POLITECNICO DI TORINO

Master of Science in Mechatronic Engineering

Master's Degree Thesis

Localization handover (indoor-outdoor) for autonomous mobile robots

Supervisors	Candidate
Prof. Alessandro RIZZO	Maria Assunta MAGGIO

July 2022

Index

List of Figures

List of Tables

Chapter 1

Introduction

Autonomous Mobile Robots (AMRs) play an increasingly fundamental role in both industrial and household fields with an expanding range of applications. In particular, the reference framework of this work refers to operations such as surveillance, transport, inspection, patrol applications, or guidance that take place generally in an heterogeneous environment where both indoor and outdoor areas are present.

Robot localization is a valuable aspect to accomplish any of the tasks recently mentioned. First and foremost, the robot must be aware of its own pose in the space in order to perform any navigation manoeuvres. The thesis addresses the problem of transitioning between two different areas, e.g. indoor and outdoor, which require different localization methodologies.

In the indoor case, the area can be considered limited, hence the position of the robot within this space can be tracked by means of various technologies (e.g, UWB sensors, WLAN, Bluetooth) that cover the overall region. This requires setting up the suitable hardware components that could be very costly depending on the application scenario.

On the other hand, satellite-based navigation systems (e.g, GLONASS, Galileo, GPS) have been widely used in outdoor localization, since the information about their position is accessible in any vast region without the need of a map or a potentially expensive infrastructure.

The transition between the two chosen approaches takes place in a sort of grey area where the signals derived from both technologies can coexist, even though their quality may be not good enough to evaluate the pose individually.

1.1 Contents organization

The thesis deals with the problem of localization handover affecting AMRs when the application scenario includes both indoor and outdoor areas. In particular, it focuses on the sensor fusion performed in the latter one and in the grey area mentioned before. The work is divided into chapters as exposed in the following.

In *Chapter 2*, a description of the issues related to the localization handover is provided. Different methodologies for indoor localization are outlined together with a brief explanation on how GPS works. Moreover, this chapter contains the state of the art for sensor fusion and the derived solution.

Chapter 3 illustrates a possible simple application scenario for a service robot.

Chapter 4 provides a detailed research on the market-available robots and their sensing equipment, that could serve to the purpose.

In *Chapter 5*, the GPS degradation in dense urban environments (e.g., urban canyons) and inside buildings is tackled.

Chapter 6 describes how the simulations on ROS 2 are set up, with particular emphasis on the **robot_localization** package used to accomplish sensor fusion.

Finally, *Chapter 7* provides a description of the designed nodes and topics to perform simulations in the created environment.

Chapter 2

Problem Description

Within the past decades, robot localization has become a crucial point for carrying out autonomous driving. In the most general case, AMRs have to navigate in an heterogeneous environment where both indoor and outdoor regions are present. The problem of localization in such areas can be tackled differently according to their own intrinsic features.

In the case of an indoor scenario, the environment is limited and can be potentially mapped before performing navigation. Moreover, an highly accurate estimation of the robot's pose is demanded since the space is more densely populated by obstacles to be avoided.

In contrast, outdoor areas can be considered unlimited and changeable, thus impossible to be previously mapped. In this case, less precision can be accepted due to the vastness of the area.

2.1 Localization technologies

Different approaches are chosen taking into account several factors, among which the application scenario (e.g, indoor/outdoor) is the most influent. In the following sections, the investigated localization technologies for indoor/outdoor environments are discussed.

2.1.1 Indoor

As indoor localization methodologies, three different approaches will be presented: *Bluetooth* network, *WLAN*, and *UWB* anchors.

Bluetooth

Bluetooth is a wireless technology standard used for exchanging data in a certain distance. According to [1], the data can be swapped in a range up to about one kilometer, though the communication between two Bluetooth devices can drop down to less than a meter due to several factors such as *radio spectrum*, *physical layer*, and *receiver sensitivity*. In the majority of applications, the range for commonly used devices is in the tens of meters.

A Bluetooth-based positioning system is a system requiring at least three receiver stations and a mobile device working also as a transmitter. The localization is performed by *trilaterion* [2], where the unknown position of the mobile device is computed measuring the distances between the known-positioned stations and the moving device. Such distances are calculated by multiplying the transmitter-to-receiver propagation time of the electromagnetic wave with the velocity of light, which requires the knowledge of the starting time at the transmitter and the time of arrival (TOA) at the receiver and, furthermore, the synchronization of their clocks. All stations are connected to a host PC which collects the measurements and computes the position of the mobile device.

The problem of synchronizing the receivers' clocks can be overcome by introducing a fourth receiver at a known position that is off the plane of the others and functions also as a transmitter. In this case, the localization is started by the additional station, called as *master station*, sending a signal to the mobile and the other receiver stations. Then, the latter ones start counting on their clocks the time passing from receiving such signal to receiving the echo response coming from the mobile. By doing so, the synchronization of the clocks in the receivers is unnecessary and it is the only remaining source of error. The working principle scheme for a differential time difference of arrival (DTDOA) system is pictured in Fig. 2.2.

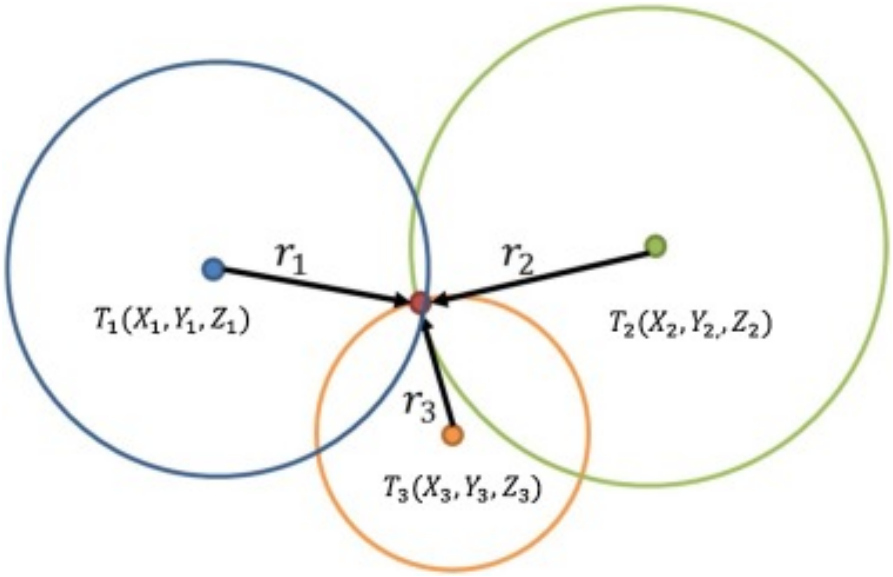

Figure 2.1: Trilateration working principle

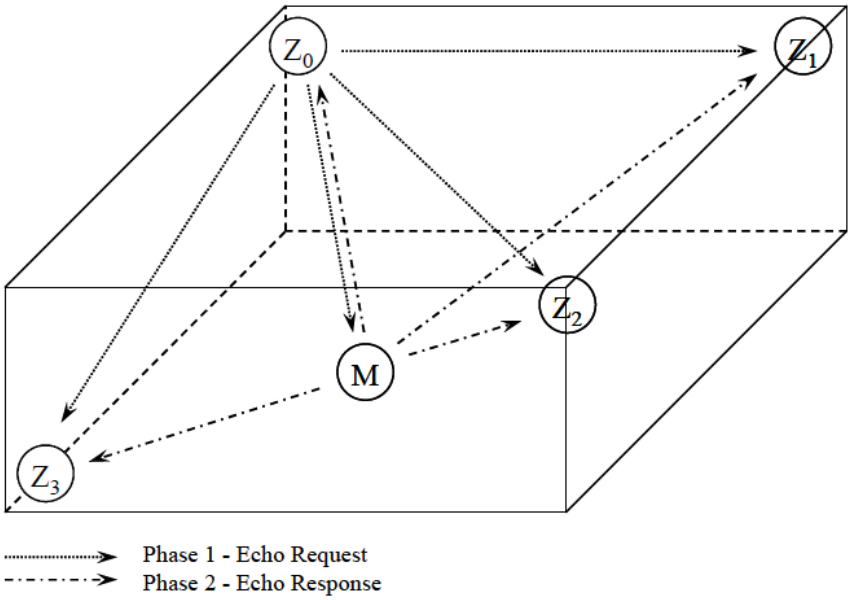

Figure 2.2: Principle of the DTDOA approach. Source [2]

Although BLE localization approach has the advantage to provide a low-cost and low-energy consumption solution, an accurate pose estimation is still challenging.

As demonstrated by [3], the achieved accuracy is between 0.07 meters and 7.81 meters, that is quite unsatisfactory for performing indoor positioning. Moreover, in the case of non-line-of-sight (NLOS) paths, the TOA measurements is less reliable due to the multipath propagation of radio signals [2].

Wireless LAN

A *WLAN* is a wireless computer network that employs high-frequency radio waves to permit communication between connected nodes. The Wireless LANs derived from the standard IEEE 802.11 are the most prevalent wireless computer networks around the world and they are known under the name of *Wi-Fi* [4].

A WLAN-based positioning system is composed by wireless access points used to track the location of the moving device. The localization can be accomplished by means of different approaches, between which the most frequent are the one based on the received signal strength indication (RSSI) and the one based on the so-called *fingerprints* [5].

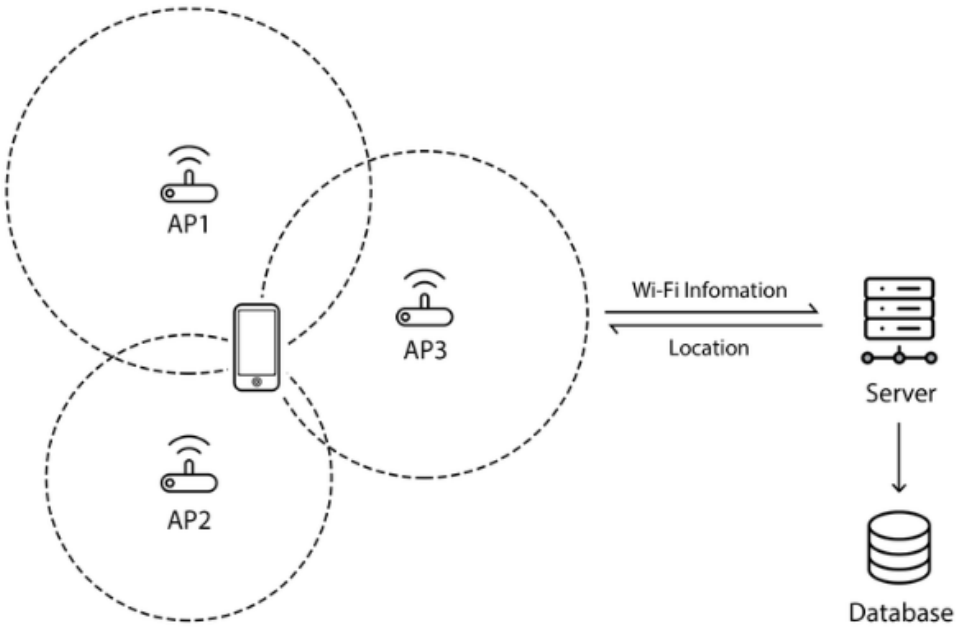

Figure 2.3: Wi-Fi positioning system. Source [6]

The RSSI localization technique is performed by exploiting the intrinsic information about the distance from the signal strength by means of a propagation model. The trilateration method is used to find the unknown position of the robot by intersecting the spheres with radius equal to the measured distances relative to the known position of the access points. However, this method cannot be considered reliable enough for our applications since the achieved accuracy is in few meters. Such drawback derives from the fluctuation of the RSSI measurements which are affected by multi-path fading and changes in the environment [5].

The fingerprinting localization method is accomplished in two phases: an offline phase, during which the signal strength and the known coordinates of the robot are recorded, and an online tracking phase, where the RSSI of the unknown-positioned robot is compared to the information stored in the fingerprint during the previous phase. The closest match is assumed as estimation of the current robot's pose. This technique provides a better estimation than the one mentioned above since the average accuracy is about half a meter [5].

Nevertheless, as stated in [7], the Wi-Fi positioning system is a more inaccurate localization technology than the BLE one. The estimate precision drops down of about 27 percent compared to the Bluetooth Low Energy solution in equivalent conditions. Furthermore, the other benefits of the BLE with respect to Wi-Fi are lower power consumption, higher scan rates, and unobtrusive less expensive transceivers.

UWB

UWB is a short-range, high-speed radio technology for wireless communication that has proved to be robust to NLOS and multipath effects. A UWB signal has either an absolute bandwidth of at least 500 MHz or a fractional (relative) bandwidth of larger than 20% [8]. Narrow pulses in the order of nanoseconds are transmitted, resulting in a very low power spectral density since the band is broad-ranging [9].

