layout [21]

It is evident that the sigma points become more and more spread out with respect to the mean the larger α gets, also weighing more the mean point and less the others. The meaning of it should fit the intuition: the further a point is from the mean, the less it should be considered and so weighed.

The first sigma point is chosen as the mean of the input, while the others are computed with the following equation, which employs the three mentioned parameters:

$$\boldsymbol{\mathcal{X}}_0 = \mu$$
$$\boldsymbol{\mathcal{X}}_i = \begin{cases} \mu + \sqrt{(n+\lambda)\Sigma}\Big|_i & i = 1 \cdots n \\ \mu - \sqrt{(n+\lambda)\Sigma}\Big|_{i-n} & i = (n+1) \cdots 2n \end{cases} \tag{3.19}$$

where λ is a parameter defined for notational convenience as $\lambda = \alpha^2(n+\kappa) - n$, where n is the dimension of the state vector $\mathbf{x}$, and the $_i$ subscript chooses the i-th row of the matrix. So basically the covariance matrix Σ is scaled by a constant, square rooted, and then added or subtracted to the mean μ to ensure the symmetry.

Then the formulation provides one set of weights for the mean and another for the covariance:

$$\begin{aligned}\mathbf{W}_0^m &= \frac{\lambda}{n+\lambda} \\ \mathbf{W}_0^c &= \frac{\lambda}{n+\lambda} + 1 - \alpha^2 + +\beta \\ \mathbf{W}_i^m = \mathbf{W}_i^c &= \frac{1}{2(n+\lambda)}\end{aligned} \tag{3.20}$$

The parameters should be chosen as follows: $0 \leq \alpha \leq 1$, $\beta \geq 0$ to incorporate knowledge of the higher order moments of the distribution ($\beta = 2$ is the optimal value for Gaussian), and $\kappa \geq 0$ to guarantee positive semidefiniteness of the covariance matrix [36]. A reasonable choice for the parameters is the following: α typically a small number to avoid sampling non-local effects, $\kappa = 3 - n$ with n the dimension of $\mathbf{x}$, and as said $\beta = 2$ [21].

With sigma points $\boldsymbol{\mathcal{X}}_{k|k-1}$ and weights so obtained, the transformed points distribution $\boldsymbol{\mathcal{Y}}_{k|k-1}$ can be obtained through the nonlinear state function f and the prior state mean and covariance are computed next using the *Unscented Transform*, whose equations are along the lines of Equation 3.18:

$$\hat{\mathbf{x}}_{k|k-1} = \sum_{i=0}^{2n} w_i^m \boldsymbol{\mathcal{Y}}_{k|k-1,i} \tag{3.21}$$

$$\hat{\mathbf{P}}_{k|k-1} = \sum_{i=0}^{2n} w_i^c (\boldsymbol{\mathcal{Y}}_{k|k-1,i} - \hat{\mathbf{x}}_{k|k-1})(\boldsymbol{\mathcal{Y}}_{k|k-1,i} - \hat{\mathbf{x}}_{k|k-1})^{\mathrm{T}} + \mathbf{Q}_k \tag{3.22}$$

In an analogous way, the measurement estimates distribution $\boldsymbol{\mathcal{Z}}_{k|k-1}$ can be obtained from the transformed sigma points through the nonlinear measurement function h, and the mean $\hat{\mathbf{z}}_{k|k-1}$ and covariance $\mathbf{S}_k$ can be calculated again by means of the unscented transform:

$$\boldsymbol{\mathcal{Z}}_{k|k-1} = h(\boldsymbol{\mathcal{Y}}_{k|k-1}) \tag{3.23}$$

$$\hat{\mathbf{z}}_{k|k-1} = \sum_{i=0}^{2n} w_i^m \boldsymbol{\mathcal{Z}}_{k|k-1,i} \tag{3.24}$$

$$\mathbf{S}_k = \sum_{i=0}^{2n} w_i^c (\boldsymbol{\mathcal{Z}}_{k|k-1,i} - \hat{\mathbf{z}}_{k|k-1})(\boldsymbol{\mathcal{Z}}_{k|k-1,i} - \hat{\mathbf{z}}_{k|k-1})^{\mathrm{T}} + \mathbf{R}_k \tag{3.25}$$

and then the characteristic equations of the Kalman filter can finally be used, with minor alterations. In fact, in this case, a new matrix is defined, the *cross covariance* $\mathbf{P}_{xz,k}$, which relates the uncertainty in the state estimation with the uncertainty in the measurement one, exactly as the expression $\hat{\mathbf{P}}_{k|k-1}\mathbf{H}^{\mathrm{T}}$ did before in the linear filter:

$$\mathbf{P}_{xz,k} = \sum_{i=0}^{2n} w_i^c(\boldsymbol{\mathcal{Y}}_{k|k-1,i} - \hat{\mathbf{x}}_{k|k-1})(\boldsymbol{\mathcal{Z}}_{k|k-1,i} - \hat{\mathbf{z}}_{k|k-1})^{\mathrm{T}} \tag{3.26}$$

which is subsequently used to evaluate the Kalman gain, which is again a factor that compares the belief in the state ($\mathbf{P}_{xz,k}$) with the belief in the measurement ($\mathbf{S}_k$) as:

$$\mathbf{K}_k = \mathbf{P}_{xz,k}\mathbf{S}_k^{-1} \tag{3.27}$$

What is left to do is to compute the residual of the measurement $\mathbf{z}_k$ with respect to the measurement estimate $\hat{\mathbf{z}}_{k|k-1}$:

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \hat{\mathbf{z}}_{k|k-1} \tag{3.28}$$

and then the posterior state $\mathbf{x}_{k|k}$ and covariance $\mathbf{P}_{k|k}$ can finally be evaluated as:

$$\mathbf{x}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\tilde{\mathbf{y}}_k \tag{3.29}$$

$$\mathbf{P}_{k|k} = \hat{\mathbf{P}}_{k|k-1} - \mathbf{K}_k\mathbf{S}_k\mathbf{K}_k^{-1} \tag{3.30}$$

Figure 3.3.2 shows how differently the sampling (Monte Carlo), the EKF, and the UKF accuracies are when dealing with an arbitrary nonlinear function:

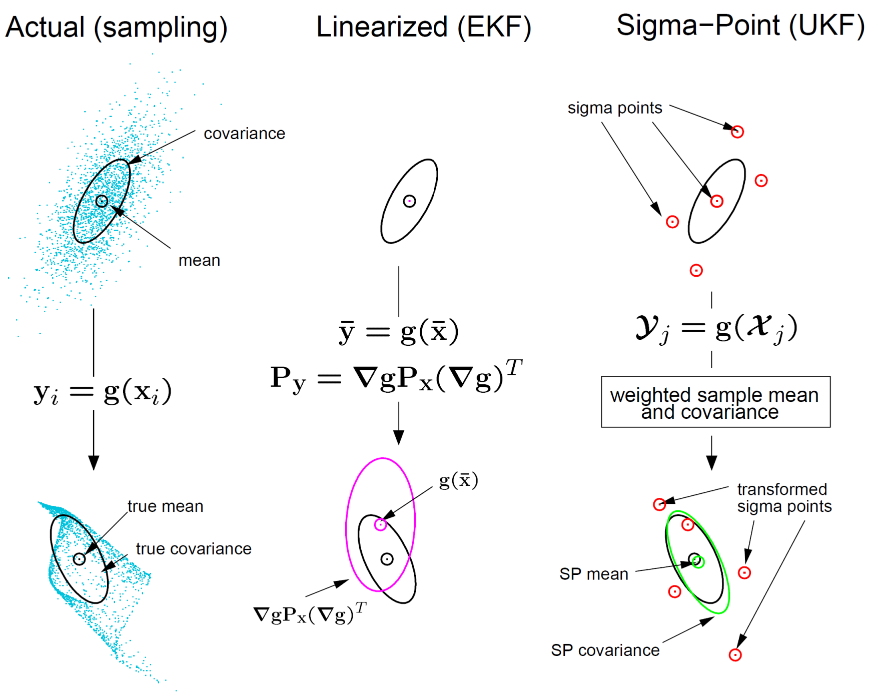

Actual (sampling)
Linearized (EKF)
Sigma−Point (UKF)
covariance
mean
sigma points
$\mathbf{y}_i = \mathbf{g}(\mathbf{x}_i)$
$\bar{\mathbf{y}} = \mathbf{g}(\bar{\mathbf{x}})$
$\mathbf{P}_\mathbf{y} = \nabla\mathbf{g}\mathbf{P}_\mathbf{x}(\nabla\mathbf{g})^T$
$\boldsymbol{\mathcal{Y}}_j = \mathbf{g}(\boldsymbol{\mathcal{X}}_j)$
weighted sample mean and covariance
true mean
true covariance
$\mathbf{g}(\bar{\mathbf{x}})$
$\nabla\mathbf{g}\mathbf{P}_\mathbf{x}(\nabla\mathbf{g})^T$
transformed sigma points
SP mean
SP covariance

Figure 3.3.2: Comparison of the accuracies of the Monte Carlo, EKF and UKF approaches in the mean and covariance propagation through a nonlinear function [36]

The picture is taken directly from Van der Merwe's Ph.D. dissertation: $\mathbf{g}$ corresponds to the state transition function f, so $\boldsymbol{\nabla}\mathbf{g}$ is its Jacobian, previously indicated as $\mathbf{F}$.

All the UKF equations are reported in Table 3.3, again with a direct comparison with the standard linear filter ones:

Linear Kalman Filter	Unscented Kalman Filter
	$\boldsymbol{\mathcal{X}} = \textbf{sigma-function}(\mathbf{x}_{k-1}, \mathbf{P}_{k-1})$
	$\mathbf{W}^m, \mathbf{W}^c = \textbf{weight-function}(\alpha, \beta, \kappa, n)$
	$\boldsymbol{\mathcal{Y}} = f(\boldsymbol{\mathcal{X}})$
$\hat{\mathbf{x}}_{k\|k-1} = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_k$	$\hat{\mathbf{x}}_{k\|k-1} = \sum_{i=0}^{2n} w_i^m \boldsymbol{\mathcal{Y}}_{k\|k-1,i}$
$\hat{\mathbf{P}}_{k\|k-1} = \mathbf{F}\mathbf{P}_{k-1}\mathbf{F}^{\mathrm{T}} + \mathbf{Q}_k$	$\hat{\mathbf{P}}_{k\|k-1} = \sum_{i=0}^{2n} w_i^c (\boldsymbol{\mathcal{Y}}_{k\|k-1,i} - \hat{\mathbf{x}}_{k\|k-1})(\boldsymbol{\mathcal{Y}}_{k\|k-1,i} - \hat{\mathbf{x}}_{k\|k-1})^{\mathrm{T}} + \mathbf{Q}_k$
	$\boldsymbol{\mathcal{Z}}_{k\|k-1} = h(\boldsymbol{\mathcal{Y}}_{k\|k-1})$
	$\hat{\mathbf{z}}_{k\|k-1} = \sum_{i=0}^{2n} w_i^m \boldsymbol{\mathcal{Z}}_{k\|k-1,i}$
$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_{k\|k-1}$	$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \hat{\mathbf{z}}_{k\|k-1}$
$\mathbf{S}_k = \mathbf{H}\hat{\mathbf{P}}_{k\|k-1}\mathbf{H}^{\mathrm{T}} + \mathbf{R}_k$	$\mathbf{S}_k = \sum_{i=0}^{2n} w_i^c (\boldsymbol{\mathcal{Z}}_{k\|k-1,i} - \hat{\mathbf{z}}_{k\|k-1})(\boldsymbol{\mathcal{Z}}_{k\|k-1,i} - \hat{\mathbf{z}}_{k\|k-1})^{\mathrm{T}} + \mathbf{R}_k$
	$\mathbf{P}_{xz,k} = \sum_{i=0}^{2n} w_i^c (\boldsymbol{\mathcal{Y}}_{k\|k-1,i} - \hat{\mathbf{x}}_{k\|k-1})(\boldsymbol{\mathcal{Z}}_{k\|k-1,i} - \hat{\mathbf{z}}_{k\|k-1})^{\mathrm{T}}$
$\mathbf{K}_k = \hat{\mathbf{P}}_{k\|k-1}\mathbf{H}^{\mathrm{T}}\mathbf{S}_k^{-1}$	$\mathbf{K}_k = \mathbf{P}_{xz,k}\mathbf{S}_k^{-1}$
$\mathbf{x}_{k\|k} = \hat{\mathbf{x}}_{k\|k-1} + \mathbf{K}_k\tilde{\mathbf{y}}_k$	$\mathbf{x}_{k\|k} = \hat{\mathbf{x}}_{k\|k-1} + \mathbf{K}_k\tilde{\mathbf{y}}_k$
$\mathbf{P}_{k\|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\hat{\mathbf{P}}_{k\|k-1}$	$\mathbf{P}_{k\|k} = \hat{\mathbf{P}}_{k\|k-1} - \mathbf{K}_k\mathbf{S}_k\mathbf{K}_k^{-1}$

Table 3.3: Unscented Kalman filter equations and linear filter comparison

The advantages of the UKF over the EKF are multiple: when it comes to the modeling of the real-world process, the EKF poses hard challenges to solve in order to be able to use it, such as the computation of the Jacobian matrices; on the other hand, the UKF does not require any type of linearization and is much simpler in terms of modeling, since its nature is to exploit the characteristics of distributions to estimate the propagation of means and covariances, and so it is able to deal with processes that can be highly nonlinear and that the EKF cannot handle. In addition to that, the UKFs present in literature are often proven to perform better that the EKFs, and it is foreseeable since the latter limits its scope to a single point, while the UKF acts on multiple points. Finally, the UKF can also be tuned through the choice of the sigma points, which is, in turn, dependent on the choice of the α, β, and κ parameters, almost arbitrarily selected by the user.

3.4 Additional Features

All the different variations of the Kalman filter just seen have some flaws, as every modeling and prediction tool does: apart from the linearization problem already discussed for the EKF, all three of them share some weaknesses that directly depend on the non-idealities characteristics of real-world processes.

The Kalman filters require an initial condition to start their processing and provide a first estimation, and that can only be arbitrarily chosen by the user; on the basis of the discordance of the initial guess from the true state, the filters can even diverge in a small amount of time. This behavior can be caused also by other factors, such as the non-Gaussianity of the measurement or process noise, which depending on the actual problem of interest could also occur. The noise can also happen to be non-additive, as instead the Kalman filters model it, and that may also contribute to faulty estimations or directly the divergence.

Among all the sources of errors in Kalman filters, the one whose effects are the most visible and most of all occurs in nearly all kinds of real-world problems, is certainly the presence of outliers in the measurements. An outlier is a measurement value that largely differs from the expected or typical value at that moment, so a point that is placed on the tails of the distribution; outliers are actually frequent in real processes, because every type of measurement system is subject to variations in temperature, pressure, humidity, light conditions and so on, that even in small in magnitude can cause even very different results.
Because of the way the Kalman filters are designed, the outliers are taken into account in the estimates, causing the filters to underestimate or overestimate the true state of the system, which can also lead to the divergence if the deviation from the expected value or the outlier occurrence rate are too high.

To deal with this flaw, some variations to the EKF and UKF have been proposed in the past years, and they all rely on a robust filtering technique, such as the Huber filter, the Student's T filter, and the M-estimation filter, or on the increase of outlier handling capacity of the Kalman filter through specific adaptiveness features that the matrices are provided to. This second group of filters is designed with the idea of adapting the covariance matrices according to the obtained measurements in order to accommodate the presence of outliers by increasing the uncertainty about them, privileging the ones that are close to the expected value. Actually, the equations of the Kalman filters of the previous sections have been reported in a specific way precisely to include the possibility of adaptiveness; in fact, one thing that is worth being highlighted is the notation used for the matrices and vectors: among the right-hand side matrices, the only ones that are reported with time steps subscripts, apart from those that are explicitly updated in the equations discussed, are the noise matrices $\mathbf{Q}_k$ and $\mathbf{R}_k$. This has been made exactly to highlight the fact that even if they are initially provided by the user, according to the characterization of the properties of the process model and the actual sensors used for the measurement, nothing forbids altering them on the basis of the specific data of each time step to provide the filter of an adaptability feature.

Nearly all the proposed modifications of the Kalman filters that can be found in the literature are based on the alteration of specific equations and/or matrices, made upon some kind of metrics that are able to provide statistical information about the goodness of a measurement. Among them, one of the most used metrics is the so-called *Mahalanobis distance*: it was first proposed by the Indian statistician P.C. Mahalanobis as a measure of the distance of a point from a distribution, and it can be seen as a multidimensional generalization of the idea of measuring how many standard deviations away the point is from the mean of the distribution. It is defined as:

$$MD(\mathbf{x}, \mathbf{Q}) = \sqrt{(\mathbf{x} - \boldsymbol{\mu}_{\mathbf{Q}})^{\mathrm{T}} \boldsymbol{\Sigma}_{\mathbf{Q}}^{-1} (\mathbf{x} - \boldsymbol{\mu}_{\mathbf{Q}})} \tag{3.31}$$

where $\mathbf{x}$ and $\mathbf{Q}$ are the multidimensional point and distribution of interest and $\boldsymbol{\mu}_{\mathbf{Q}}$ and $\boldsymbol{\Sigma}_{\mathbf{Q}}$ are respectively the mean and the positive-definite covariance matrix of the distribution $\mathbf{Q}$.

From the theory of hypothesis testing, it can be verified that the Mahalanobis distance squared applied to a Gaussian distribution obeys to the χ^2 distribution with n degrees of freedom, where n is the number of dimensions of the normal distribution. It means that supposing a significance level α ranging from 0 to 1, $MD(\mathbf{x}, \mathbf{Q})^2$ has just an α probability of being higher of a certain threshold $\chi^2_{n,\alpha}$ which is tabulated and dependent on α and the degree of freedom n of the distribution. This property can be usefully exploited to obtain a piece of statistical information about the trust level that should be put in a certain measurement, which has so to exhibit a Mahalanobis distance squared smaller than the threshold value.

Among the techniques that realize adaptive filtering by making use of the Mahalanobis distance, here it is chosen to report the three of them which have also been implemented in the Kalman filters designed in the experimental work, showing interesting results.

The first technique described is the simplest one to use when the Mahalanobis distance is involved, the thresholding technique. It is as straightforward as the name suggests: a measurement can be either retained or discarded based on the information provided by the Mahalanobis distance, which, instead of considering the whole measurement vector, is computed measurement by measurement in this case, as it is also done in other proposed techniques and in some of the ones that will be discussed later. The formula used is a direct derivation of the multidimensional case, treating the multiplication of a vector by its transpose as the matrix way to compute its square and the inverse of a matrix as a way to make a ratio out of it:

$$MD(\mathbf{z}_k, \mathbf{S}_k)_i = \sqrt{\frac{(\mathbf{z}_{k,i} - \hat{\mathbf{z}}_{k,i})^2}{\mathbf{S}_{k_{i,i}}}} \tag{3.32}$$

where $\mathbf{S}_{k_{i,i}}$ is i-th element of the diagonal of the measurement uncertainty matrix corresponding to the i-th measurement $\mathbf{z}_{k,i}$ and $\hat{\mathbf{z}}_{k,i}$ is the i-th measurement estimate.

The corresponding degree of freedom is then reduced to one, since the measurements are treated value by value, and the significance level can be chosen arbitrarily, although a significance level of 0.05 is often used as the cutoff between significant and non-significant results, which in this case results in a $\chi^2_{1,0.05}$ threshold value of 3.841, according to the χ^2 tables.
Then the algorithm simply cuts off each measurement that results greater than the threshold, actually reducing the dimension of the measurement vector. This clearly implies a collective reduction of dimension for all the matrices that directly involve the measurement vector in their calculations ($\mathbf{S}_k$ and $\mathbf{K}_k$ for the EKF, to which $\mathbf{P}_{xz,k}$ is added in the case of UKF), effectively downscaling the order of the system. The technique is effective because of its simplicity, allowing the filter not to be affected by the outliers' influence without a real increase in complexity; its drawback on the other hand is again its unsophistication, since it completely ignores the data of some sensors and makes the filter rely only on a subset of them, leading to a loss of accuracy in some cases.

A more advanced approach to outlier detection is the one proposed in 2007 by Ting, Theodorou, and Schaal [34] in 2007, whose advantage is the lack of need for manual parameter tuning by the user, which is often required in other proposed Kalman filter variations. It makes use of a weight w_k applied to the measurement noise matrix $\mathbf{R}_k$ in order to privilege the most reliable measurements by associating them with a higher weight, while down-weighing the ones that turn out to be too far from the expected value. The equations of the "Weighted Robust" Kalman filter, as it is called by the authors, are the following:

Prediction:

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_k \tag{3.33}$$

$$\hat{\mathbf{P}}_{k|k-1} = \mathbf{Q}_k \tag{3.34}$$

Update:

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_{k|k-1} \tag{3.35}$$

$$\mathbf{S}_k = \mathbf{H}\hat{\mathbf{P}}_{k|k-1}\mathbf{H}^{\mathrm{T}} + \frac{1}{w_k}\mathbf{R}_k \tag{3.36}$$

$$\mathbf{K}_k = \hat{\mathbf{P}}_{k|k-1}\mathbf{H}^{\mathrm{T}}\mathbf{S}_k^{-1} \tag{3.37}$$

$$\mathbf{x}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\tilde{\mathbf{y}}_k \tag{3.38}$$

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\hat{\mathbf{P}}_{k|k-1} \tag{3.39}$$

and the two differences with the general Kalman filter equations are in $\hat{\mathbf{P}}_{k|k-1}$ and $\mathbf{S}_k$ matrices: for the state covariance matrix $\hat{\mathbf{P}}_{k|k-1}$, this filter eliminates its explicit dependency on $\hat{\mathbf{P}}_{k-1}$ by imposing that at each timestep the measurement noise matrix is set equal to the process noise matrix $\mathbf{Q}_k$; For $\mathbf{S}_k$ instead, it performs a rescaling of the measurement noise matrix $\mathbf{R}_k$ at each timestep through the weight w_k defined as:

$$w_k = \frac{a_{w_k} + \frac{1}{2}}{b_{w_k} + (\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_{k|k-1})^{\mathrm{T}}\mathbf{R}_k^{-1}(\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_{k|k-1})} \tag{3.40}$$

where it is evident the use of the squared Mahalanobis distance at the denominator, which has the effect of lessening the weight given to a measurement too distant from the expected value. The parameters a_{w_k} and b_{w_k} are prior scale parameters for the weight w_k: they have to be selected so that the initial average weight is equal to 1 with some confidence, meaning that the algorithm starts assuming that most measurement data are inliers. That can be achieved with a choice of a_{w_k} and b_{w_k} both equal to 1, so that the maximum value of w_k is capped at 1.5. It is worth noticing that differently from the thresholding technique, the Weighted Robust Kalman filter does never cut off completely an outlier measurement, instead it makes it less influent by increasing its related uncertainty. The computational complexity of this filter is in the same order as that of the standard Kalman filter.

Another technique that performs well in presence of outliers, for example in NLoS condition in the positioning field, is the one proposed by Li, Wang, and Khoshelham [22] in 2018. It is very similar to the previously discussed one, but it performs the rescaling of the $\mathbf{R}_k$ matrix in another way and also extends the procedure to the state covariance matrix $\mathbf{P}_{k|k-1}$. The algorithm computes the Mahalanobis distance $MD_{k,i}$ for each measurement $\mathbf{z_{k,i}}$ as for the thresholding technique, but then the scaling factor applied to the measurement noise matrix is the ratio of $MD_{k,i}$ and the threshold value of the χ^2 distribution, so that:

$$\mathbf{R}_{k_{i,i}} = \begin{cases} \mathbf{R}_{0_{i,i}} & MD_{k,i} < \chi^2_{1,\alpha} \\ \left(\frac{MD_{k,i}}{\chi^2_{1,\alpha}}\right) \cdot \mathbf{R}_{0_{i,i}} & MD_{k,i} \geq \chi^2_{1,\alpha} \end{cases} \tag{3.41}$$

where $\mathbf{R}_{0_{i,i}}$ indicates the i-th element of the diagonal of the initial user-provided $\mathbf{R}$ matrix, so the one at time zero, and $\chi^2_{1,\alpha}$ indicates the threshold value of the χ^2 distribution with one DoF and significance value α.
In an analogous way, the authors propose an adaptation of the state covariance matrix $\hat{\mathbf{P}}_{k|k-1}$ to overcome abnormal system states, when instead the measurements are correct, which can be caused by sudden variations of the state, unknown biases or even incorrect knowledge of process statistics:

$$\hat{\mathbf{P}}'_{k|k-1} = \begin{cases} \hat{\mathbf{P}}_{k|k-1} & MD_{k,i} < \chi^2_{1,\alpha} \\ \left(\frac{MD_{k,i}}{\chi^2_{1,\alpha}}\right) \cdot \hat{\mathbf{P}}_{k|k-1} & MD_{k,i} \geq \chi^2_{1,\alpha} \end{cases} \tag{3.42}$$

observing that the Mahlanobis distance of a single measurement causes the whole $\hat{\mathbf{P}}_{k|k-1}$ matrix to adapt. To detect abnormal system states, the aforementioned paper introduces a robustness factor that employs knowledge about the sensors' error behavior to discover if the actual error comes from a wrong measurement or a wrong process model. This computation surely moderately increases the complexity of the filter, but the adaptiveness equations introduce only a small overhead per se.

The third and last technique that will be reported is the one discussed by Echeverri, Medeiros et al. [14] in 2018. This approach is not dissimilar to the previous one, but its advantage, mainly with respect to the thresholding technique, is to redefine the $\mathbf{R}_k$ matrix by means of a sigmoid function to allow for smoother transitions between the various weights. In particular, the employed equation is the following:

$$w_{k,i} = \frac{1}{1 + e^{-(MD_{k,i} - \xi)}} \tag{3.43}$$

where $MD_{k,i}$ indicates the Mahalanobis distance of the single measurement $\mathbf{z}_{k,i}$ computed as in Equation 3.32, and ξ is the analogue of the $\chi^2_{1,\alpha}$ described before. Then, these weights are used to redefine the measurement noise matrix as follows:

$$\mathbf{R}_k = \mathbf{\Delta}\mathbf{w}_k \tag{3.44}$$

where $\mathbf{\Delta}$ is defined as the diagonal matrix that reflects the a priori knowledge of the errors of the various sensors, so the resulting matrix $\mathbf{R}_k$ is a diagonal matrix as well. This allows the user to furtherly scale the uncertainty of a single sensor more or less than the others if their error trends are known to be different; if otherwise the sensors behave similarly and are all equally modeled, $\mathbf{\Delta}$ is simply the identity matrix, and $\mathbf{R}_k$ is equal to $diag(\mathbf{w}_k)$. In the paper the technique is applied in a slightly different way, since the sensors they used were of different types and organized in clusters, each one having its own Kalman filter, and then a final filter was in charge of the sensor fusion; so, the Mahalanobis distance was applied in the vector sense as in Equation 3.31 and the $\mathbf{\Delta}$ matrix was implemented in the final filter to reflected the knowledge about each cluster, along with other scaling factors derived from other methods specific of the application.

Part II

Case Study: Rover localization inside a Martian ground simulation facility

Chapter 4

State of the Art

Accomplishing the task of positioning is surely one of the most important requirements when dealing with mobile robotics. If the use case is related to open-air activities, the various available GNSSs have a predominant role in providing a reliable estimation of the global position of a robot, but they cannot be so useful in indoor activities, due to all the limitations that have been discussed. Even in outdoor environments, there are not always the ideal conditions for the satellite systems to work as they should, especially when the area of action is very large and so the environmental conditions such as the extension of the vegetation do change from one point to another. Another challenging case is when the area of interest is not defined a priori, and so the environmental circumstances are little to completely unknown; this happens when the robot is demanded to perform a first exploration of the surroundings, so chances are that in many zones there would be no coverage of satellite signals.

To overcome the limitations present in GNSSs-challenging or denied environments, many approaches have been followed through the years and are well documented in the literature. Most of them make use of inertial measurements and cameras to analyze the surrounding environment up to a certain radius from the robot: Visual Odometry (VO) involves Inertial Measurement Units (IMU) to report a body's specific force, angular rate, and occasionally the orientation of the body and compare this information with the deviation of motion present in the sequences of images captured by on-board cameras; Simultaneous Localization And Mapping (SLAM), instead, relies on a live map that is incrementally-built by the algorithm during the exploration, and then compares the previously acquired data with the current information to keep track of its position within the map.
The advantages of the use of IMUs and cameras certainly are their low weight and low power requirement, but they also experience drawbacks when the light conditions or the reflective properties of the environment change (for the cameras) or in general when the navigation is performed for a long time.

When it comes to space exploration, though, the main limitation in performing localization is the total lack of global positioning systems and the impossibility to rely on magnetometers, which are extremely useful to acquire orientation data. The case study in particular is focused on handling the positioning task in a particular space

setup: with the constant increase of interest in space exploration, it is not temporally remote the day in which mankind will be able to realize the first extraterrestrial outpost on the Lunar or Martian soil; this will pose a whole range of challenges in building a network able to provide information to astronauts about their surroundings and the precise location of instrumentation, vehicles, or any other targets of interest.
With this goal in mind, an excellent solution would be the employment of preinstalled, fixed radiofrequency anchors: this solution would perform better than other approaches since it is able to fully exploit the characteristics of a structured environment, and the fact that its aim would not be to perform a first exploration of the environment, so it could make full use of pre-existent, reliable knowledge of known surroundings. If the user is able to provide the system with precise information about the positions of the anchors, a technology such as UWB would be ideal to be employed because of all the advantages in terms of accuracy, fidelity, low cost, and robustness.

Dardari et al. [11] developed in 2020 a real-time locating system (RTLS) with ultra-low power UWB tags within the project "LOST" (Localization of Objects in Space through RF Tags) funded by ESA; their purpose was to investigate suitable technologies to localize objects deployed or floating inside the International Space Station or future space stations, aimed at tracking every tagged object present in the environment to avoid potentially dangerous situations and to allow astronauts not to waste their extremely valuable time searching for lost tools. The characteristics of the system were high accuracy (close to 1 cm) and medium coverage (a few tens of meters) but the most important was the battery-less UWB tags: in fact, the system exploited the UHF technology to deliver power to the tags, which then utilized UWB pulses to perform the localization through fixed reference nodes. They performed a wide test campaign of 2D localization in a room of size $10 \times 7\,\mathrm{m}^2$, using a TOTAL laser station as ground truth, obtaining average errors of 4 cm and discovering good robustness to metallic obstacles with diameters of tens of centimeters. They also performed a final demonstrator in ESA/ESTEC RObotic Labs by mounting the developed system on the Mars Rover prototype, observing quite large errors in the z-direction because of the physical area constraints, an error below 3 cm on the x-y plane and an overall localization error which remained below the 10 cm in 95% of cases.

