POLITECNICO DI TORINO

Master's Degree
in Mechatronic Engineering

Master's Degree Thesis

Hybrid Localization Solutions Enabling Autonomous Navigation of Robotic Platforms

Supervisor
Prof. Michele TARAGNA
Eng. Francesco SOTTILE

Canditate
Sibasish MUKHERJEE

Academic Year 2024-2025

Abstract

Recently, autonomous navigation of mobile robotic platforms has received increasing attention from the research community. Both outdoor and indoor localization is very crucial as it enables the navigation of Autonomous Mobile Robots. Autonomous Mobile Robots can be employed for a wide range of applications including logistic operations, automated inventory management in warehouses, inspection, and monitoring tasks in industrial plants, precision agriculture, and critical infrastructure monitoring without the need for humans.

This research aims not only to provide positioning solutions for autonomous navigation but also to improve and optimize their accuracy and reliability for autonomous navigation. The typical solution for indoor positioning is Ultra-Wide Band (UWB), and for outdoor positioning is Global Navigation Satellite Systems (GNSS); these positioning solutions was integrated with IMU (also called Hybrid solution) using the help of ESKF (Error State Kalman Filter) for attitude estimation and a more accurate positioning estimation. The goal of this thesis is to design a hybrid localization algorithm that combines both UWB/GNSS and IMU data, enabling autonomous operations of robotic platform in desired environments.

The designed algorithm was tested using both simulated data and real measurement data and optimized iteratively. The indoor localization tests were conducted in the Robotic Laboratory provided by LINKS Foundation, while the outdoor localization tests took place on the sidewalks within the Politecnico di Torino area. These tests demonstrated the algorithm's capability to support autonomous navigation in dynamic environments.

Acknowledgements

First and foremost I would like to express my deepest gratitude to my thesis supervisor, Professor Michele Taragna and my co-supervisor, Eng. Francesco Sottile, for their invaluable guidance, support, and patience throughout this research. Their insightful feedback and encouragement have been instrumental in shaping this work.

I would also like to thank LINKS Foundation for providing me with the opportunity to work in their research lab, where I gained invaluable experience.

I would also like to extend my appreciation to the authors and researchers whose work has laid the foundation for my study. Their contributions to the field have been a great source of inspiration.

A special thanks to my colleagues and friends, who have supported me throughout this journey. Your discussions, encouragement, and companionship have made this experience all the more enriching.

Lastly, I am deeply grateful to my family for their unconditional love and unwavering support. Their belief in me has been my greatest motivation.

This thesis would not have been possible without all of you. Thank you.

Table of Contents

List of Tables

List of Figures

Acronyms

AMR	Autonomous Mobile Robot
ROVER	Remote Operated Vehicle for Exploration
UAV	Unmanned Aerial Vehicle
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
IMU	Inertial Measurement Unit
MEMS	Micro-Electro-Mechanical Systems
INS	Inertial Navigation System
UWB	Ultra-Wideband
TOA	Time Of Arrival
OWR	One-Way time-of-flight Ranging
TWR	Two-Way time-of-flight Ranging
EKF	Extended Kalman Filter
ESKF	Error State Kalman Filter
EPF	Extended Particle Filter
SPI	Serial Peripheral Interface
I^{2}C	Inter-Integrated Circuit
MCU	Microcontroller Unit
ENU	East-North-Up Frame
ECEF	Earth-Centered Earth-Fixed Frame
LC	Loosely Coupled
TC	Tightly Coupled
LOS	Line Of Sight
DOP	Dilution of Precision
HDOP	Horizontal Dilution of Precision

RMSE	Root Mean Square Error
STD	Standard Deviation
GT	Ground Truth
S1	Scenario 1
S2	Scenario 2

Chapter 1

Introduction

1.1 Context and Motivation

Autonomous Mobile Robots (AMRs) have become the need of the hour. In fields such as industrial manufacturing, medical science, social science, agriculture, education, and space exploration, AMRs have made impressive strides. For autonomous mobile robots, navigation is a difficult task. The robot goes through several stages, including vision, localization, cognition, and motion control, in order to navigate successfully. The robot interprets its sensors to derive useful data during the perception phase. The robot uses data from external sensors to assess its current location in the work environment during the localization phase. The robot plans the actions required to get to the destination during the cognition phase. By altering its motor outputs, the motion control phase enables the robot to follow its intended path. In this thesis, we will take a closer look at the localization phase.[5].

For outdoor localization, Global Navigation Satellite Systems (GNSS) signals (which include GPS, GLONASS, Galileo and BeiDou) are the go-to localization techniques. However, in indoor conditions, GNSS is of little to no use because of signal loss due to building materials and hence for indoor localization it is imperative to adapt to other techniques, most common being Ultra-Wideband (UWB) which is a type of radio frequency (RF) based navigation [6]. However, these old school localization methods in most cases are not extremely precise, do not have a very high refresh rate and do not provide the pose estimation of a robotic system, something that is essential for autonomous localization.