Similarly to other existing wireless localization technologies, UWB positioning is carried out by measuring either the directions or the distances. In the former case, the angle of arrival (AOA) between two nodes is computed, while, in the latter one, the robot's position is derived from the received signal strength (RSS) or the time of arrival/time difference of arrival (TOA/TDOA) [10]. Figure 2.4 provides a

visual understanding of the possible different approaches.

The intrinsic features of UWB-based localization systems lead to an anti-multipath solution that can achieve an accuracy of about few centimeters [11]. An additional benefit of such systems is that recent progress in commercial modules has improved the working range up to 60 meters [12] in line-of-sight (LOS), resulting in a wider coverage area.

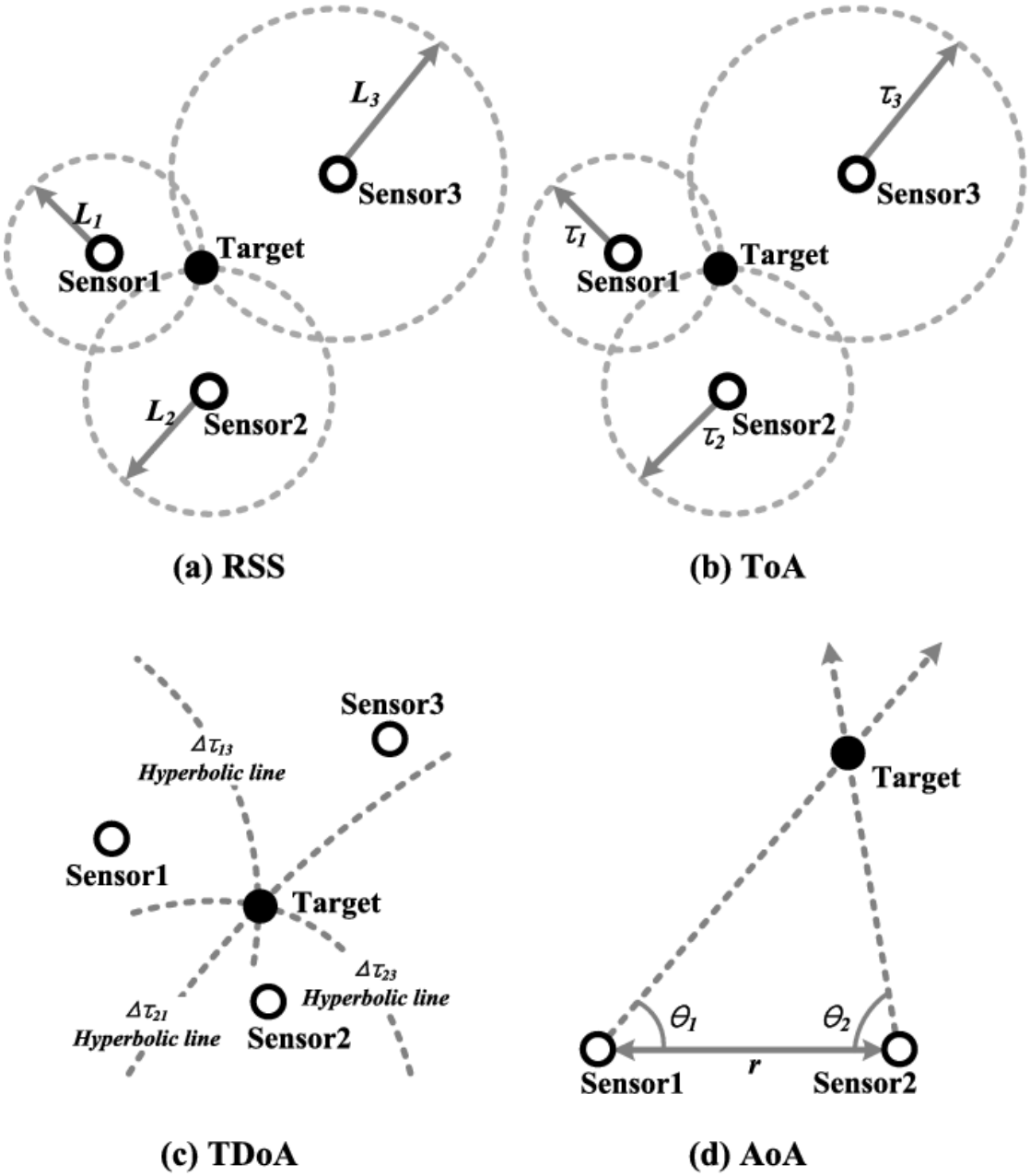

Figure 2.4: Different approaches for performing UWB positioning. Source [13]

2.1.2 Outdoor

The widely used localization technology for the outdoor is the Global Positioning System (GPS), the functioning of which is depicted in the following.

Global Positioning System

GPS is one of the global navigation satellite systems (GNSS). It is "owned by the United States government and operated by the United States Space Force", as mentioned in [14]. It is a radio-navigation system capable of providing position and time information to any GPS receiver on Earth in the line-of-sight of at least four GPS satellites. The required minimum number of satellites results from the four unknown quantities in the mathematical formulation: three position coordinates and the receiver clock error [14].

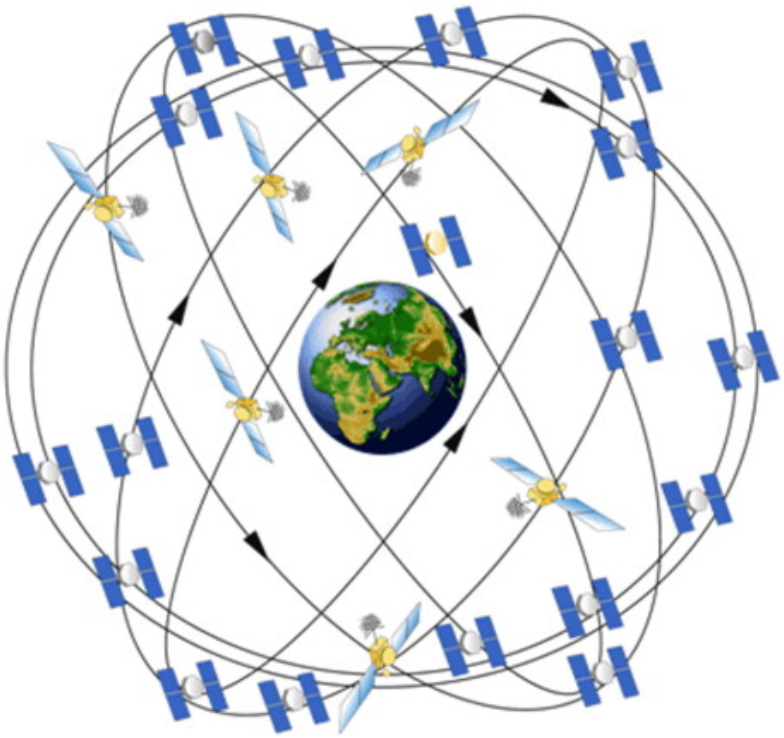

Figure 2.5: GPS constellation of 24 satellites

As outlined in [15], a signal containing the time of transmission is broadcasted repeatedly by each GPS satellite. The time of signal reception is stored by the GPS receiver and, then, used for computing the time passed from the signal transmission. Such measurement of time is converted in a so-called *pseudorange* by multiplication with the speed of light.

Once the pseudoranges are computed, the position of the GPS receiver can be

easily obtained through trilateration since the satellites' coordinates are known. Indeed, the signal broadcasted is encoded together with the so-called *Navigation Message*, that contains the *broadcast ephemeris* from which satellites' coordinates can be derived. The *Ephemeris Algorithm* is the algorithm used for transforming the orbit parameters into coordinates in a geocentric system, known as WGS-84, that is composed as follows (see Fig. 2.6):

- the origin is at the Earth centre of mass;
- Z axis points towards the North Pole;
- X axis is such that the Prime Meridian lays on the XZ plane;
- Y axis is chosen to form a right-handed orthogonal coordinate system together with the X and Z axes.

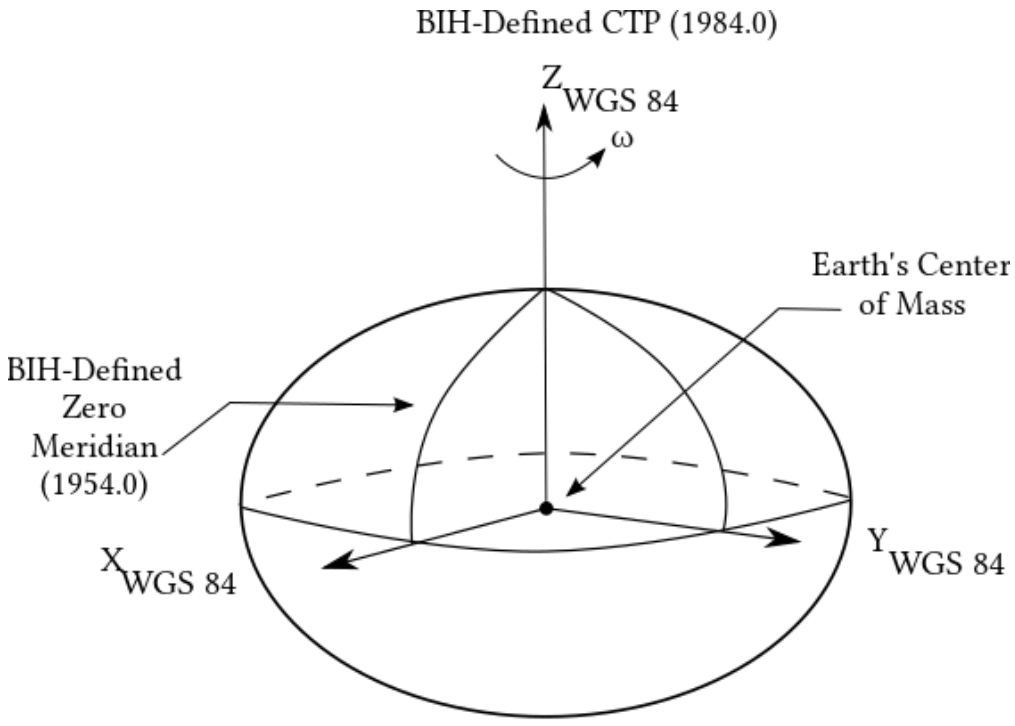

Figure 2.6: WGS-84 Reference Frame. Source [16]

2.2 State of the art

Nowadays, localization can be accomplished by fusing the data coming from different sensors, exploiting the properties of each sensor since they all have drawbacks. For instance, wheel encoders suffers from the so-called *wheel slippage*, leading to an error that increases over time. On the other hand, GPS provides a slower data-rate compared to dead-reckoning technologies and it is prone to degradation due to urban canyons or blocking satellites' signals. Nonetheless, an accurate pose estimation of the robot can be provided by performing sensor fusion that compensates for the errors coming from different sensors.

The problem of localization in mixed indoor-outdoor scenarios has been addressed by several works. [17] combines together WPAN-based, WLAN-based, and GNSS-based technologies according to the area where the robot is. However, such solution does not provide the needed accuracy and it also requires an infrastructure not always present.

The work of Agrawal and Konolige [18] analyses the results of a fusion between visual odometry and GPS by means of an Extended Kalman Filter (EKF). Whenever visual odometry fails, the EKF is provided with inertial measurements. Unfortunately, the EKF is not optimal in the case of nonlinear transformations.

Moreover, EKF makes independence assumptions that can lead to erroneous pose estimations since measurements referring to the same variables of the robot state are strongly correlated. This is the case of [19], where the authors merge a laser-based SLAM method with GPS, odometry, and IMU measurements by means of a Kalman filter.

In [9], an accurate positioning is performed combining UWB and GPS measurements by means of a probabilistic method called Monte Carlo Localization algorithm. Also [20] proposes a MCL approach in addition to the widely used Kalman filter. However, it also assumes that the robot is moving straightforward between two consecutive GPS measurements in order to provide the orientation without the use of a compass (e.g, in environments with variable magnetic fields). Such strong assumption can be discarded as Urcola et al. described in [21], where the orientation is computed only when several GPS measurements prove that the robot is moving

in a straight line.

Finally, a solution for overcoming the issue of GPS outages is outlined in [22]. This work employs a multi-layer perceptron neural network (MLP-NN) that provides predictions of the robot pose whenever GPS becomes unavailable. The neural network is trained when GPS signals are provided.

As the most common approaches to sensor fusion, the following sections illustrates the working principle of the Monte Carlo Localization algorithm and the Extended Kalman Filter.