Another interesting project was developed by Ni and Barton [28] in 2005. The system they designed at NASA Johnson Space Center (JSC) aimed to provide aid in inspection around the IIS and the Space Shuttle through a free-flying video camera known as Mini-AERCam (Autonomous Extra-vehicular Robotic Camera) equipped with UWB technology to implement the tracking system. They made use of TDoA measurements to develop a 2D tracking system that reached accuracies of tens of centimeters within a planar orbit of a 100 m radius. Their UWB test setup was designed to avoid the receivers' synchronization problem by using four transmitter antennas and only one receiver, which could scan four delayed versions of the received signal from as many antennas within a time window less than 100 ns long; the tracking resolution obtained was less than 1 foot in a 15 $\times$ 15 squared feet lab environment.

In the same year, Ni et al. [26] continued the efforts of the previous project re-adapting it to the tracking need of the SCOUT vehicle performing outdoor tests near the Meteor Crater in Arizona. They obtained simulation results carried out at a range of 610 m which showed an average tracking error of below 3 m, less than 0.5% of the tracking range; the outdoor tracking performance was encouraging as well, being less than 1% at ranges up to 2000 feet.

What is closest to the project developed in this thesis is the work carried out in 2010 by Ni, Arndt et al. [27], who designed a positioning system for a robotic control system in the "Moonyard" facility at Honeywell Defense & System in Arizona. They designed a UWB TDoA high-resolution 3D proximity tracking system for the localization of rovers and astronauts during early exploration missions when satellite navigation systems are not available. Simulations were performed utilizing five receivers uniformly distributed in a given squared space of approximately 7 m side in an optimum configuration called "Twisted Square Configuration", where a receiver is placed in the origin of the reference system, and the remaining four lay on the vertices of a "Twisted Square" (two on the x-z plane and two in the y-z plane) at a distance from the origin defined as the radius of the configuration baseline; they observed a small stable error when the target was located inside the baseline and an exponentially increasing error outside of it.
For the field test, they kept the "One-Receiver-Multiple-Antennas" design to avoid the degradation in time resolution introduced by synchronization errors among receivers, obtaining the time-delayed versions of the UWB pulses as in the previous work, keeping the Twisted Square Configuration" adopted in the simulations; the center antenna was set at height of 4 feet, while the other four were set at 2 and 6 feet in pairs, at a radius of 20 feet. They experienced average tracking accuracy of less than 1 inch in x and y-directions, while on the z-axis it exceeded 2 inches; they conjectured that the tracking accuracy degradation was due to the limited span of the z-dimension: in fact, further simulations performed at an increased span in the z-direction (40 feet) demonstrated the alignment of the vertical tracking error with the one exhibited in the x-y plane.

Chapter 5

Experimental Work

This work aims to offer a low-power, low-cost, modular, easily scalable solution to provide an extraterrestrial outpost with a local positioning system; it is also conceived, though, to serve as ground truth for the research activities of TAS-I Robotics, offering a reliable and precise position reference to aid the development of different mapping and exploration algorithms without interfering with all the RF communication signals already present. The chapter will go through a brief description of the RoXY facility and the equipment used during the field tests and the environment setup; it will follow a detailed discussion of the UWB sensors employed in the system, mentioning their functioning algorithm and the hardware developed to deliver the final setup, including the power adapter PCB and the 3D-printed outdoor detachable case. Subsequently, the ROS2 framework will be presented along with the main nodes developed to implement the positioning algorithm and to integrate it within the rover system; furthermore, the simulation environment will be detailed, providing an overview of the preexisting tools and the developed ones that have been used to test and tune the algorithms, as well as the obtained results. Finally, it will be described the UWB antennas' calibration strategy and the carrying out of the field tests, their results, and a comparison with the simulations.

5.1 TAS-I Torino RoXY setup and equipment

The whole system has been developed in such a way as to fully exploit the properties of the Rover eXploration facilitY (RoXY) at TAS-I Turin: the latter, as visible in Figure 5.1.1, is a technological area dedicated to robotic systems design, development, validation, and verification. It covers an area of about $700\,\mathrm{m}^2$, including a Mars playground, a control room, and a workshop. The outdoor playground extends to an area of $22 \times 26\,\mathrm{m}^2$ that reproduces Mars-like planetary morphology in terms of color, landscape, boulders, smaller rocks, and slopes. The control room hosts the software development, validation, and verification infrastructure, including CC cameras and communication antennas, while the workshop provides a secure area where to store robots and materials, and also to perform integration, tests, and maintenance activities. The perimeter is surrounded by uniform background walls, which isolate the terrain from external interferences like people and vehicles.

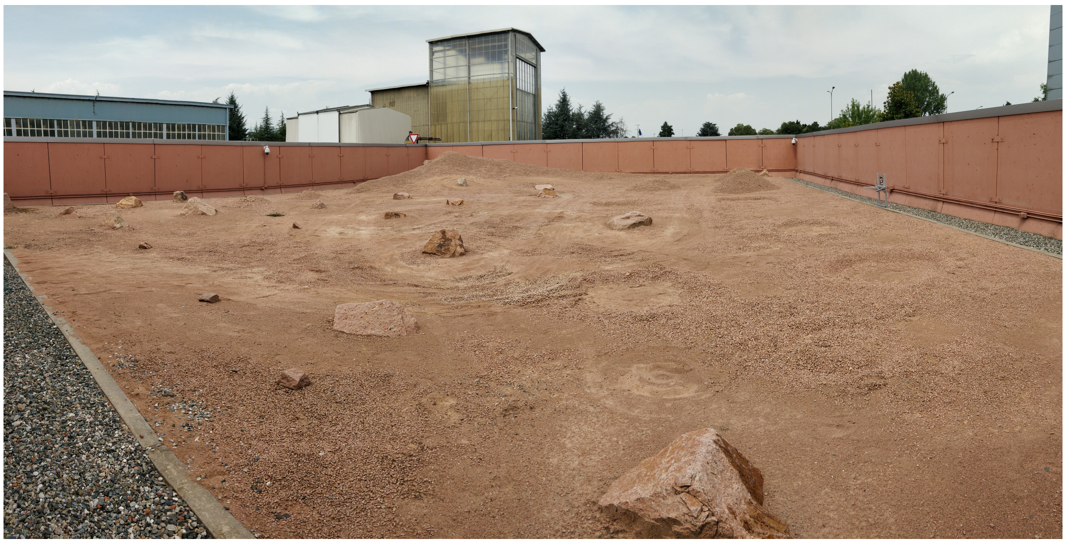
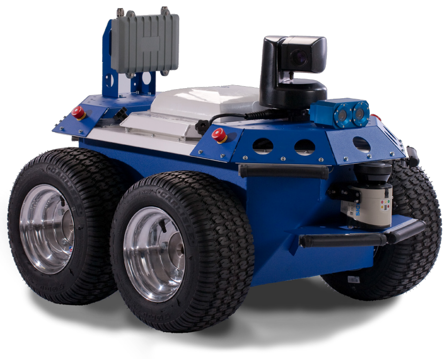


Figure 5.1.1: Perspective view of Rover eXploration facilitY at TAS-I Turin

Moreover, the workshop office box can be transported via truck to be relocated to the field to provide on-site logistics during field test campaigns. The facility is equipped with different power supply rails that run all along the perimeter walls: besides a 220 V rail with standard sockets, there also are 48 V and 24 V AC rails, and the latter has been specifically employed as the power delivery bus chosen for this project. The following subsections report a brief description and a specification table for each significative tool and equipment that has been used, leaving aside everything that concerns the UWB anchors, which will be presented in detail in section 5.2.

5.1.1 Adept MobileRobots Seekur Jr Rover

Seekur Jr by Adept MobileRobots is an all-terrain, all-weather four-wheel skid-steer robot meant for laboratory experimentation or outdoor operation. The actual rover in the RoXY facility has been heavily customized with respect to the stock one, retrofitting it with a modern suite of on-board hardware and keeping only the mechanical parts: it has been provided with an Intel i7 on Versalogic VL-EBXe-41SJF Copperhead primary OBC, and an Nvidia Jetson Xavier NX as secondary OBC, also embedding a LucidLabs Helios2 Time-of-Flight (ToF) camera and a StereoLabs Zed2 stereocamera with a built-in IMU, used for visual odometry and autonomous navigation. Its core software has been totally rewritten to implement a wide ROS2 system to organically control locomotion, cloud point processing, and sensor fusion. It can be controlled remotely with a joypad or laptop, with position goal auto-reaching. Its versatility makes it an ideal solution for R&D purposes, enabling easy sensors integration and algorithm testing without the additional complications that arise with real space exploration rovers. For the purpose of this project, a custom stand has been modeled and 3D-printed to be attached to the top of the rover, in its center, in order to support the UWB target tag. The technical specs of the standard commercial Seekur Jr are reported in Table 5.1.


(a)

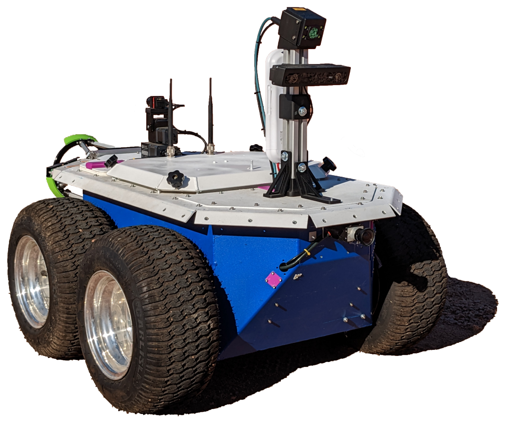

(b)

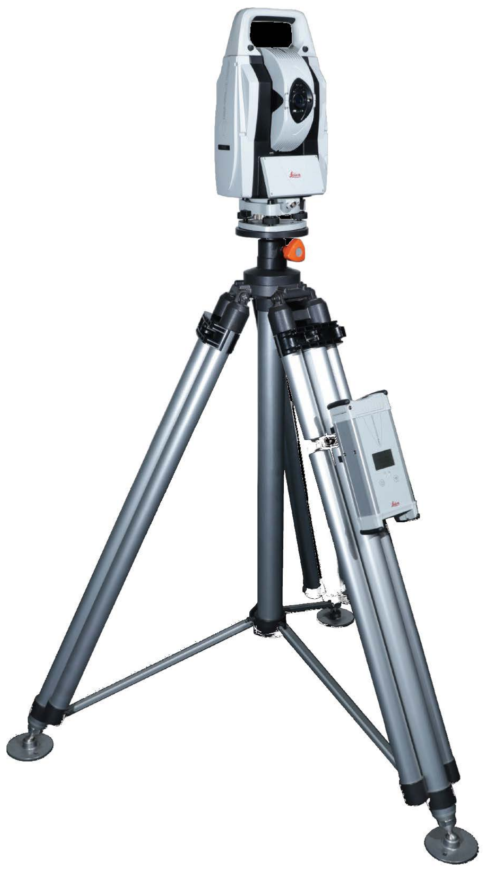
Figure 5.1.2: Adept MobileRobots Seekur Jr standard version **(a)** and RoXY Lab customized version **(b)**

Dimensions	1151 mm × 835 mm × 494 mm
Weight + Payload	77 kg + 40 kg
Body material & IP ratings	Powder-coated aluminum IP-54
Tires	16 inch pneumatic
Max forward/backward speed	1.2 m/s
Turn & Swing radius	0 cm, 52 cm
Run time	3 hours
Operating temperature	(-5°C , +35°C)

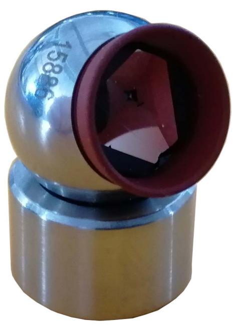
Table 5.1: Adept MobileRobots Seekur Jr technical specifications

5.1.2 Leica Absolute Tracker AT403

Leica Absolute Tracker AT403 is a portable laser tracker with continuous measurements and reflector tracking features. It allows a precise and trustworthy ground truth for the anchors' location, which has been exploited in both anchors' calibration and final placement, employing prisms with extremely precise dimensions and offsets. Measurements are gathered by means of SpatialAnalyzer® software, which also enables a quick and simple generation of reports and ASCII files that can be used for data analysis and post-processing, such as the creation of planes, the directional removal of offsets and the table of reciprocal distances. The laser tracker's technical specifications are listed in Table 5.2.



(a)



(b)

Figure 5.1.3: Leica AT430 mounted on a tripod with its control unit **(a)** and the 0.5” prism used in anchors’ placement **(b)**

Dimensions (L x W x H)	290 mm × 221 mm × 188 mm
Weight	7.3 kg
Measurement angle	Horizontal: ±360°, Vertical: ±145°
Accuracy	±15 µm + 6 µm/m
Laser class	2
Laser type	635 nm, < 1 mW
Operating temperature	(-15°C , +45°C)
Operating relative humidity	< 95%

Table 5.2: Leica AT403 technical specifications

5.1.3 Bosch GLM 250 VF Professional Laser Measure

Bosch GLM 250 VF Professional is a portable laser tool able to perform a wide range of measurements, including standard length, surface, volume, and indirect height. It is provided with a wide optical lens and a bubble level, and ensures built-in dust and splash water protection, making it ideal for outdoor environments. It has been employed during field tests to measure single-point positions to be compared with the result provided by the UWB system. Technical specifications follow in Table 5.3.

Figure 5.1.4: Bosch GLM 250 VF Professional Laser Measure

Dimensions (L x W x H)	120 mm × 66 mm × 37 mm
Measurement range	(0.05 , 250) m
Measurement time	Typ: < 0.5 s, Max: 4 s
Accuracy	Typ: $\pm$ 1 mm
Laser class	2
Laser type	635 nm, $<$ 1 mW
Dust and splash protection	IP-54

Table 5.3: Bosch GLM 250 VF Professional technical specifications

5.1.4 Advanced Navigation Spatial GNSS/INS

Spatial is a ruggedized miniature GNSS-aided Inertial Navigation System (INS) and Attitude and Heading Reference System (AHRS) that provides accurate position, velocity, acceleration, and orientation under the most demanding conditions. It combines temperature-calibrated accelerometers, gyroscopes, magnetometers, and a pressure sensor with an advanced GNSS receiver that supports GPS, Galileo, GLONASS and BeiDou. The sensors are coupled in a sophisticated AI neural network fusion algorithm to deliver accurate and trustworthy navigation and orientation; it also supports Real-Time Kinematics (RTK) technology for real-time 2 cm position accuracy. It has been exploited as a reliable benchmark for the UWB positioning system to be compared to. Its technical specs are listed in Table 5.4.

Figure 5.1.5: Advanced Navigation Spatial GNSS/INS

Dimensions (L x W x H)	40 mm × 30 mm × 24 mm
Horizontal accuracy	2 cm (with L1 RTK)
Vertical accuracy	3 cm (with L1 RTK)
Update rate	10 Hz
Communication interface	RS232
Environmental protection	IP67 & MIL-STD-810G

Table 5.4: Advanced Navigation Spatial GNSS/INS technical specifications

5.2 UWB anchors

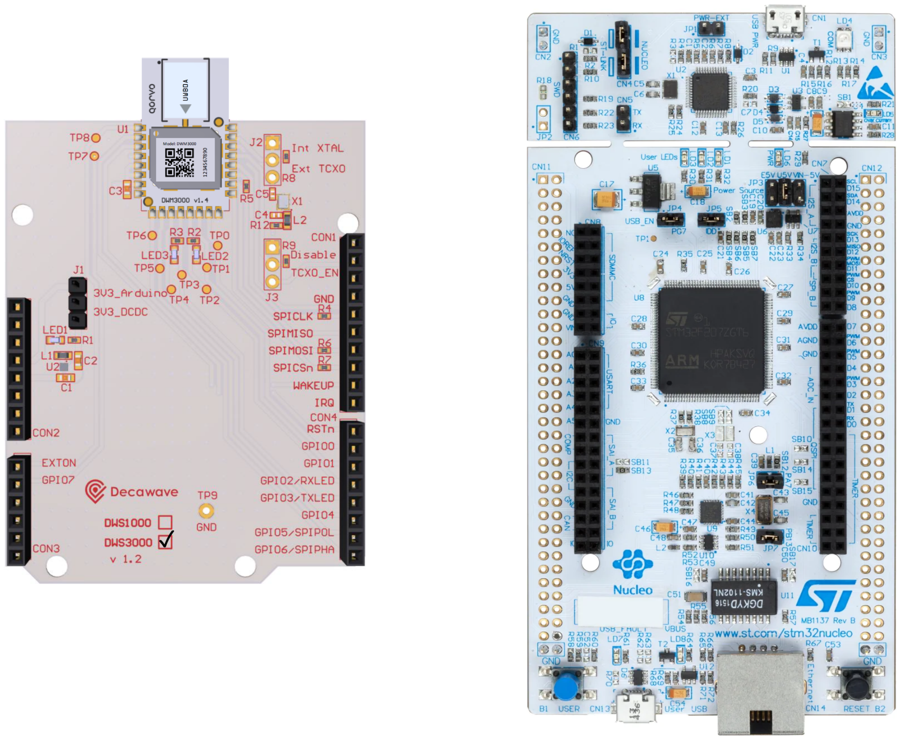
5.2.1 Qorvo Decawave DWM3000EVB + STM32 Nucleo-F429ZI

The localization system designed for this thesis relies on UWB anchors composed of a microcontroller and a UWB antenna peripheral. The choice for the latter fell on the Qorvo ® (Decawave) DWM3000 Evaluation Board, an evolution from the previous DWM1000 model that has been optimized in terms of power consumption, SPI speed, and cost. One of its drawbacks is the limited API compatibility: despite being provided with an Arduino-compatible shield, the API supplied by the manufacturer, and at the time being the only one that constitutes a reliable and complete solution, has been designed so that the compatibility is ensured only for a small set of microcontrollers; among them, the STMicroelectronics Nucleo-F429ZI development board has been chosen as the MCU for the UWB peripheral, to keep the system as low cost as possible as well as because of previous experiences with STM boards.

The DWM3000EVB is an Arduino form-factor compatible shield designed for the evaluation of the DWM3000 UWB transceiver module, which integrates the antenna, power management and clock circuitry in one low-power component, simplifying the integration. It can be used in two-way ranging or TDoA location systems to locate assets to a precision of 10 cm and supports data rates of up to 6.8 Mbps; it supports both UWB Channels 5 and 9, allows programming the transmitter output power and it is fully compliant with FCC & ETSI UWB spectral masks. Table 5.5 sums up the main technical characteristics.

Dimensions (L x W x H)	78 mm × 53 mm × 14 mm
Operating bands	UWB Channel 5 (6.5 GHz) and 9 (8 GHz)
Data rate	6.8 MBps (IEEE 802.15.4-2015 compliant)
Spectral density	-41.3 dBm/MHz (Max)
UWB antenna	Qorvo® DW3110 IC
Ranging technique	SS-TWR, DS-TWR, TDoA
Ranging precision	10 cm (2D)

Table 5.5: Qorvo ® DWM3000 Evaluation Board technical specifications



(a) (b)

Figure 5.2.1: Qorvo ® DWM3000 Evaluation Board **(a)** and STMicroelectronics Nucleo-F429ZI development board **(b)**

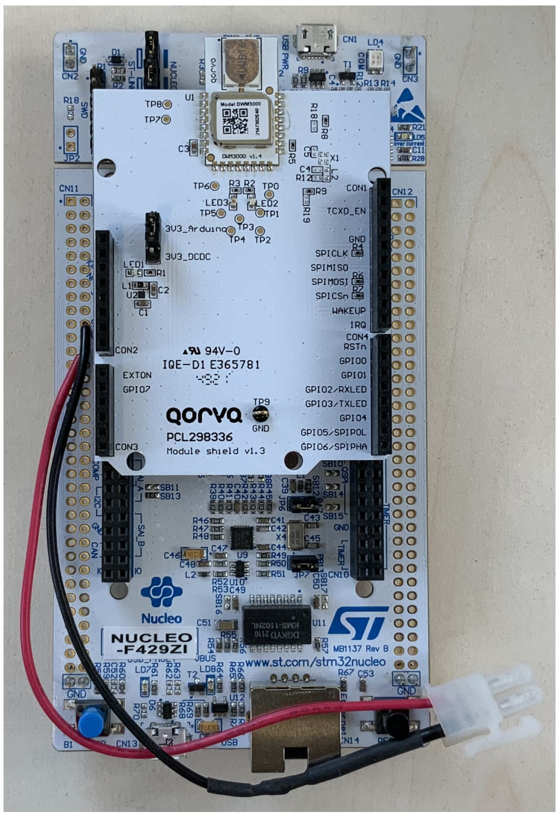


Figure 5.2.2: Assembled UWB anchor with soldered power supply cable

The STM32 Nucleo-F429ZI board is one of the two MCU development platforms suggested by the UWB antenna manufacturer to be used along with the DWM3000EVB since completely compatible with its software package provided by Qorvo ®. It integrates the ST-LINK Debugger/Programmer which allows to easily and quickly load the application software on the platform or flash and update the firmware on the MCU; it is also provided with ST Zio connector, which extends the Arduino® Uno V3 connectivity, and ST morpho headers that constitute an easy means of expanding the functionality of the Nucleo open development platform with a wide choice of specialized shields.

The DWM3000 manufacturer provides a set of APIs designed to allow the user to take confidence with the peripheral and to test the different configuration modes that the UWB antenna supports, detailing every function and parameter in *DW3xxx Device Driver - Application Programming Interface (API) Guide*.
The device driver is essentially a set of low-level functions providing a means to exercise the main features of the transceiver without having to deal with the details of accessing the device directly through its SPI interface register set. It is provided as source code in C programming language to allow it to be ported to any target microprocessor system with an SPI interface. In addition to that, the guide comes with a handful of example codes meant to show the user how to properly use the main functions and choose the desired use configurations of the IC; some of the examples also provide boilerplate code to implement specific actions and ranging communications: specifically, they show how to perform transmissions, receptions, blinking, and also complete TWR communications, both the Single-Sided and Double-Sided versions.

For this specific application and environment, the SS-TWR ranging mode has been employed, because more than one reason: first, the actual localization algorithm is performed on the rover's OBC and on a laptop mounted on top of it, so the final information about the ranges has to be obtained by the onboard UWB tag, meaning that the DS-TWR cannot be used without adjustments. In fact, there are only two cases in which the DS version could be used: the first assumes that a fourth additional message (with respect to the two needed by the SS) was sent from the anchors to the tag to inform it of the ranging calculation results, and that should happen for each anchor, leading to a doubling of messages sent and consequent time spent in the ranging activity; in the second one instead, each anchor should be equipped with a communication cable to send the ranging results to a central computer in the control room, increasing by a lot the complexity of the system, because of the synchronization needed, and also the total length of the path that the information has to run to get to the rover, because of the further communication to be performed from the control room to the moving robot.
In addition to the previous motivation, there's another one that has led to prefer the SS-TWR over the DS-TWR: in subsection 2.2.1 it has been seen how the usual performances of SS-TWR are worse than the DS version because of its poor accuracy due to the clock offset between the two nodes, but it has also been anticipated that additional adjustments could have been taken to increase them; in fact, among

the functions provided by the examples code, there is a specific one called *Carrier Integrator Diagnostic* that allows to read the clock offset and improve the accuracy of the SS-TWR range estimation. The results are so good that the same API guide states that, since the introduction of this compensation feature, they are no more recommending to prefer the use of the DS-TWR over the SS one because the overall performance is now comparable between the two.

With this premise, the implementation of the SS-TWR exchange is the following: each anchor has its own ID number, which is embedded in the response message, and waits indefinitely for a poll message from the tag to start the exchange; when it is captured, the anchor sends its response message with its ID and all the timestamps recorded by the application, including the calculated/predicted transmission timestamp for the response message itself. The tag, on the other side, is in charge of initiating the exchange: it embeds the anchor ID in the poll message to communicate with a specific one at a time, but it also loops the anchors' IDs to continuously get a range from each one; the poll message contains the timestamp of the transmission, and when the response message is acquired (and if corresponding to the expected one, in order to verify the reliability), all the timestamps are fed to the TWR distance formula, after the response one has been corrected from its clock offset. Finally, the tag application increases the current anchor ID to begin the exchange with the next anchor, and when all the anchors have been interrogated, a message is built with each anchor ID, a separator, and the corresponding estimated range, and then sent to the calculator that implements the localization algorithm through a serial communication via USB.

The application code also involves many parameters and constants that require proper calibration to be performed for each anchor, in order to get the best possible results. More than the others, the constant that sets the antenna delay is particularly critical because it directly affects the computed distance, with indirect proportionality, and so requires precise tuning: in section 5.5 it will be discussed the calibration strategy chosen to get to reliable results, that was actually a crucial point and quite a challenge.

5.2.2 Power Supply PCB adapter

As previously anticipated, the RoXY facility is equipped with multiple power rails, but they all supply alternating current, at least at 24 V. However, the STM32 Nucleo-F429ZI only accepts three types of power sources: it can be powered by a micro-B USB cable through the micro-USB port on the integrated ST-LINK (U5V), allowing powering and programming/debugging at the same time, or it can be run on external voltages through two different pins (E5V and V_{IN}), accepting respectively 5 ± 0.25 V and $7\text{ V} \leq V_{IN} \leq 12\text{ V}$, and obviously all three intended as DC. This created the necessity to design a power adapter from the 24 V AC power rail to at least the maximum voltage supported by the MCU board, 12 V DC, which in this case was also a lucky number: the power adapter only needs to rectify the signal and halve its amplitude.

With these requirements, a simple power adapter printed circuit board has been designed, employing commercial components to ease the procurement timelines, the assembling, and the integration with the UWB anchor. The circuit is composed of a diode rectifier bridge and a 12 V voltage regulator in cascade, with two filtering capacitors of 470 µF and 0.1 µF respectively upstream and downstream of the regulator, to provide both high and low-frequency filtering. The schematic is shown in Figure 5.2.3.

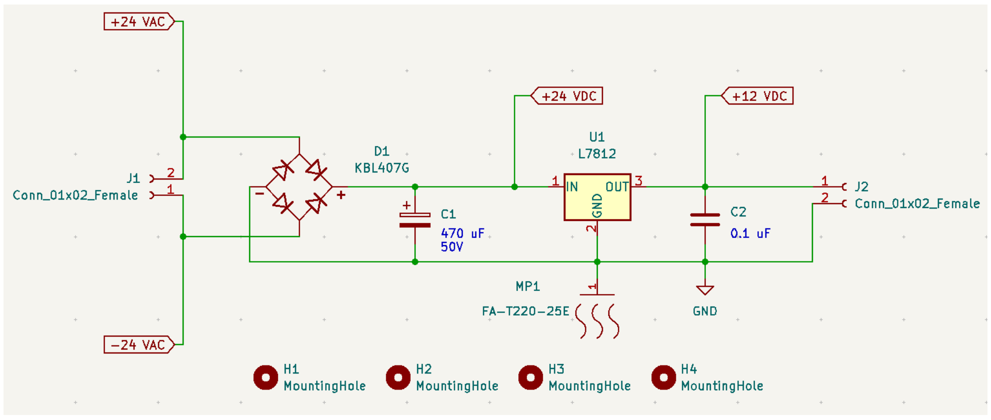


Figure 5.2.3: Power adapter circuit schematic

The schematic has then been transposed on a real circuit with the aid of the open-source KiCad EDA software: it has been provided with 2x1 micro-fit Molex connectors for both the input and output port and a large dissipator for the voltage regulator, arranged with all the discrete components footprints on the smallest area possible, given a width equal to the one of the Nucleo-F429ZI. The traces have been posed on the layer in order to be as far from each other as possible, while a wide metal plate on a layer below has been used as ground, and also connected through many vias to the upper layer also acting as a ground, increasing the ground area in order to provide shielding against external interferences. A picture taken from the PCB view of KiCad depicting the footprints, the traces, and the various layers is shown below.

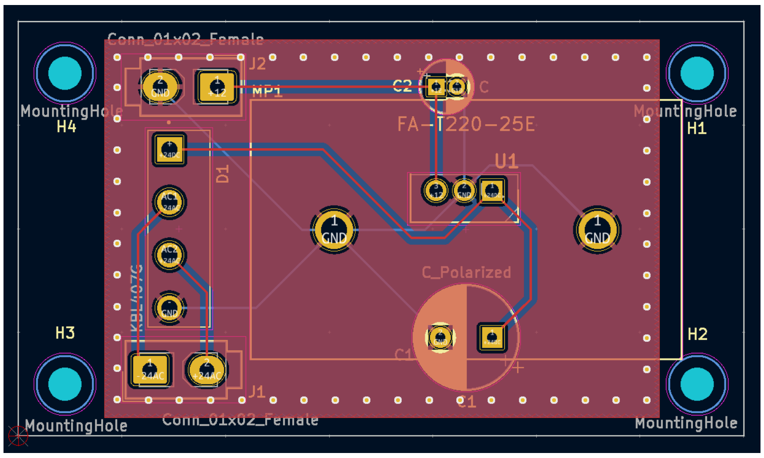
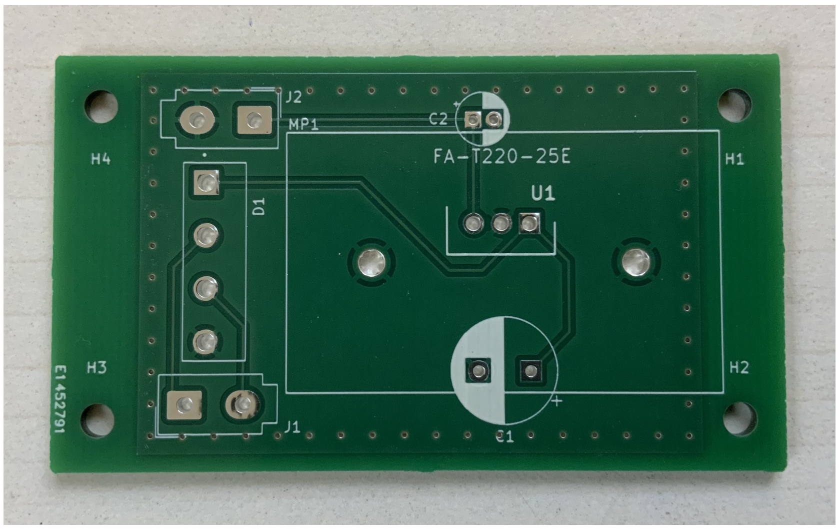
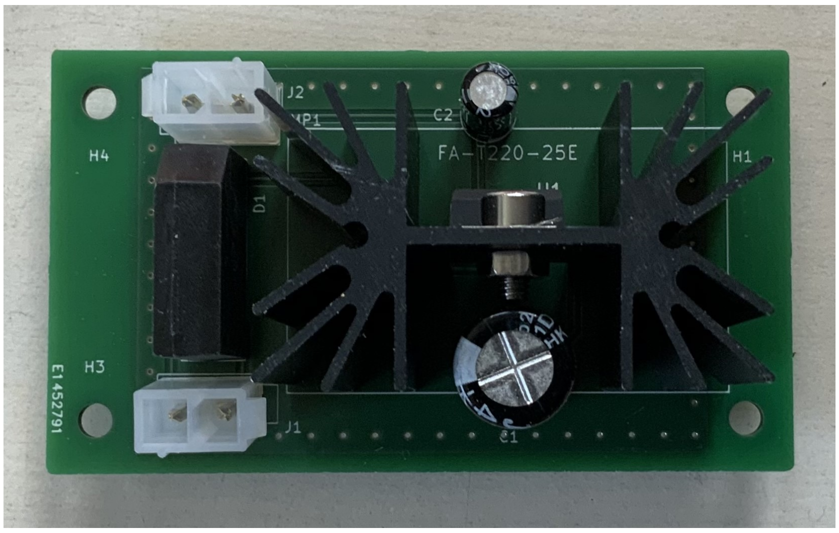


Figure 5.2.4: PCB masks and views

The PCBs production has been entrusted to Eurocircuits and kindly borne from the Politecnico di Torino Interdepartmental Centre for Service Robotics (PIC4SeR). The PCBs have been hand-soldered with the various through-hole components, and then continuity and validation tests have been performed for each one of them, showing an output voltage that has never overcome the Nucleo board limit of 12 V. Figure 5.2.5 shows the physical PCB right following the production and after the components soldering.