Inertial navigation systems (INS) are self-contained and, in contrast to GNSS/UWB receivers, can deliver location, velocity, and attitude estimates quickly, typically 100 to 300 times per second. INS systems depends on Newton's equations of motion, which state that if an object's initial location, velocity, and attitude are known, then all subsequent positions, velocities, and attitudes can be calculated by integrating the object's angular rates and accelerations. However, sensor biases and low frequency noise are worsened by the INS's integrative nature. As a result, unaided INS may diverge [7].

To solve this issue, the proposed solution is to integrate GNSS/UWB with MEMS IMU. In fact, it has been demonstrated that a navigation system with both high update rate and good accuracy may be achieved by fusing the INS's short-term accuracy and rapid update rate with the GNSS/UWB's long-term accuracy [8], [9]. An overview of the fusion has been described in Figure 1.1.

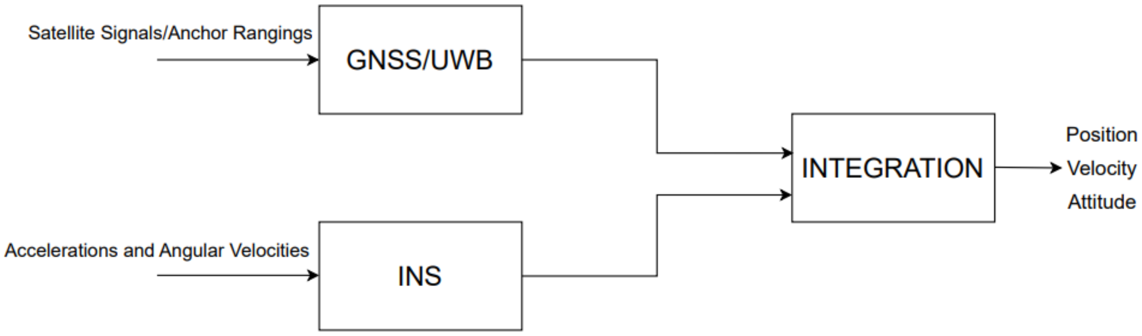


Figure 1.1: Hybrid Solution Schematics

1.2 System Design Outline

Integrating the two streams of data is however, not very trivial. The two sets of data are acquired at different frequencies and start times and so synchronization of the two clocks are needed. Once done, the two datasets needed to be transformed to a common coordinate system, which in this thesis was the East-North-Up (ENU) frame, a type of navigation frame.

To proceed with the integration we have to select between loosely coupled, tightly coupled, open or close loop. In this thesis, for indoor localization both loosely and tightly coupled solutions were studied, while for outdoor localization, only loosely coupled solutions were studied. In a loosely coupled system the GNSS/UWB operates independently from the INS, whereas in a tightly coupled system, one can estimate the UWB position directly within the state estimation process, incorporating raw range measurements into the fusion algorithm.

Finally for the state estimation, the proposed solution is to use the Error-State Kalman Filter (ESKF). ESKF is a variant of the Kalman Filter that estimates errors to the current state rather than estimating the full state directly. This type of filter was used because of the nonlinear nature of the system, and it provides better numerical stability with the large state.

1.3 Thesis Organization and Contributions

This thesis has been structured in 8 chapters.

- Chapter 2 - Introduces the coordinate frames used in this thesis and how they are analyzed.
- Chapter 3 - Describes the Kalman filters that were used while carrying out this thesis, which helped us to process the hybrid UWB/GNSS with IMU data.

- Chapter 4 - Provides us an overview of the localization techniques and what one can expect going forward in the thesis.
- Chapter 5 - Describes the IMU which was used while carrying out this thesis and how the integration between UWB/GNSS data was carried out.
- Chapter 6 - Describes the lab setup and the experimental procedure. It then presents the results and provides an analysis of the findings.
- Chapter 7 - Details the outdoor experimental setup and testing process. The chapter concludes with the presentation of results and a comprehensive evaluation of the findings.
- Chapter 8 - Presents the conclusion and further work

Chapter 2

Coordinate Systems

2.1 Introduction to Coordinate System

The measured quantities in inertial navigation systems are acquired in body frame, and then converted into other frames. A minimal inertial navigation system uses two frames at the very least. The vehicle accelerations are measured by the accelerometers in relation to the IMU body frame. A fixed rotation matrix is required to convert the accelerations from the IMU body frame into the fixed navigation frame which will be introduced later on.

On the other hand, gyros measure the vehicle angular rates in relation to the IMU body frame of reference and like in the case of accelerations, a fixed rotation matrix converts the angular rates measured by the gyros in the IMU body frame of the gyros into angular rates in the vehicle body frame. The gyro data are utilized to produce a rotation matrix, which transfers the accelerations in the platform frame into the navigation frame of interest. The navigation system processes this matrix to determine its position and velocity of the vehicle. [7].