2.2.1 Monte Carlo Localization algorithm

MCL algorithm offers a probabilistic description of the estimation of the robot pose by using particles to represent the possible states. It is based on a recursive Bayesian filter which working principle is to evaluate a probabilistic density function (pdf) recursively over time taking into account the incoming measurements from the sensors.

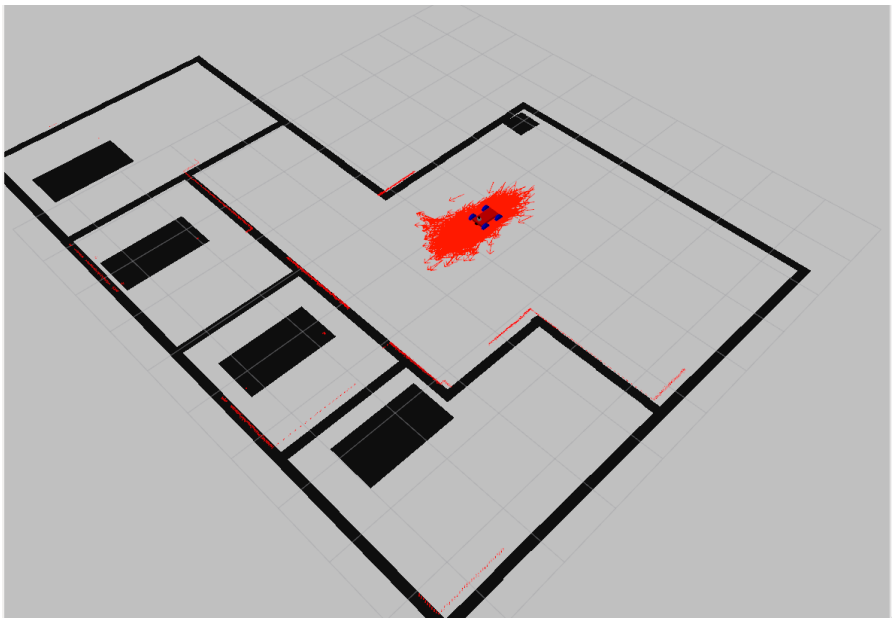

Figure 2.7: Monte Carlo Localization for mobile robots in ROS. Source [23]

Typically, there are no information about the initial robot state, thus a uniform

distribution over the state space is considered. Then, the posterior distribution is derived from the sensors' measurements and a weight coming from the observation likelihood function is assigned to each particle. Whenever a new measurement occurs, the weights are updated giving higher weights to the particles that better match the sensor readings [24].

2.2.2 Extended Kalman Filter

A discrete-time, nonlinear, time-variant dynamic system is described by the set of equations below

$$x(k+1) = f(k, x(k), u(k)) + v1(k) \quad (2.1)$$
$$z(k) = h(k, x(k)) + v2(k) \quad (2.2)$$

where $v1(k)$ and $v2(k)$ represent the process and observation noises respectively. These noises are assumed to be white noises with zero mean value, variance $V1(k)$ and $V2(k)$ respectively and covariance $V12(k)$.

The EKF is used to estimate the state of a nonlinear system. Such an estimate is performed in two steps:

1. the *prediction* step, during which the predicted state estimate $\hat{x}(k+1|k)$ is computed;
2. the *correction* step, during which the state estimate is updated by exploiting the information coming from the last measurement $z(k)$.

In the following, the equations of an EKF are provided:

$$\hat{x}(k+1|k) = f(k, \hat{x}(k|k-1), u(k)) + \hat{K}(k)e(k) \quad (2.3)$$
$$\hat{z}(k|k-1) = h(k, \hat{x}(k|k-1)) \quad (2.4)$$
$$\hat{x}(k|k) = \hat{x}(k|k-1) + \hat{K}_0(k)e(k) \quad (2.5)$$
$$e(k) = z(k) - z(k|k-1) \quad (2.6)$$

$$\begin{aligned}
\hat{K}(k) &= (\hat{A}(k|k-1)P(k)\hat{C}(k|k-1)^T \\
&\quad +V12(k))(\hat{C}(k|k-1)P(k)\hat{C}(k|k-1)^T + V2(k))^{-1} \qquad (2.7)\\
P(k+1) &= \hat{A}(k|k-1)P(k)\hat{A}(k|k-1)^T + V1(k) \\
&\quad -\hat{K}(k)(\hat{C}(k|k-1)P(k)\hat{C}(k|k-1)^T + V2(k))\hat{K}(k)^T \qquad (2.8)\\
\hat{K}_0(k) &= P(k)\hat{C}(k|k-1)^T(\hat{C}(k|k-1)P(k)\hat{C}(k|k-1)^T \\
&\quad +V2(k))^{-1} \qquad (2.9)
\end{aligned}$$

where $P(k)$ is the prediction error variance matrix computed by means of the Difference Riccati Equation (DRE), while $\hat{K}(k)$ and $\hat{K}_0(k)$ are the predictor gain matrix and the filter gain matrix respectively. The matrices $\hat{A}(k|k-1)$ and $\hat{C}(k|k-1)$ derive from

$$\hat{A}(k|k-1) = \left.\frac{\partial f(k,x,u)}{\partial x}\right|_{u=u(k),\, x=x(k|k-1)} \qquad (2.10)$$

$$\hat{C}(k|k-1) = \left.\frac{\partial h(k,x)}{\partial x}\right|_{x=x(k|k-1)} \qquad (2.11)$$

2.3 Proposed solution

Localization will be tackled by means of UWB anchors for indoor areas, while GPS will be used to track the robot outdoor. Additionally, information coming from the wheel encoders and Inertial Measurement Unit (IMU) are exploited in an EKF to enhance the pose estimation, especially in those regions where the robot is transitioning from indoor to outdoor and vice versa, or, even worse, when both UWB and GPS signals are not available.

The Extended Kalman Filter method is chosen in order to avoid additional complexity of the system. The accuracy of the solution will be evaluated in further works. If required, a different solution will be investigated.

This work focuses on the robot localization in outdoor and transitioning areas. Similarly to [25], the proposed solution is composed of two steps: first, a Kalman filter is used for merging the information derived from the dead-reckoning technologies

(i.e., wheel odometry and IMU); then, a weighting sum between the GPS data and the output of the Kalman filter is performed. The parameter that accounts for the reliability of the GPS signal can be defined as σ. The higher is the confidence of the GPS signal, the greater σ will be (anyway, $\sigma \leq 1$ always).

The GPS information can be considered more or less precise according to few parameters such as the attenuation of the signal or the variance of the latitude/longitude data. The varying parameter σ can be modeled as a continuous function of these characteristics of the received signal or, more easily, defining a lookup table where σ assumes defined values depending on the scenario the robot may be. As first approach, the varying parameter depends on the variance of GPS readings. When the robot is detected to be indoor, the GPS information will be considered no more reliable and, thus, discarded.

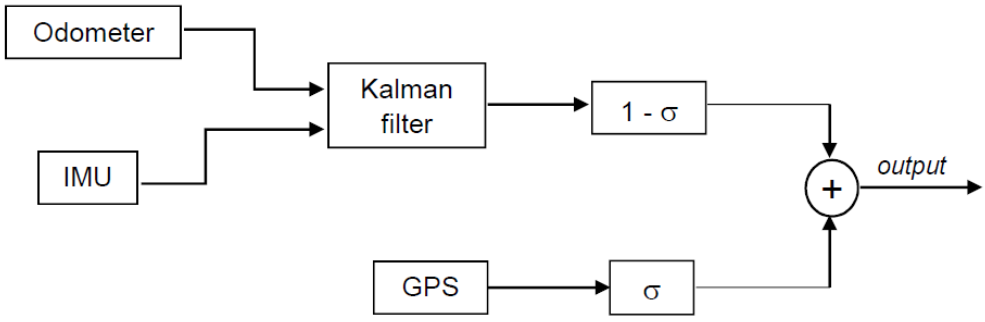

Figure 2.8: Sensor fusion scheme for outdoor and transitioning areas

Chapter 3

Prototype Scenario

A possible scenario is an application of service robotics in a warehouse, which includes both an indoor area (e.g., an industrial hall) and an outdoor area, such as a courtyard where the goods are stored temporarily before being delivered.

Figure 3.1 depicts an example of such scenario. Four different areas can be determined in the total environment:

- **A**, a part of the outdoor region where the GPS signal is assumed to have good quality and no data are sensed from the UWB tag on the robot;
- **B**, a part of the outdoor region where both the localization technologies (i.e., GPS and UWB) provide information on the robot's pose even though possibly degraded;
- **C**, a portion of the B area where the UWB signal is blocked due to the presence of metal in the wall;
- **D**, the indoor area where only the UWB anchors provide reliable information about robot's position.

The sensed data are merged with the information derived from the dead-reckoning localization technologies, such as the IMU and the wheel encoders.

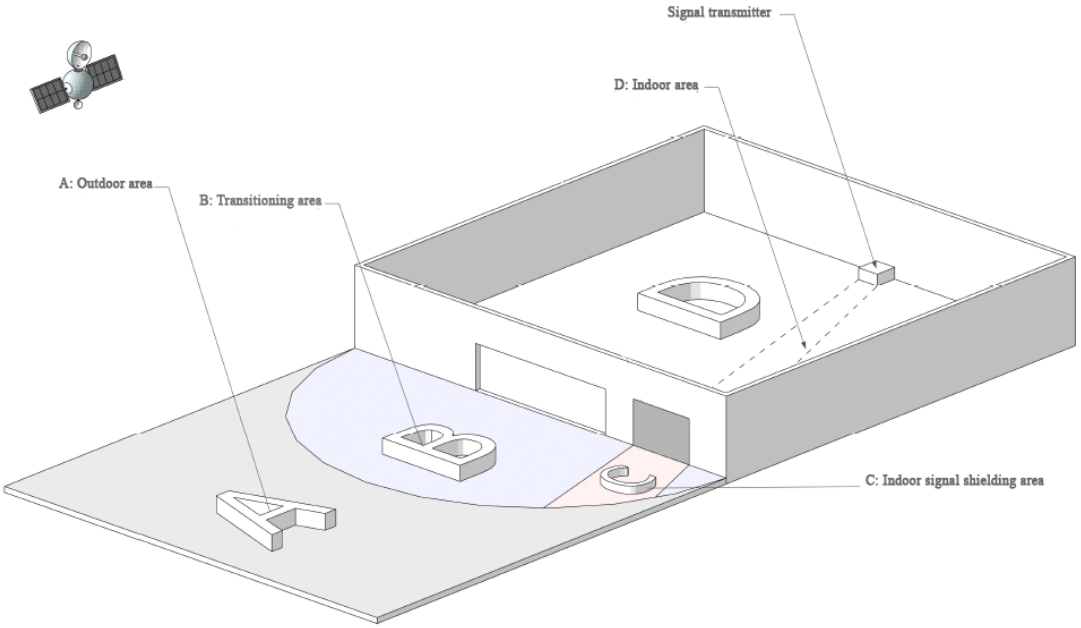

Figure 3.1: Example of application scenario

Chapter 4

Scouting on Hardware

4.1 Required sensors

In order to accomplish the proposed solution, the robot has to be provided with the following hardware:

- an Inertial Measurement Unit;
- wheel encoders;
- an UWB tag (together with the UWB anchors that require to be installed in the environment);
- a GPS receiver.

4.2 Scouting on robot

4.2.1 Dogs

Xiaomi cyberdog

Dimensions: 771x355x400 mm

Operating time: about 1 h

Charging time: 160 min

Price: 1500 €

ROS compatible

Figure 4.1: Xiaomi cyberdog

It is equipped with 11 sensors, including an Inertial Measurement Unit, GPS modules, an Ultrasonic Sensor, and a geomagnetic sensor. At the front there is an Intel Realsense D450 depth camera together with an AI Interactive camera and a dual-view super camera. It's rated to carry up to 3 kg in payload and its maximum walking speed is about 3 m/s. There are 3 USB Type-C ports (for fast charging, downloading, peripheral expansion) and an HDMI port useful for connecting other sensors. It also supports XiaoAi voice control, but only for Chinese.

Unitree Go1 Pro

Dimensions: 645x280x400 mm

Operating time: 1-2 h

Charging time: about 90 min

Price: 3050 €

ROS compatible

Figure 4.2: Unitree Go1

The robot walks alongside its human master. It is equipped with 5 sets Fish-eye Stereo Depth Cameras and 3 sets of Ultrasonic sensors, mounted on the front and on both sides. The robot is provided with an Inertial Measurement Unit. There are multiple external interfaces, such as 3 HDMI ports and 3 USB ports reserved for development, a Gigabit Ethernet port, and an Integration expansion interface. The maximum payload is 3 kg.