(a)



(b)

Figure 5.2.5: Physical PCB **(a)** and final soldered circuit **(b)**

A 3D model of both the Nucleo board with the DWM3000EVB mounted on top and the soldered PCB (Figure 5.2.6) has been acquired and exploited to aid the design of the anchor case that will be presented afterward, in order to reduce at the minimum the hand-taken measurement of the various component.

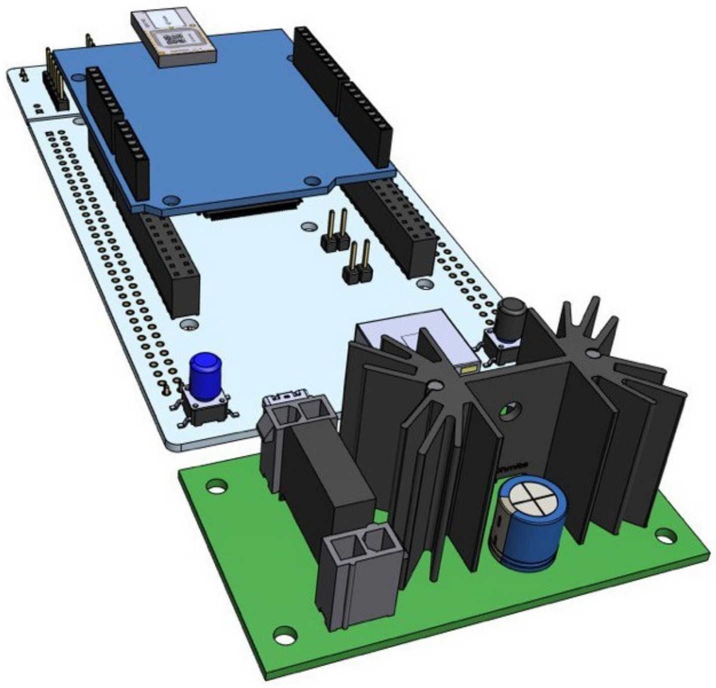


Figure 5.2.6: 3D model of the UWB anchor and the power adapter PCB

5.2.3 UWB anchor 3D-printed case

The UWB system proposed in this thesis effort is meant to become part of the RoXY facility as a reliable system to provide position ground truth to assist the research activities of the Robotics team of TAS-I Turin. This means that it has to be provided with a suitable structure to allow it to be easily turned on and off, to resist outdoor meteorological conditions such as snow and rain, and to be ready and active with no time waste: to accomplish this task, a great effort has been put in the development of a case for the UWB anchors that could contemporarily be adequate for outdoor condition, be attached to the perimeter walls and remain fixed but leaving the option to detach it to access the board to reprogram it, and that could easily fit both the Nucleo MCU and the PCB.

The modeling of such a structure has been performed with the aid of a semi-professional 3D modeling CAD called Shapr3D run on an iPad Pro: it has been chosen because of prior experience with the software and because of the rapidness and easiness in its use, which makes it an ideal choice for the rapid prototyping. In fact, thanks to the availability of two Prusa i3 MK3S+ 3D printers available at the facility, it has been possible to investigate many designs for the case, and also to test specific portions of it to tune the perfect dimensions and tolerances to achieve

the best fit between all the pieces.
The design of the case has been conceived as a single piece that could be both thick enough to protect the board and the PCB from external agents and thin so that it would not constitute a real obstacle for the UWB signal transmissions: that was made possible by the employment of PETG material and special 3D honeycomb construction patterns that allow the structure to be at the same time robust and weather resistant while maintaining a thickness not superior to 3 mm, which also reduces to less than 2 in the small area portion right in front of the UWB antenna, to furtherly reduce the impact of the case; so, while it would not be incorrect to state that a UWB transmission with such a cased antenna would constantly be in NLoS conditions, the just discussed feature makes its performance de facto comparable to the LoS ones.

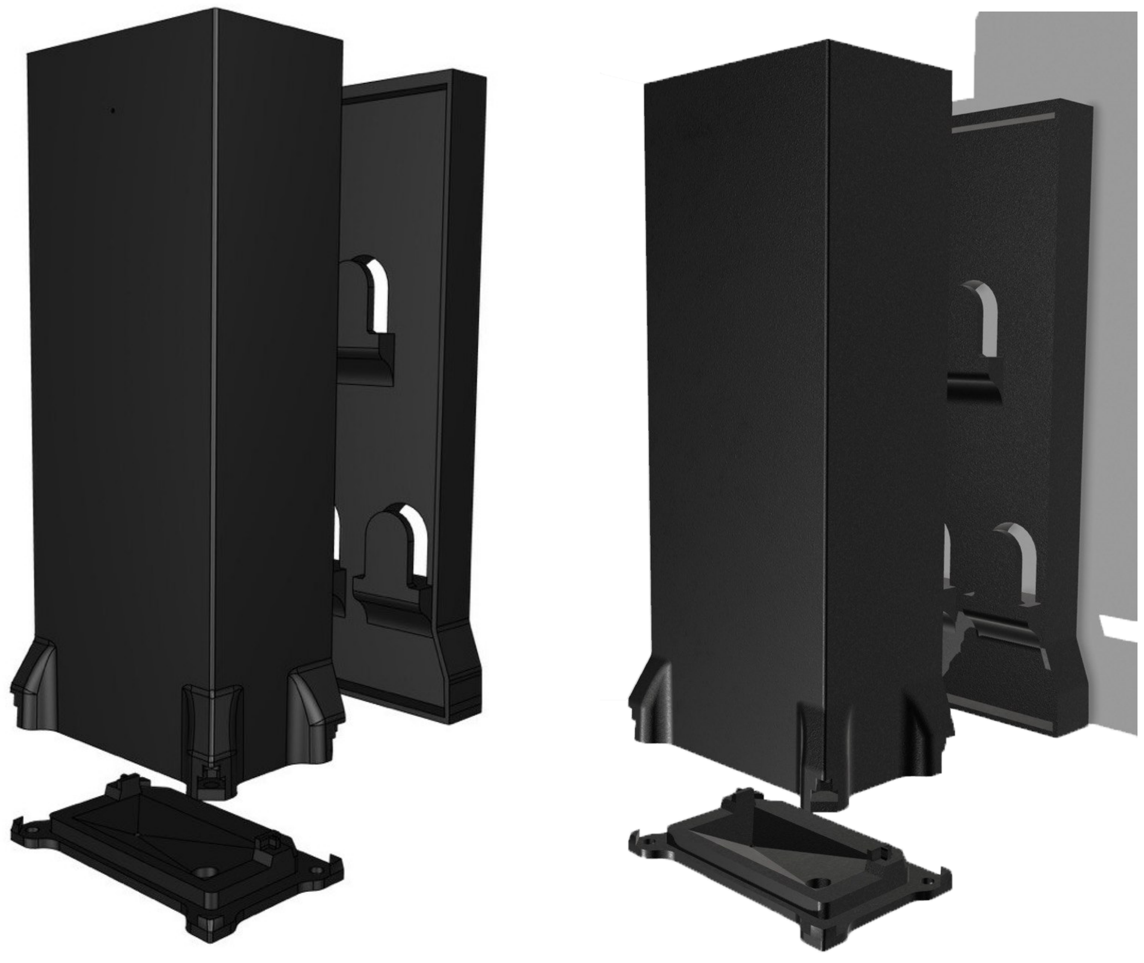

(a) (b)

Figure 5.2.7: Case design expanded in its three components **(a)** and its render **(b)**

The design is composed of three pieces, the anchor case, a cap, and a backplate, as visible in Figure 5.2.7 where are shown both the model and its render, which provides a better view of the details.

The actual case has a parallelepiped shape, with a top layer slightly thicker and the honeycomb infill denser than the others, to ensure higher stiffness and robustness on the impact surface of the weather agents, and offer less space to be filled should a crack let some water seeping inside. It is also provided with bumps at the bottom corners designed to provide the necessary space for inserting four nuts to screw in the cap and simultaneously provide a sloping surface for water to flow outwards while preventing it from ending up inside due to slits. It also presents three slots on its back, apt to host the three corresponding supports laying on the backplate, to realize the detachability feature. Finally, the inside lateral walls expose a rail suited to slide the board and the PCB along the whole length of the case without letting them slip down when the case is put in place upright, and with proper offsets from the wall to keep them still and centered. A detail of the case is shown in Figure 5.2.8.

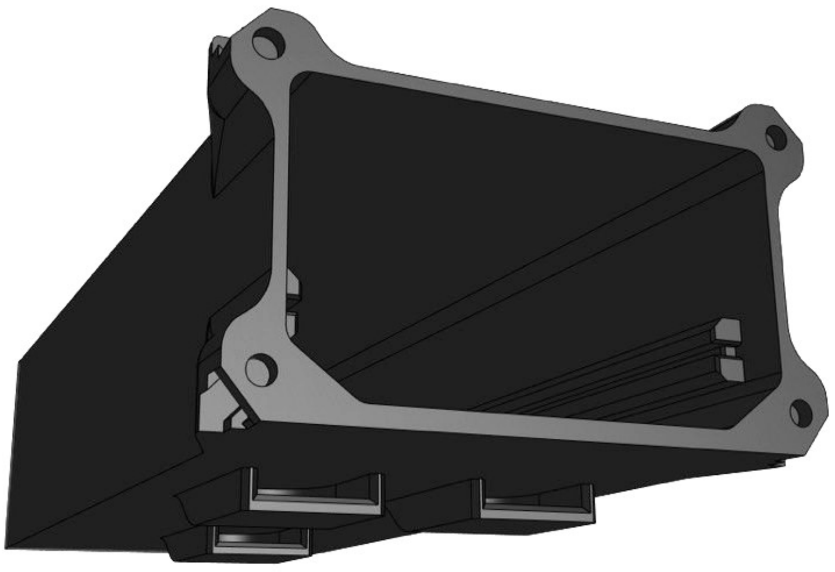


Figure 5.2.8: Detail of the bottom part of the case, showing the inside

For what concerns the cap, it has been designed with the following features: it presents four holes in the corners, where the screws are to be put, with the furthest corners extruded to helps inserting the cap in the right position, and its side flaps slightly wider than the case to allow the user to easily detach it. In addition, a platform whose perimeter corresponds to the bottom opening of the case is extruded upward to fit seamlessly into the case and provide an additional measure of security against some water that might seep in; on either side of the cap, there are two segments shaped exactly like the case's internal rails, designed to fit precisely into them and hold the boards in place. Finally, there are two holes, a larger one through which to run the power cables and then seal them with hot glue or silicone, and a much smaller one located at the bottom at the apex of a sloping pyramidal surface whose purpose is to drain away any condensation that might build up inside. The cap is displayed in detail in Figure 5.2.9a right below.

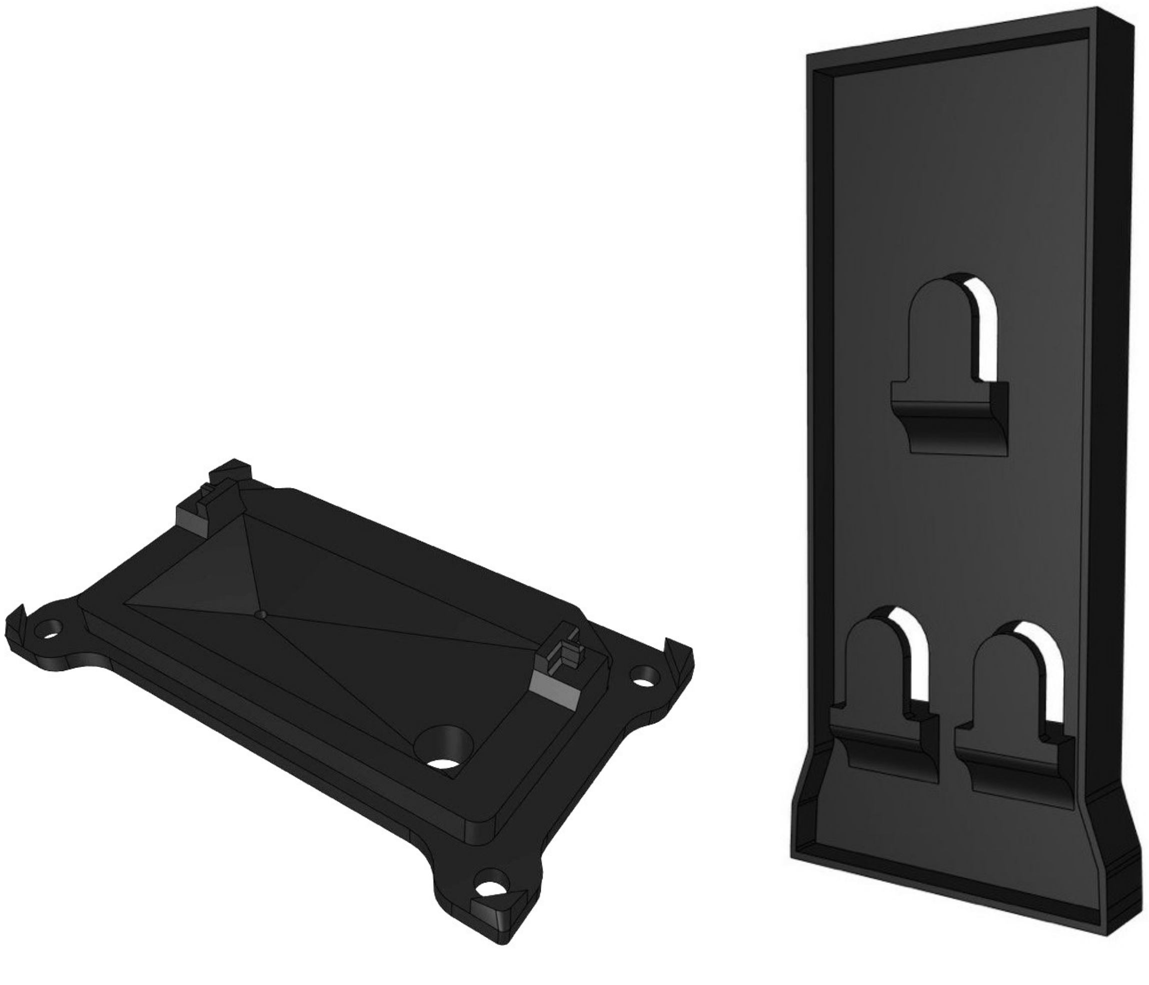

(a) (b)

Figure 5.2.9: Cap **(a)** and Backplate **(b)** in perspective view

Lastly, the backplate is the part that is meant to be attached to the wall: it presents three supports perfectly corresponding with the inside of the slots on the back of the case, providing a stiff grip and a large contact area to hold the case firmly in place. Its back surface has been glued to the perimeter wall of the facility to avoid making permanent damage to it, but the piece is designed so that two holes can be made above and below the first support, to easily screw it to the walls. Figure 5.2.9b shows the complete view of the backplate.

The 3D-printed case is then reported in Figure 5.2.10 below, showing all three pieces and the final assembled structure, along with the four nuts inserted in their case slots and the power cable glued on the cap.

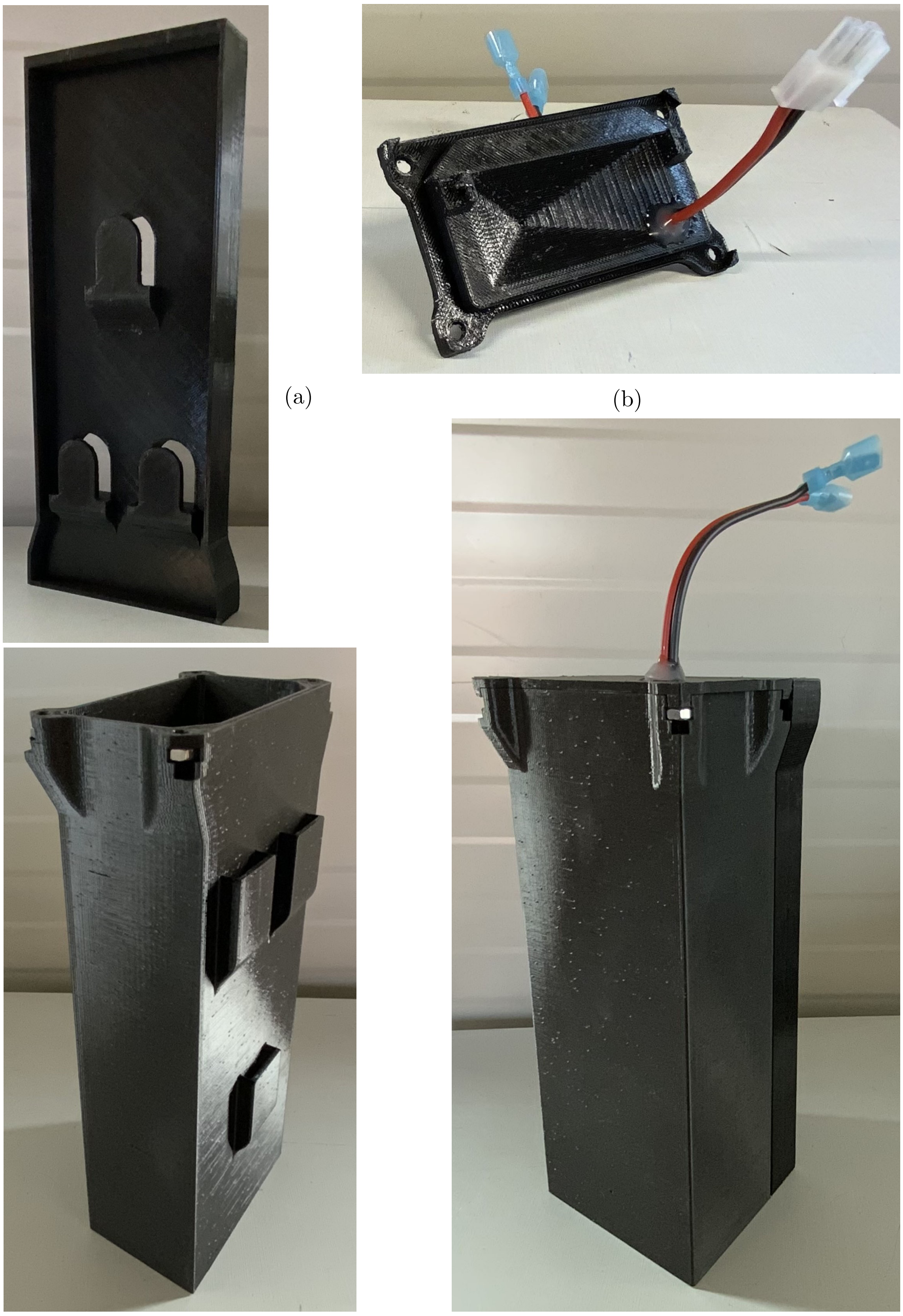


Figure 5.2.10: 3D-printed PETG backplate **(a)**, cap **(b)**, anchor case **(c)**, and assembled structure **(d)**

Last of all, another structure has been designed and realized thanks to rapid prototyping, and that is the rover stand for the UWB tag. Its design is quite simple, as it only needs to provide a suitable physical interface to place the Nucleo board and screw it in, while also offering stability and robustness not to let the tag suffer the mechanical solicitations due to the rover movement. So, it is provided with suitable holes for screws and slots to insert the bottom pins and the Ethernet port, to let the Nucleo board adhere firmly on a large contact area, and with vertical edge supports on both sides to almost completely eliminate the oscillations in movement. The design is shown in the following Figure 5.2.11:

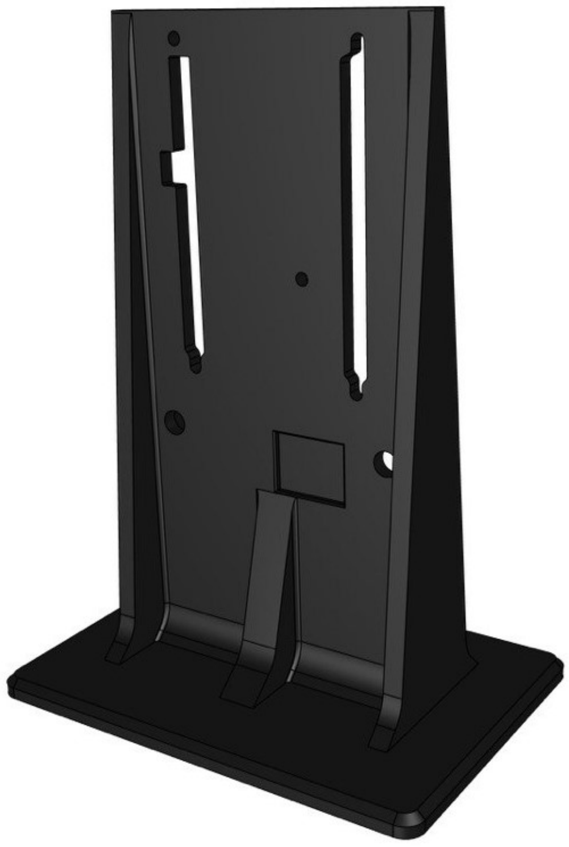


Figure 5.2.11: 3D model of rover stand

5.3 ROS2 Framework and nodes

ROS2 (Robot Operating System 2) is an open-source software framework designed for developing and operating robotics systems. It is the successor to the original ROS framework and was released in 2014 by the Open Robotics organization. ROS2 is designed to address the limitations of the original ROS framework and provide a more flexible, scalable, and robust platform for developing complex robotics systems. It supports multi-language programming, including C++ and Python, and employs the Data Distribution Service (DDS) middleware to enable communication between nodes, the fundamental functional units of a ROS2-based system, providing a standard interface for exchanging data to make it easier to integrate different components of a robotics system; the communication system involves the use of standard or customized ROS2 messages sent by the nodes, broadcasted on the network on a specific topic, which collect all the information about a single specific subject, to which other nodes can subscribe.

Among its advantages, it is surely worth mentioning the large scalability potential, which means that it can be used to develop robotics systems of different sizes and complexity, but also the modularity feature is crucial, having ROS2 a small core and a set of independent modules that can be added as needed, because it allows to more easily develop and test individual components of a robotics system, and also enabling easy customization and extension of the framework without impacting the complexity of the internal communication system.

The potential of the ROS2 framework has been exploited to create a network of nodes that cooperate to acquire the data from the ranging phase, elaborate them within the filters algorithm, and then provide a set of metrics and visualizable data to estimate the goodness of the system. The chosen programming language is C++, for both prior experience with it and the not marginal difference in the execution and communication speed that it provides, acting at a much lower level than Python; however, a few nodes that do not directly intervene in the localization algorithm, and therefore don't need to be extremely optimized for speed, have been written in Python because of the greater easiness of implementation due to ROS2 specific structures.

The nodes that are explicitly involved in the localization algorithm are presented as follows:

- *SerialReader*: this node employs the serial message read function available in the Asio module of the well-known Boost C++ library to acquire the information about the ranges estimated by the UWB anchor and sent serially via USB cable. It receives the ranges with the respective anchor ID one by one, then packs them all in one single ROS2 custom message in vector form, and sends it through the related topic. Before sending the message, the node discards the bad measurement, namely the ones that have reported errors or timeout, replacing their range with a negative number.

- *ExtendedKF*: this is the node responsible for computing the position estimation by means of the Extended Kalman Filter equations. Its basic functioning involves the collection of the ranges from the subscription to the corresponding topic and the detection of wrong or absent measurements to modify adequately the order of the filter (i.e. the size of the measurement matrix and the related ones) or increase the corresponding entry of the uncertainty matrix **R** to infinite to actually discards it from the calculation, based on the desired approach; then it uses a vector parameter containing the exact 3D positions of the anchors to compute the residual vector to fed to the standard EKF equations. When the state and covariance are estimated, it broadcast them as a 3D point with covariance, that can be displayed on the monitor, and adds it to the previous ones to form a path that is also sent through the network to be displayed; the code also collects some of the vectors and matrices used in or computed by the filter's equations to send them over another topic in order to use them to calculate some metrics.

  For what concerns the f and h functions implemented in the filter, the models discussed in [20] have been taken into consideration: in the article, the authors carried out a comparison between three models, the P-model, the PV-model and the PVA-model, which stand for respectively Position, Position-Velocity and Position-Velocity-Acceleration; they discovered that the most complex model, the PVA one, actually presents poorer performances than the other two, which are more or less equivalent to each other. In order to keep the filter as simple as possible, the choice has fallen on the P-model, which only takes into account the update of the position of the robot, while the PV and PVA ones also take into account the laws of dynamics to compute the velocity and the acceleration. Since no control input actually intervenes in the estimation of the position given only the UWB ranges, the state transition function f actually reduces to the identity function only, while the measurement function h implements the trilateration presented in subsubsection 1.1.2.2. Hence, considering a 3D position state estimation vector $\hat{\mathbf{x}} = [\hat{x}_0 \quad \hat{x}_1 \quad \hat{x}_2]$ and a measurement estimation vector $\hat{\mathbf{z}} = [\hat{z}_0 \quad \hat{z}_1 \quad \hat{z}_2 \cdots \hat{z}_n]$ obtained from the ranges of n anchors, each one with its position $\mathbf{p}_i = [p_{i,0} \quad p_{i,1} \quad p_{i,2}]$, the **F** and **H** matrices employed in the EKF are the following:

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{H} = \begin{bmatrix} \frac{\partial h_1(\hat{\mathbf{x}})}{\partial \hat{\mathbf{x}}}\Big|_{\hat{x}_0} & \frac{\partial h_1(\hat{\mathbf{x}})}{\partial \hat{\mathbf{x}}}\Big|_{\hat{x}_1} & \frac{\partial h_1(\hat{\mathbf{x}})}{\partial \hat{\mathbf{x}}}\Big|_{\hat{x}_2} \\ \frac{\partial h_2(\hat{\mathbf{x}})}{\partial \hat{\mathbf{x}}}\Big|_{\hat{x}_0} & \frac{\partial h_2(\hat{\mathbf{x}})}{\partial \hat{\mathbf{x}}}\Big|_{\hat{x}_1} & \frac{\partial h_2(\hat{\mathbf{x}})}{\partial \hat{\mathbf{x}}}\Big|_{\hat{x}_2} \\ \vdots & \vdots & \vdots \\ \frac{\partial h_n(\hat{\mathbf{x}})}{\partial \hat{\mathbf{x}}}\Big|_{\hat{x}_0} & \frac{\partial h_n(\hat{\mathbf{x}})}{\partial \hat{\mathbf{x}}}\Big|_{\hat{x}_1} & \frac{\partial h_n(\hat{\mathbf{x}})}{\partial \hat{\mathbf{x}}}\Big|_{\hat{x}_2} \end{bmatrix} = \begin{bmatrix} \frac{\hat{x}_0 - p_{i,0}}{\hat{z}_0} & \frac{\hat{x}_1 - p_{i,1}}{\hat{z}_0} & \frac{\hat{x}_2 - p_{i,2}}{\hat{z}_0} \\ \frac{\hat{x}_0 - p_{i,0}}{\hat{z}_1} & \frac{\hat{x}_1 - p_{i,1}}{\hat{z}_1} & \frac{\hat{x}_2 - p_{i,2}}{\hat{z}_1} \\ \vdots & \vdots & \vdots \\ \frac{\hat{x}_0 - p_{i,0}}{\hat{z}_n} & \frac{\hat{x}_1 - p_{i,1}}{\hat{z}_n} & \frac{\hat{x}_2 - p_{i,2}}{\hat{z}_n} \end{bmatrix}$$

where $\hat{z}_i = \sqrt{(\hat{x}_0 - p_{i,0})^2 + (\hat{x}_1 - p_{i,1})^2 + (\hat{x}_2 - p_{i,2})^2}$ as required by the trilateration method for the estimation of the range given the estimated state and the anchor positions.

The node has been provided with several additional features, meant to increase the effectiveness of the algorithm: the Mahalanobis distance-based thresholding technique, as well as the modifications proposed in [34], [22], [14] and discussed in section 3.4 have been implemented in the code in an optional form, which means that they can be activated and deactivated freely by the user. In addition to that, also a smoother has been coded, to perform a moving average of the positions in a user-resizable window and reduce the output noise increasing also the visual performance. Lastly, another additional function called "error map" has been added, this time to balance the error introduced by the UWB anchors, but it will be discussed later in section 5.5.

To be able to compare the various features, it has been exploited ROS2's ability to launch multiple instances of the same node with different names and parameters. This feature has been crucial to allow the simultaneous visualization of the performance of the different versions of the filters.

- *UnscentedKF*: similar to the previous node, this one is in charge of estimating the tag location employing the Unscented Kalman Filter equations. The structure and the additional function seen in the EKF node are reused here, where the only changes consist in the implemented equations, which obviously refer to the UKF. Differently from what is done in the EKF, the UKF do not approximate the estimation of state and measurements with a Jacobian matrix, instead, it directly employs the f and h function to perform an exact calculation. The state transition function f is analogous to the one in the EKF to implement the P-model, so it simply is $\hat{\mathbf{x}} = f(\mathbf{x}) = \mathbf{x}$, while the measurement function h represents the distance computation and is again $\hat{z}_i = h(\boldsymbol{\mathcal{Y}}_i, \mathbf{p}_i) = \sqrt{(\mathcal{Y}_{i,0} - p_{i,0})^2 + (\mathcal{Y}_{i,1} - p_{i,1})^2 + (\mathcal{Y}_{i,2} - p_{i,2})^2}$, where $\mathcal{Y}_{i,k}$ indicates the k-th component of the i-th sigma point and p_i is again the i-th anchor.