2.2 Coordinate Frames

The various coordinate frames and associated nomenclature used in this thesis are defined in this section. Every coordinate system is a Cartesian orthogonal system.

2.2.1 Earth-Centered Earth-Fixed Frame (ECEF)

This coordinate system revolves with the earth and has its origin at its center of the earth. Figure 2.1 shows the definition of the axes directions: the x-axis points towards the point where the prime meridian and the equator cross, the z-axis points towards the mean polar axis, and the y-axis completes the right-hand coordinate.

2.2.2 Local Geodetic Frame (Navigation Frame)

In everyday life, this coordinate system is frequently referred to as the north, east, and down directions. Fitting the tangent plane to a fixed point on the geodetic reference ellipse yields the result.This point will serve as the coordinate system's origin, and the x-, y-, and z-axes all point in the direction of true north, east, and the earth's interior, respectively, to complete the right-hand coordinate system. This frame is sometimes also referred to as the ENU frame(East-North-Up frame).The figure below (Figure 2.1) provides further clarification of the previously discussed concept.

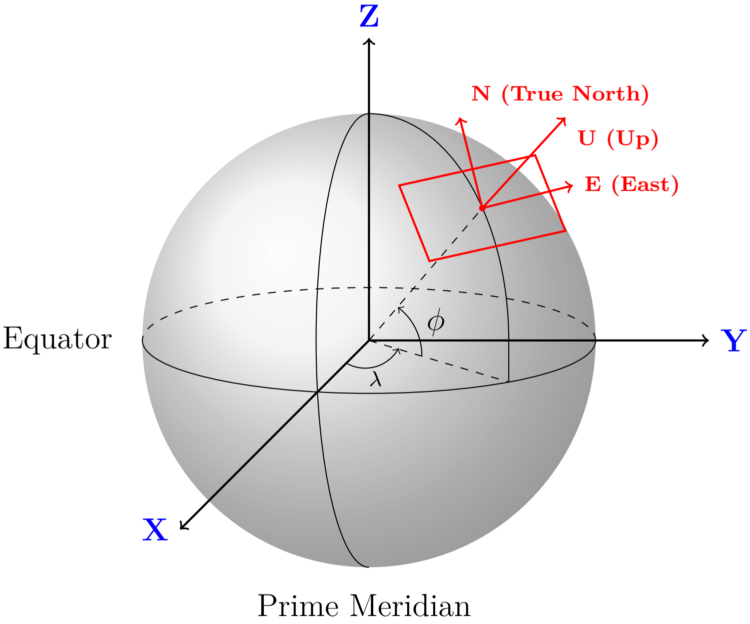


Figure 2.1: ECEF and Local Geodetic Frame

2.2.3 Vehicle Body Frame

It is the frame associated to the body moving in our robotic platform. A generic term "vehicle" is being used but in the case of indoor localisation it is an UAV/Turtlebot (Figure 2.2), in the case of outdoor localisation it is an AgileX rover. The x-axis points forward, the z-axis descends through the vehicle, and the y-axis completes the right-hand coordinate system. The coordinate system originates at the vehicle's centre of gravity.

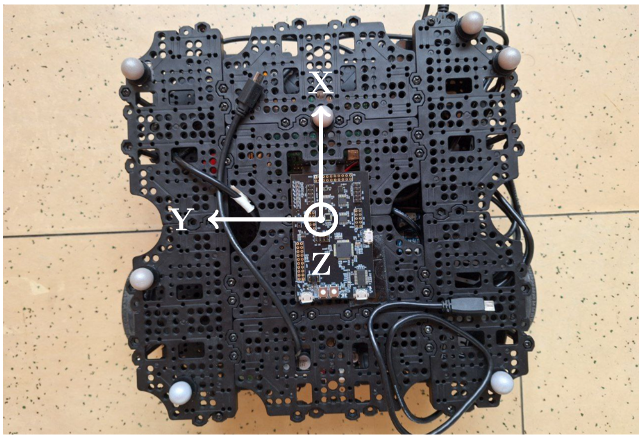


Figure 2.2: Vehicle body frame of a TurtleBot

2.2.4 IMU Body Frame

It is the frame associated to the IMU sensor used during the course of the thesis. In this thesis,the coordinate axes of the IMU body frame is perfectly aligned with the coordinate axes of the Vehicle body frame. In cases where they cannot be perfectly aligned, then we would need an additional rotation matrix.