Alphadog C100

Dimensions: 480x300x400 mm

Operating time: 2.5 h (average)

Charging time: -

Price: 4940 €

ROS compatible

Figure 4.3: Alphadog C100

Unlike an ordinary robot dog with remote control, it moves freely in the environment with the help of 5G mobile internet. It is not equipped with a camera or onboard sensors. However, they can be added thanks to the external ports on the side (RJ45, USB, WIFI). The payload capacity is up to 5 kg. It moves with a speed up to 2.5 m/s and it is capable of climbing slopes up to 20°.

4.2.2 Rovers

Turtlebot3

Dimensions: 281x306x141 mm

Operating time: 2 h

Charging time: 2.5 h

Price: 1500 €

ROS compatible

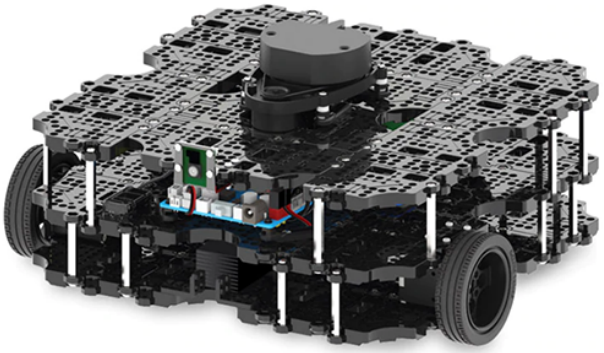

Figure 4.4: Turtlebot3 WafflePi

It is equipped with a Raspberry Pi Camera for perception, a 360° LiDAR for SLAM and navigation, and a 9-axis IMU (3- axis accelerometer, 3-axis gyroscope, 3-axis magnetometer). It is modular, compact and customizable. Different peripherals can be connected to the robot platform by means of several ports and pins, such as a CAN, three UART, 18 pins GPIO, and a 32 pins Arduino. It also provides an USB port for PC connection. It's rated to carry up to 30 kg and the maximum translational velocity is 0.26 m/s. The sealed Dynamixel actuators in the XW range are waterproof. They are IP68 certified, which guarantees also full dust protection. The connectors and cables are also waterproof to ensure the actuator is not damaged in any way. However, the standard platform is provided with servos of XM430 series.

Leo Rover

Dimensions: 410x460x270 mm

Operating time: up to 4 h

Charging time: about 2 h

Price: 2760 € (4800 € assembled)

ROS compatible

Figure 4.5: Leo Rover

The robot is equipped with a 2 MP CMOS OV2710 camera. It offers a wide range of waterproof external connectors (SPI, USB, 3 x timer output/counter input/IO, 2 x 7V/2A, 1 x 24V/3A, 1 x 5V/1A), allowing you to add extra modules. It can be customized with accessories such as grippers, a camera, sensors, an IMU, and so on. It is waterproof and guarantees protection from the dust. The protection rating is in fact IP66 (not certified). It can handle loads of up to 5 kg.

RR100 EDU

Dimensions: 860x659x801 mm

Operating time: about 5 h

Charging time: 80 min

Price: 29000 €

ROS compatible

Figure 4.6: RR100 EDU

The standard version of the RR100 robot is equipped with an Intel Realsense D435 depth camera, an IMU, a GPS, a RSLIDAR- 16 LiDAR, and a WIFI router. Mounting rails on top can be used by users to add other sensors. It comes with external USB and Ethernet, in addition to USB, Ethernet, 5V, 12V and 24V connections inside the robot. The RR100 is designed to be outdoors in all weathers (IP54). It can climb 20° slopes and overcome obstacles up to 13 cm high. It has a maximum payload of 50 kg (100 kg when the robot is equipped with 5 motors).

Agilex Scout Mini

Dimensions: 627x550x252 mm

Operating time: -

Charging time: 2 h

Price: about 7000 €

ROS compatible

Figure 4.7: Agilex Scout Mini

It is an all-terrain high-speed mini UGV with four-wheel differential drive. The CAN interface on top of the robot platform can be used to install peripherals, such as GPS, camera, laser range finder and so on. It can carry up to 10 kg of payload with a maximum speed of 10 km/h. It can overcome obstacles up to 7 cm high and climb slopes up to 30°. Waterproof plug-in components are adopted for the electrical interfaces, allowing the use of the rover even under severe operating conditions.

4.2.3 Recap on robots' features

Table 4.1 and Table 4.2 compare the features of the mentioned robots within dogs and rovers respectively.

	Xiaomi cyberdog	**Unitree Go1 Air**	**Unitree Go1 Pro**	**Unitree Go1 Edu**	**Alphadog C100**
ROS	✔	✔	✔	✔	✔
GPS	✔	✖	✖	✖	✖
IMU	✔	✖	✖	✖	✖
LiDAR	✖	✖	✖	✔	✖
Laser Rangefinder	✖	✖	✖	✖	✖
Camera	Intel Realsense D450 depth camera	Single wide-angle stereo depth camera	5 wide-angle stereo depth cameras	5 wide-angle stereo depth cameras	✖
Peripherals	USB-C, HDMI	3xUSB, 3xHDMI, Gigabit Ethernet, Integration expansion interface	3xUSB, 3xHDMI, Gigabit Ethernet, Integration expansion interface	3xUSB, 3xHDMI, Gigabit Ethernet, Integration expansion interface	RJ45, USB, WiFi
Payload	3 kg	3 kg	3 kg	10 kg	5 kg
Operating time	about 1 h	1-2 h	1-2 h	1-2 h	2.5 h
Charging time	160 min	about 90 min	about 90 min	about 90 min	–
Protection rating	–	–	–	–	–
Price	1500 €	2400 €	3050 €	7400 €	4940 €

Table 4.1: Comparison between dogs

	Turtlebot3 WafflePi	**Leo Rover**	**RR100 EDU**	**Agilex Scout Mini**
ROS	✔	✔	✔	✔
GPS	✖	✖	✔	✖
IMU	✔	✖	✔	✖
LiDAR	✔	✖	✔	✖
Laser Rangefinder	✖	✖	✔	✖
Camera	Raspberry Pi camera	2 MP CMOS OV2710 camera	Intel Realsense D435 depth camera	✖
Peripherals	CAN, 3xUART, GPIO 18 pins, Arduino 32 pins, USB	SPI, USB	USB, Ethernet, mounting rails	CAN
Payload	30 kg	5 kg	50 kg	10 kg
Operating time	2 h	up to 4 h	about 5 h	–
Charging time	2.5 h	about 2 h	80 min	2 h
Protection rating	–	IP66 (not certified)	IP54	✔
Price	1500 €	2760 €	29000 €	7000 €

Table 4.2: Comparison between rovers

Chapter 5

GPS Degradation

GPS signal provides position with a user accuracy of few meters under open sky [26]. Nevertheless, many factors can worsen the GPS performance in indoor and urban environments. Common causes are:

- signal blockage due to the presence of tall buildings, foliage, or bridges;
- signal reflection off walls or buildings, causing the well-known multipath effect.

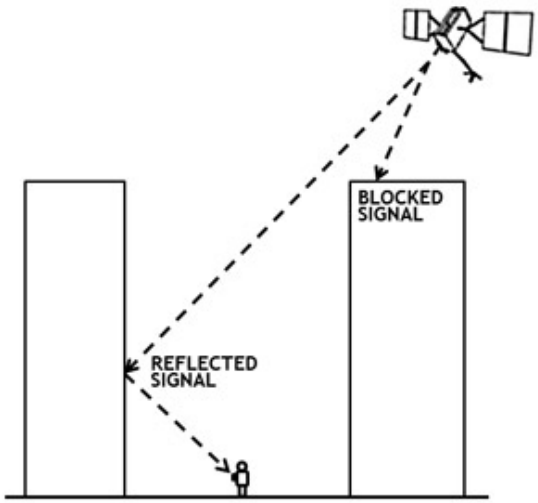

Figure 5.1: Reflected and blocked signals

As stated in [27], the main challenge for urban and indoor environments is the need

of GPS receivers to operate at low signal to noise power ratios (SNRs). The SNRs of GPS signals depend on the elevation angle of the satellite and they are deeply influenced by the presence of objects in the line of sight. In particular, Figures 5.2, 5.3, and 5.4 demonstrate how the SNRs degrade under foliage and even more inside a hotel.

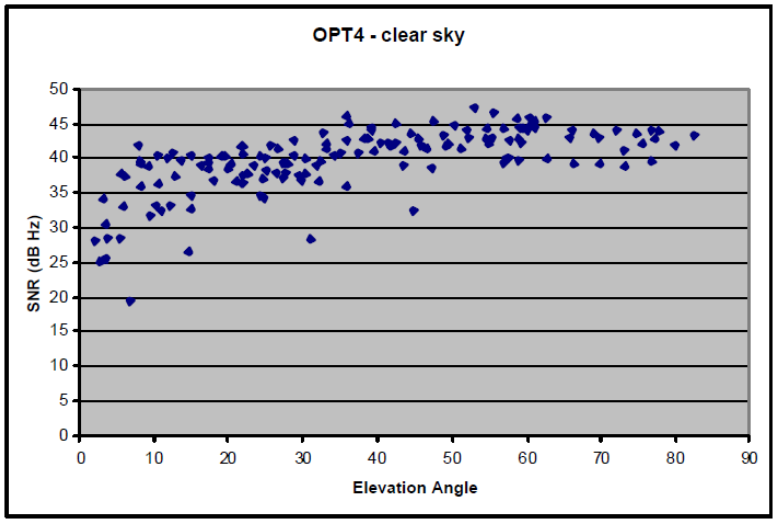

Figure 5.2: SNR vs. elevation for 11 satellites viewed from the roof of a building. Source [27]

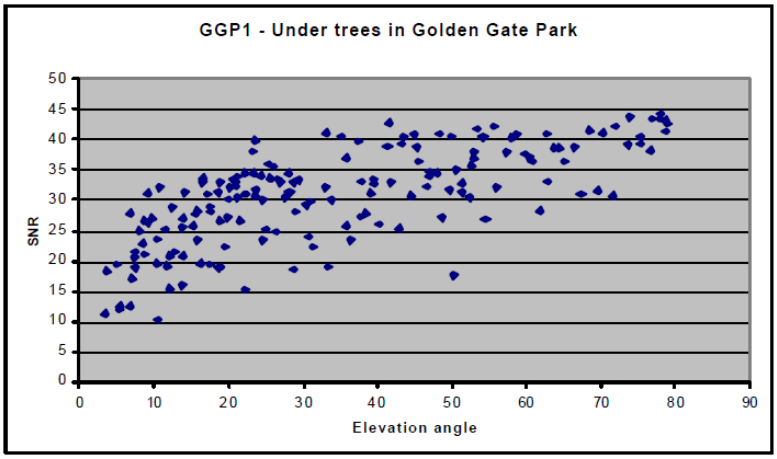

Figure 5.3: SNR vs. elevation for 11 satellites viewed from under foliage in a park. Source [27]

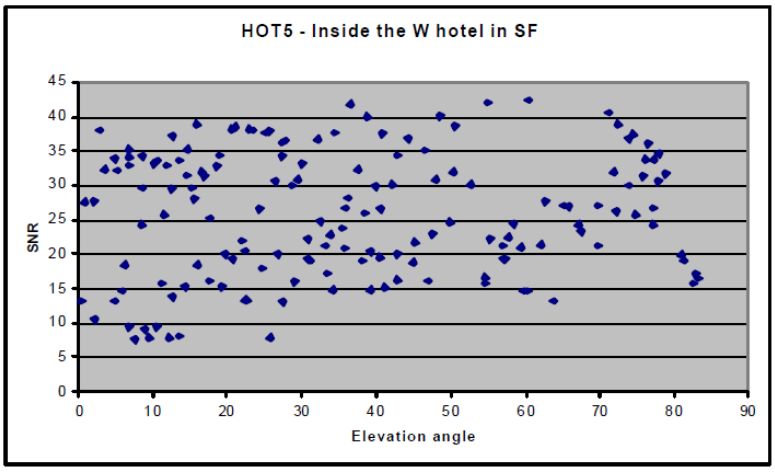

Figure 5.4: SNR vs. elevation for 11 satellites viewed from inside the W Hotel in San Francisco. Source [27]

Moreover, the received signal in urban and indoor environments may derive from multiple reflections. The multipath phenomenon introduces large errors since reflected, scattered and/or diffracted signal components can exceed direct signal (see Fig. 5.5).