  A slight modification to the equations, with respect to their standard form seen in section 3.3, has been made following the ones proposed in [30]. In particular, the change affects the first part of the update step, where the measurements are first estimated through the transformed sigma points: the authors re-compute the sigma points after the estimation of the new state and covariance, using exactly them to obtain a new sigma point distribution, which is then fed to the Unscented Transform that calculates the estimate of the measurements; it has been seen that this small changes positively influences the accuracy of the filter. For the sake of clarity, the modified UKF equations are reported in Table 5.6:

$$\boldsymbol{\mathcal{X}}_{k-1} = \textbf{sigma-function}(\mathbf{x}_{k-1}, \mathbf{P}_{k-1})$$

$$\mathbf{W}^m, \mathbf{W}^c = \textbf{weight-function}(\alpha, \beta, \kappa, n)$$

$$\boldsymbol{\mathcal{Y}}_{k-1} = f(\boldsymbol{\mathcal{X}}_{k-1})$$

$$\hat{\mathbf{x}}_{k|k-1} = \sum_{i=0}^{2n} w_i^m \boldsymbol{\mathcal{Y}}_{k-1,i}$$

$$\hat{\mathbf{P}}_{k|k-1} = \sum_{i=0}^{2n} w_i^c (\boldsymbol{\mathcal{Y}}_{k-1,i} - \hat{\mathbf{x}}_{k|k-1})(\boldsymbol{\mathcal{Y}}_{k-1,i} - \hat{\mathbf{x}}_{k|k-1})^{\mathrm{T}} + \mathbf{Q}_k$$

$$\boldsymbol{\mathcal{Y}}_{k|k-1} = \textbf{sigma-function}(\hat{\mathbf{x}}_{k|k-1}, \hat{\mathbf{P}}_{k|k-1})$$

$$\boldsymbol{\mathcal{Z}}_{k|k-1} = h(\boldsymbol{\mathcal{Y}}_{k|k-1})$$

$$\hat{\mathbf{z}}_{k|k-1} = \sum_{i=0}^{2n} w_i^m \boldsymbol{\mathcal{Z}}_{k|k-1,i}$$

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \hat{\mathbf{z}}_{k|k-1}$$

$$\mathbf{S}_k = \sum_{i=0}^{2n} w_i^c (\boldsymbol{\mathcal{Z}}_{k|k-1,i} - \hat{\mathbf{z}}_{k|k-1})(\boldsymbol{\mathcal{Z}}_{k|k-1,i} - \hat{\mathbf{z}}_{k|k-1})^{\mathrm{T}} + \mathbf{R}_k$$

$$\mathbf{P}_{xz,k} = \sum_{i=0}^{2n} w_i^c (\boldsymbol{\mathcal{Y}}_{k|k-1,i} - \hat{\mathbf{x}}_{k|k-1})(\boldsymbol{\mathcal{Z}}_{k|k-1,i} - \hat{\mathbf{z}}_{k|k-1})^{\mathrm{T}}$$

$$\mathbf{K}_k = \mathbf{P}_{xz,k} \mathbf{S}_k^{-1}$$

$$\mathbf{x}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \tilde{\mathbf{y}}_k$$

$$\mathbf{P}_{k|k} = \hat{\mathbf{P}}_{k|k-1} - \mathbf{K}_k \mathbf{S}_k \mathbf{K}_k^{-1}$$

Table 5.6: Modified Unscented Kalman filter equations [30]

- *MetricsPublisher*: this node is the collector of the messages from both *ExtendedKF* and *UnscentedKF* sent over the metrics topic. It utilized the state vector $\mathbf{x}$, the state estimation vector $\hat{\mathbf{x}}$, the state covariance matrix $\mathbf{P}$, the measurement vector $\mathbf{z}$, the measurement estimation vector $\hat{\mathbf{z}}$ and the measurement covariance matrix $\mathbf{S}$ to compute some indicators of the goodness of the algorithm. In particular, $\mathbf{x}$ and $\hat{\mathbf{x}}$ are employed in the evaluation of the Root-Mean-Square Error (RMSE) and of the Normalized Estimation Error Square (NEES) considering also $\mathbf{P}$, while $\mathbf{z}$, $\hat{\mathbf{z}}$ and $\mathbf{S}$ are needed to compute the Normalized Innovation Squared (NIS): the RMSE is useful to evaluate the quality of estimation in terms of its variation and distortion, while the NEES helps to check the consistency of a state estimator with a hypothesis test, following it the chi-squared distribution, and the NIS serves the same purpose but for what concerns the consistency on the measurement estimation. Furthermore, since

Kalman filters are known to be more accurate on the xy-plane rather than in the z-direction, a bi-dimensional version of RMSE and NEES are calculated too. In addition to that, the node also computes and outputs the Mahalanobis distance of both the state and the measurement.

In the development of this thesis project, the coding of the nodes just described has been performed in parallel with the other activities discussed in the previous subsections, but it ended up terminating much before that the other project components did. This happened for various reasons related to the numerous iterations and the literal time needed for the 3D rapid prototyping (several tens of hours), as well as to the provisioning time to get all the Nucleo boards, to print and then solder the PCBs, and so on. So, there has been a moment in which the software part of the localization system was ready to be tested, but the complete field test setup was still missing. To get around this problem, and also to be provided with a handier way to test the algorithms and tune the parameters than perform countless field tests, a great effort has been put into the development of a simulator of some sort.
Despite ROS2 being provided with its own simulator, called *Gazebo*, it has been chosen not to involve it in the system development process: it is due to the high complexity of Gazebo and its difficult customization to adapt it to own specific purposes, but also and especially to the fact that it represents a huge overkill for the necessities of this project; in fact, Gazebo is meant to simulate mostly the physical properties and behaviors of the object that are loaded in it, including the gravity, the collisions, and so on, while the type of simulator that was needed in this case resembled more the simple robot simulator used in the ROS2 tutorials to explore the characteristics of the framework, which is called *TurtleSim*.
Made on these premises, the simulator has been developed to possess the following features: it had to represent the rover moving through the 3D space and be controllable, preferably by keyboard for convenience; it had to be able to spawn both the anchors and the RoXY terrain to resemble a verisimilar environment with respect to the field test, and the ranges acquired from the anchors needed to be affected by a controllable noise, in order to fully simulate the behavior of the real ones; lastly but more importantly, all these features had to be implemented in such a way to be compatible with the input and output interfaces of the UWB system, which means that the ranges had to be provided in the exact same format as the one generated by the real anchors, and the visualization of the simulation had to be consistent with the output of the real system.

To accomplish this task, the visualizer provided with the ROS2 framework, *RViz2*, has been used, and four more nodes have been developed:

- *AnchorsRoxySpawner*: this node is the simplest among those deputed to the implementation of the simulator, but surely the most useful one for what concerns its visual performance. Its task is to read a parameter file containing the positions of the anchors and broadcast as many corresponding frames; in ROS2, a frame represents a reference system, often associated with an object, that allows the ROS2's transform library, *tf2*, to keep track of its position over time. *tf2* maintains the relationship between coordinate frames in a tree structure buffered in time and lets the user transform points, vectors, etc. between any two coordinate frames at any desired point in time. For what concerns the anchor, their position is fixed so the static version of the transform broadcaster is used.

Furthermore, the node is also in charge to broadcast the 3D representation of the anchors, which is associated with the respective frames, and it is made possible by the STL file of the anchor case model. Similarly, the node allows the visualization of the RoXY terrain thanks to a 3D scan of the playground obtained by a stereocamera: it is extremely important to provide a good verisimilarity with the real field conditions and to guide effectively the simulated rover across a playground that also considers the third dimension.

- *RangingSim*: this node is the one that reproduces the functioning of the UWB system. It exploits the transformed frames broadcasted by the previous node to obtain the exact distance between them and the frame corresponding to the rover position, which will be discussed in the next node, and it is worth specifying that the node does not need to perform any calculation to get the ranges, because *tf2* directly provides them to the network. To make the simulation as similar as possible to the real case, the white Gaussian noise generator function of the standard library of C++ is employed: it allows easily to add random noise with user-defined standard deviation to the exact ranges to simulate the real noise conditions; furthermore, an optional parameter is added, to allow the user to arbitrarily increase the noise (through its standard deviation) even for a specific amount of time, and this has been done to test the effectiveness of the various outlier rejection features of the filters. Finally, the node sends a message containing all the ranges through the same interface of the real UWB tag, in order to be compatible with the input of the filters.

- *PasquaControl*: this node serves as a controller to teleoperate the rover sending control inputs to it (its name origins from the codename given to the modified Seekur Jr rover present in the RoXY facility, *Pasqualone*). Along with the next discussed node, they are the only nodes written in Python, as was mentioned before that there would be: this was due to both the greater easiness of using Python, when the speed limit is not crucial, and the fact that a public ROS2 node used to control the ROS2 tutorial simulator mentioned earlier, *TurtleSim*, has been employed as boilerplate to the development of this node.

  The code, however, has been almost completely revisited, since the nature of the two simulators is quite different: it captures the keyboard inputs provided by the user to move the robot and transforms them into directional velocity commands; the keys, in fact, are bound to specific sets of values, which serve as the increment to the current command. At the launch of the node, the command is set to zero, but the user can vary it by adding constant increments; the velocity increments are designed to not allow the instantaneous acceleration, in such a way that, within a single command, the user can only increase or decrease the speed or move the rover into the space at the currently set speed changing the direction of the velocity.

The node actually allows a 5DoF control, since the command is composed of linear and angular components; this had to be done in order to provide the rover with the ability to move naturally across the 3D space as a real one would do: in fact, the simulated one is able to perform smooth turns in the xy-plane but also to climb hillocks and slopes. So, the controller is able to singularly (or contemporarily) modify the linear and angular components of the velocity commands, specifically the x-y-z components and the roll-yaw components (imposing that the rover front direction corresponds to the y one, it has been assumed that a pitch variation was not significant or useful for the verisimilarity purpose): this means that the rover is able to contemporarily change its position and orientation, to actually perform climbing and turning.

The node also provides commands to reset the increments to their default values, while also ensuring that no negative speed command could be sent (to reverse, for instance, only the y direction has to be negative, while the velocity remains positive. Finally, the linear and angular direction components are multiplied by the relative linear and angular speed values to obtain a twist command, which in the end acts as a 6D velocity increment in the three linear and three angular directions (but with the pitch increment fixed to zero). All the commands are sent with an appropriate message through the reserved topic.

- *PasquaSim*: this is the node that actually simulates the rover moving across space. As already said, it has been written in Python, this time also because a code skeleton previously developed and used by the Robotics team in TAS-I was provided; its content, however, has been greatly modified to satisfy the different needs of this simulator. The code is able to perform various actions: its main function is to spawn into the simulated playground a frame corresponding to the position of the rover and to move it across the space using the commands sent by the previously seen node, but also allows to create a path accumulating all the positions' history, to nicely plot it on the visualizer, and to reset the position to the user-defined default value, clearing the path.

  The code implements a simple dynamic model to recompute position and orientation of the rover as a set frequency: it multiplies the received speed increments by the simulation time (the inverse of the update frequency) to get the position, and then again multiplies it by a rotation matrix, to also obtain the new orientation. It also provides an upper and lower bound to the climb degree and checks that the rover position does not go below the ground, to ensure the correspondence with the reality and the actual limits of the rover, and alerts the user when the rover is brought back to a horizontal position.

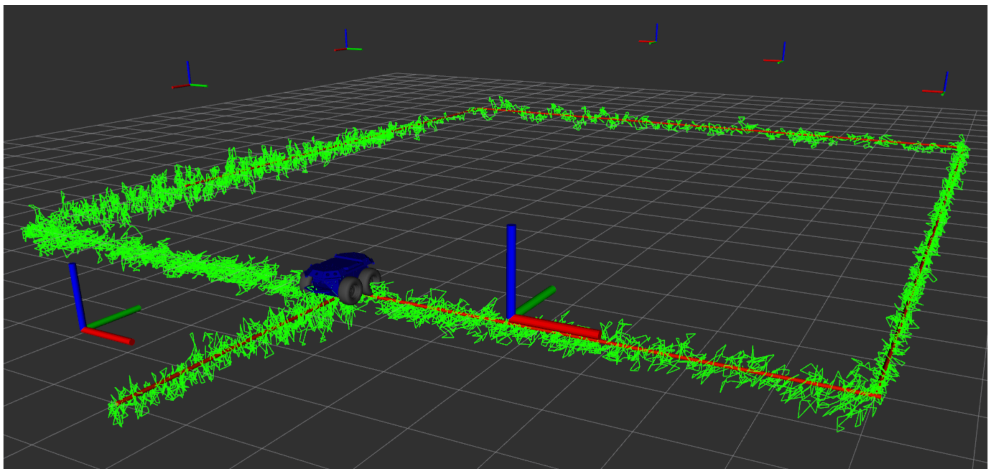
  Finally, it broadcasts the new pose of the rover and the updated path and also moves the robot frame in the *tf2* system. The frame is then used in the visualizer to spawn the 3D model of the rover in the playground area, to achieve a realistic and good-looking visual performance.

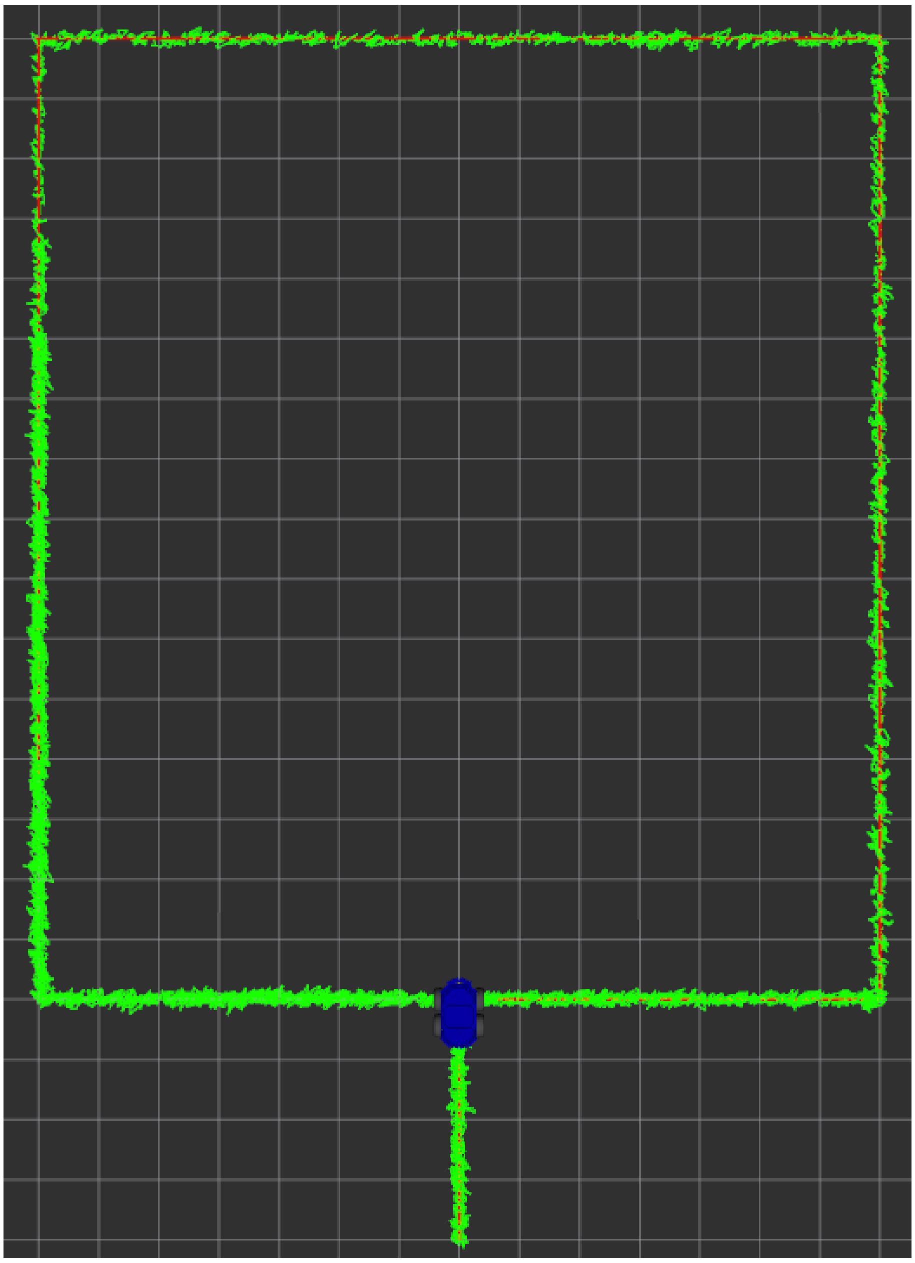
Both the visual and algorithmic performance of this simulator was observed to be really good, having it made a good job in aiding the testing of the computational system, as will be presented in the next section.

5.4 Simulations and Results

The simulator made it much easier to perform many different simulations of both the EKF, UKF, and their respective variations, and thanks to a ROS2 tool called *ROSbag* it was also possible to save and store the input data (so the position of the simulated rover and the ranges) to deterministically run the simulation over and over, in order to be able to capture all the images and data for an accurate analysis of all the filters. In particular, five types of paths have been simulated: the first two were drawn to be regular shapes, a square and a circle, and lay on a single horizontal plane without taking into account the playground; the other three paths instead run across the RoXY terrain, with the second a little longer than the first and with the third passing through ditches and knolls along the way to simulate the change of height. It is worth mentioning that early simulations have been performed only aiming to test and tune the algorithms, they employed different anchors positions and statistical parameters since at the time the real ones were yet to be obtained; for this reason, they have been repeated after having acquired all the missing data, and those are the ones reported in this section: specifically, only a subset of images and data will be shown, while the remaining ones can be found in Appendix A.

The following figures show a top and a perspective view of the complete route of the squared path and the EKF performance:


(a)

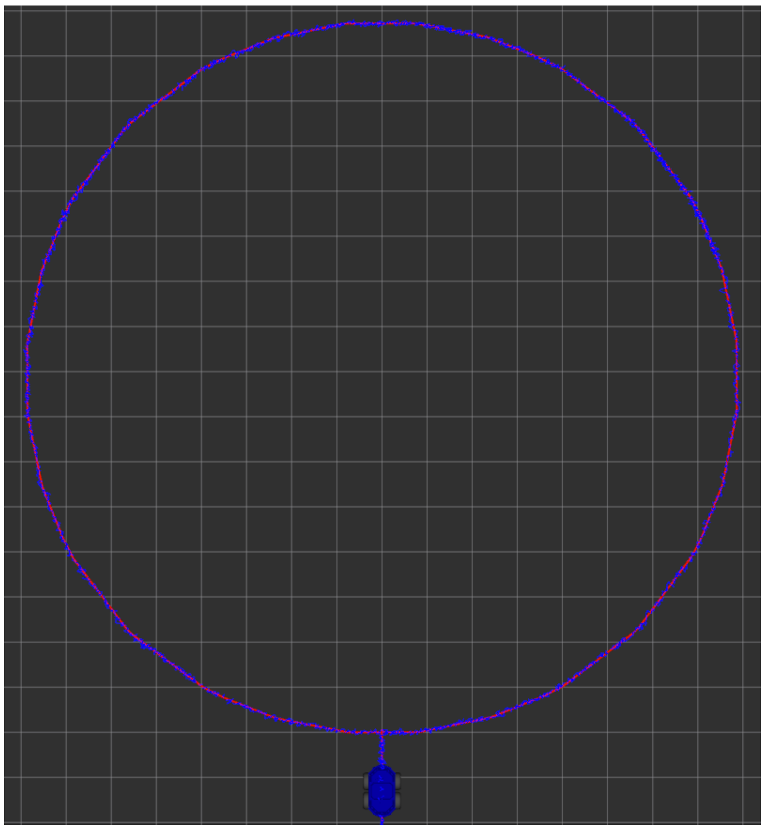


(b)

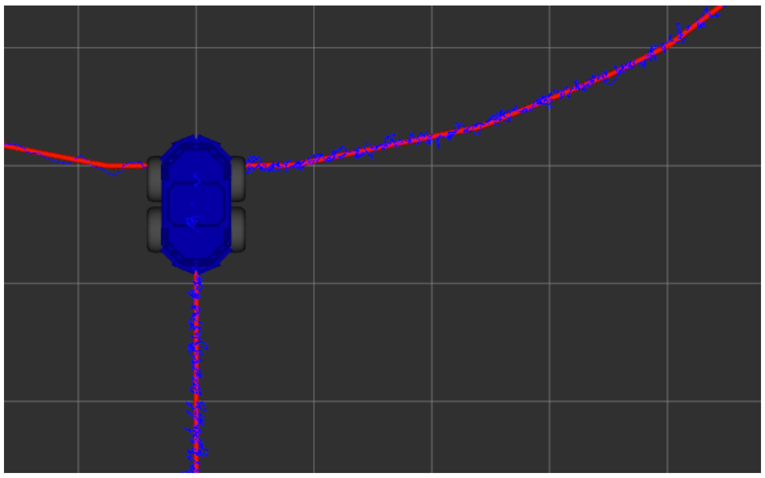
Figure 5.4.1: EKF top and perspective view - Square

In the pictures, it is visible the rover model and the anchors' frames that indicate their location, while the red line represents the exact path and the green one is the EKF. It is clear that the filter is able to track successfully the rover movements, with an evident poorer performance in the z-direction: this behavior will be common for all the paths and filters and it is due to the much more limited span of the anchors in that dimension, being them at only three different heights spaced 0.5 m one from each other, while on the xy-plane the wider distribution allows greater accuracies.

The UKF behavior is instead shown on the circular path in Figure 5.4.2:



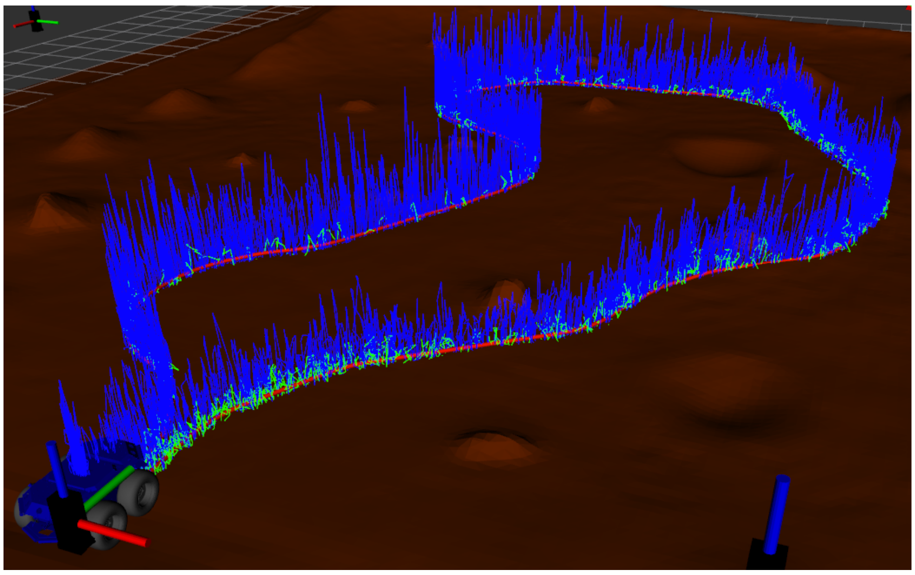
(a)



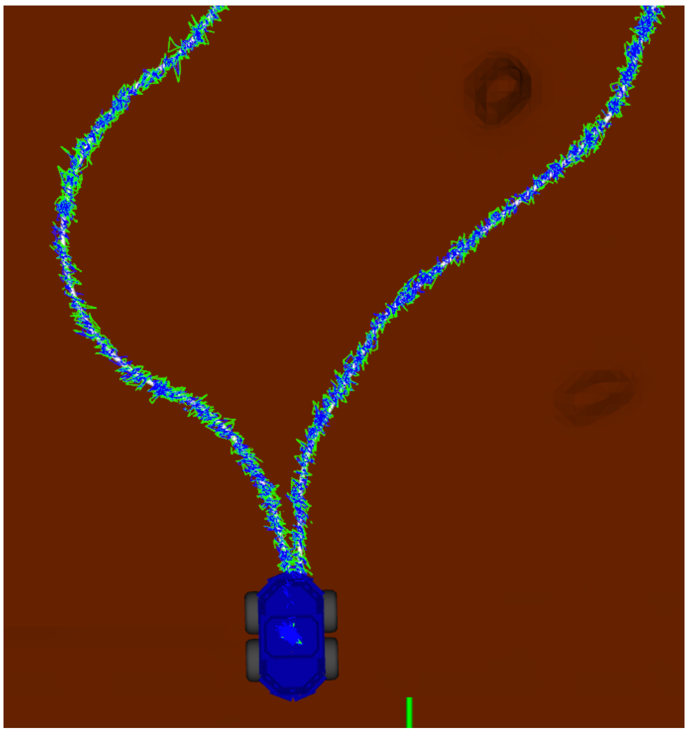
(b)

Figure 5.4.2: UKF top view and detail - Circle

The performance of the UKF is visibly better than the EKF on the xy-plane, with accuracies in the order of a few centimeters. Although filter variations have also been tested on these paths, their regularity does not allow differences in performance to be effectively highlighted, which will therefore be shown with more complex paths. So, it follows the analysis of the first more complex path that also involves the RoXY simulated terrain, where also the anchor cases are visible:



(a)



(b)

Figure 5.4.3: EKF (*green*) vs UKF (*blue*) perspective and top view detail

These pictures make evident that the UKF performs better than the EKF in the horizontal plane, but it suffers from the poor z-dimension anchors distribution much more: this is probably due to the fact that the sigma point distribution accentuates the lack of information in the vertical direction, while the linearization that occurs in the EKF reduce the entity of the error. As it will be shown better later, among the four filter algorithm modifications, only the matrix redefinition and rescaling techniques (hereafter referred to as Redef and Rescal version for convenience) have been found to actually bring improvements to the basic filters performance, while the weighted-outlier-robust (WOR) and thresholding techniques exhibit more or less the same accuracy as the unmodified algorithms. The following figures depict the comparison between the basic EKF and UKF performance and the Redef and Rescal versions:

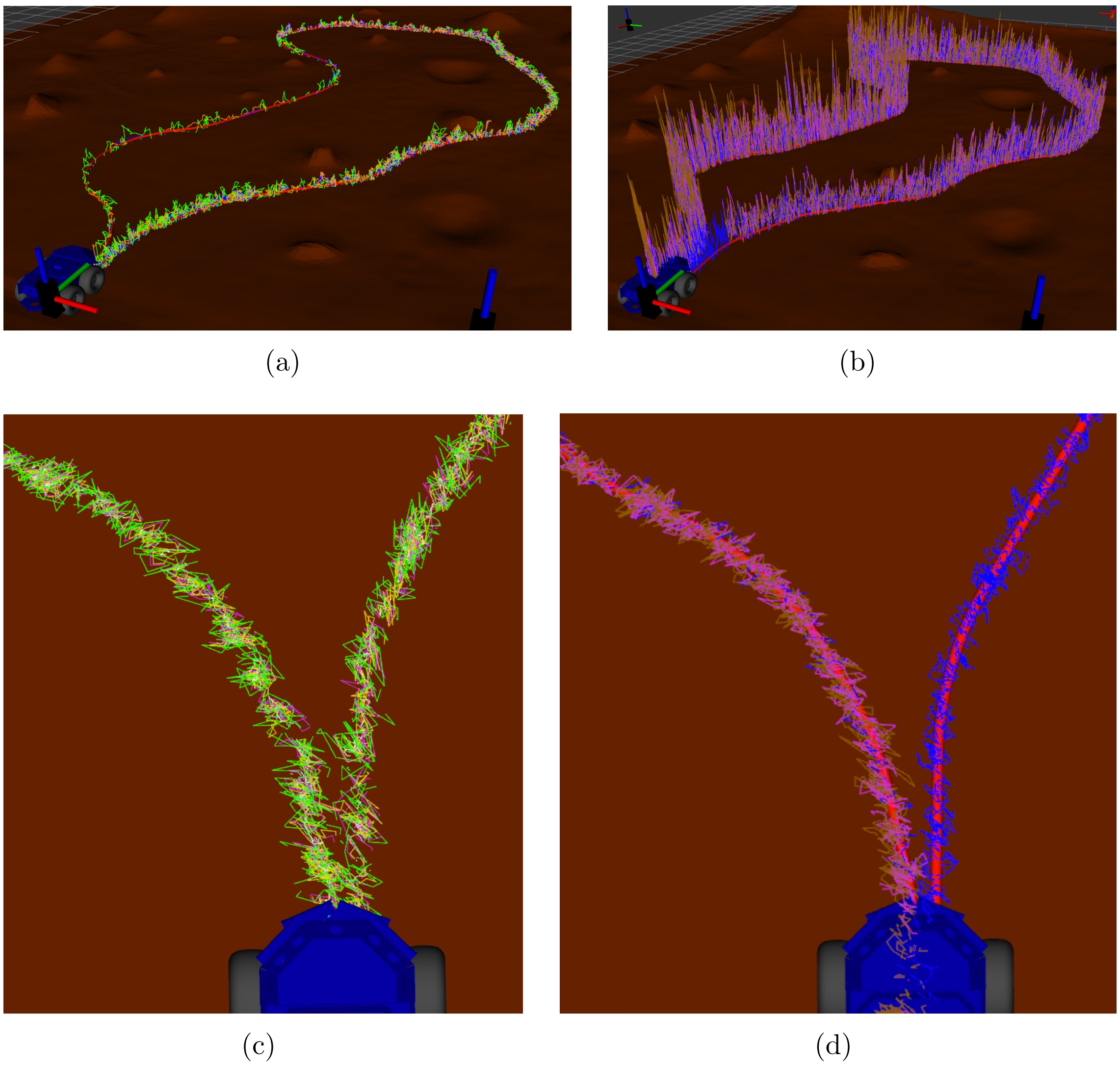
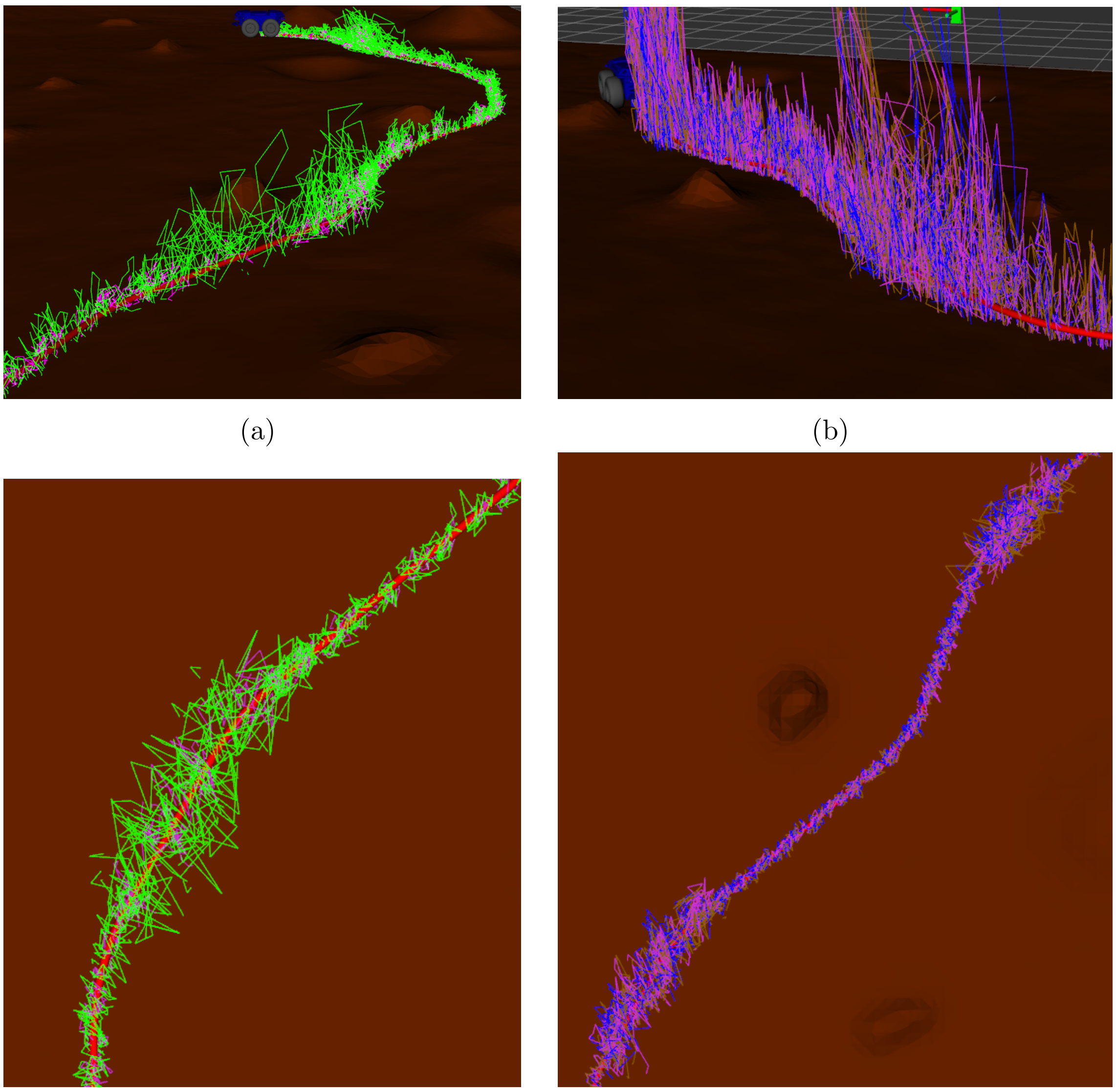