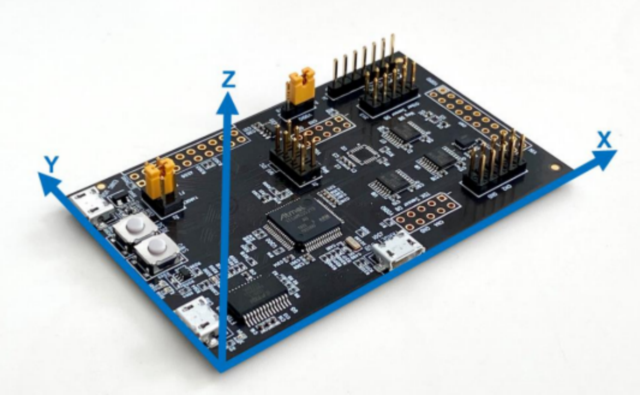


Figure 2.3: IMU body frame

2.3 Coordinate Frame Transformation

Converting a vector from one coordinate system to another is frequently required in navigation systems. In our analysis, this is essential for transitioning between the body frame and the navigation frame. Two distinct approaches to obtaining a mathematical equation for the rotation matrix connecting two orthogonal, Cartesian coordinate systems are discussed in this section.

2.3.1 Projection

In the first technique, a vector is projected onto the orthonormal bases of the two coordinate systems in order to find a mathematical formula for the rotation matrix related to coordinate systems. As shown in Figure 4.2, let $\{\mathbf{i}_a, \mathbf{j}_a, \mathbf{k}_a\}$ and $\{\mathbf{i}_b, \mathbf{j}_b, \mathbf{k}_b\}$ be the unit vectors spanning two orthonormal coordinate systems with the same origin. The unit vectors spanning coordinate system A can therefore be used to express vector P, that is:

$$P = x_a\mathbf{i}_a + y_a\mathbf{j}_a + z_a\mathbf{k}_a \tag{2.1}$$

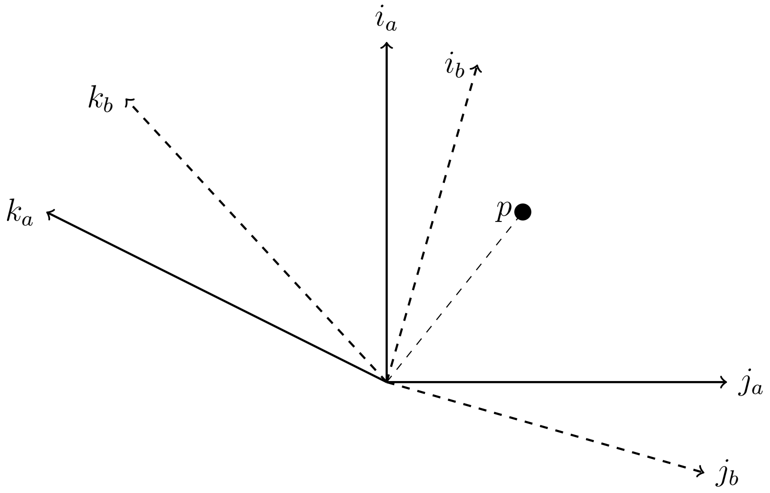


Figure 2.4: The directional cosine matrix connecting the two orthonormal coordinate systems can be found by projecting the vector p onto their bases.

Since the coordinate systems are orthogonal, the vector P can be projected onto coordinate system B by taking the dot product of P with the unit vectors of the B-frame.

$$\begin{aligned} x_b &= \langle \mathbf{i}_b, \mathbf{P} \rangle = \langle \mathbf{i}_a, \mathbf{i}_b \rangle x_a + \langle \mathbf{j}_a, \mathbf{i}_b \rangle y_a + \langle \mathbf{k}_a, \mathbf{i}_b \rangle z_a \\ y_b &= \langle \mathbf{j}_b, \mathbf{P} \rangle = \langle \mathbf{i}_a, \mathbf{j}_b \rangle x_a + \langle \mathbf{j}_a, \mathbf{j}_b \rangle y_a + \langle \mathbf{k}_a, \mathbf{j}_b \rangle z_a \\ z_b &= \langle \mathbf{k}_b, \mathbf{P} \rangle = \langle \mathbf{i}_a, \mathbf{k}_b \rangle x_a + \langle \mathbf{j}_a, \mathbf{k}_b \rangle y_a + \langle \mathbf{k}_a, \mathbf{k}_b \rangle z_a \end{aligned}$$

The cosine of the angle between vectors $\mathbf{i_k}$ and $\mathbf{j_l}$, denoted as $\cos(\theta_{ik,jl})$, is the scalar product $\langle \mathbf{i_k}, \mathbf{j_l} \rangle$. The angle $\theta_{ik,jl}$ between the two vectors is expressed in this form. This relationship can be written in matrix form as:

$$\mathbf{P}_b = \mathbf{R}_a^b \mathbf{P}_a \tag{2.2}$$

Where

$$\mathbf{R}_a^b = \begin{bmatrix} \langle \mathbf{i_a}, \mathbf{i_b} \rangle & \langle \mathbf{j_a}, \mathbf{i_b} \rangle & \langle \mathbf{k_a}, \mathbf{i_b} \rangle \\ \langle \mathbf{i_a}, \mathbf{j_b} \rangle & \langle \mathbf{j_a}, \mathbf{j_b} \rangle & \langle \mathbf{k_a}, \mathbf{j_b} \rangle \\ \langle \mathbf{i_a}, \mathbf{k_b} \rangle & \langle \mathbf{j_a}, \mathbf{k_b} \rangle & \langle \mathbf{k_a}, \mathbf{k_b} \rangle \end{bmatrix} \tag{2.3}$$