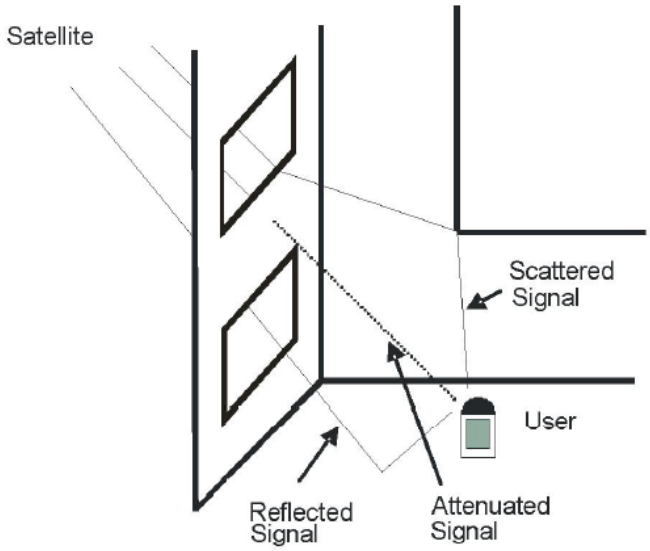

Figure 5.5: Example of indoor GPS signal. Source [28]

5.1 Degradation modeling

Ma et al. [29] provide a statistic model known as *Urban Three-State Fade Model* (UTSFM) describing the fading distribution of GPS signals. According to this model, a GPS signal can be statistically described by a composite amplitude probability density function that is compounded by three pdf:

- a Ricean pdf for clear line-of-sight (CLOS) signals;
- a Rayleigh pdf for blocked signals;
- a Loo's function for shadowed signals.

The overall probability density function is

$$f(\alpha, v) = C(\alpha) f_{Ricean}(v) + S(\alpha) f_{Loo}(v) + B(\alpha) f_{Rayleigh}(v) \tag{5.1}$$

where α is the elevation angle of the satellites, $C(\alpha) + S(\alpha) + B(\alpha) = 1$, and C($\alpha$), S($\alpha$) and B($\alpha$) are weight coefficients indicating the relative magnitude of each component at a certain elevation range. The pdf for each component of the GPS signal are:

$$f_{Ricean}(v) = 2Kv\ exp\left[-K(v^2+1)\right] I_0(2Kv) \tag{5.2}$$

$$f_{Rayleigh}(v) = 2Kv\ exp\left[-Kv^2\right] \tag{5.3}$$

$$f_{Loo}(v) = 8.686\sqrt{\frac{2}{\pi}}\frac{Kv}{\sigma} \int_0^\infty \frac{1}{z} exp\left(-\frac{(20log(z)-m)^2}{2\sigma^2} - K(v^2+z^2)\right) I_0(2Kvz)dz \tag{5.4}$$

where K is the ratio of the direct power received to the multipath power, v is the received voltage relative to the clear path voltage, I_0 is the 0^{th} order modified Bessel function. As showed in Loo's function, shadowed signals consist of attenuated LOS signals, which fade is logarithmic, and multipath signals, which fade according to Rayleigh's function.

Chapter 6

Simulation Environment

As mentioned in 2.3, an Extended Kalman Filter is used to evaluate the robot pose in two steps: by predicting the pose using a nonlinear dynamic model and, then, by correcting such prediction exploiting the sensed information. For this work, the robot under study is a Turtlebot3 WafflePi, which is already equipped by wheel encoders and an IMU. A GPS receiver is also considered as mounted on the robot in order to simulate the proposed solution.

The simulations are conducted in *ROS2 Foxy* by using the **robot_localization** package, developed to perform sensor fusion for localization.

6.1 robot_localization package

The *robot_localization* package provides a collection of state estimation nodes, i.e. "nonlinear state estimator for robots moving in 3D space" as explained in [30]. The package comprises two state estimation nodes, called *ekf_localization_node* and *ukf_localization_node*, and the node *navsat_transform_node* for integrating GPS data with other sensed information. The former two nodes implement the Extended Kalman Filter and the Unscented Kalman Filter respectively. They both share the following common features:

- **continuos estimation**, since the pose will be provided by exploiting the internal motion model in case of a lasting shortage of sensors' measurements;

- **support for multiple ROS message types**, namely *nav_msgs/Odometry*, *sensor_msgs/Imu*, *geometry_msgs/PoseWithCovarianceStamped*, or *geometry_msgs/TwistWithCovarianceStamped* messages;
- **arbitrary number of fused sensors**;
- **per-sensor input customization** to exclude irrelevant data from a particular sensor.

6.1.1 ekf_localization_node

The *ekf_localization_node* implements an EKF that predicts the robot's state by means of an omnidirectional motion model and corrects the prediction using the data derived from the sensors. The state is composed by 15 variables:

$$\begin{bmatrix} x & y & z \\ \phi & \theta & \psi \\ \dot{x} & \dot{y} & \dot{z} \\ \dot{\phi} & \dot{\theta} & \dot{\psi} \\ \ddot{x} & \ddot{y} & \ddot{z} \end{bmatrix}$$

where ϕ, θ, and ψ represent the roll, pitch, and yaw angles respectively.

For each sensor, the variables need to be defined as *true* or *false* in order to take into account only the ones set to *true* for performing sensor fusion. Another important parameter is the process covariance matrix that represents the amount of uncertainty introduced by the internal motion model during the prediction phase of the filter. It has to be set for each sensor, too [31].

6.1.2 navsat_transform_node

The *navsat_transform_node* is used to convert GPS data, that are usually provided in terms of latitude, longitude, and altitude, into coordinates in the robot's world frame (see Appendix B for further details on frames). As depicted in Fig. 6.1, it requires three inputs for transforming GPS data into the robot's world frame:

- a *nav_msgs/Odometry* message, containing the estimated current position of the robot in its world frame;

- a *sensor_msgs/Imu* message, carrying an earth-referenced estimate of the robot's heading;
- a *sensor_msgs/NavSatFix* message, providing geographic coordinate expressed as a latitude/longitude pair.

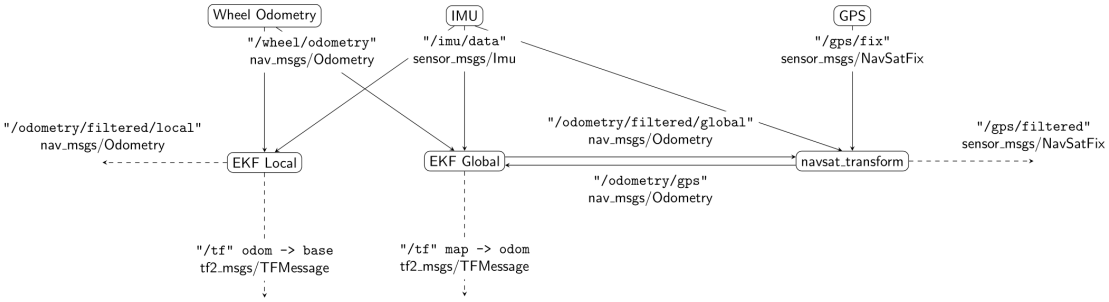

Figure 6.1: Example setup for integrating GPS data with navsat_transform_node [32]

Chapter 7

Simulation Results

The goal of this work is to build a simulation environment that can be used to demonstrate the validity of the intended sensor fusion. For this purpose, the *robot_localization* ROS package described in the previous chapter is exploited.

As a first step, the prototype scenario has been created in the **Gazebo** simulator. The intended environment is composed by a simple factory made through the Gazebo Editor. The model of the building is included in a Gazebo world together with the model of a Turtlebot3. Any of the three versions of the robot (*Burger*, *Waffle*, and *WafflePi*) can be deployed by assigning '*burger*', '*waffle*', or '*wafflepi*' to the variable *TURTLEBOT3_MODEL*.

Then, two simulations are conducted:

- a local EKF estimation node providing sensor fusion between IMU and wheel encoders data;
- a global state estimate of robot pose obtained merging the data from the GPS.

7.1 Local EKF

By means of a python launching file, a single instance of an EKF state estimation node is run. The node named as *ekf_filter_node* receives data from IMU and wheel encoders by subscribing to the topics */imu/data* and */wheel/odometry* respectively.

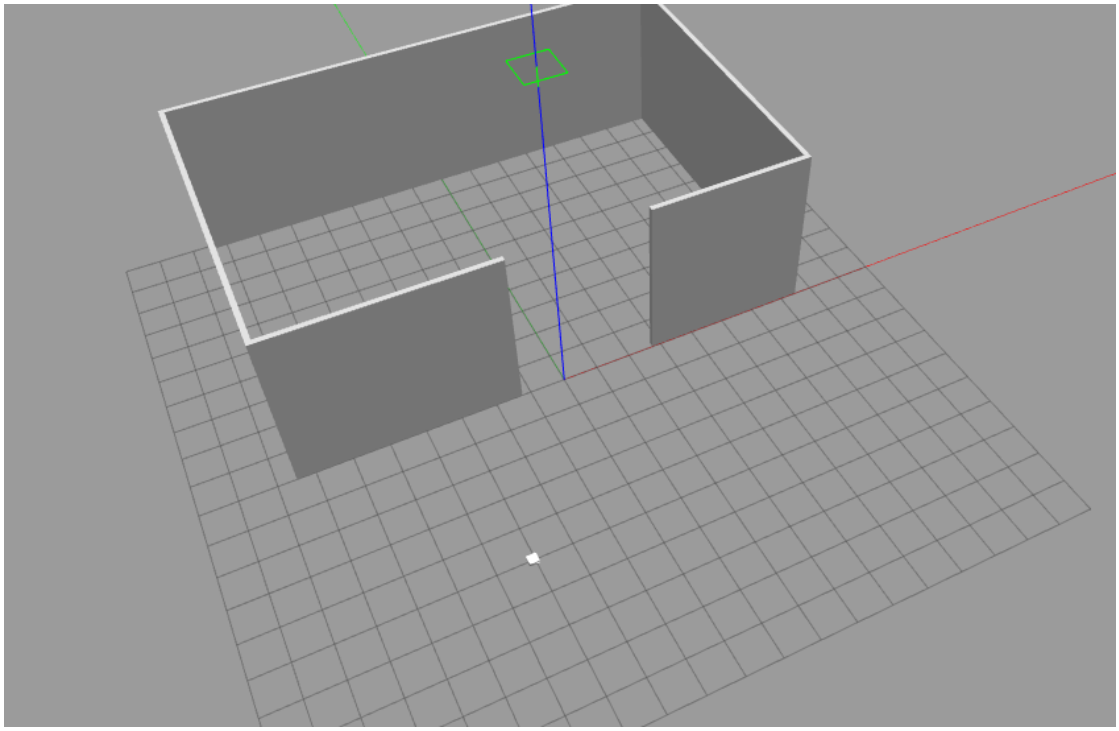

Figure 7.1: Gazebo world

The derived information are combined by the EKF using the configuration pictured in Table 7.1:

Sensor	**x**	**y**	**z**	ϕ	θ	ψ	$\dot{x}$	$\dot{y}$	$\dot{z}$	$\dot{\phi}$	$\dot{\theta}$	$\dot{\psi}$	$\ddot{x}$	$\ddot{y}$	$\ddot{z}$
Odometry	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0
IMU	0	0	0	1	1	1	0	0	0	1	1	1	1	1	1

Table 7.1: Sensor configuration (0 = false, 1 = true) for local EKF node

The obtained estimation of the robot pose is published on the */odometry/local* topic, as Figure 7.3 shows.