(a) (b) (c) (d)

Figure 5.4.4: EKF and UKF vs Redef and Rescal perspective (a and b) and top view detail (c and d)

The Redef and Rescal EKF are respectively the purple and yellow lines, while the UKF versions are in violet and light brown color. It is visible, even if not markedly, their better performance again on the xy-plane. It is important to state that these algorithms are not meant to improve the basic filters in general, instead they are designed to perform better in presence of outliers; so, the next pictures report their behavior when a 10x noise is added to the ranging measurements only in a small section of the route:



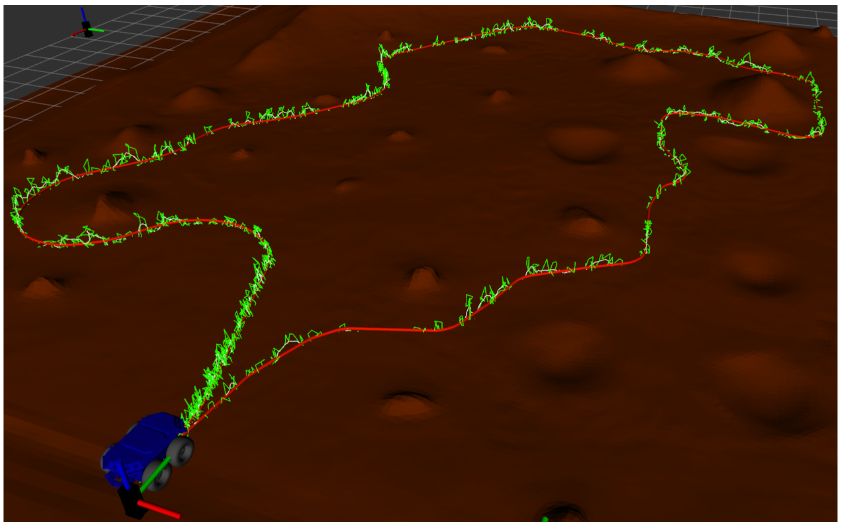
(a) (b) (c) (d)

Figure 5.4.5: EKF and UKF vs Redef and Rescal perspective (a and b) and top view detail (c and d) with 10x added noise

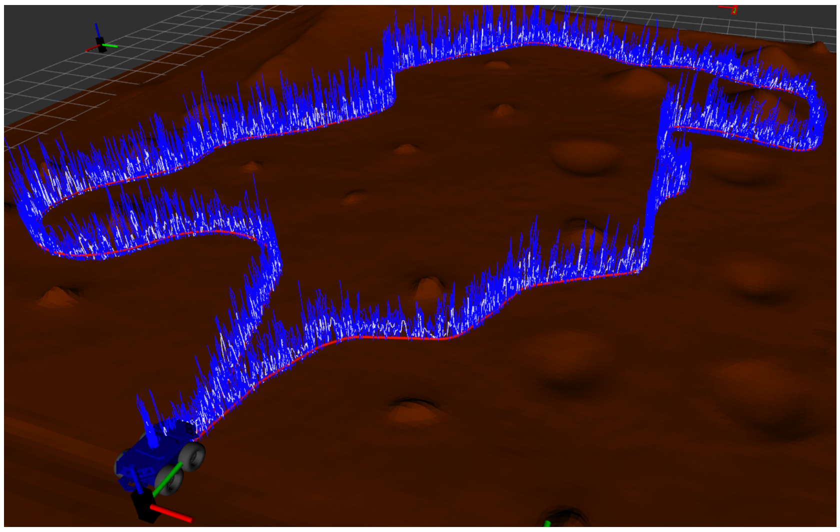
While for the UKF there are no remarkable improvements, the EKF exhibits a quite better performance on the xy-plane, with improvements of up to 15 cm with respect to the basic version, but this time it is the z-direction that reports far superior results, as it is evident from Figure 5.4.5a where the higher noise is almost completely rejected. In the experimental test section, even superior filter performances will be shown in presence of real outliers.

In addition to the already discussed features, smoothing has also been implemented as a moving average with a window of user-configurable size; despite being a simple and well-known technique, it is rather effective, so that in some sections of the path it is indistinguishable from the exact track of the robot, as visible in Figure 5.4.6a and Figure 5.4.6c:

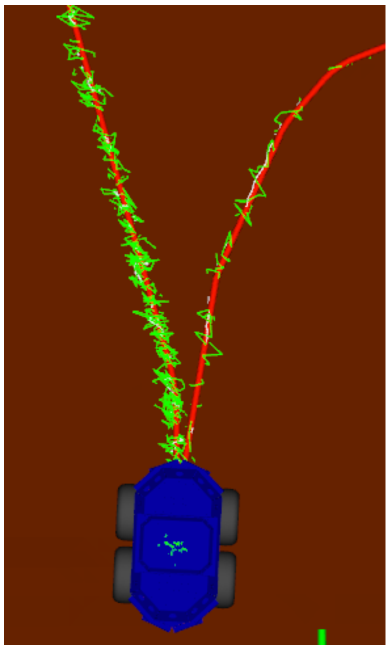
(a)

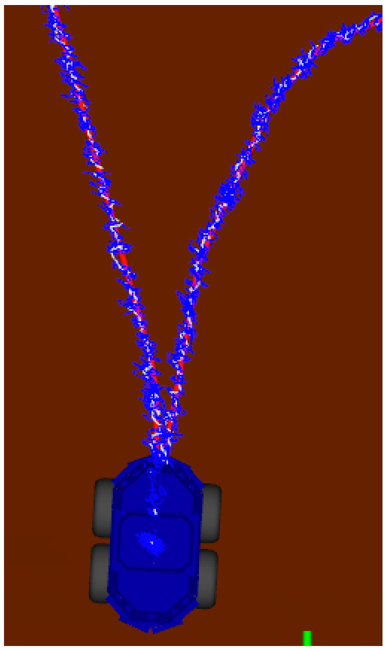


(b)



(c)



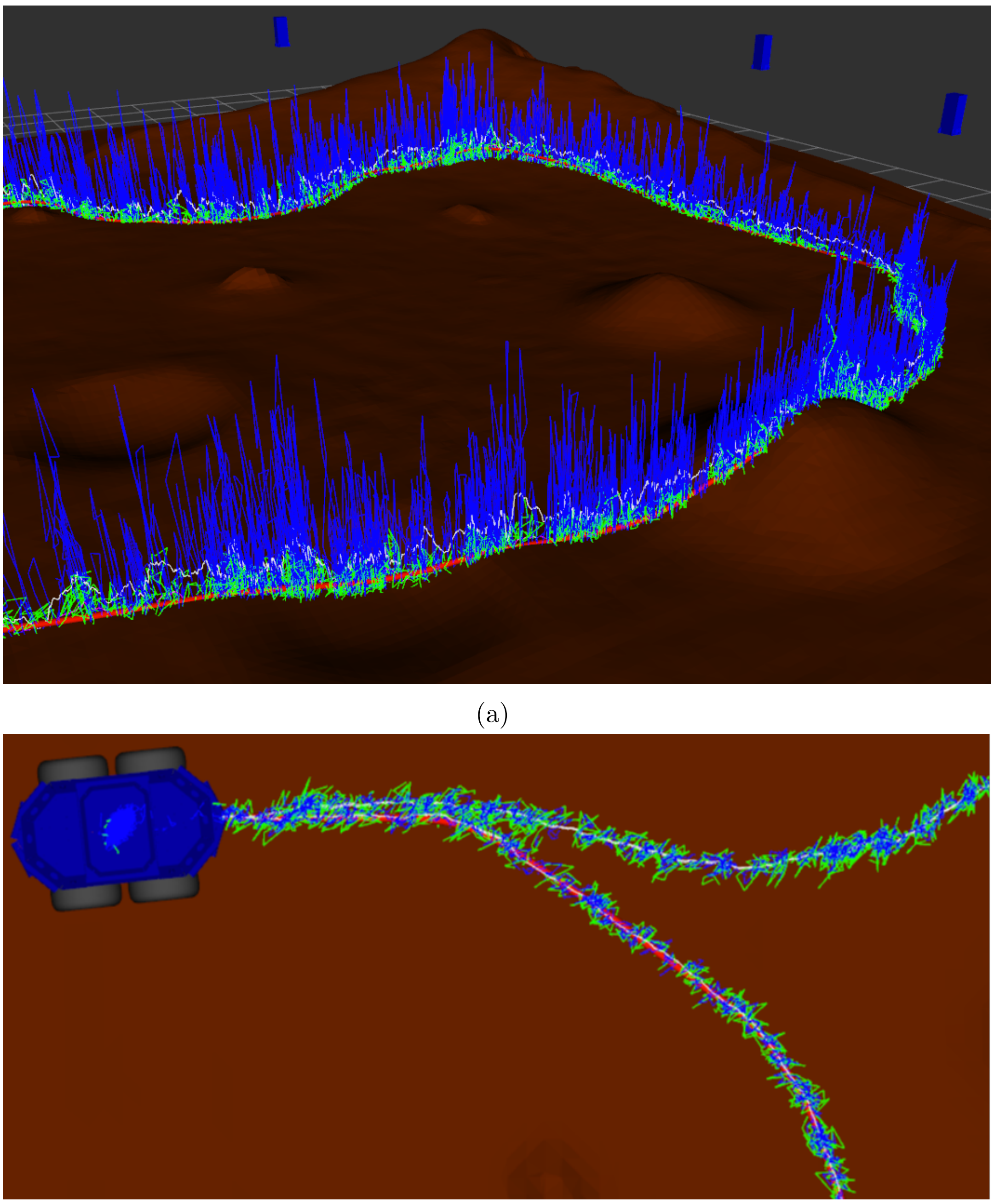


(d)

Figure 5.4.6: EKF and UKF vs Smooth perspective (a and b) and top view detail (c and d)

The smoothed versions (in white), here produced with a window size of 50, demonstrate a much higher accuracy in all three dimensions, with particular regard to the EKF version which is so accurate that it is covered by the exact track for a large part of the route.

Lastly, the third path is shown in the following pictures, to highlight the height changes:



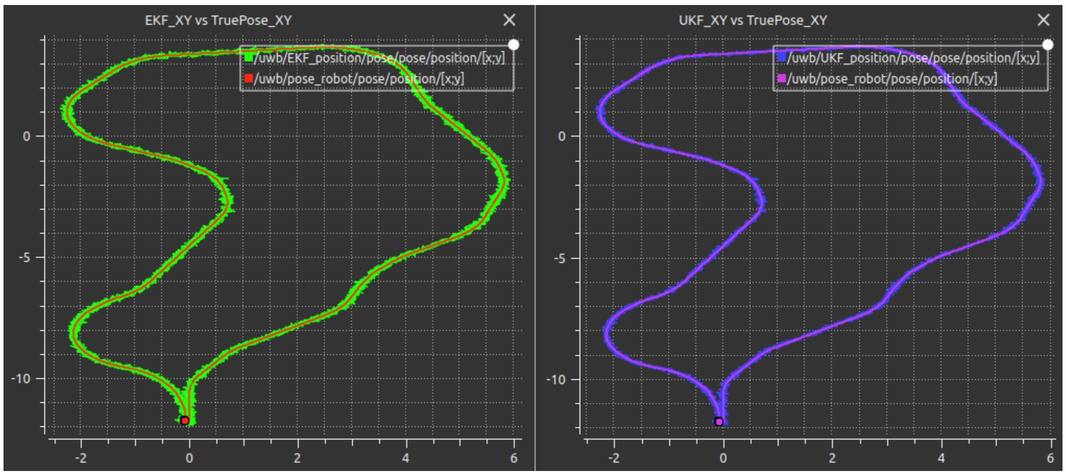
(a)

(b)

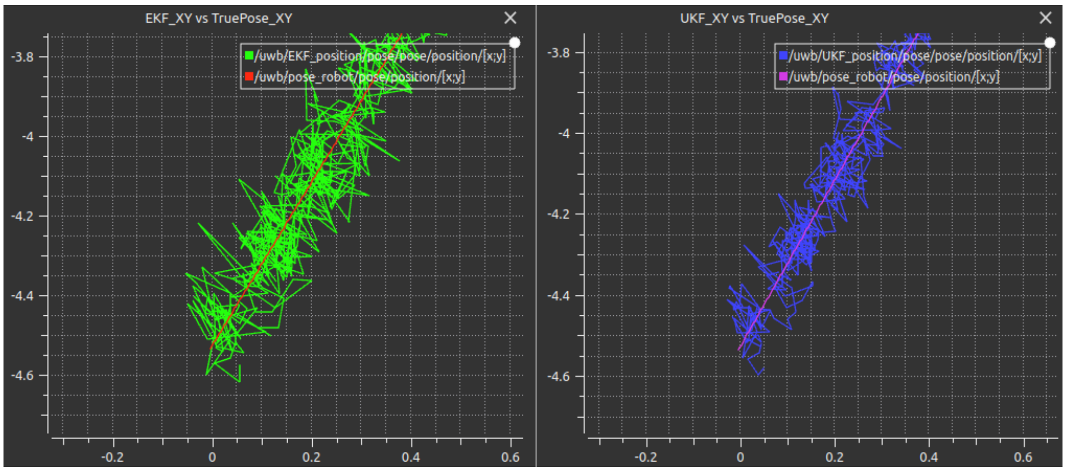
Figure 5.4.7: EKF vs UKF vs Smooth perspective **(a)** and top view **(b)** of a height-changing path

In the figures, there are depicted the EKF, UKF, and their smoothed versions with the same color code as before: the rover runs through a ditch and then a hillock in the bottom part of Figure 5.4.7a and the filters are able to track its movement also in the z-direction, even if with less accuracy.

Numerical data has been acquired from simulations thanks to a visualization tool called *PlotJuggler* that is able to work seamlessly with ROS2 data. The metrics used to evaluate the filter performances mainly are the RMSE and NEES, which directly provide information about the system state, and also the 2D and 3D error between the ground truth and the estimation. The data reported hereafter refer to the path depicted in Figure 5.4.3a. The following picture highlights the paths of EKF and UKF with respect to the exact reference path:



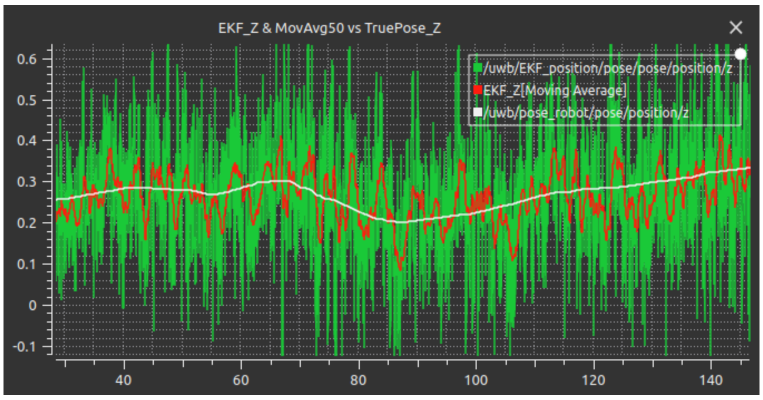
(a)



(b)

Figure 5.4.8: EKF vs UKF paths **(a)** and detailed view **(b)**. Values in meters

Figure 5.4.8b reveals that the two-dimensional distribution of the poses estimated by the UKF is quite narrower than its EKF counterpart. This behavior is actually reversed for what concerns the vertical direction, as shown in Figure 5.4.9:

(a)


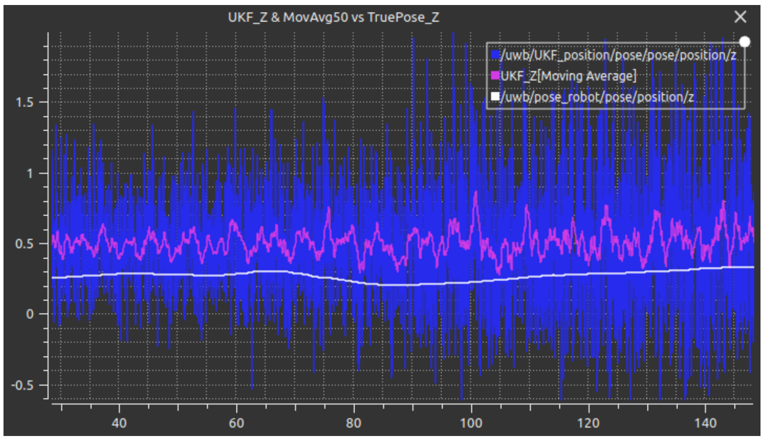
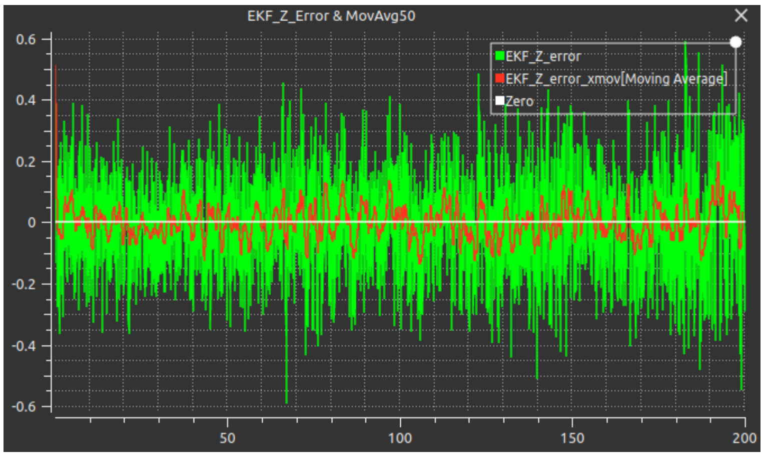
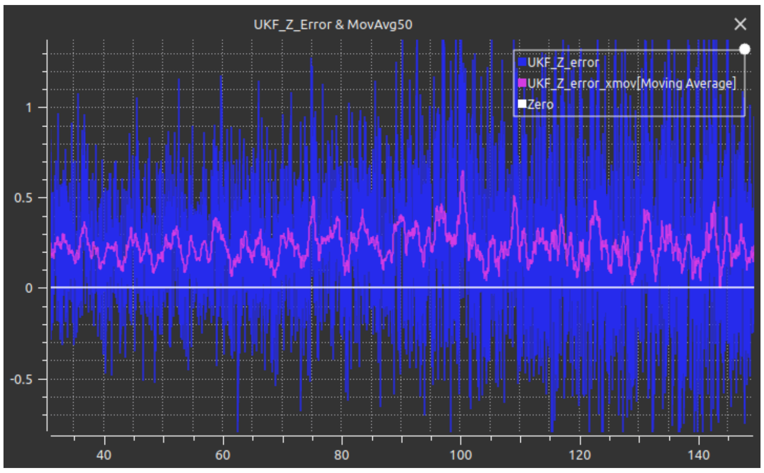
(b)


Figure 5.4.9: EKF vs UKF vertical position estimation. Values in meters

In both pictures, there are also provided the smoothed plots obtained with the same moving average employed in the previous pictures. From the first picture, one can notice that the error in the z-direction is much higher than the horizontal counterpart, as expected, while the second one reveals a far worse raw curve of the UKF with respect to the EKF and an almost constant offset in its estimation, as already observed and discussed in Figure 5.4.3. For better analysis and thanks to the auxiliary functionalities of *Plotjuggler*, the errors both in the vertical direction, on the xy-plane, and relative to the 3D distance were calculated and plotted:

(a)




(b)

Figure 5.4.10: EKF vs UKF vertical position estimation error. Values in meters

and for sake of completeness, a 20 cm offset version of Figure 5.4.10b is reported hereafter, to emphasize the comparable results, if not better in some parts, with the EKF for what concerns the moving average if the offset is taken into account:

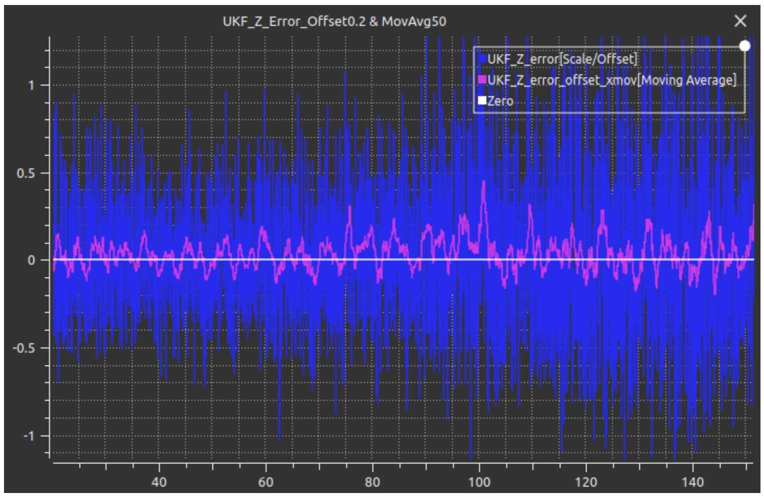


Figure 5.4.11: UKF vertical position estimation error with 20 cm offset. Values in meters

In addition, for both EKF and UKF, two-dimensional and three-dimensional distances absolute errors between the curves and the ground truth have been calculated, always alongside their moving average. From Figure 5.4.12 it is crystal clear how much the out-of-scale error in the vertical direction highly affects the three-dimensional error: for the EKF, the error actually doubles going from 6 cm in 2D to more than 11 cm in the 3D case, while the UKF exhibits both a lower error of less than 5 cm in the xy-plane and a larger 3D error of approximately 30 cm, caused indeed by the aforementioned vertical offset.

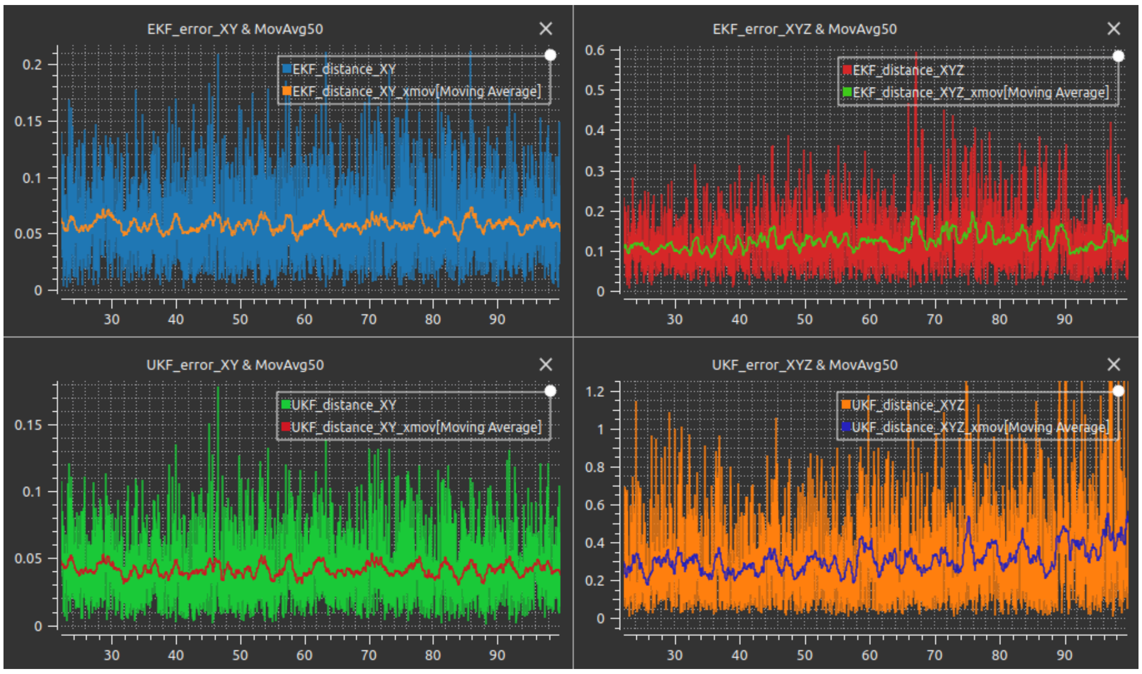
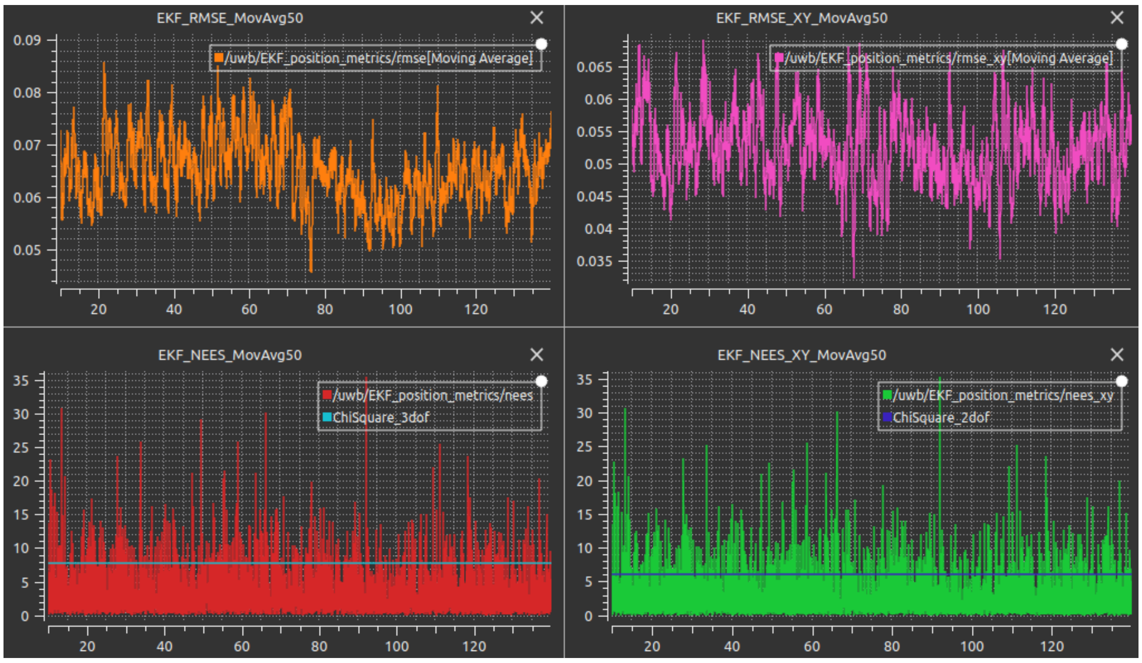


Figure 5.4.12: EKF and UKF bidimensional and tridimensional absolute errors. Values in meters

Finally, the RMSE and NEES metrics are discussed, with Figure 5.4.13a and Figure 5.4.13b showing the two metrics (RMSE averaged) concerning respectively the EKF and the UKF:



(a)