Or

$$\mathbf{R}_a^b = \begin{bmatrix} \cos(\theta_{ia,ib}) & \cos(\theta_{ja,ib}) & \cos(\theta_{ka,ib}) \\ \cos(\theta_{ia,jb}) & \cos(\theta_{ja,jb}) & \cos(\theta_{ka,jb}) \\ \cos(\theta_{ia,kb}) & \cos(\theta_{ja,kb}) & \cos(\theta_{ka,kb}) \end{bmatrix} \tag{2.4}$$

The rotation matrix $\mathbf{R}_a^b$ is a so-called directional cosine matrix. Although $\mathbf{R}_a^b$ has nine elements, it possesses only three degrees of freedom and can be uniquely described by three Euler angles, which are collected in the vector $\boldsymbol{\theta}$.

The directional cosine matrix is an orthonormal matrix, meaning that:

$$\mathbf{R}_a^b (\mathbf{R}_a^{b)^T} = (\mathbf{R}_a^{b)^T} \mathbf{R}_a^b = \mathbf{I},$$

where **I** is the identity matrix. Hence, the inverse of the rotation matrix is the same as its transpose [7].

2.3.2 Plane Rotation

Plane projection is a rather convenient method for constructing the rotation matrix. When two coordinate frames share the same origin, their transformation can be expressed using plane rotations, where one frame is rotated around a vector **v**. The overall rotation matrix between two frames connected by multiple plane rotations can be found by multiplying the individual rotation matrices in a certain order. The order depends on wether the rotations are about the current frame or a fixed frame.[7].

For the following example, let us assume three successive rotations:

1. A rotation of angle $\boldsymbol{\gamma_1}$ around the z-axis of Coordinate System A.
2. Then a rotation of angle $\boldsymbol{\gamma_2}$ around the y-axis of the **current frame**.
3. A final rotation of angle $\boldsymbol{\gamma_3}$ around the x-axis of the **current frame**.

In Coordinate System A, we begin with an arbitrary point $\{x_a, y_a, z_a\}$. The first rotation, denoted by $\boldsymbol{\gamma_1}$, is performed around the z-axis of Coordinate System A.

The rotation matrix for the first rotation is given by:

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \cos(\gamma_1) & -\sin(\gamma_1) & 0 \\ \sin(\gamma_1) & \cos(\gamma_1) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_a \\ y_a \\ z_a \end{bmatrix} \tag{2.5}$$

The coordinates $\{x', y', z'\}$ represent the mapping of the coordinates $\{x_a, y_a, z_a\}$ in the intermediate frame. This is further illustrated in Figure 2.5a. Following this, another set of rotations, $\boldsymbol{\gamma_2}$, is applied around the y-axis of the current frame. This set of rotation is shown in Figure 2.5b.

$$\begin{bmatrix} x'' \\ y'' \\ z'' \end{bmatrix} = \begin{bmatrix} \cos(\gamma_2) & 0 & -\sin(\gamma_2) \\ 0 & 1 & 0 \\ \sin(\gamma_2) & 0 & \cos(\gamma_2) \end{bmatrix} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} \tag{2.6}$$

Finally, a third and last rotation, denoted by $\boldsymbol{\gamma_3}$, is performed around the current x-axis (Figure 2.5c). This rotation completes the transformation, resulting in the mapping of the point $\{x_a, y_a, z_a\}$ from Frame A to Frame B, giving the new coordinates $\{x_b, y_b, z_b\}$.

$$\begin{bmatrix} x_b \\ y_b \\ z_c \end{bmatrix} = \begin{bmatrix} x''' \\ y''' \\ z''' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\gamma_3) & -\sin(\gamma_3) \\ 0 & \sin(\gamma_3) & \cos(\gamma_3) \end{bmatrix} \begin{bmatrix} x'' \\ y'' \\ z'' \end{bmatrix} \tag{2.7}$$

Multiplying the rotation matrices in equations 2.5, 2.6 and 2.7 in a certain order of rotations yields the rotation matrix from coordinate system A to coordinate system B. The rotation matrix $\mathbf{R}_a^b$ is found as follows.