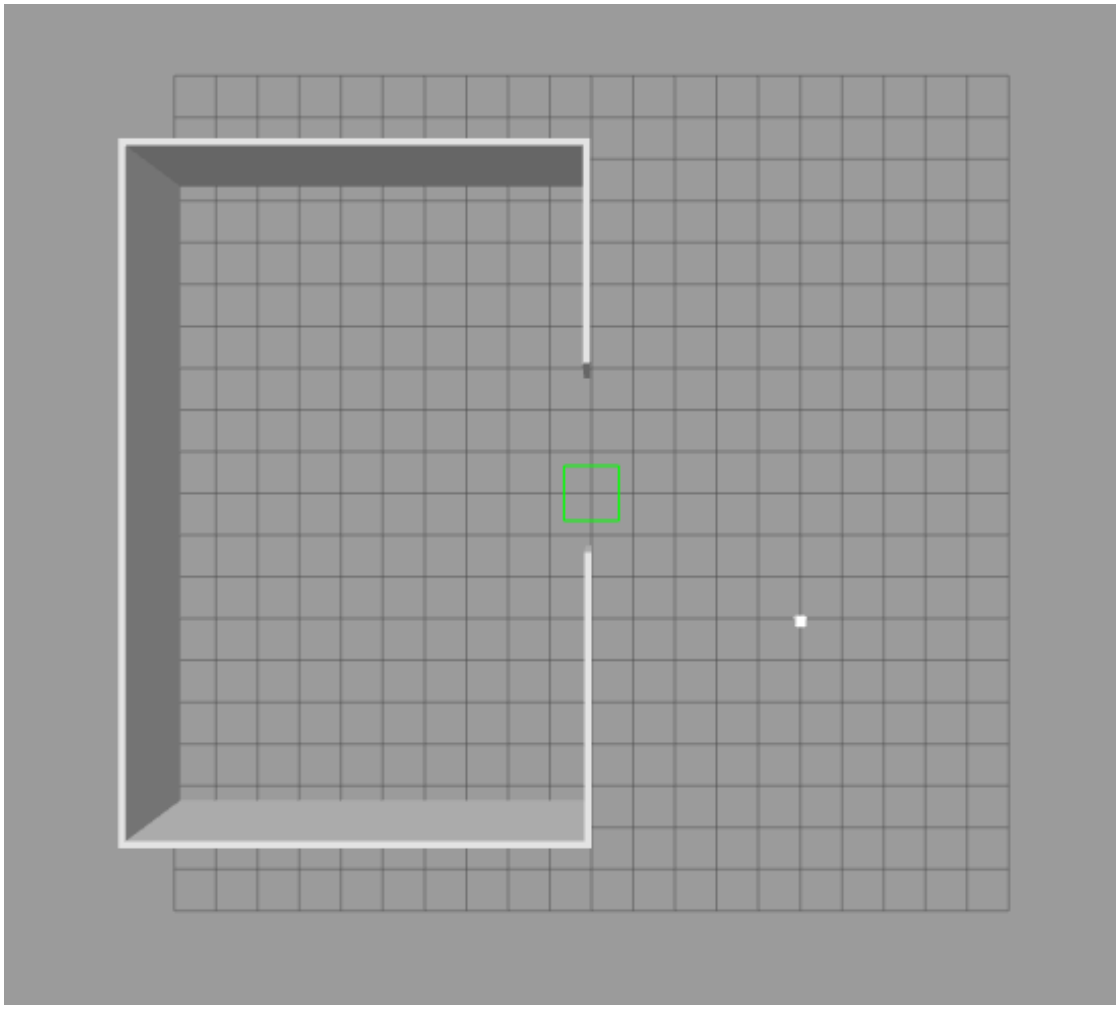

Figure 7.2: Gazebo world top view

7.2 Global EKF

In order to perform sensor fusion using also the information provided by GPS sensor, two state estimation nodes are launched:

- an EKF node fusing only continuous data, i.e. data from odometry and IMU;
- another EKF node that deals with the discrete jumps characterizing GPS data.

The *navsat_transform_node* communicates with the *turtlebot3_gps* node, which

publishes *NavSatFix* messages on the */gps/fix* topic. The raw GPS coordinates expressed as latitude/longitude need to be transformed in the robot world frame. For this purpose, the *navsat_transform_node* requires three sources of data:

- the GPS information (see *sensor_msgs/NavSatFix Message* in Appendix C);
- an earth-referenced heading (see *sensor_msgs/Imu Message* in Appendix C);
- the robot's current pose in its world frame, that is specified through its initial location (see *nav_msgs/Odometry Message* in Appendix C).

Once GPS coordinates are converted into the robot's world frame, these data are fused with the ones coming from IMU and wheel encoders through the global state estimation node which has the following configuration vectors:

Sensor	**x**	**y**	**z**	$\boldsymbol{\phi}$	$\boldsymbol{\theta}$	$\boldsymbol{\psi}$	$\dot{\boldsymbol{x}}$	$\dot{\boldsymbol{y}}$	$\dot{\boldsymbol{z}}$	$\dot{\boldsymbol{\phi}}$	$\dot{\boldsymbol{\theta}}$	$\dot{\boldsymbol{\psi}}$	$\ddot{\boldsymbol{x}}$	$\ddot{\boldsymbol{y}}$	$\ddot{\boldsymbol{z}}$
Odometry	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0
IMU	0	0	0	1	1	1	0	0	0	1	1	1	1	1	1
GPS	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7.2: Sensor configuration (0 = false, 1 = true) for global EKF node

The obtained estimation of the robot pose is published on the */odometry/global* topic. Appendix A provides the ROS graph representing the described simulation environment.

7.3 RViz validation

The two simulations are validated by means of the 3D visualization tool **RViz**. Once the turtlebot is controlled using keyboard teleoperation, the RViz tool shows the pose derived from the topics described in the previous sections. Figure 7.4 represents the estimated motion, that coincides with the commands provided.

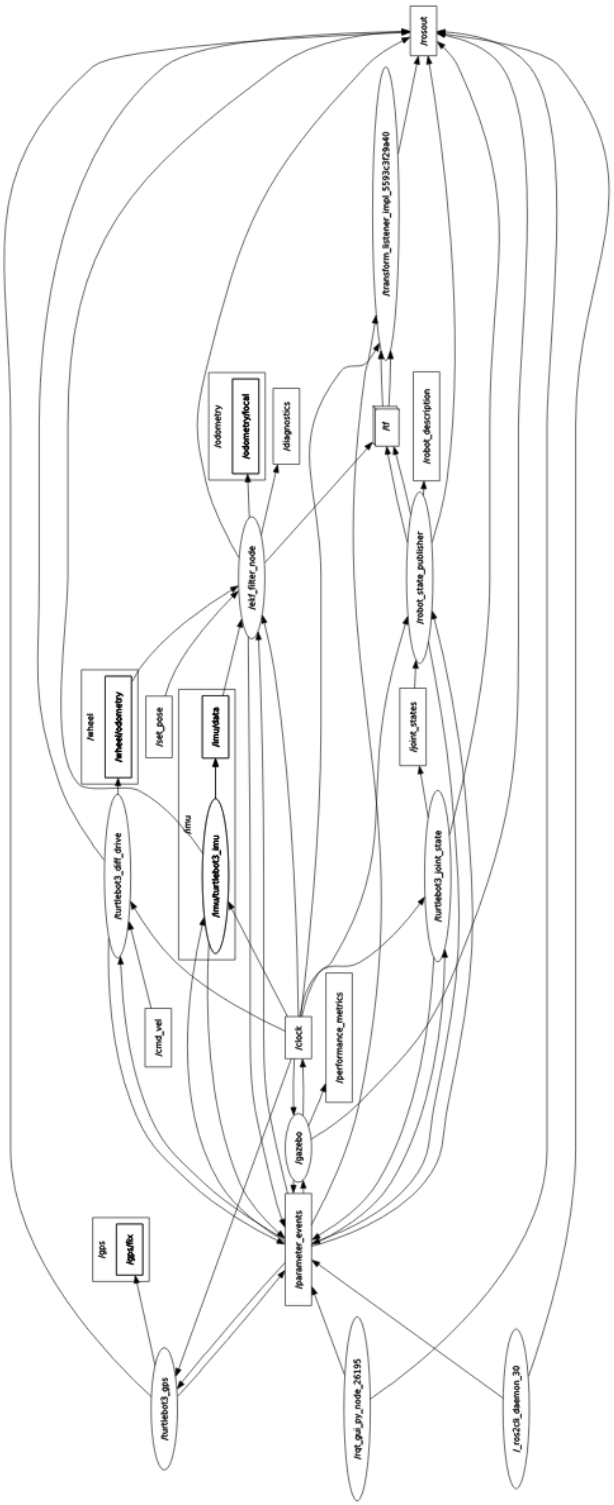

Figure 7.3: ROS graph for the local EKF

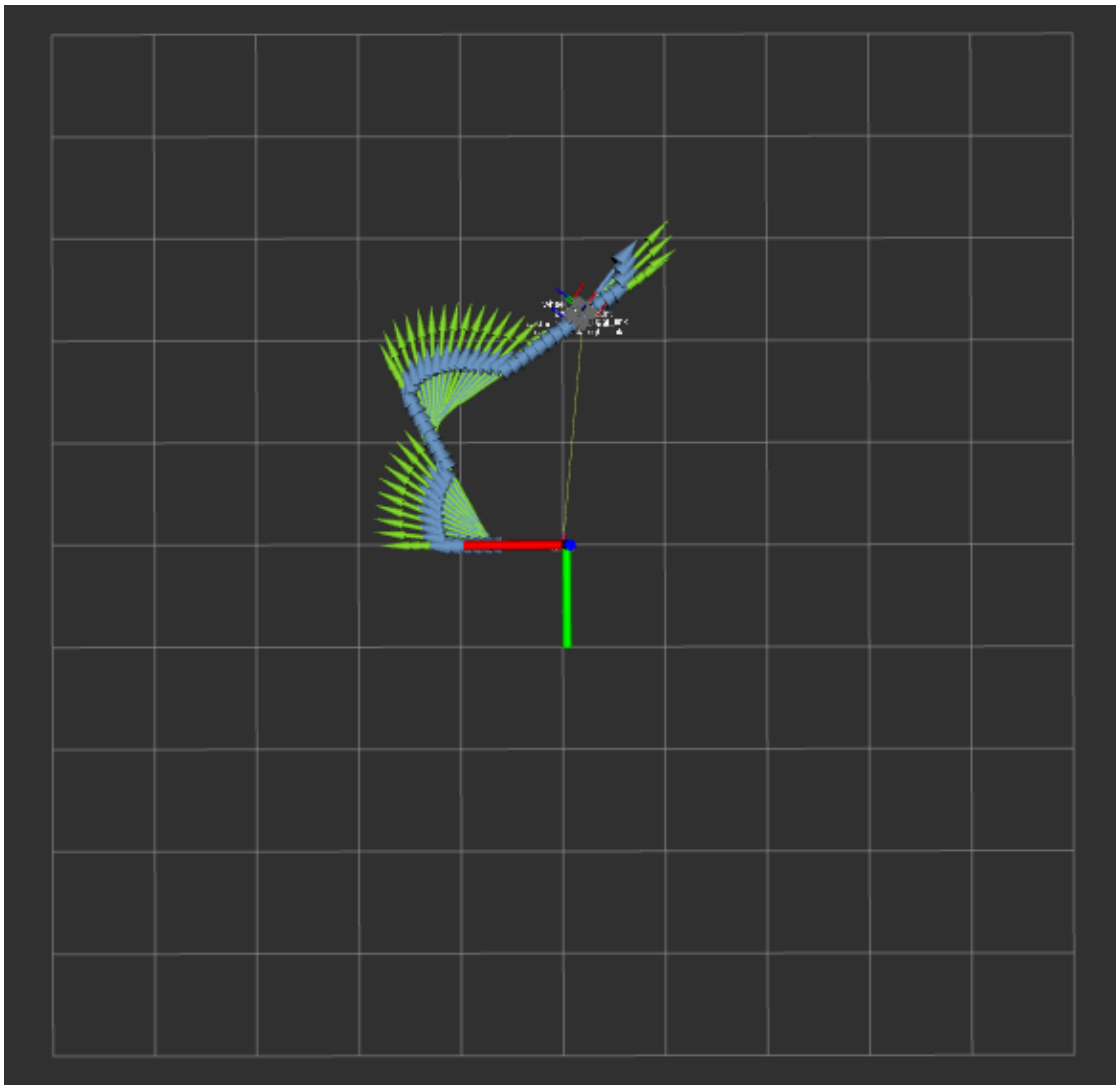

Figure 7.4: RViz validation of EKF sensor fusion

Chapter 8

Conclusions and future works

The thesis addresses the problem of localization handover for scenarios requiring different localization technologies for indoor and outdoor areas. Particularly, it focuses on the sensor fusion performed in the outdoor and in the transitioning area, where GPS and UWB signals can coexist. However, such signals can be remarkably degraded due to the presence of the walls of the building.

In order to provide a realistic solution to the problem, a detailed research on the robots available on the market is conducted. Among the investigated robots which satisfy the requirements, **Turtlebot3** is chosen as the least expensive, expandable mobile platform. Nevertheless, a different robot can be preferred according to company needs.

Once the factors affecting the quality of GPS signals are investigated, a solution involving an Extended Kalman Filter is proposed in order to avoid additional complexity. First, the EKF is used to fuse the information derived from wheel encoders and IMU; then, the output of the Kalman filter is combined with GPS data by means of a weighting sum. The weight can be defined according to the variance of GPS readings.

Finally, the last goal of this work is setting up the simulation environment in ROS.

The *robot_localization* package is exploited to perform simulated localization.

8.1 Next works

Once the simulation environment is set up, the proposed solution shall be implemented in ROS, taking into account also the issues affecting the sensors (e.g, wheel slippage, foliage coverage, etc...). Simulations for evaluating the performance of sensor fusion between IMU, wheel odometry, UWB, and GPS need to be conducted. Finally, the proposed solution shall be deployed and integrated on a *Turtlebot3*.

Bibliography

[1] Bluetooth SIG, Inc. *Bluetooth*. 2022. URL: https://www.bluetooth.com/ (cit. on p. 4).

[2] Gunter Fischer, Burkhart Dietrich, and Frank Winkler. «Bluetooth indoor localization system». In: (Jan. 2004) (cit. on pp. 4–6).