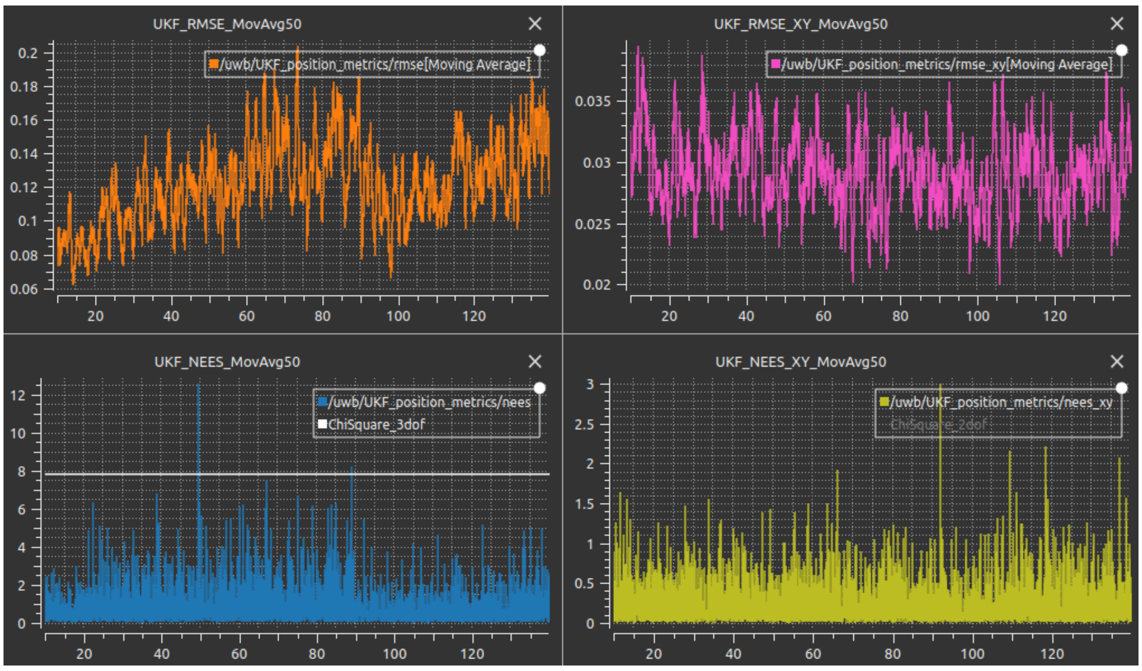
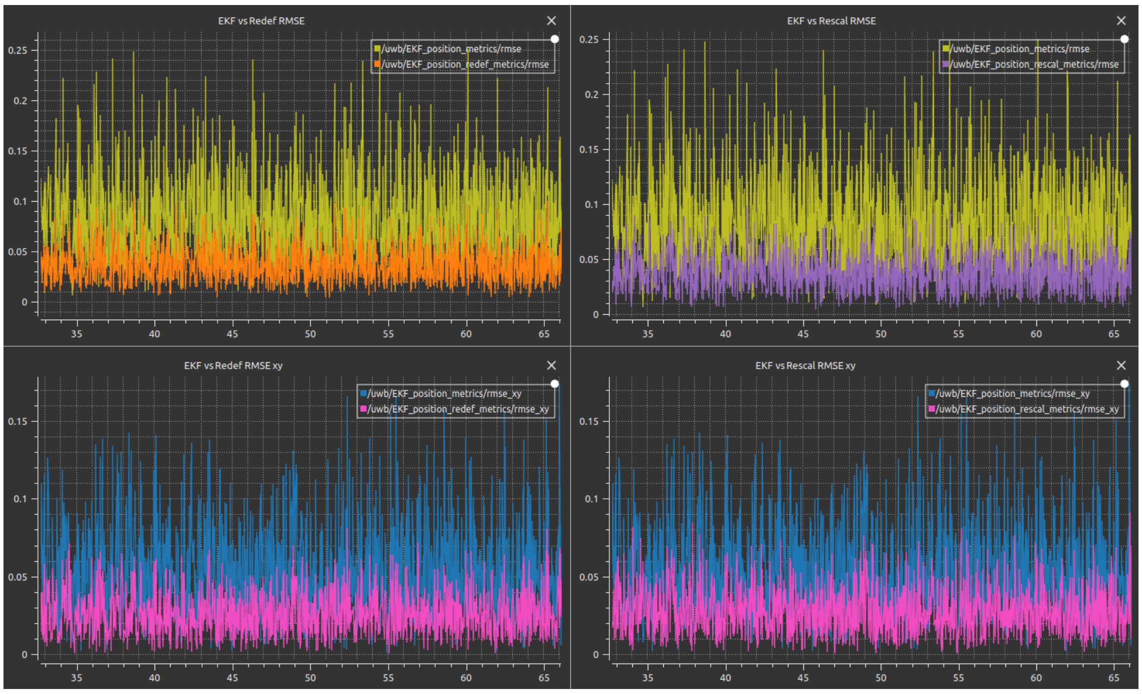
(b)


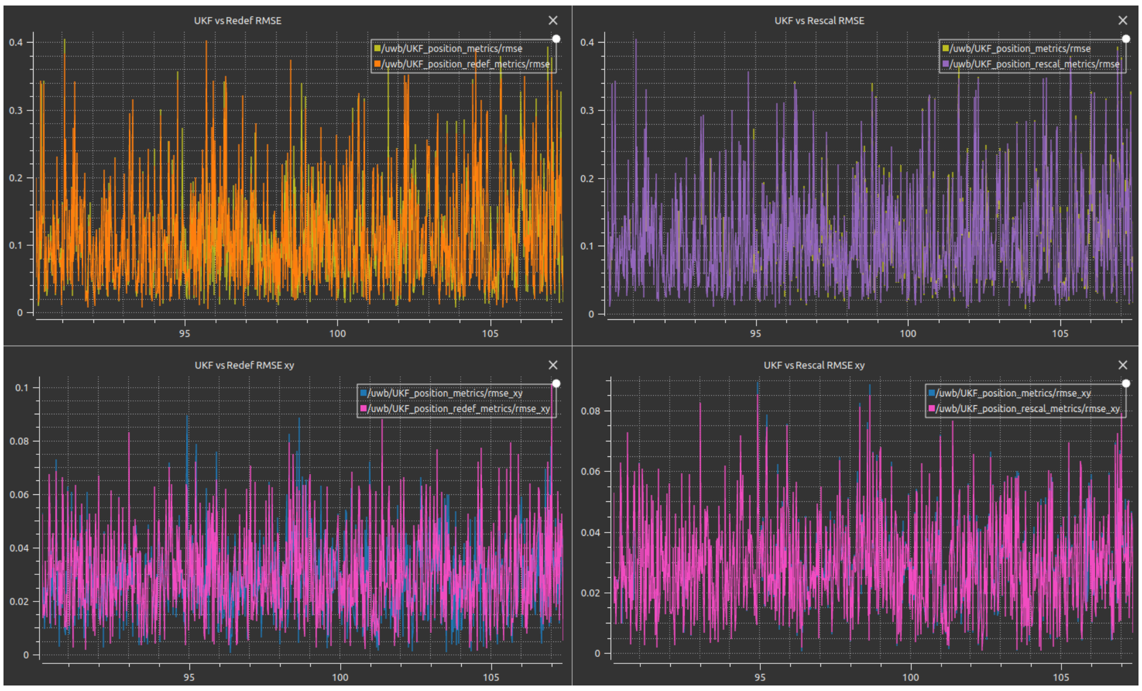
Figure 5.4.13: EKF and UKF Root-Mean-Square Errors (RMSE) and Normalized Estimation Errors Square (NEES). Values in meters

Comparing the two pictures, much information can be obtained: first of all, the overall higher RMSE of 13 cm average exhibited by the UKF is surely due to its wider distribution in the vertical direction with respect to the EKF, where it is almost stable at 6.5 cm; in fact, considering only the bidimensional case, the UKF performs better, remaining approximately constant around 3 cm, while the EKF presents an average of 5.5 cm. Furthermore, the NEES curves offer insights into the estimation goodness, comparing it with the correspondent χ-squared value, in this case considered at 3DoF and 2DoF for the 3D and 2D curves: the EKF produces a quite less reliable estimate if compared to the UKF, and this behavior originated from both the linearization error of the EKF and the fact that an error on the xy-plane, where the distribution of the anchor is wider, is weighted more than a correspondent error in the vertical direction, where a worse estimation is not so much due to the model but to the nodes distribution instead. In practice, however, the overall performance of the UKF is not a match for the EKF, because of the former totally out-of-scale vertical error. In Appendix A there are reported these same metrics but with respect to the ground truth position, obviously displaying the same behavior just discussed, but with little higher errors.

A last analysis has been performed on the improvement brought by the filter variations: the complete set of curves can be found in Appendix A, while the following part of the section will report only the Redef and Rescal RMSE metrics and the Smoothed performance, having them the most interesting results. Figure 5.4.14 highlights the differences between the EKF and UKF variations behavior:



(a)



(b)

Figure 5.4.14: EKF **(a)** and UKF **(b)** RMSEs vs Redef and Rescal RMSEs. Values in meters

The EKF variations seem to perform much better, displaying more than a halving of the RMSE value, than the correspondent UKF ones, which do not observe changes at all; actually, paying attention to the scales of the plots, the truth is a little different: it is surely true that the EKF benefits from the implementation of matrix adaptiveness, but its absolute performances in 2D "only" reach almost the same level of the UKF ones, which were better already without adaptiveness, while only in 3D the difference is really evident because of all the causes already discussed. The reason behind this behavior is to be found in the intrinsic noise introduced by the ranging activity, rather than in the error contributed by the filter: in fact, the filter variations exhibit a mean RMSE of approximately 3 cm for the UKF two-dimensional case, which also extends to the 3D case for what concerns the EKF, and which almost corresponds to the user-defined noise artificially injected in the ranging simulation and constituting a global minimum for the error performance obtainable with the system.

The performances of the smoothed versions of the filters have not been reported herein, since their accuracy fully mirrors the behavior of the moving average curves appearing in Figure 5.4.9 with the respective errors, but they can be found in Appendix A. However, it is possible to say that, despite being a fairly simple technique, the results are rather good especially in the UKF case, where the distribution of estimated poses in the z-direction is more than five times less wide than the basic version with respect to their respective mean values.

5.5 Anchors' calibration

Before deepening into the discussion of the tests performed in the RoXY facility playground, it is important to report how the anchors' calibration has been carried out, since it was quite a challenging task. As mentioned earlier, the configuration code for the DMW3000 module involves the setting of many parameters, such as the CPU processing time, to tune to the minimum possible until errors arise, the inter-ranging delay period, the inter-frames delay, the receive response timeout to wait before raising timeout exception, and so on; the most critical one, though, is the TX and RX antenna delay, which acts as a correction factor due to delay introduced by internal circuits, external components and the specific board that runs the code, that has to be applied to the Time of Flight, so to the estimated range, to which is inversely proportional.

It has been noticed that its variation led to quite different estimated measurements, even considering their mean, during initial tests at a precise tag-anchor distance of 1 m, measured using the Bosch laser tool: specifically, considering that the default value of antenna delay as provided by the manufacturer is 16385, slight variations in the range of tens led to mean value changes of ± 2 cm. So, after the first round of calibration tests, the anchors presented very different antenna delay values, even of hundreds of units, but showed an almost perfect ranging estimation at 1 m, with mean errors of a few millimeters. Unfortunately, however, when it has been tried to measure a longer distance of 2 m, the results were completely wrong, such that the required antenna delay to get back to an acceptable precision differed again by hundreds of units. This evidence led to a complete rethinking of the calibration procedure, which during its carrying out has been changed repeatedly until a reliable strategy was found, and also gave the opportunity to perform another important action, error mapping.

The new calibration strategy was the following: in order to get the most reliable values, the calibration has been performed in almost the final setup conditions, placing a total of 8 anchor cases on the external perimeter walls, precisely 7 on the left wall, spaced not completely evenly but at almost 4 m distance to each other, and 1 on the bottom one; in particular, the latter was reserved to the tag and has been obtained from a wrong 3D printed case, which was opened up to expose the antenna as on the rover it would have been, while the other seven were exactly reproducing the final setup conditions, including the presence of the anchor case. To calibrate an anchor, it has been placed in each one of the seven positions in a round fashion, to be able to get its behavior at different distances, which varied in a range between almost 5 and 24 meters, with a further 3 m distance obtained moving the tag to the first position and the anchor to the second one and another 28 m distance obtained later in time, after the placement of all the anchors in the final configuration. All the positions have been acquired precisely by means of the Leica laser tracker, and their relative distances have been computed by the SpatialAnalyzer® software. Here follows the software representation:

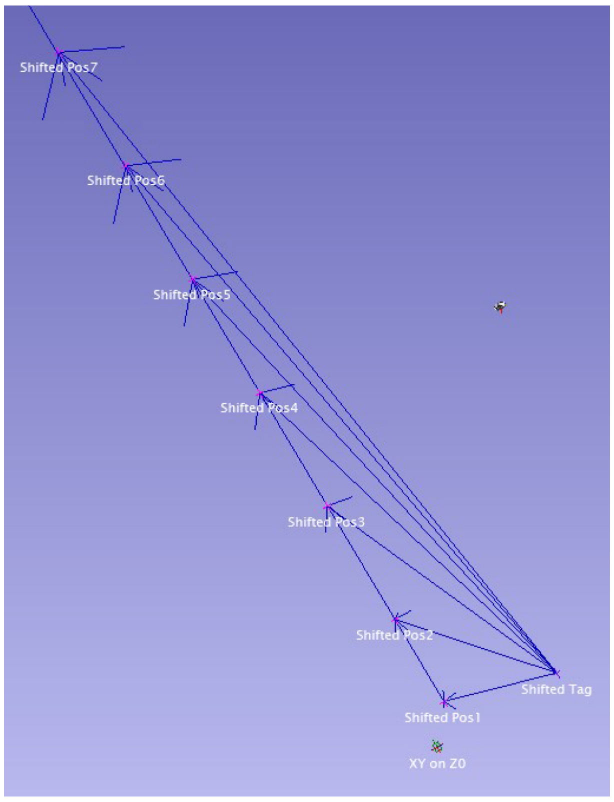


Figure 5.5.1: Calibration positions acquired with Leica AT403 and SpatialAnalyzer

Many ranging exchanges were performed: each calibration round consisted of 20000 measurements then considered in groups of 1000, and mean value, variance, standard deviation, maximum error, and average error has been computed for each group. Then, the best, average, and worst cases have been evaluated per group, and saved in a spreadsheet. The procedure was repeated for all the different positions, and an aggregate table has been produced from the data: the points obtained from the average errors of the cases with respect to the relative distances have been plotted and interpolated with a 3rd-degree polynomial regression, a 5th-degree one and a 3rd-degree spline.

A complete round of measurements for a single anchor has been performed a few times, in order to see how the interpolated curve reflected the changes in the antenna delay values, and the trial stopped at the value that provided the flattest curve with a mean value closer to zero. Then the procedure has been repeated for each anchor, with the representation of one ranging measurements distribution for a single range and anchor shown in Figure 5.5.2 to highlight the excellent fit with a Gaussian distribution, while a more detailed dataset is reported in Appendix B: in particular, there are reported the statistics of the first three anchors, and both the ranges distributions and the error map functions of the first six anchors.

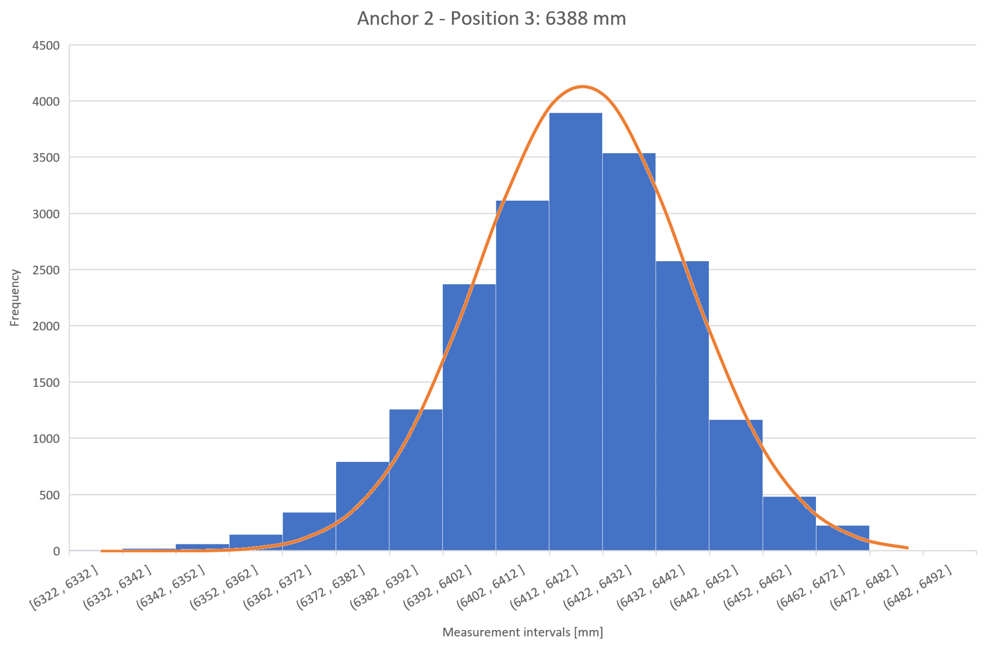


Figure 5.5.2: Ranging measurements distribution of Anchor 2 at 6.388 m

Fortunately, this different strategy led to far better results, such that many anchors actually shared almost the same antenna delay value (approximately 16360) correspondent to the best curve fit, and also variations in this value did not generate the difference of measurements experienced in the previous calibration setup, which was evidently unfit for the order of magnitude of the ranges that the antennas were meant to measure. However, even if the interpolated curves were very similar in shape and quite similar in values, they were far from being flat or zero-meaned, as one of them reported in the following picture clearly highlights:

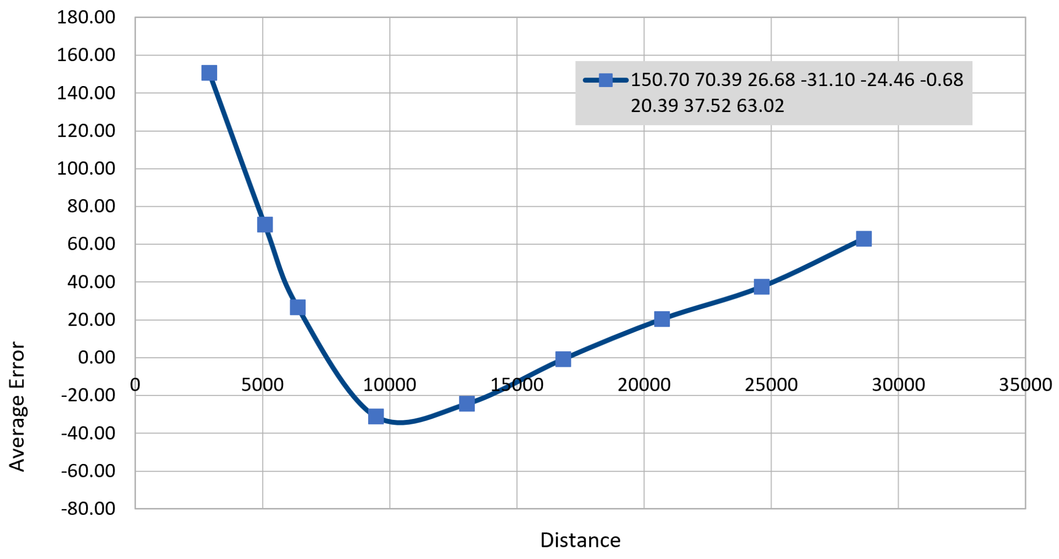


Figure 5.5.3: Error map of a UWB anchor, values in mm

It is a spline interpolation, and it is evident that at small distances, which 1 meter clearly was, the error curve is rather steep and the sensor performances are then very poor. Nevertheless, a similar curve has been generated for each anchor and the three interpolation curves mentioned before have been implemented into the filters' code in order to compensate for the raw measurement at a certain distance with the interpolated error and then provide it to the localization algorithm.

5.6 Field tests and Results

The final setup of this UWB system consists of 12 anchors distributed on the RoXY perimeter walls as evenly as possible in groups of 3, paying attention not to cover the Line-of-Sight and thus avoiding placing any of them behind the large slope in the top left corner of the terrain. Precise measurements have been performed again by means of the Leica laser tracker, post-processing them to get rid of the offset introduced by the prism and to refer them all to a common origin chosen and measured in the bottom left corner; the SpatialAnalyzer® software representation is reported in Figure 5.6.1, where the central marker represents the Leica tool position, the bottom left cross indicates the chosen origin, and the lines that connect the anchor markers are the orthogonal representations of the planes on which the same side anchors lay, which provide the normal direction to be considered to remove the prism offset :

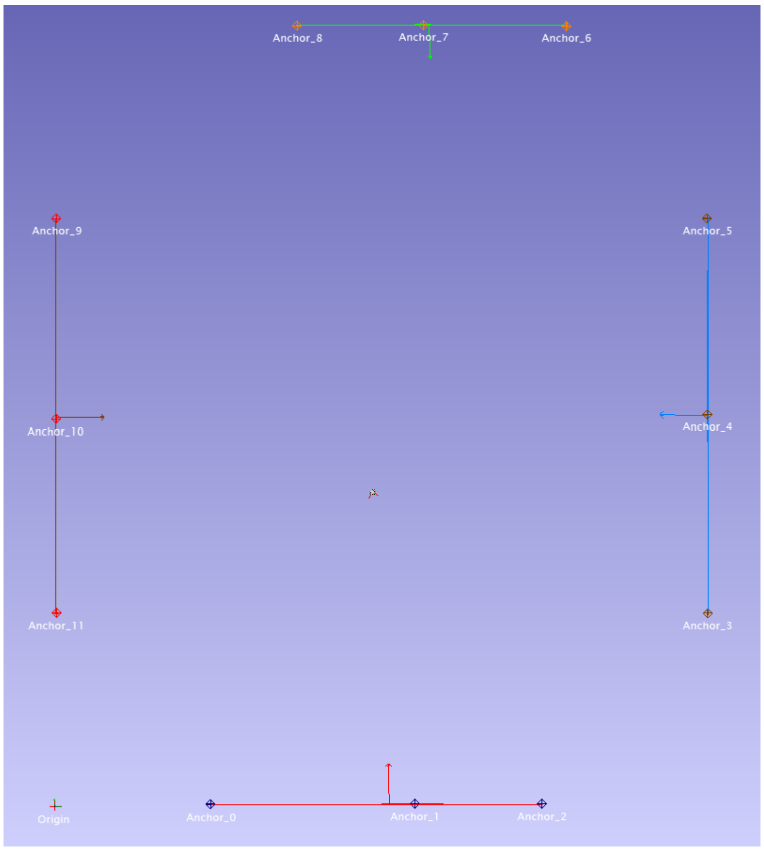


Figure 5.6.1: Anchor positions acquired with Leica AT403 and SpatialAnalyzer

Then, the rover has been equipped with the UWB tag mounted on the 3D printed rover stand and the AN Spatial GPS just next to it, all of them powered via a USB hub connected to the rover. Figure 5.6.2 depicts the final rover setup:

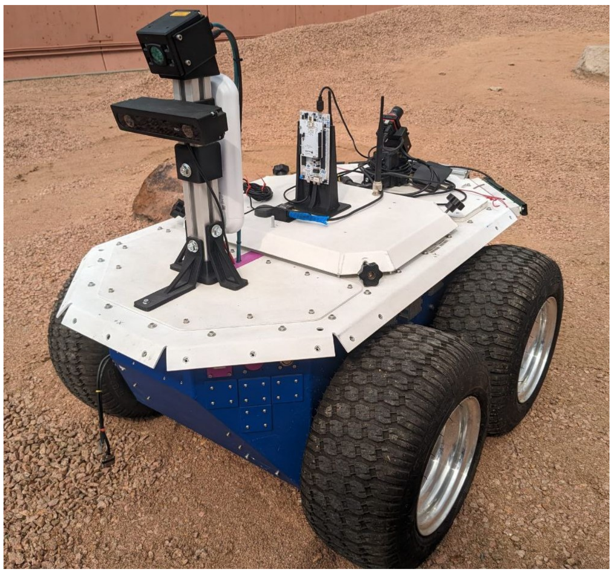


Figure 5.6.2: Rover final setup

Because of technical issues with the ROS2 environmental configurations due to unsolvable differences of the reference frames in the locomotion and sensor-fusion rover localization system (whose reference frame changes at every run) and the UWB system, it was not possible to exploit Plotjuggler to obtain nice data plots as shown in the simulation section; to overcome this limitation, the paths estimated by the own robot localization and the UWB system have been only displayed using the RViz tool, with a denser grid to approximately estimate the errors, while in order to obtain more precise numerical data, single-point positions have been estimated by the UWB system with a ground truth provided by the Bosch laser tool accurate measurements.

Many paths with different lengths have been tested, with no one following a regular shape due to the irregular terrain configuration, of which three of them are herein reported; Figure 5.6.3 displays a direct comparison between the EKF and UKF two-dimensional performance with respect to the path obtained by the robot sensor-fusion system involving the stereocamera, ToF camera, wheel odometry, and GPS:

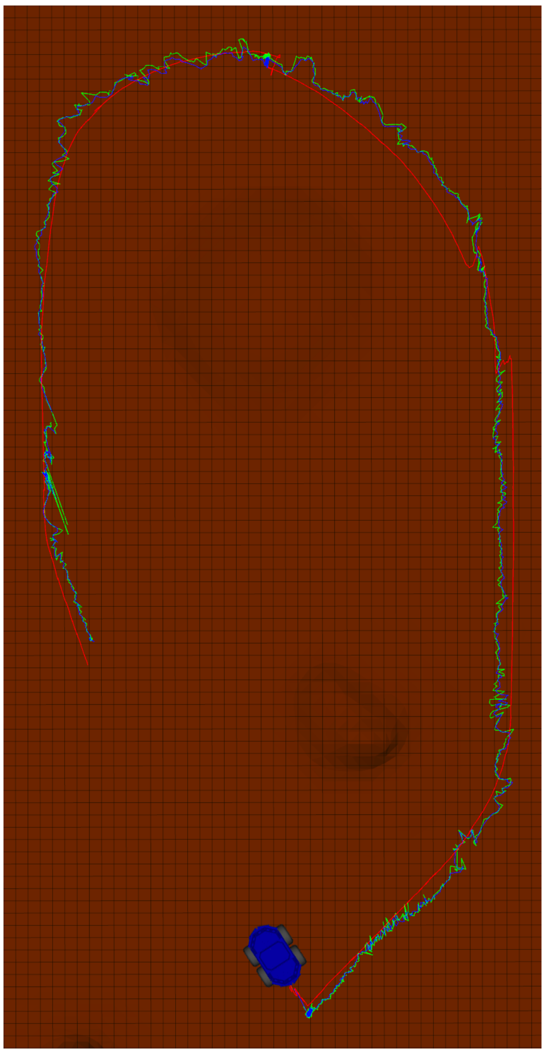


Figure 5.6.3: EKF (*green*) vs UKF (*blue*) vs Reference (*red*) - First path, top view

Much information can be grasped from the picture: first of all, the red line that should act as a reference is undoubtedly almost unaffected by noise, but does not actually represent the exact path of the rover since in some small sections it experiences drifts and shifts (top left corner); nevertheless, it is enough reliable to be taken as a reference, and in doing so, considering that each square of the grid has 10 cm side length, both the filters seem to produce quite a good tracking of the robot, with estimation errors not above the 10 cm for most of the path. It should be noted that the paths are much less noisy than they were in the simulation pictures, and this is due to the different updating frequency of the real system, approximately 20 Hz, from the simulated one, 50 Hz, the former therefore generating fewer estimations and so less noise per unit time. Furthermore, both the filters have produced some outlier estimations during the field test as it was foreseeable, quite different from the increased noise test performed in the simulations, and they will be discussed shortly.

A whole different situation is observed considering the vertical position estimation, shown in Figure 5.6.4:

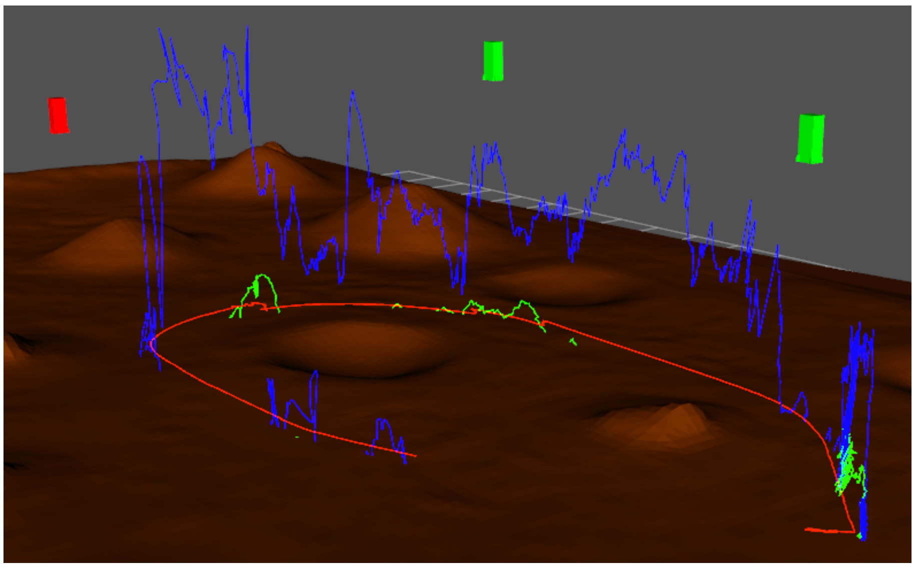


Figure 5.6.4: EKF vs UKF vs Reference - First path, perspective view

In the top view picture, the ground level has been artificially lowered in order to show the complete paths as generated by the filter, but in reality the vertical estimation is such that the EKF and UKF paths are respectively below and very much above the ground, so they can be seen whole from this perspective view. The errors are of a few tens of centimeters for the EKF, while they get to the order of a meter for the UKF: this phenomenon was expected and it is due to the fact that the vertical uncertainty of real system estimations is much higher than their correspondent horizontal one, again because of the scarce distribution of the anchors in the vertical direction; differently from the simulation, where the artificial noise was injected equally in the three direction and definitely followed a Gaussian distribution much more ideal than the real noise does, the vertical fluctuations are far more accentuated.

To highlight the effectiveness of the outlier rejection of the filter variations, the second path is reported hereafter:

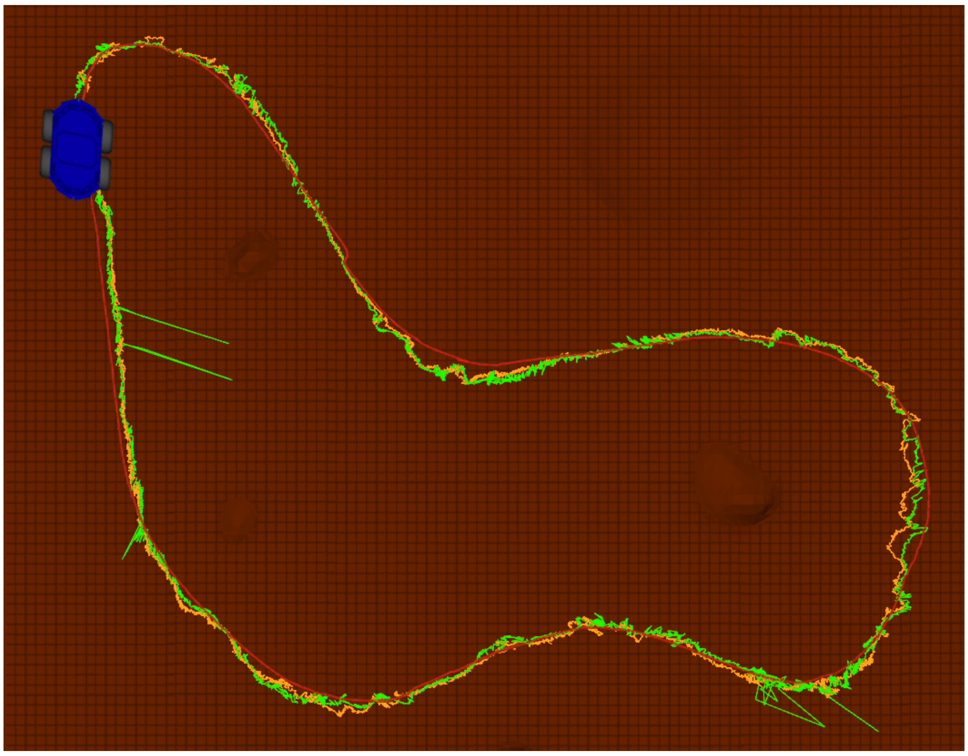
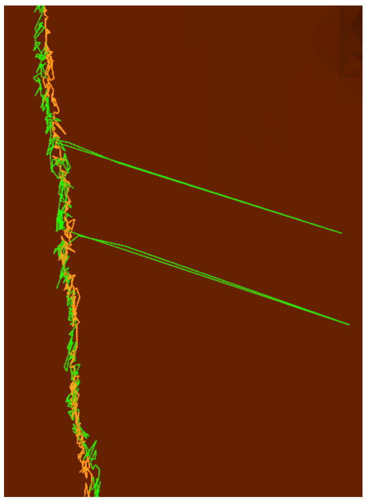
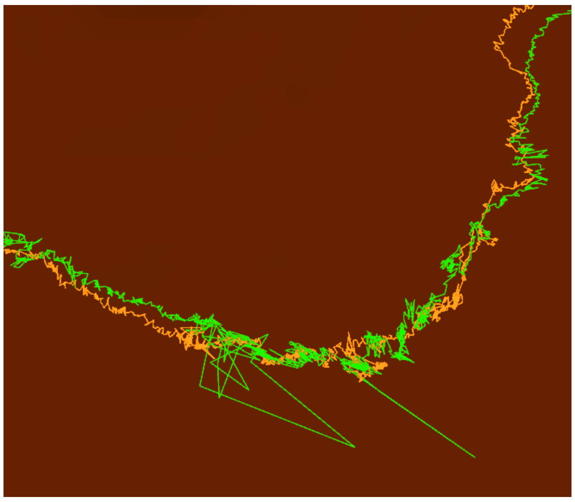


Figure 5.6.5: EKF (*green*) vs Redef (*orange*) vs Reference (*red*) - Second path, top view

The output of the modified filter, the Redef one in this picture but the Rescal one is pretty much equivalent, is not so different for what concerns the errors with respect to the reference path, but the real improvement verifies exactly in the sections affected by outliers: the filter is able to correctly identify the presence of outliers and successfully reject them, as it is more clearly visible from Figure 5.6.6.