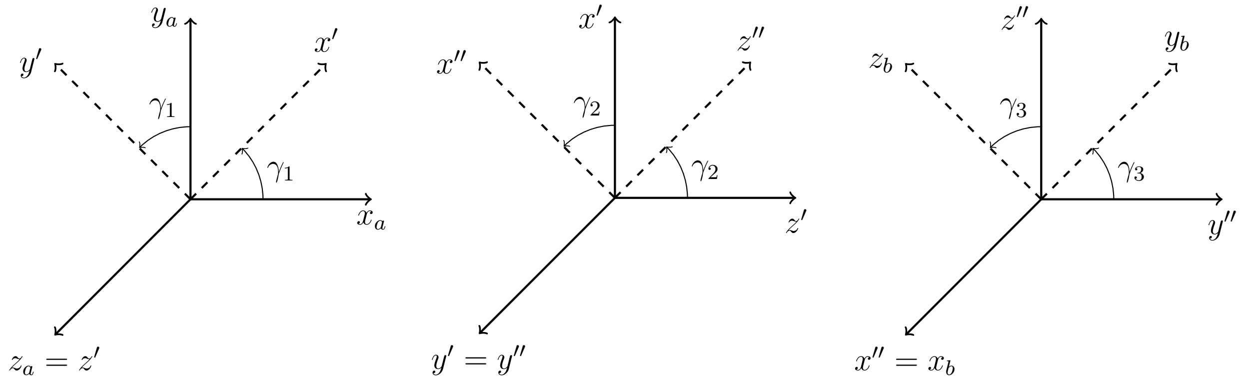
$$\mathbf{R}_a^b = \mathbf{R}_x(\gamma_3) \cdot \mathbf{R}_y(\gamma_2) \cdot \mathbf{R}_z(\gamma_1) \tag{2.8}$$

Multiplying all the rotation matrices and finding the transpose, we get:

$$\mathbf{R}_a^b = \begin{bmatrix} \cos\gamma_2\cos\gamma_1 & \cos\gamma_2\sin\gamma_1 & -\sin\gamma_2 \\ \sin\gamma_3\sin\gamma_2\cos\gamma_1 - \cos\gamma_3\sin\gamma_1 & \sin\gamma_3\sin\gamma_2\sin\gamma_1 + \cos\gamma_3\cos\gamma_1 & \sin\gamma_3\cos\gamma_2 \\ \cos\gamma_3\sin\gamma_2\cos\gamma_1 + \sin\gamma_3\sin\gamma_1 & \cos\gamma_3\sin\gamma_2\sin\gamma_1 - \sin\gamma_3\cos\gamma_1 & \cos\gamma_3\cos\gamma_2 \end{bmatrix} \tag{2.9}$$

It is important to note that if the subsequent rotations are based on a fixed frame A then the order of multiplying with rotation matrices will reverse. In that case, the rotation matrix, $\mathbf{R}_a^b$ will be as follows:

$$\mathbf{R}_a^b = \mathbf{R}_z(\gamma_1) \cdot \mathbf{R}_y(\gamma_2) \cdot \mathbf{R}_x(\gamma_3) \tag{2.10}$$



(a) Rotation around z. (b) Rotation around y'. (c) Rotation around x''.

Figure 2.5: Successive rotations around coordinate axes.

Chapter 3

Kalman Filters

3.1 Introduction to Kalman Filters

Rudolf E. Kálmán developed the state estimation method known as Kalman filtering in 1960. It is utilised in numerous application domains, such as spacecraft navigation, robotic motion planning, signal processing, and wireless sensor networks, because of its ability to extract valuable information from noisy data and its low computing and memory needs. The typical use of Kalman filtering is focused on robotic applications and state estimation. Kalman filters are designed assuming that the noise is Gaussian, which sometimes creates a common misconception that Kalman filters can only be used with Gaussian noise[10].

In our thesis, two special types of Kalman filters will be used- Extended Kalman Filter (EKF) and Error State Kalman Filter (ESKF).

3.2 Extended Kalman Filter

Kalman filters, are perfect when we are dealing with systems which behave linearly and are normally distributed. Unfortunately for us, these conditions rarely stand true when we try to deal with real life situations, in which linearization is necessary. In these cases, we must adapt to Extended Kalman Filter (EKF). In this thesis, EKF will be exclusively used for UWB based position estimation.