[3] M Fachri and A Khumaidi. «Positioning Accuracy of Commercial Bluetooth Low Energy Beacon». In: *IOP Conference Series: Materials Science and Engineering* 662.5 (Nov. 2019), p. 052018. DOI: 10.1088/1757-899x/662/5/052018. URL: https://doi.org/10.1088/1757-899x/662/5/052018 (cit. on p. 6).

[4] wikipedia.org. *Wireless LAN*. 2022. URL: https://en.wikipedia.org/wiki/Wireless_LAN (cit. on p. 6).

[5] wikipedia.org. *Wi-Fi positioning system*. 2022. URL: https://en.wikipedia.org/wiki/Wi-Fi_positioning_system (cit. on pp. 6, 7).

[6] Weixin Huang, Yuming Lin, and Wu Mingbo. «Spatial-Temporal Behavior Analysis Using Big Data Acquired by Wi-Fi Indoor Positioning System». In: Apr. 2017. DOI: 10.52842/conf.caadria.2017.745 (cit. on p. 6).

[7] Xiaojie Zhao, Zhuoling Xiao, Andrew Markham, Niki Trigoni, and Yong Ren. «Does BTLE measure up against WiFi? A comparison of indoor location performance». In: *European Wireless 2014; 20th European Wireless Conference*. 2014, pp. 1–6 (cit. on p. 7).

[8] Hailiang Xiong and Julian Cheng. «Investigation of short-range high precision 3D localization via UWB radio». In: *2014 IEEE Global Communications*

Conference. 2014, pp. 4090–4095. DOI: 10.1109/GLOCOM.2014.7037448 (cit. on p. 7).

[9] J. Gonzalez, J. L. Blanco, C. Galindo, A. Ortiz-de-Galisteo, J.A. Fernandez-Madrigal, F.A. Moreno, and J.L. Martinez. «Combination of UWB and GPS for indoor-outdoor vehicle localization». In: *2007 IEEE International Symposium on Intelligent Signal Processing*. 2007, pp. 1–6. DOI: 10.1109/WISP.2007.4447550 (cit. on pp. 7, 12).

[10] S. Gezici, Zhi Tian, G.B. Giannakis, H. Kobayashi, A.F. Molisch, H.V. Poor, and Z. Sahinoglu. «Localization via ultra-wideband radios: a look at positioning aspects for future sensor networks». In: *IEEE Signal Processing Magazine* 22.4 (2005), pp. 70–84. DOI: 10.1109/MSP.2005.1458289 (cit. on p. 7).

[11] Lukasz Zwirello, Tom Schipper, Marlene Harter, and Thomas Zwick. «UWB Localization System for Indoor Applications: Concept, Realization and Analysis». In: *Journal of Electrical and Computer Engineering* 2012 (Mar. 2012). DOI: 10.1155/2012/849638 (cit. on p. 8).

[12] Wang Shule, Carmen Martínez Almansa, Jorge Peña Queralta, Zhuo Zou, and Tomi Westerlund. «UWB-Based Localization for Multi-UAV Systems and Collaborative Heterogeneous Multi-Robot Systems». In: *Procedia Computer Science* 175 (2020). The 17th International Conference on Mobile Systems and Pervasive Computing (MobiSPC),The 15th International Conference on Future Networks and Communications (FNC),The 10th International Conference on Sustainable Energy Information Technology, pp. 357–364. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2020.07.051. URL: https://www.sciencedirect.com/science/article/pii/S1877050920317324 (cit. on p. 8).

[13] Byung Gyu Yu, Geunhaeng Lee, Hong Gul Han, Won-Sang Ra, and Tae Wook Kim. «A Time-Based Angle-of-Arrival Sensor Using CMOS IR-UWB Transceivers». In: *IEEE Sensors Journal* 16 (2016), pp. 5563–5571 (cit. on p. 9).

[14] wikipedia.org. *Global Positioning System*. 2022. URL: https://en.wikipedia.org/wiki/Global_Positioning_System (cit. on p. 10).

[15] Geoffrey Blewitt. «Basics of the GPS Technique : Observation Equations». In: 2000 (cit. on p. 10).

[16] wikipedia.org. *World Geodetic System*. 2022. URL: https://en.wikipedia.org/wiki/WGS84 (cit. on p. 11).

[17] Jonggu Kang, Daeyoung Kim, Eunjo Kim, Youngsoo Kim, Seong-eun Yoo, and Daehan Wi. «Seamless mobile robot localization service framework for integrated localization systems». In: *2008 3rd International Symposium on Wireless Pervasive Computing*. 2008, pp. 175–179. DOI: 10.1109/ISWPC.2008.4556191 (cit. on p. 12).

[18] M. Agrawal and K. Konolige. «Real-time Localization in Outdoor Environments using Stereo Vision and Inexpensive GPS». In: *18th International Conference on Pattern Recognition (ICPR'06)*. Vol. 3. 2006, pp. 1063–1068. DOI: 10.1109/ICPR.2006.962 (cit. on p. 12).

[19] E. B. Pacis, B. Sights, G. Ahuja, G. Kogut, and H. R. Everett. «An adaptive localization system for outdoor/indoor navigation for autonomous robots». In: *Unmanned Systems Technology VIII*. Ed. by Grant R. Gerhart, Charles M. Shoemaker, and Douglas W. Gage. Vol. 6230. International Society for Optics and Photonics. SPIE, 2006, pp. 716–727. DOI: 10.1117/12.668520. URL: https://doi.org/10.1117/12.668520 (cit. on p. 12).

[20] Matthias Hentschel, Oliver Wulf, and Bernardo Wagner. «A GPS and laser-based localization for urban and non-urban outdoor environments». In: *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2008, pp. 149–154. DOI: 10.1109/IROS.2008.4650585 (cit. on p. 12).

[21] Pablo Urcola, Maite Lorente, J. Villarroel, and Luis Montano. «Seamless Indoor-Outdoor Robust Localization for Robots». In: vol. 253. Jan. 2014, pp. 275–287. ISBN: 9783319036526. DOI: 10.1007/978-3-319-03653-3_21 (cit. on p. 12).

[22] Sofia Yousuf and Muhammad Bilal Kadri. «Robot Localization in Indoor and Outdoor Environments by Multi-sensor Fusion». In: *2018 14th International Conference on Emerging Technologies (ICET)*. 2018, pp. 1–6. DOI: 10.1109/ICET.2018.8603597 (cit. on p. 13).

[23] GitHub, Inc. *Monte Carlo Localization for mobile robots in ROS*. 2022. URL: https://github.com/topics/monte-carlo-localization?o=desc&s=updated (cit. on p. 13).

[24] Sebastian Thrun, Dieter Fox, Wolfram Burgard, and Frank Dellaert. «Robust Monte Carlo localization for mobile robots». In: *Artificial Intelligence* 128.1 (2001), pp. 99–141. ISSN: 0004-3702. DOI: https://doi.org/10.1016/S0004-3702(01)00069-8. URL: https://www.sciencedirect.com/science/article/pii/S0004370201000698 (cit. on p. 14).

[25] Sofia Yousuf and Muhammad Bilal Kadri. «Sensor fusion of INS, odometer and GPS for robot localization». In: *2016 IEEE Conference on Systems, Process and Control (ICSPC)*. 2016, pp. 118–123. DOI: 10.1109/SPC.2016.7920715 (cit. on p. 15).

[26] GPS.gov. *GPS Accuracy*. 2022. URL: https://www.gps.gov/systems/gps/performance/accuracy/ (cit. on p. 29).

[27] Per K. Enge, Rod Fan, Anil Tiwari, Andrew Chou, Wallace Mann, Anant Sahai, Jesse Stone, and B. Roy. «Improving GPS Coverage and Continuity: Indoors and Downtown». In: 2001 (cit. on pp. 29–31).

[28] Pavel Puricer and Pavel Kovar. «Technical Limitations of GNSS Receivers in Indoor Positioning». In: *2007 17th International Conference Radioelektronika*. 2007, pp. 1–5. DOI: 10.1109/RADIOELEK.2007.371487 (cit. on p. 31).

[29] Changlin Ma, Gyu-In Jee, Glenn Macgougan, Gérard Lachapelle, Scott Leland Bloebaum, Geoffrey F Cox, Lionel Jacques Garin, and John Shewfelt. «GPS Signal Degradation Modeling». In: 2001 (cit. on p. 32).

[30] ROS.org. *robot_localization wiki*. 2022. URL: http://docs.ros.org/en/melodic/api/robot_localization/html/index.html (cit. on p. 33).

[31] ROS.org. *State Estimation Nodes wiki*. 2022. URL: http://docs.ros.org/en/melodic/api/robot_localization/html/state_estimation_nodes.html (cit. on p. 34).

[32] ROS.org. *Integrating GPS Data wiki*. 2022. URL: http://docs.ros.org/en/melodic/api/robot_localization/html/integrating_gps.html (cit. on p. 35).

[33] ROS.org. *REP 103*. 2022. URL: https://www.ros.org/reps/rep-0103.html (cit. on p. 51).

[34] ROS.org. *REP 105*. 2022. URL: https://www.ros.org/reps/rep-0105.html (cit. on pp. 51, 53).

[35] wikipedia.org. *Geodetic Datum*. 2022. URL: https://en.wikipedia.org/wiki/Geodetic_datum#Local_east.2C_north.2C_up_.28ENU.29_coordinates (cit. on pp. 52, 53).

[36] wikipedia.org. *Local tangent plane coordinates*. 2022. URL: https://en.wikipedia.org/wiki/Local_tangent_plane_coordinates (cit. on p. 52).

[37] wikipedia.org. *Earth-centered, Earth-fixed coordinate system*. 2022. URL: https://en.wikipedia.org/wiki/Earth-centered,_Earth-fixed_coordinate_system (cit. on p. 53).

[38] ROS.org. *nav_msgs/Odometry Message*. 2022. URL: http://docs.ros.org/en/api/nav_msgs/html/msg/Odometry.html (cit. on p. 55).

[39] ROS.org. *sensor_msgs/Imu Message*. 2022. URL: http://docs.ros.org/en/api/sensor_msgs/html/msg/Imu.html (cit. on p. 55).

[40] ROS.org. *geometry_msgs/PoseWithCovarianceStamped Message*. 2022. URL: http://docs.ros.org/en/api/geometry_msgs/html/msg/PoseWithCovarianceStamped.html (cit. on p. 56).

[41] ROS.org. *geometry_msgs/TwistWithCovarianceStamped Message*. 2022. URL: http://docs.ros.org/en/api/geometry_msgs/html/msg/TwistWithCovarianceStamped.html (cit. on p. 56).

[42] ROS.org. *sensor_msgs/NavSatFix Message*. 2022. URL: http://docs.ros.org/en/api/sensor_msgs/html/msg/NavSatFix.html (cit. on p. 57).