(a)





(b)

Figure 5.6.6: EKF vs Redef - Second path, top view detail

The same happens with the UKF versions, which, however, present the exact same estimation of the basic filters almost all along the path:

(a)

(b)

(c)

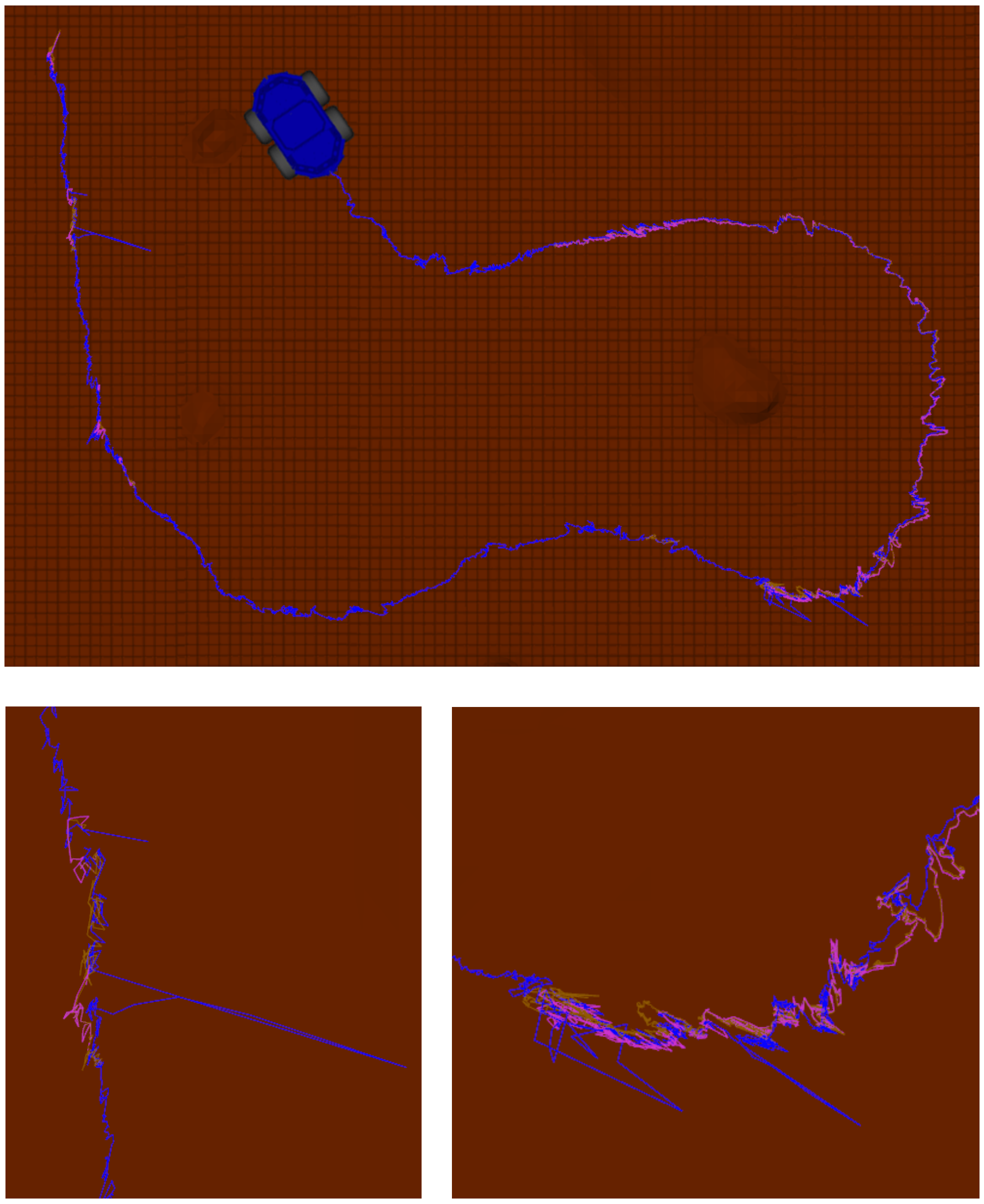


Figure 5.6.7: UKF (*blue*) vs Redef (*purple*) vs Rescal (*light brown*)- Second path, top view **(a)** and details (b and c)

In this case, the paths of both the outlier rejection algorithms are reported, and one can see how both of them completely reject the erroneous estimations.

For sake of completeness, the UKF smoothed version of the second path is reported in Figure 5.6.8:

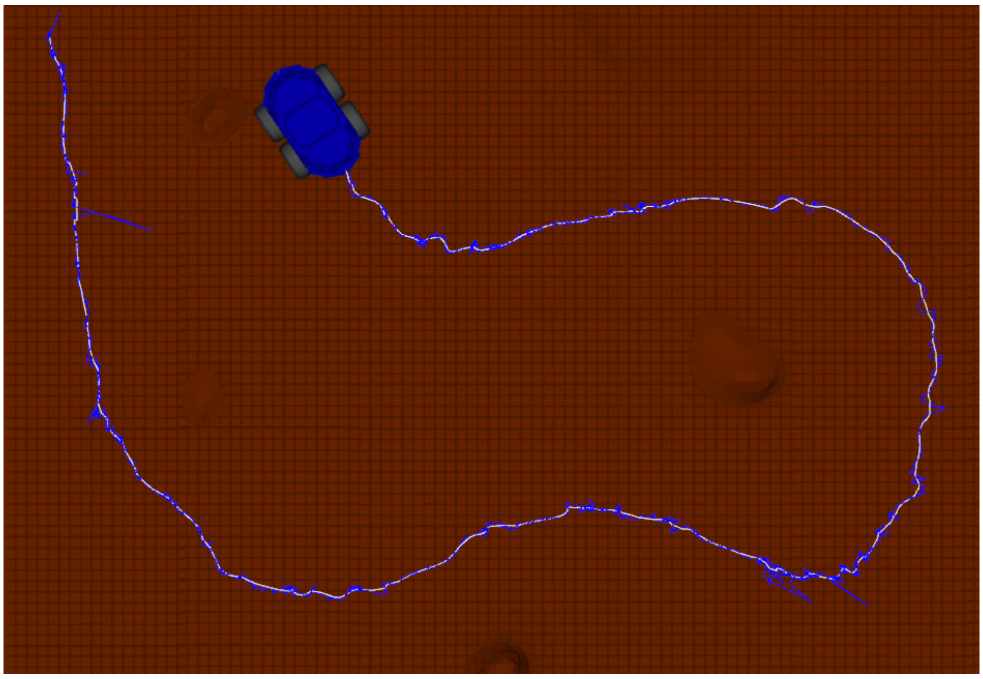


Figure 5.6.8: UKF vs Smooth - Second path, top view

It is interesting to note that apart from obviously increasing precision and reducing the noise of the basic UKF, the smoothing also constitutes a valid and simple technique to get rid of outliers but only provided that their occurrence is locally limited in time and randomly distributed in space, otherwise they would still influence the average path and so the final result.

The error map-aided measurement correction discussed in the previous section has been tested within the third path, which also was the longest and more complex one. Figure 5.6.9 shows the complete path with the correspondent EKF estimation, while the detailed view of Figure 5.6.10 reveals the effect of measurement compensations on the position estimations: it is notable how in the top central section of the path, what should be the reliable reference route actually experience unusual drifts (probably due to exceptionally high background light conditions, which highly affects the stereovision) while the UWB system provides instead quite a straight path.

For what concerns the error map feature, from Figure 5.6.10 it can be noticed, even if the paths are hardly distinguishable, that there is no real improvement in the filter estimation: this makes sense if one considers that the maximum error correction is a bit more than 15 cm and the filter already works out discrepancies of that entity, so it is hypothesized that the noise injected by the filter naturally overcomes the eventual reduction that the ranging error correction provides. This behavior is partly reassuring, meaning that the filter works well enough to admit a certain range of ranging errors without affecting too much the performance, but on the other hand it nullifies all the attempts to improve sensors accuracy.

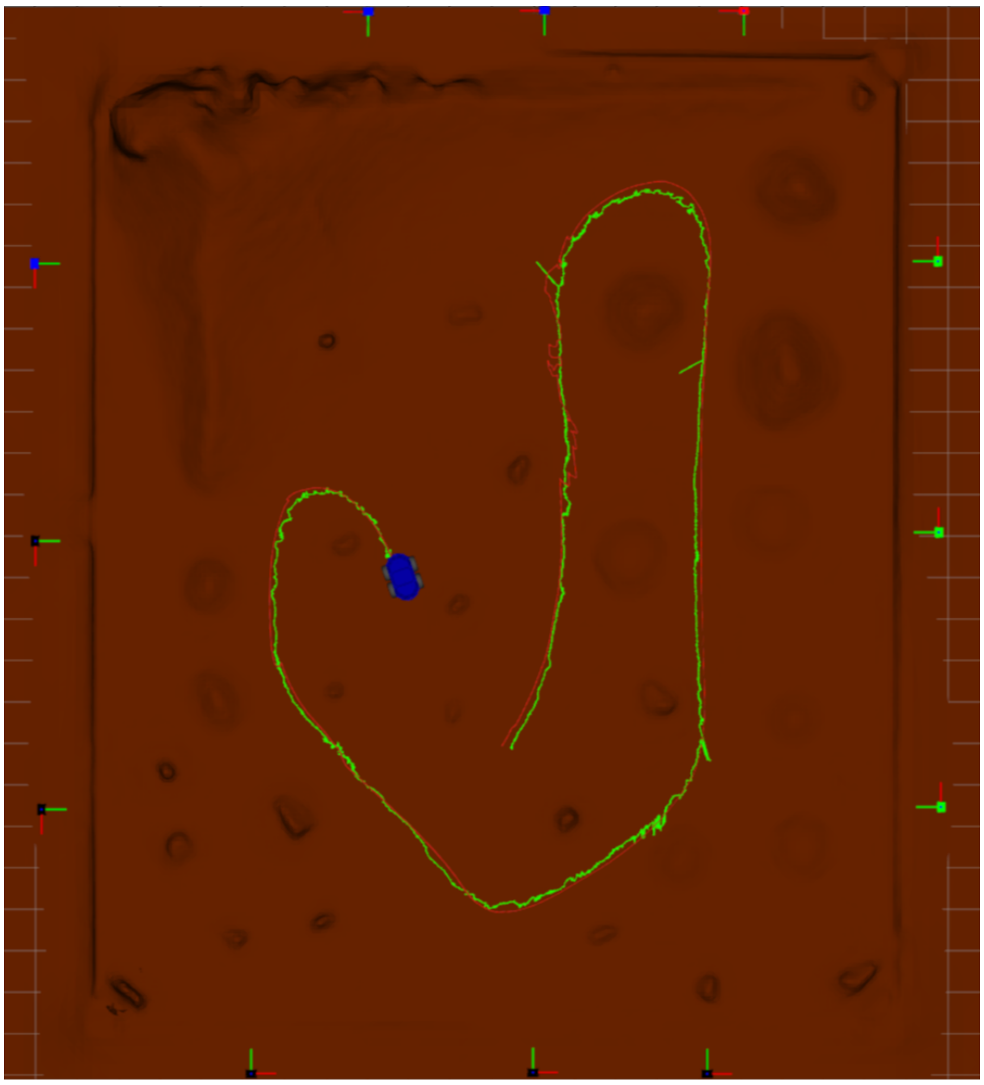


Figure 5.6.9: EKF vs Reference - Third path, top view

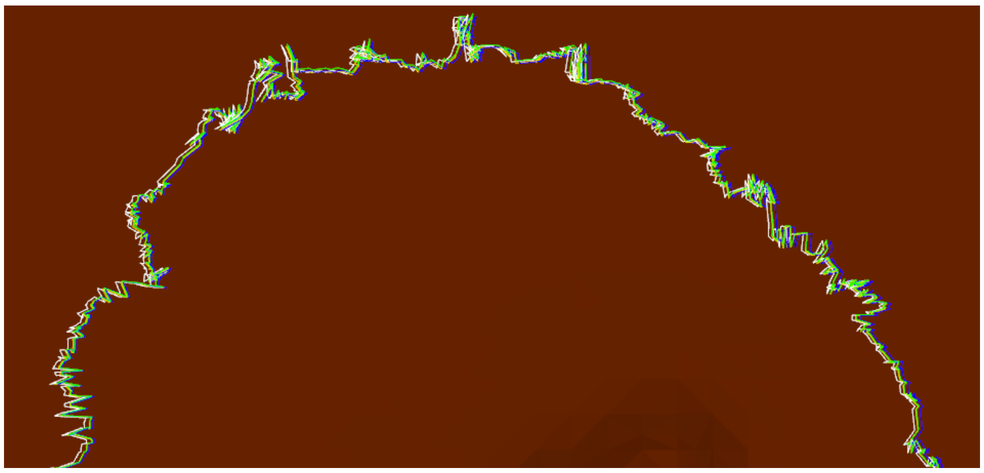


Figure 5.6.10: EKF vs Poly3 (*yellow*) vs Poly5 (*blue*) vs Spline3 (*white*) - Third path, top view detail

As anticipated at the beginning of the section, some measurements have been acquired at individual points to numerically compare the EKF and UKF results, and to obtain information about the filters' accuracy and ability to provide stable estimations through time. The nine positions that were taken into account, almost all at the same height, are shown in Figure 5.6.11 and the data that follow only concern the two-dimensional position in the RoXY terrain, disregarding the height component that has already been observed not to be either accurate or reliable. For each position, 100 pose estimations (for a total of 5 seconds) have been retained and analyzed in terms of mean value, standard deviation, maximum error, and average error, which are all reported in Table 5.7 for the EKF and in Table 5.8 for the UKF.

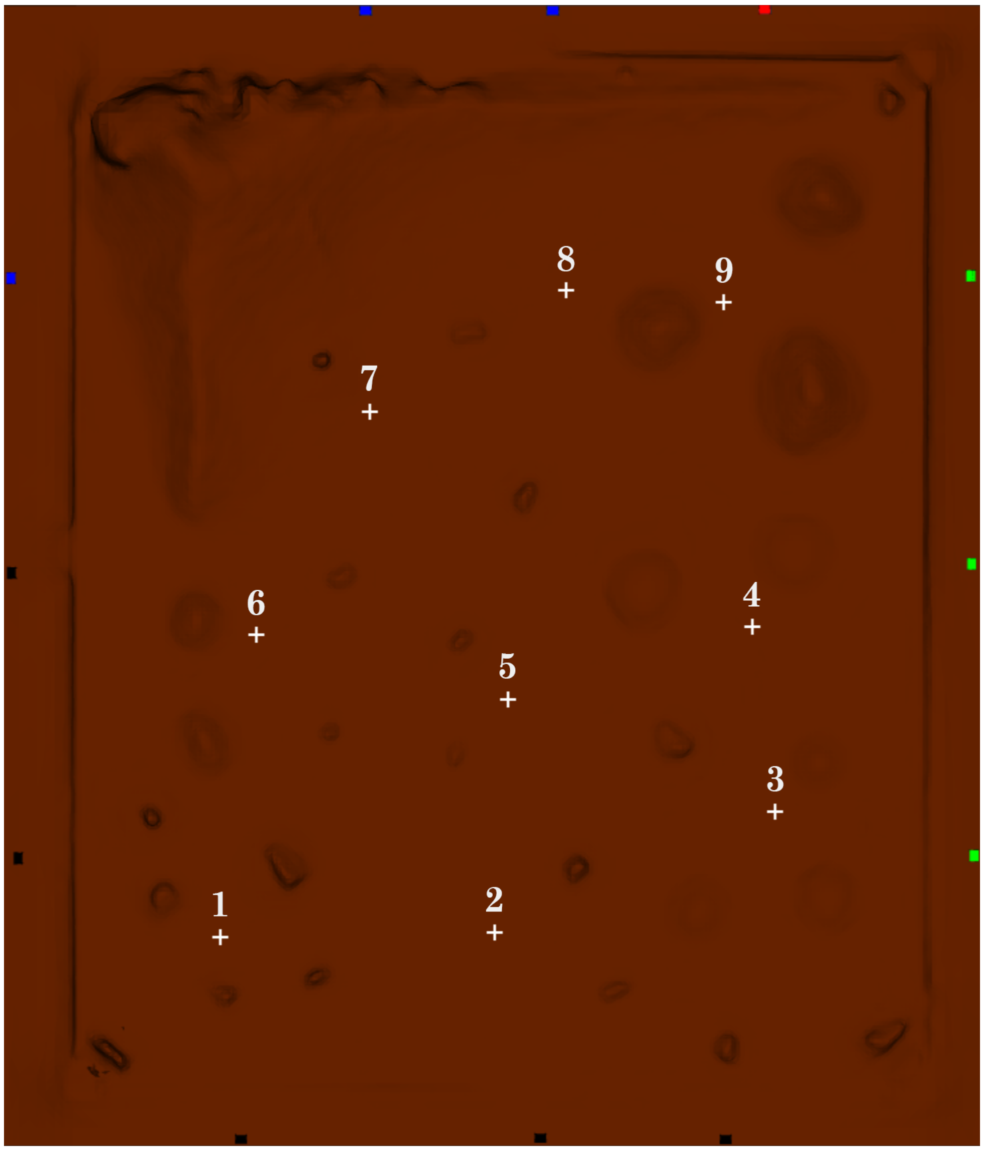


Figure 5.6.11: Positions considered for the single point measurements

EKF								
	Reference		**Mean Value**		**Standard Deviation**		**Distance Error of Mean**	**Max Distance Error**
	x [m]	**y [m]**	**x [m]**	**y [m]**	**x [m]**	**y [m]**	**[m]**	**[m]**
Position 1	4.941	5.079	4.875	5.013	0.007	0.007	0.093	0.109
Position 2	10.218	4.963	10.201	4.973	0.008	0.007	0.020	0.037
Position 3	16.919	7.187	16.934	7.081	0.010	0.012	0.107	0.133
Position 4	17.129	12.225	17.213	12.292	0.021	0.034	0.108	0.198
Position 5	10.933	10.030	10.919	10.074	0.008	0.008	0.046	0.071
Position 6	5.389	11.250	5.317	11.323	0.011	0.009	0.103	0.134
Position 7	7.414	16.196	7.309	16.313	0.012	0.030	0.157	0.245
Position 8	11.750	19.697	11.771	19.760	0.011	0.011	0.067	0.086
Position 9	15.638	20.598	15.750	20.705	0.045	0.011	0.155	0.241

Table 5.7: EKF metrics with single point measurements

UKF								
	Reference		**Mean Value**		**Standard Deviation**		**Distance Error of Mean**	**Max Distance Error**
	x [m]	**y [m]**	**x [m]**	**y [m]**	**x [m]**	**y [m]**	**[m]**	**[m]**
Position 1	4.941	5.079	4.850	4.999	0.006	0.005	0.121	0.134
Position 2	10.218	4.963	10.194	4.972	0.006	0.005	0.026	0.037
Position 3	16.919	7.187	16.938	7.085	0.009	0.013	0.104	0.124
Position 4	17.129	12.225	17.223	12.272	0.014	0.023	0.105	0.161
Position 5	10.933	10.030	10.892	10.068	0.006	0.006	0.056	0.075
Position 6	5.389	11.250	5.296	11.283	0.009	0.007	0.099	0.120
Position 7	7.414	16.196	7.315	16.296	0.010	0.022	0.141	0.200
Position 8	11.750	19.697	11.774	19.763	0.007	0.009	0.070	0.093
Position 9	15.638	20.598	15.757	20.697	0.036	0.007	0.155	0.232

Table 5.8: UKF metrics with single point measurements

Several conclusions can be drawn from these data: first, it can be seen from the standard deviations that the filters are both capable of providing continuous estimates of a fixed position that do not deviate much from the mean value, noting in fact that only in a few cases the value does exceed 1.5 cm for the EKF, while the UKF is even more accurate. In addition, the data on the error distances of both filters are in agreement with what was evident from the paths shown in the previous figures, noting in particular that the mean error of neither filter ever exceeds 16 cm, as well as the maximum error at each position is always below 25 cm.

Conclusions and Future Improvements

This thesis work presented the full-scale design of a UWB-based local positioning system aimed at assisting the research activities of the Robotics team at the TAS-I RoXY facility in Turin, by providing a reliable absolute position estimation free from the drifts commonly found in techniques such as odometry and SLAM. The project also had the higher goal of proposing a low-cost solution for local positioning in GNSS-denied zones, as the Moon and Mars are until extraterrestrial global positioning systems such as LCNS (Lunar Communications and Navigation System) will be finalized. The development started from the integration of the Qorvo UWB module within the ST microcontroller board and its programming, paying particular attention to the computational effort and time duration of the ranging algorithm, and performing a fine-tune of all the parameters involved through an extensive calibration activity.
Furthermore, in order to comply with the 24 V AC power supply present in the facility, a PCB power converter was designed to provide the 12 V DC required by the ST board. The bare boards were assembled, soldered, and tested in the facility workshop. In addition, given the necessity of a fixed setup for the system to permanently integrate it within the facility, a wall-mounted waterproof case for the UWB anchors was modeled with a 3D CAD and rapidly prototyped with the aid of Prusa 3D printers; the case was designed in order to be detachable from its wall support for reprogramming purposes and also to contain the power converter PCB, in order to exploit the pre-existent power supply rail of the facility.

The localization algorithm was implemented on the ROS2 framework by means of both an Extended Kalman Filter and an Unscented Kalman Filter, to highlight the different accuracies that they provide, and they were also enforced with further features of outlier rejection, smoothing, and ranging error correction. To be able to efficiently and repeatedly test their performances with and without the additional features and to tune the parameters, a simulator was built employing many ROS2 tools in order to be compatible with the filters algorithms and to reproduce the essential test characteristics, such as the ranging noise, the motion control of the rover through space, and a faithful replica of the RoXY terrain. The simulation test configuration was made as similar as possible to the one of real field tests, so a total of 12 anchors were spawned on the perimeter of the playground, in groups of 3 and at three different heights.

The results showed that the filters were able to successfully track the rover's movement in the three dimensions, achieving 2D RMSE errors of approximately 5 cm and 3 cm for the EKF and UKF, respectively, which increases to 7 cm and 14 cm if the third dimension is taken into account, meaning that the estimation of the latter is better carried out by the EKF than by the UKF. The poor performance in the vertical estimation is well-known in the literature of UWB positioning systems, and it is due to the scarce distribution of the anchors in the z-dimension with respect to the always wider span in the xy-plane. Selectively increasing the injected ranging noise for brief sections of the paths, the outlier rejection features were tested and they were observed to deliver more accurate position estimation, as also did the smoothing version of the basic filters.

The field tests were performed by placing the UWB anchors in the same positions that were simulated before, precisely measuring them with the Leica laser tracker, and by mounting the UWB tag on a dedicated 3D-printed support placed on top of the rover and retrieving the data directly from the rover OBC through the connection with the control room workstation. The reference path used to compare the filters' performance was obtained with a sensor-fusion algorithm developed by the TAS-I Robotics team, which involved data from a stereocamera, a Time-of-Flight camera, wheel odometry, and a Real-Time-Kinematics-aided GPS.
The results highlighted an effective tracking of the rover position by both the filters, with errors obviously higher than the simulation in both the two-dimensional and three-dimensional cases, the latter being too much unreliable. On the xy-plane, the errors of both filters were noticed to be within 10 cm and 15 cm, while the vertical errors were totally out of scale. This was hypothesized to occur because of the higher noise of the actual sensors with respect to the simulated ones, and all the real-life undesired effects that were not taken into account during the simulations, such as signal reflections, antenna orientation, and so on. The occurrence of real outliers in the estimation made the outlier rejection techniques much more effective than they were in the simulations, showing a successful detection and rejection of wrong estimations in all cases. The same was observed enabling the smoothing features, also because the outlier occurrence was rarely experienced, which furtherly validates the effectiveness of the UWB system.

Future improvements of this kind of system will include the employment of UAVs acting as movable UWB anchors aimed at increasing the span in the vertical direction, thus providing a better estimation of the height component of the position in three-dimensional space. Also, the filters could be provided with further techniques designed to increase the robustness of the algorithms to divergence, which occasionally occurred, mainly with the UKF. Furthermore, a more complex sensor-fusion system could exploit the UWB ranging ability to provide even more precise localization, combining relative visual information and odometry with an absolute system as the UWB is.

Bibliography

[1] A. Alvarez, L. Celis, E. Lopez, D. Rakic *et al.*, “Ultra-Wideband: Past, Present and Future,” Presented at EUWB consortium (Whitepaper) https://www.researchgate.net/publication/301626050_Ultra-Wideband_Past_Present_and_Future, 2011.

[2] T. W. Barrett, “History of Ultra Wideband Communications and Radar: Part I, UWB Communications,” *Microwave Journal*, vol. 44, no. 1, pp. 22–56, 2001.

[3] T. W. Barrett, “History of Ultra Wideband Communications and Radar: Part II, UWB Radars and Sensors,” *Microwave Journal*, vol. 44, no. 2, pp. 22–52, 2001.

[4] G. Breed, “A Summary of FCC Rules for Ultra Wideband Communications,” *High Frequency Electronics*, pp. 42–44, 2005.

[5] R. Brena, J. García-Vázquez, C. Galván Tejada, D. Munoz, C. Vargas-Rosales, J. Fangmeyer Jr, and A. Palma, “Evolution of Indoor Positioning Technologies: A Survey,” *Hindawi: Journal of Sensors*, vol. 2017, 2017.

[6] D. Burghal, A. T. Ravi, V. Rao, A. A. Alghafis, and A. F. Molisch, “A Comprehensive Survey of Machine Learning Based Localization with Wireless Signals,” arXiv, 2020.

[7] Cabinet Office, Government Of Japan, “Overview of the Quasi-Zenith Satellite System (QZSS),” http://qzss.go.jp/en.

[8] M. Cardullo, “Genesis of the Versatile RFID Tag,” RFID Journal, https://www.rfidjournal.com/genesis-of-the-versatile-rfid-tag.

[9] Chinese Government, “BeiDou Navigation Satellite System,” http://en.beidou.gov.cn/SYSTEMS/System.

[10] Combain Mobile AB, “Combain positioning solutions,” https://combain.com/.

[11] D. Dardari, N. Decarli, A. Guerra, M. Fantuzzi, D. Masotti, A. Costanzo, D. Fabbri, A. Romani, M. Drouguet, T. Feuillen, C. Raucy, L. Vandendorpe, and C. Craeye, “An Ultra-Low Power Ultra-Wide Bandwidth Positioning System,” *IEEE Journal of Radio Frequency Identification*, vol. 4, no. 4, pp. 353–364, 2020.

[12] A. Deshmukh and S. Bodhe, "Performance Evaluation of Constant Envelope OFDM working in 60 GHz Band," *International Journal of Engineering and Technology*, vol. 4, pp. 24–36, 2012.

[13] M.-G. Di Benedetto, T. Kaiser, A. F. Molish, I. Oppermann, C. Politano, and D. Porcino, *UWB Communication Systems: A Comprehensive Overview.* Hindawi Publishing Corporation, 2006.

[14] A. F. Echeverri, H. Medeiros, R. Walsh, Y. Reznichenko, and R. Povinelli, "Real-time Hierarchical Bayesian Data Fusion for Vision-based Target Tracking with Unmanned Aerial Platforms," in *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2018, pp. 1262–1270.

[15] S.-H. Fang, Y.-C. Cheng, and Y.-R. Chien, "Exploiting Sensed Radio Strength and Precipitation for Improved Distance Estimation," *IEEE Sensors Journal*, vol. 18, no. 16, pp. 6863–6873, 2018.

[16] S. Gezici, Z. Sahinoglu, H. Kobayashi, and H. V. Poor, *Ultra Wideband Geolocation,* in *Ultra Wideband Wireless Communication* (eds H. Arslan, Z.N. Chen and M.-G. Di Benedetto). John Wiley & Sons, Ltd, 2006, ch. 3, pp. 43–75.

[17] M. Hämäläinen, V. Hovinen, and L. Hentilä, *UWB Theory and Applications.* John Wiley & Sons, Ltd, 2004.

[18] C. Isaia and M. P. Michaelides, "A Review of Wireless Positioning Techniques and Technologies: From Smart Sensors to 6G," *Signals*, vol. 4, no. 1, pp. 90–136, 2023.

[19] Italian Space Agency (ASI), "Galileo," https://www.asi.it/tlc-e-navigazione/galileo.

[20] R. Khan, S. U. Khan, S. Khan, and M. U. A. Khan, "Localization Performance Evaluation of Extended Kalman Filter in Wireless Sensors Network," *Procedia Computer Science*, vol. 32, pp. 117–124, 2014.

[21] R. R. Labbe Jr., *Kalman and Bayesian Filters in Python.* Creative Commons Attribution 4.0 International License, https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python/tree/master/, 2020.

[22] X. Li, Y. Wang, and K. Khoshelham, "A Robust and Adaptive Complementary Kalman Filter Based on Mahalanobis Distance for Ultra Wideband/Inertial Measurement Unit Fusion Positioning," *Sensors (MDPI)*, vol. 18, no. 10, 2018.

[23] M. P. Michaelides and C. G. Panayiotou, "SNAP: Fault Tolerant Event Location Estimation in Sensor Networks Using Binary Data," *IEEE Transactions on Computers*, vol. 58, no. 9, pp. 1185 – 1197, 2009.

[24] National Coordination Office for Space-Based Positioning, Navigation, and Timing, "Official U.S. government information about the Global Positioning System (GPS) and related topics," https://www.gps.gov/systems/gps, 2021.