The system is modeled by the following non-linear equation (3.1) which is observed by the measurement model (3.2):

$$\boldsymbol{x}_k = f(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}) + \boldsymbol{w}_k \tag{3.1}$$

$$\boldsymbol{z}_k = h(\boldsymbol{x}_k) + \boldsymbol{v}_k \tag{3.2}$$

The process starts by executing what is known as the Time Update, which propagates the state estimate $\boldsymbol{x}$ and its covariance matrix $\boldsymbol{P}$ over time.

$$\boldsymbol{x}_k^- = f\left(\hat{\boldsymbol{x}}_{k-1}^+, \hat{\boldsymbol{u}}_{k-1}\right) + \boldsymbol{q}_k \tag{3.3}$$

$$\boldsymbol{P}_k^- = \boldsymbol{F}_k \boldsymbol{P}_{k-1}^+ \boldsymbol{F}_k^T + \boldsymbol{Q}_k \tag{3.4}$$

where:

- $\boldsymbol{q}_k$: The mean vector of the system state noise vector $\boldsymbol{w}_k$, which has a Gaussian distribution.
- $\boldsymbol{Q}_k$: The covariance matrix of the system state noise vector $\boldsymbol{w}_k$.
- $\boldsymbol{F}_k$: Based on the prior a posteriori state estimate $\hat{\boldsymbol{x}}_{k-1}^{+}$, the Jacobian of the system transition function f.

Following the discovery of these a priori values, the algorithm moves on to the "Measurement Update" adjusting the estimations and determining a weight factor for the measures based on their covariance matrix and the estimated state:

$$\boldsymbol{S}_k = \boldsymbol{H}_k \hat{\boldsymbol{P}}_k^{-} \boldsymbol{H}_k^{T} + \boldsymbol{R}_k \tag{3.5}$$

$$\boldsymbol{K}_k = \hat{\boldsymbol{P}}_k^{-} \boldsymbol{H}_k^{T} \boldsymbol{S}_k^{-1} \tag{3.6}$$

$$\boldsymbol{s}_k = \boldsymbol{z}_k - h\left(\hat{\boldsymbol{x}}_k^{-}\right) \tag{3.7}$$

$$\hat{\boldsymbol{x}}_k^{+} = \hat{\boldsymbol{x}}_k^{-} + \boldsymbol{K}_k \boldsymbol{s}_k \tag{3.8}$$

$$\hat{\boldsymbol{P}}_k^{+} = \left(\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k\right) \hat{\boldsymbol{P}}_k^{-} \tag{3.9}$$

where:

- Variables with $(\hat{\cdot})$ are estimates, $(^{-})$ are prediction and $(^{+})$ are correction.
- S_k is the covariance matrix of the innovation vector.
- R_k is the covariance matrix of the measurement noise vector v_k.
- K_k is the Kalman filter gain.
- s_k is the innovation vector.
- H_k is the Jacobian of the observation function h, computed around the a priori state estimate $\hat{x}_k^{-}$.

The performance of the KF depends on a few factors such as how the system is modeled and how well the initial parameters are chosen. In the following section, we will explore what are state models and measurement models, as well as the specific model selected for this study [11].

3.2.1 State Model

The state model is used to describe the system dynamics. There are a few type of models that could be used in our scenario which are: P model (Position), PV model (Position-Velocity) and PVA model (Position-Velocity-Acceleration). For this study, only P and PV model were studied. The models that are displayed here are in three dimensions (n=3) [11].

P model

This model can be thought of as a static version of the KF; that is, it only updates the measurements. The model usually serves two purposes: tracking where due to low sampling frequency, a dynamic model will not work well and estimating certain measurement parameters while knowing the precise location. In case of tracking (our case), the matrix $\boldsymbol{Q_k}$ which is one of the parameters greatly affects the KF, where lower values allow us to have a smooth output, but in turn we get a high settling time. For this reason, a variable $\boldsymbol{Q_k}$ is given as a function of the time interval between two measurements Δt_k and the standard deviation σ_a of a normally distributed acceleration vector. The state is modeled as follows:

$$\boldsymbol{x_k} = \begin{bmatrix} x & y & z \end{bmatrix}^T \tag{3.10}$$

$$\boldsymbol{x_k} = f(\boldsymbol{x_{k-1}}, 0) = I_{n\times n}\boldsymbol{x_{k-1}} \tag{3.11}$$

$$Q_k = [\Delta t_k I_{n\times n}][\Delta t_k I_{n\times n}]^T \sigma_a^2 \tag{3.12}$$

PV model

It is a dynamic KF that makes the assumption that the velocity is almost constant between the Δt_k estimate intervals. Once more, process noise is crucial, and if we take into account minimal gaussianly-distributed acceleration noise, we will get a smooth tracking. Slow reaction or even divergence will be observed on non-linear manoeuvres (where velocity is no longer constant); yet, some of these manoeuvres can be followed with noisier tracking if process noise is set well. The expression for it's state vector is:

$$\boldsymbol{x_k} = \begin{bmatrix} x & y & z & v_x & v_y & v_z \end{bmatrix}^T \tag{3.13}$$

$$\boldsymbol{x_k} = f(\boldsymbol{x_{k-1}}, 0) = \begin{bmatrix} I_{n\times n} & \Delta t_k I_{n\times n} \\ 0_{n\times n} & I_{n\times n} \end{bmatrix} \boldsymbol{x_{k-1}} \tag{3.14}$$

$$Q_k = \begin{bmatrix} \frac{1}{2}\Delta t_k^2 I_{n\times n} \\ \Delta t_k I_{n\times n} \end{bmatrix} \begin{bmatrix} \frac{1}{2}\Delta t_k^2 I_{n\times n} \\ \Delta t_k I_{n\times n} \end{bmatrix}^T \sigma_a^2 \tag{3.15}$$