Appendix A

ROS Graph

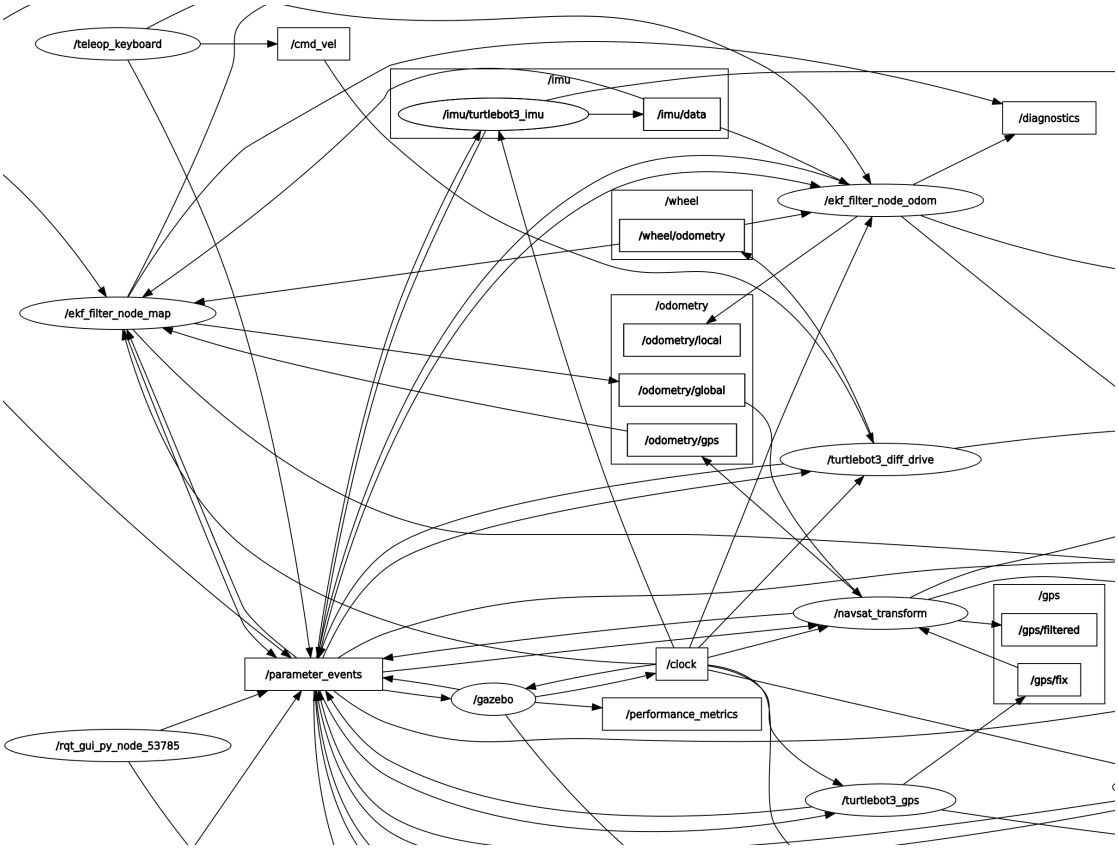

Figure A.1: ROS Graph zoomed

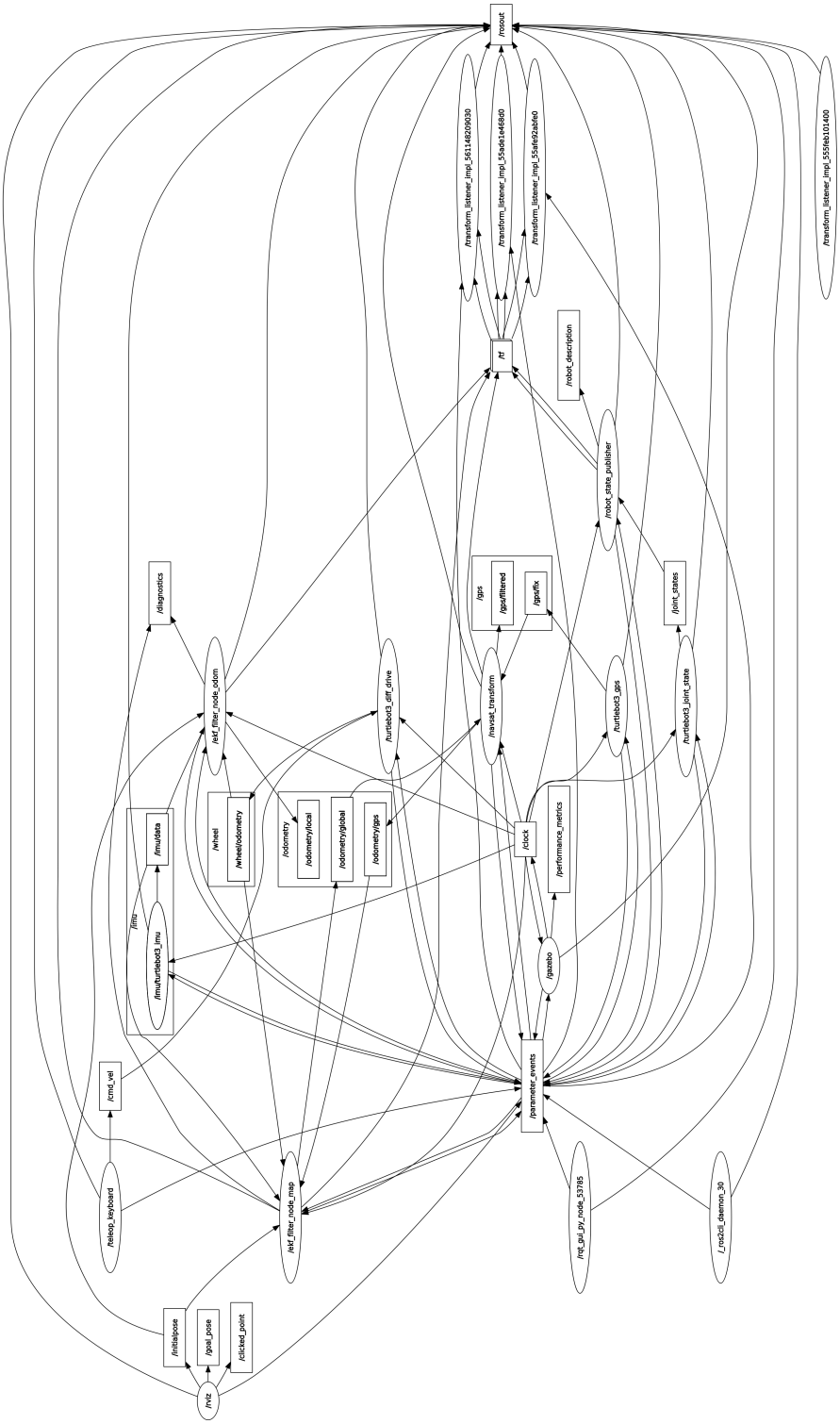

Figure A.2: ROS Graph

Appendix B

ROS Enhancement Proposal

REP 103 *Standard Units of Measure and Coordinate Conventions* [33] and **REP 105** *Coordinate Frames for Mobile Platforms* [34] are the crucial documents defining conventions for localization and navigation in ROS. While REP 103 specifies the units of measure and coordinate conventions, REP 105 is a reference for coordinate frames conventions used for mobile robots.

B.1 REP 103

B.1.1 Units of Measure

Base Units	
Quantity	**Units**
length	meter
mass	kilogram
time	second
current	ampere

Table B.1: REP 103 Base Units of Measure Conventions

Derived Units	
Quantity	**Units**
angle	radian
frequency	hertz
force	newton
power	watt
voltage	volt
temperature	celsius
magnetism	tesla

Table B.2: REP 103 Derived Units of Measure Conventions

B.1.2 Coordinate Frame Conventions

All coordinate systems shall comply with the right hand rule and the axis orientation described below:

Frames	x	y	z
body frame	forward	left	up
short-range locations (ENU [35])	east	north	up

Table B.3: REP 103 Axis Orientation

In the case of outdoor systems or cameras, it is generally provided a second frame with either the "_ned" or "_optical" suffix respectively. The "_ned" frame for outdoor system is defined since it is preferable to be compliant with the *NED* convention [36].

Frames	x	y	z
camera "_optical"	right	down	forward
outdoor "_ned"	north	east	down

Table B.4: REP 103 Suffix Frames

B.2 REP 105

A shared convention for coordinate frames allows ROS developers to integrate and re-use drivers, models, libraries, and other software components. *REP 105* provides information about the frames used for localizing mobile platforms in the environment. The defined tf2 coordinate frame tree for mobile robots is pictured below:

(earth) -> map -> odom -> base_link

base_link is the coordinate frame attached to the mobile robot frame. Its preferred orientation is stated in *REP 103* as X forward, Y left, and Z up.

The **odom** frame is a world-fixed frame. The robot pose in such a frame changes smoothly, without discrete jumps, guaranteeing continuity. However, the pose in the odom drifts over time making it reliable only for short-term local reference. Indeed, "the odom frame is computed based on an odometry source, such as wheel odometry, visual odometry or an inertial measurement unit" [34].

The **map** frame is a world-fixed frame, too. On the other hand, it is not a continuous frame, thus the pose of a mobile platform can change in discrete jumps. This makes the map frame an accurate long-term global reference since the robot pose is computed repeatedly from sensor measurements ruling out any drift. Even though *REP 103* defines ENU (see [35]) as preferred axis orientation, it may be more helpful to align the map in structured environments with a more appropriate frame.

Finally, **earth** is a global world-fixed frame that is used to make multiple robots interact in different map frames. Therefore, it is neglected when only a map frame is needed. It is compliant with the Earth-centered, Earth-fixed coordinate system (ECEF [37]).

Figure B.1 pictures the different frames ECEF, ENU, and latitude/longitude frame.

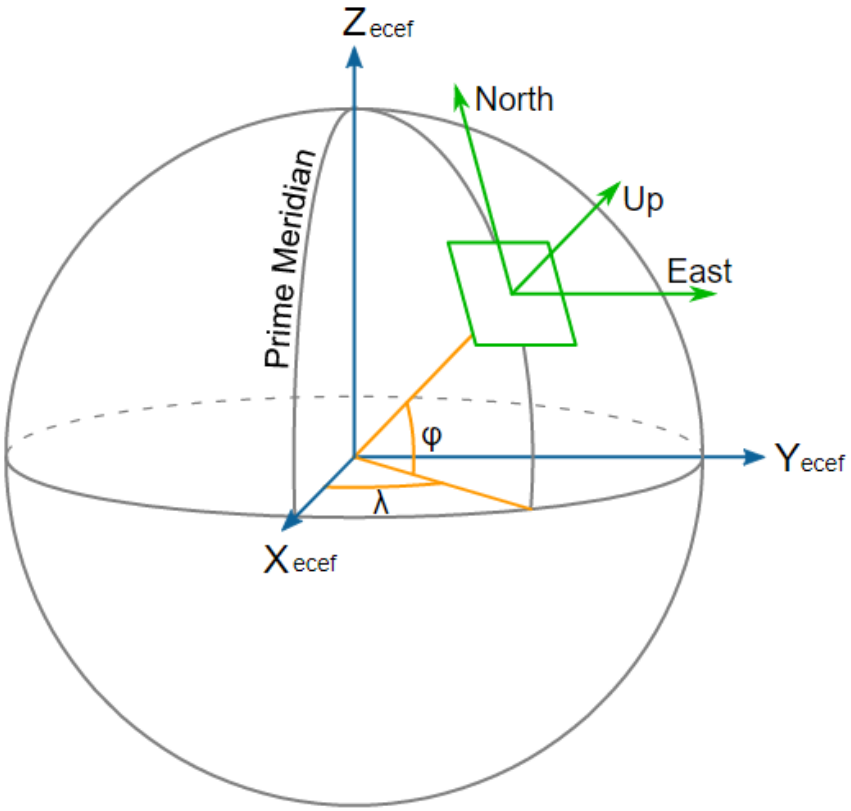

Figure B.1: Visual comparison between ECEF (blue), ENU (green), and latitude/longitude (yellow) frames of reference

Appendix C

ROS messages

C.1 nav_msgs/Odometry Message

It is defined in the *nav_msgs/Odometry.msg* file.

It "represents an estimate of a position and velocity in a free space. The pose in this message should be specified in the coordinate frame given by header.frame_id. The twist in this message should be specified in the coordinate frame given by the child_frame_id", as stated in [38].

Compact Message Definition

std_msgs/Header header
string child_frame_id
geometry_msgs/PoseWithCovariance pose
geometry_msgs/TwistWithCovariance twist

C.2 sensor_msgs/Imu Message

It is defined in the *sensor_msgs/Imu.msg* file.

It is used to collect data coming from an Inertial Measurement Unit (IMU). It provides information about the orientation, the angular velocity, and the linear acceleration together with their corresponding covariance matrices. See [39] for

further details.

Compact Message Definition

std_msgs/Header header
geometry_msgs/Quaternion orientation
float64[9] orientation_covariance
geometry_msgs/Vector3 angular_velocity
float64[9] angular_velocity_covariance
geometry_msgs/Vector3 linear_acceleration
float64[9] linear_acceleration_covariance

C.3 geometry_msgs/PoseWithCovariance-Stamped Message

It is defined in the *geometry_msgs/PoseWithCovarianceStamped.msg* file.

It "expresses an estimated pose with a reference coordinate frame and timestamp", as stated in [40].

Compact Message Definition

std_msgs/Header header
geometry_msgs/PoseWithCovariance pose

C.4 geometry_msgs/TwistWithCovariance-Stamped Message

It is defined in the *geometry_msgs/TwistWithCovarianceStamped.msg* file.

It "represents an estimated twist with reference coordinate frame and timestamp", as stated in [41].

Compact Message Definition

std_msgs/Header header
geometry_msgs/TwistWithCovariance twist

C.5 sensor_msgs/NavSatFix Message

It is defined in the *sensor_msgs/NavSatFix.msg* file.

It provides Navigation Satellite data specified using the WGS 84 reference ellipsoid. It contains the information regarding the satellite fix status, the estimation of latitude, longitude, and altitude, and the position covariance. See [42] for further details.

Compact Message Definition

uint8 COVARIANCE_TYPE_UNKNOWN=0
uint8 COVARIANCE_TYPE_APPROXIMATED=1
uint8 COVARIANCE_TYPE_DIAGONAL_KNOWN=2
uint8 COVARIANCE_TYPE_KNOWN=3
std_msgs/Header header
sensor_msgs/NavSatStatus status
float64 latitude
float64 longitude
float64 altitude
float64[9] position_covariance
uint8 position_covariance_type