[25] D. Neirynck, E. Luk, and M. Mclaughlin, “An Alternative Double-Sided Two-Way Ranging Method,” in *13th Workshop on Positioning, Navigation and Communications (WPNC)*, 2016.

[26] J. Ni, D. Arndt, P. Ngo, C. Phan, and J. Gross, “Ultra-wideband two-cluster tracking system design with angle of arrival algorithm,” in *2006 IEEE Conference on Radar*, 2006, pp. 37–43.

[27] J. Ni, D. Arndt, P. Ngo, C. Phan, K. Dekome, and J. Dusl, “Ultra-wideband Time-Difference-Of-Arrival High Resolution 3D Proximity Tracking System,” in *IEEE/ION Position, Location and Navigation Symposium*, 2010, pp. 37–43.

[28] J. Ni and R. Barton, “Ultra-wideband Time-Difference-Of-Arrival High Resolution 3D Proximity Tracking System,” in *2005 IEEE/ACES International Conference on Wireless Communications and Applied Computational Electromagnetics*, 2005, pp. 31–34.

[29] O. A. Oumar, M. F. Siyau, and T. P. Sattar, “Comparison between MUSIC and ESPRIT direction of arrival estimation algorithms for wireless communication systems,” in *The First International Conference on Future Generation Communication Technologies.* IEEE, 2012, pp. 99–103.

[30] Y. Pei, S. Gao, G. Hu, Y. Zhao, and K. Jia, “Mahalanobis Distance based Adaptive Unscented Kalman Filter and Its Application in GPS/MEMS-IMU Integration,” in *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, 2019, pp. 2649–2655.

[31] Roscosmos State Corporation Applied Consumer Center, “About GLONASS,” https://glonass-iac.ru/en/about_glonass.

[32] SinoDefence.com, “Compass (BeiDou 2) Satellite Navigation System,” http://www.sinodefence.com/space/spacecraft/beidou2.

[33] J. Stefański, “Hyperbolic Position Location Estimation in the Multipath Propagation Environment,” in *Wireless and Mobile Networking.* Springer Berlin Heidelberg, 209, pp. 232–239.

[34] J.-A. Ting, E. Theodorou, and S. Schaal, “A Kalman Filter for Robust Outlier Detection,” in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems.* San Diego, CA: IEEE/RSJ International, 2007, pp. 1514–1519.

[35] U R Rao Satellite Centre, Department of Space, Government of India, “IRNSS - Indian Regional Navigation Satellite System,” https://www.ursc.gov.in/navigation/irnss.

[36] R. Van der Merwe, “Sigma Point Kalman Filters for Probabilistic Inference in Dynamic State-Space Models,” Ph.D. dissertation, OGI School of Science & Engineering, Oregon Health & Science University, USA, 2004.

[37] J. J. Wang, J. G. Hwang, and J. G. Park, "A novel indoor ranging algorithm based on received signal strength and channel state information," in *IPIN (Short Papers/Work-in-Progress Papers).* IEEE, 2019, pp. 32–39.

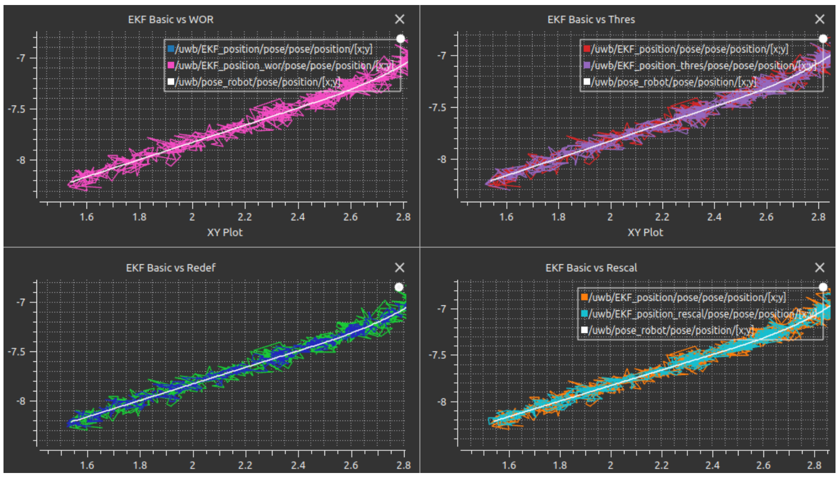
[38] Wikipedia, "Galileo positioning system," https://it.wikipedia.org/wiki/Sistema_di_posizionamento_Galileo, 2023.

[39] F. Zafari, A. Gkelias, and K. K. Leun, "A Survey of Indoor Localization Systems and Technologies," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2568 – 2599, 2019.

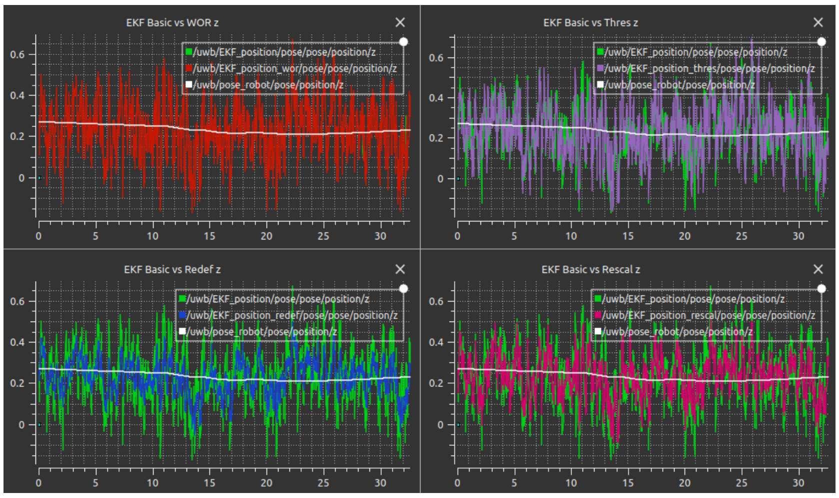
[40] R. Zekavat and R. M. Buehrer, *Handbook of Position Location: Theory, Practice, and Advances*, 1st ed. Wiley-IEEE Press, 2011.

Appendix A

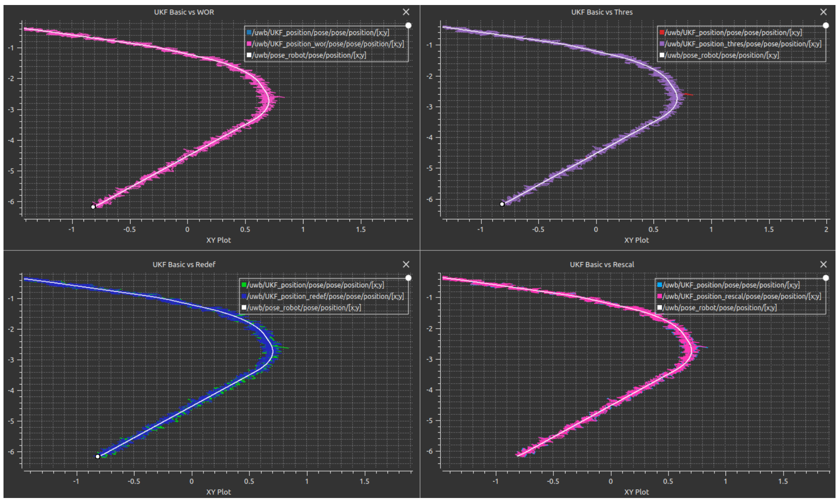
Simulation Data



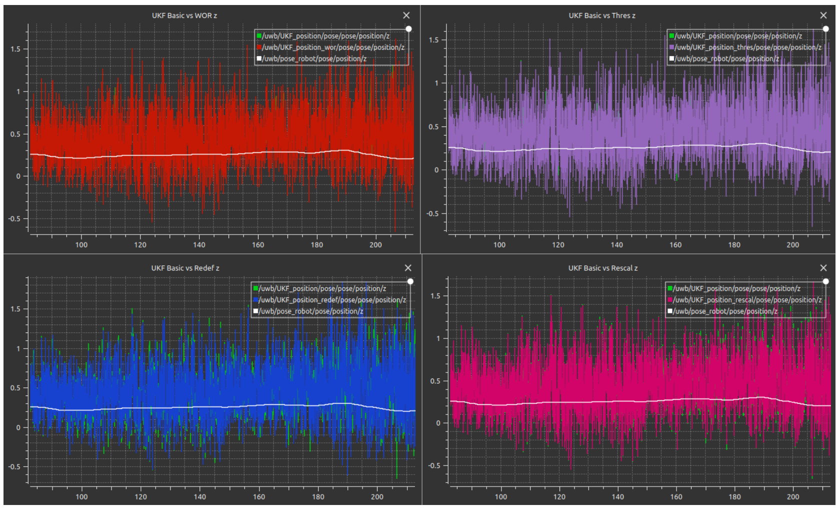
(a) EKF vs outlier rejections vs Reference (xy)



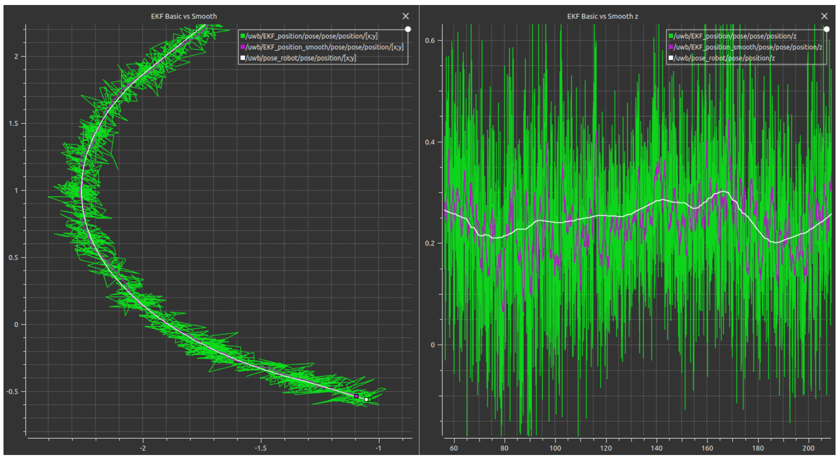
(b) UKF vs outlier rejections vs Reference (z)



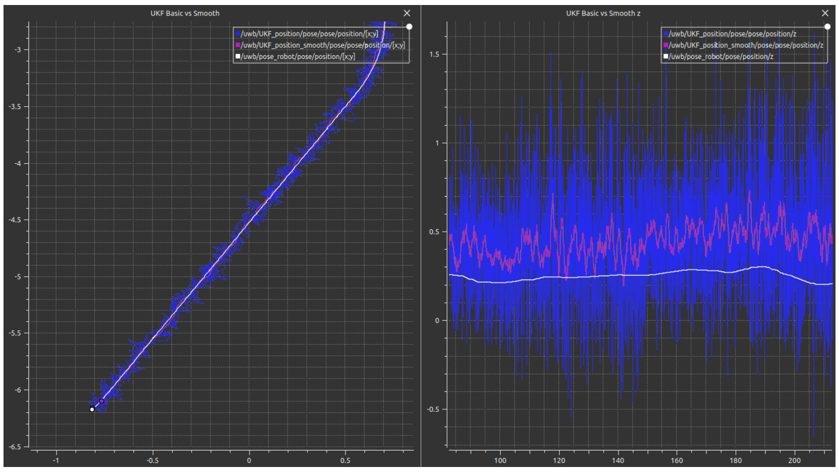
(a) UKF vs outlier rejections vs Reference (xy)



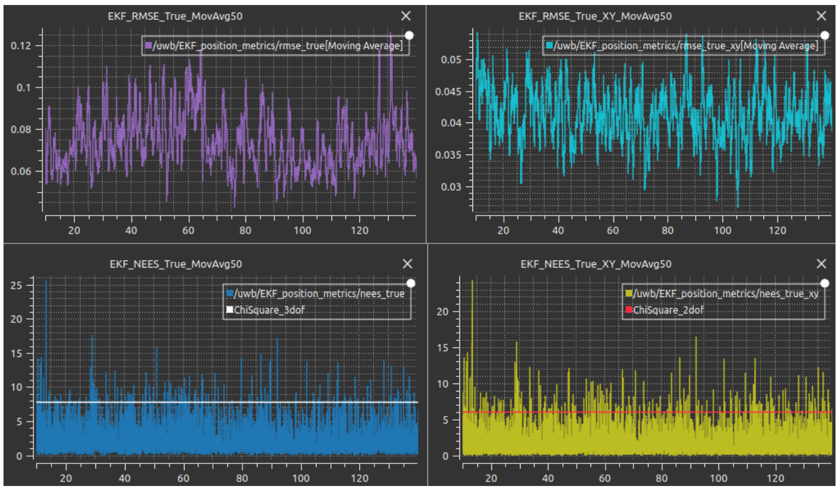
(b) UKF vs outlier rejections vs Reference (z)



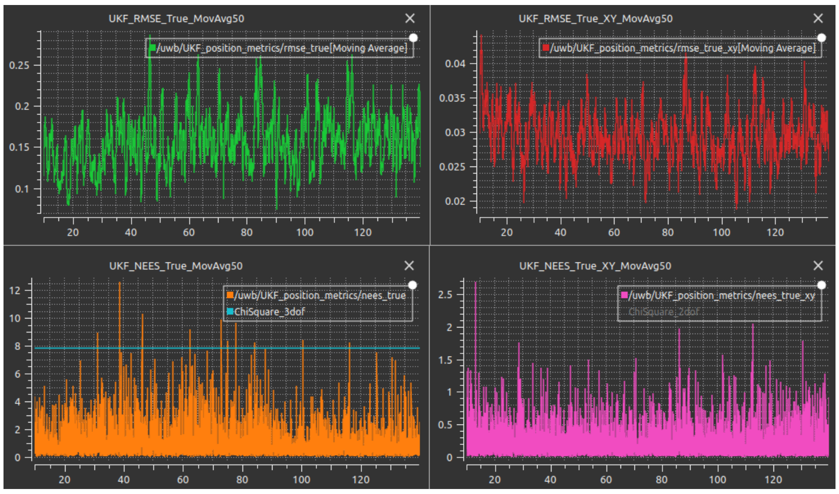
(d) EKF vs Smooth vs Reference



(e) UKF vs Smooth vs Reference



(a) EKF metrics referred to Reference



(b) UKF metrics referred to Reference

Appendix B

Error Map Data

Case	Error [mm]	Mean [mm]	Standard Deviation [mm]	Max Error [mm]	Average Error [mm]	Reference Distance [mm]	Overall Average Absolute Error [mm]
Best	147.10	3060.10	20.76	229	147.10		
Average	150.74	3063.74	21.24	217	150.74	2913	
Worst	154.77	3067.77	23.36	229	154.77		
Best	56.68	5148.68	20.39	123	56.74		
Average	59.37	5151.37	22.95	127	59.43	5092	
Worst	61.77	5153.77	25.28	123	61.81		
Best	25.37	6413.37	20.18	94	27.79		
Average	28.08	6416.08	21.95	94	30.37	6388	
Worst	30.20	6418.20	23.92	103	33.00		
Best	-22.91	9430.09	24.53	101	27.99		
Average	-25.38	9427.62	24.37	106	29.25	9453	
Worst	-28.08	9424.92	24.40	101	31.53		
Best	-11.26	13023.74	20.19	90	18.53		44.52
Average	-13.10	13021.90	21.57	85	20.29	13035	
Worst	-15.59	13019.41	22.60	90	22.21		
Best	10.03	16840.03	16.23	55	15.12		
Average	10.55	16840.55	16.11	59	15.41	16830	
Worst	11.55	16841.55	17.16	64	16.88		
Best	20.74	20715.74	17.89	73	23.32		
Average	21.49	20716.49	17.92	77	23.64	20695	
Worst	22.27	20717.27	17.66	73	23.97		
Best	45.71	24661.71	20.55	120	45.75		
Average	47.49	24663.49	19.15	112	47.56	24616	
Worst	48.89	24664.89	19.05	120	49.00		
Best	71.03	28706.03	22.72	135	71.04		
Average	77.62	28712.62	22.99	144	77.64	28635	
Worst	85.91	28720.91	23.93	145	85.94		

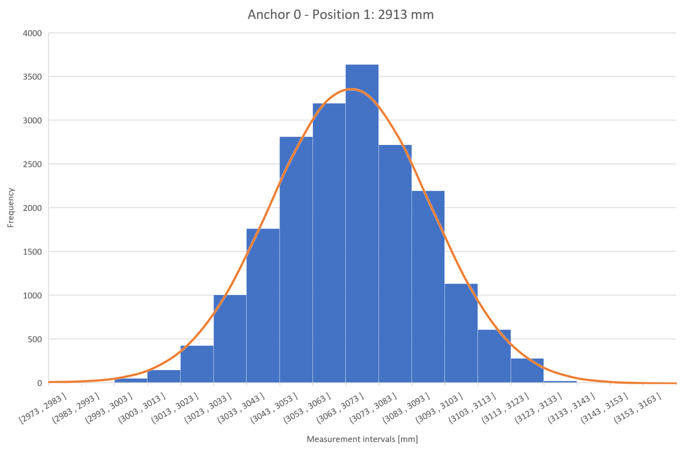
Table B.1: Anchor 0 error data

Case	Error [mm]	Mean [mm]	Standard Deviation [mm]	Max Error [mm]	Average Error [mm]	Reference Distance [mm]	Overall Average Absolute Error [mm]
Best	147.15	3060.15	22.62	210	147.15		
Average	150.64	3063.64	21.88	217	150.64	2913	
Worst	154.30	3067.30	21.09	220	154.30		
Best	70.48	5162.48	24.24	142	70.49		
Average	71.71	5163.71	23.44	143	71.72	5092	
Worst	73.04	5165.04	23.17	151	73.04		
Best	32.99	6420.99	23.06	94	34.47		
Average	36.60	6424.60	24.28	109	38.10	6388	
Worst	39.34	6427.34	24.04	112	40.43		
Best	-42.84	9410.16	25.38	138	43.59		
Average	-45.17	9407.83	26.69	142	45.70	9453	
Worst	-47.63	9405.37	27.00	129	47.95		
Best	-18.09	13016.91	15.31	81	19.95		54.08
Average	-20.33	13014.67	15.92	72	22.26	13035	
Worst	-22.08	13012.92	16.71	71	23.85		
Best	16.82	16846.82	15.85	64	19.30		
Average	17.96	16847.96	15.76	70	20.09	16830	
Worst	18.53	16848.53	16.11	64	20.85		
Best	36.71	20731.71	19.52	92	37.01		
Average	38.32	20733.32	19.81	94	38.61	20695	
Worst	39.68	20734.68	20.21	92	40.05		
Best	51.24	24667.24	18.32	102	51.29		
Average	51.88	24667.88	18.56	105	51.91	24616	
Worst	53.29	24669.29	18.77	102	53.31		
Best	81.67	28716.67	24.37	154	81.67		
Average	86.93	28721.93	25.67	154	86.93	28635	
Worst	90.91	28725.91	25.40	163	90.91		

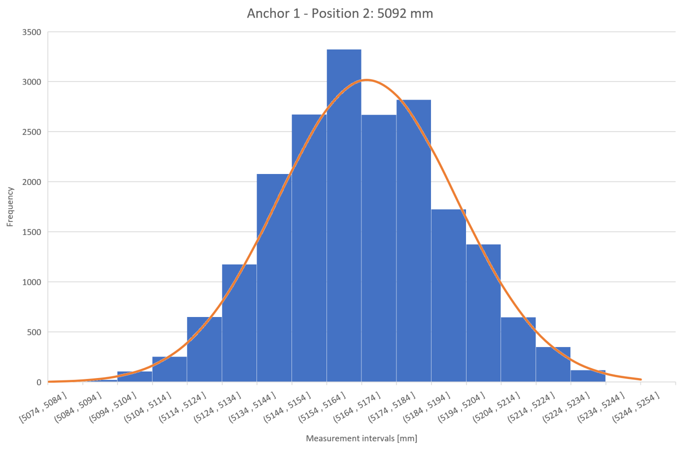
Table B.2: Anchor 1 error data

Case	Error [mm]	Mean [mm]	Standard Deviation [mm]	Max Error [mm]	Average Error [mm]	Reference Distance [mm]	Overall Average Absolute Error [mm]
Best	146.80	3059.80	22.60	220	146.80		
Average	150.70	3063.70	22.48	221	150.70	2913	
Worst	153.30	3066.30	22.77	229	153.30		
Best	66.60	5158.60	21.87	142	66.65		
Average	70.39	5162.39	22.96	143	70.41	5092	
Worst	73.47	5165.47	25.09	133	73.48		
Best	24.73	6412.73	21.95	103	27.94		
Average	26.68	6414.68	20.21	85	28.90	6388	
Worst	28.96	6416.96	19.82	94	30.65		
Best	-28.59	9424.41	24.20	120	31.62		
Average	-31.10	9421.90	24.84	115	33.79	9453	
Worst	-32.94	9420.06	24.48	129	34.69		
Best	-21.25	13013.75	20.98	71	25.46		45.24
Average	-24.46	13010.54	21.38	84	27.72	13035	
Worst	-27.28	13007.72	21.88	81	30.51		
Best	0.00	16830.00	16.98	58	13.22		
Average	-0.68	16829.32	17.21	58	13.64	16830	
Worst	-1.93	16828.07	16.94	58	13.45		
Best	12.91	20707.91	16.20	73	16.93		
Average	20.39	20715.39	16.95	72	22.43	20695	
Worst	24.10	20719.10	17.90	83	25.83		
Best	34.89	24650.89	19.46	92	35.60		
Average	37.52	24653.52	20.82	112	37.95	24616	
Worst	40.42	24656.42	22.08	120	40.75		
Best	55.84	28690.84	24.20	135	56.00		
Average	63.02	28698.02	23.43	138	63.05	28635	
Worst	69.76	28704.76	25.29	154	69.76		

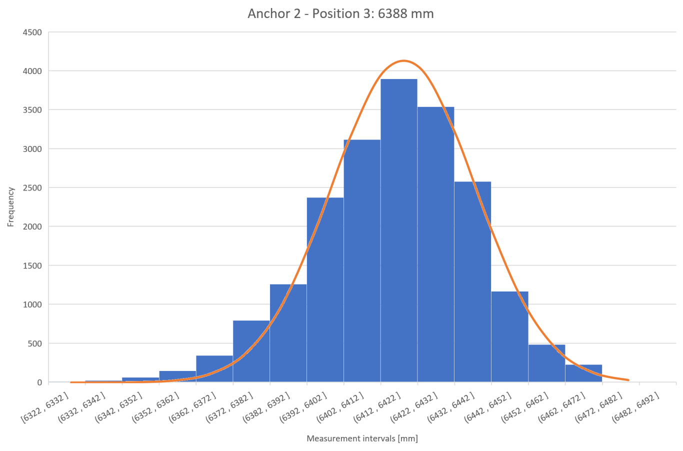
Table B.3: Anchor 2 error data



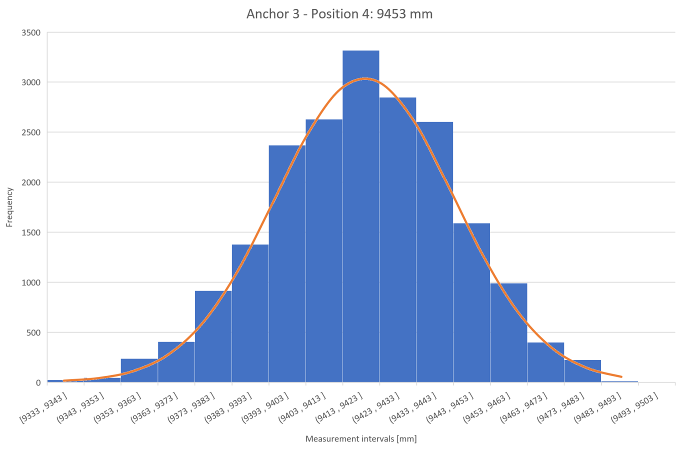
(a)



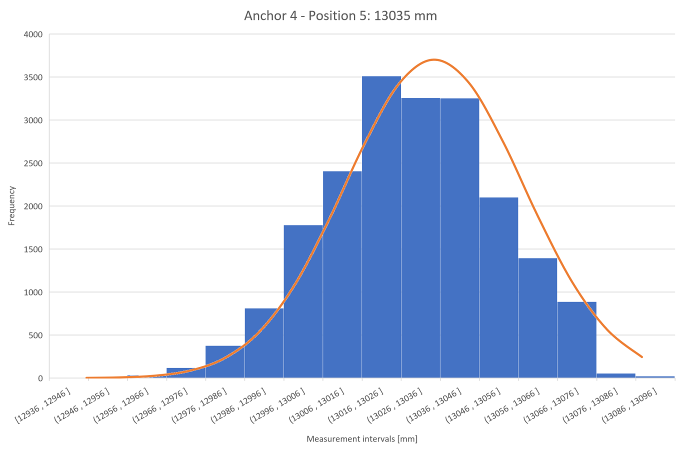
(b)



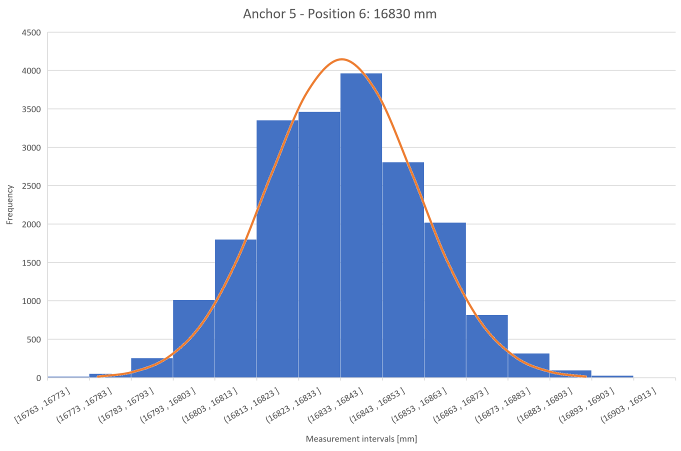
(c)



(d)

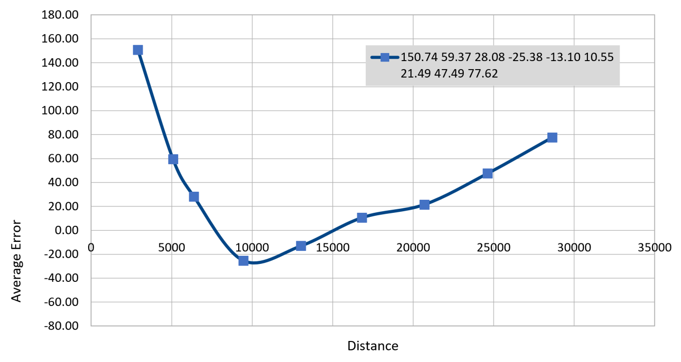


(e)

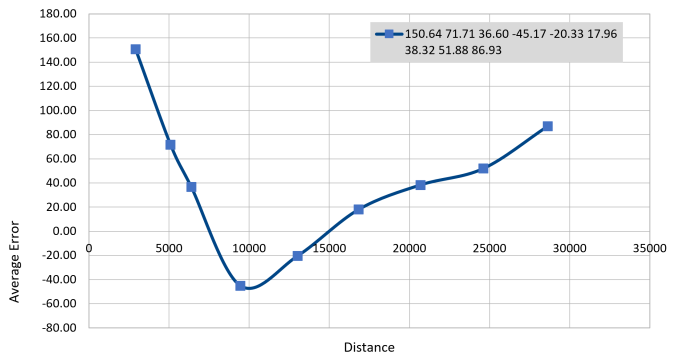


(f)

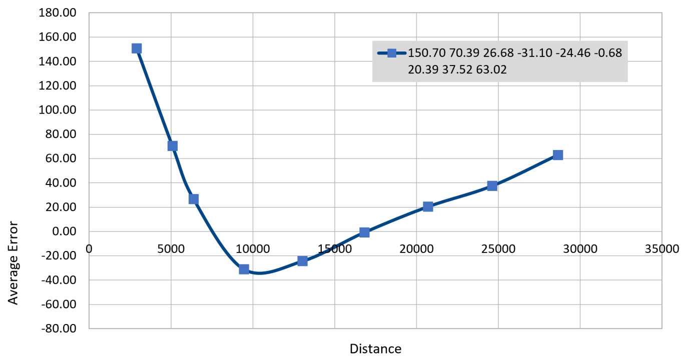
Figure B.1: Anchors 0 - 5 ranges distribution at the first 6 positions



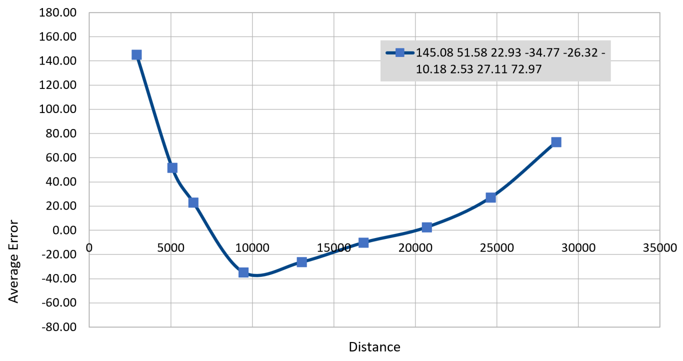
(a) Anchor 0



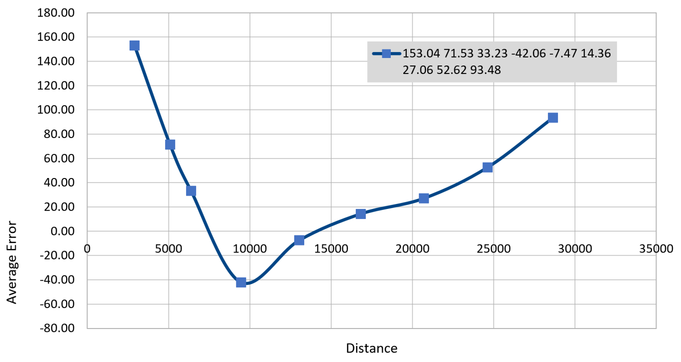
(b) Anchor 1



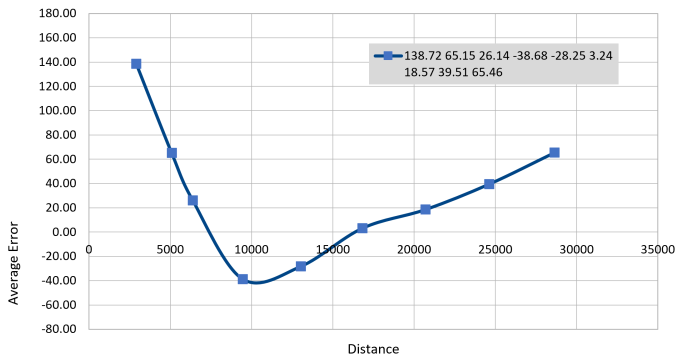
(c) Anchor 2



(d) Anchor 3



(e) Anchor 4



(f) Anchor 5

Figure B.2: Anchors 0 - 5 spline error functions, values in mm