3.2.2 Measurement Model

The measurement model, which follows the KF design, explains all of the variables and phenomena associated to the measurements as well as their relationship to the state vector. The approach or techniques employed as constraints are taken into consideration while formulating the measurement equation. Time of arrival, time difference of arrival, angle of arrival, and received signal strength (more about them in section 4.1.1) are some of the most often employed for position tracking. The majority of them can be transformed into approximate locations or distances. In this thesis, only distance measurements measurement model was studied as we worked with time of arrival.

Distance Measurements

Tracking systems typically rely on some form of distance estimation. These measurements can be represented as a vector $\mathbf{z}_k = [z_1, z_2, \ldots, z_N]$, where each z_i corresponds to an individual measurement. Although these estimates may have uncertainties, they generally follow a Gaussian distribution, making the EKF a suitable choice for state estimation.

$$z_i = d_i + n_i, \quad n_i \sim \mathcal{N}(0, \sigma^2) \tag{3.16}$$

$$d_{\mathrm{i},k} = \sqrt{(x_k - x_{\mathrm{i},k})^2 + (y_k - y_{\mathrm{i},k})^2 + (z_k - z_{\mathrm{i},k})^2} \tag{3.17}$$

$$h(x_k) = \begin{bmatrix} \sqrt{(x_k - x_{\mathrm{ref1},k})^2 + (y_k - y_{\mathrm{ref1},k})^2 + (z_k - z_{\mathrm{ref1},k})^2} \\ \sqrt{(x_k - x_{\mathrm{ref2},k})^2 + (y_k - y_{\mathrm{ref2},k})^2 + (z_k - z_{\mathrm{ref2},k})^2} \\ \vdots \end{bmatrix} \tag{3.18}$$

$$H_k = \left.\frac{\partial h}{\partial x}\right|_{x=x_k} = \begin{bmatrix} \frac{x_k - x_{\mathrm{ref1},k}}{d_{\mathrm{ref1},k}} & \frac{y_k - y_{\mathrm{ref1},k}}{d_{\mathrm{ref1},k}} & \frac{z_k - z_{\mathrm{ref1},k}}{d_{\mathrm{ref1},k}} & 0 & 0 & \cdots \\ \frac{x_k - x_{\mathrm{ref2},k}}{d_{\mathrm{ref2},k}} & \frac{y_k - y_{\mathrm{ref2},k}}{d_{\mathrm{ref2},k}} & \frac{z_k - z_{\mathrm{ref2},k}}{d_{\mathrm{ref2},k}} & 0 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \tag{3.19}$$

$$R_k = \mathrm{diag}\left(\begin{bmatrix} \sigma^2_{d_{\mathrm{ref1},k}} & \sigma^2_{d_{\mathrm{ref2},k}} & \cdots \end{bmatrix}\right) \tag{3.20}$$

where, Hk is linearized measurement model (3.2), whose size depends on the model being used. Rk is the measurement noise covariance matrix. In order to adjust the a priori estimate, all of the distance measurements that are available at each step must be used. Sometimes, if the a priori estimate is good, no significant improvement is noticed [11].

3.3 Error State Kalman Filter

Error State Kalman Filter (ESKF), also known as the Error State Extended Kalman Filter (ES-EKF). The main difference between EKF and ESKF is that in ESKF the error are modeled separately from the state. The non-linear system dynamic equation is:

$$\boldsymbol{z}_{\mathrm{k+1}} = c(\boldsymbol{z}_k, \boldsymbol{u}_k) + \boldsymbol{w}_k \tag{3.21}$$

The term $\boldsymbol{z}_k$ is the state, which can be further broken down into:

$$\boldsymbol{z_k} = \begin{bmatrix} x_k & v_k & \theta_k \end{bmatrix} \tag{3.22}$$

where x_k represents the position, v_k represents the velocities and θ_k are the euler angles. The $\boldsymbol{u}_k$ represents the input of the system which are the accelerations and angular velocities:

$$\boldsymbol{u_k} = \begin{bmatrix} a_k & \omega_k \end{bmatrix} \tag{3.23}$$

The vector $\boldsymbol{w}_k$ is the process noise which are the external factors such as disturbances and unmodeled forces etc. The main concept of this filter is separating the state vector $(\boldsymbol{z}_k)$ and the input vector $(\boldsymbol{u}_k)$ into two parts as follows:

$$\boldsymbol{z}_k = \boldsymbol{z}_k^{\mathrm{n}} + \boldsymbol{z}_k^{\mathrm{err}} \tag{3.24}$$

$$\boldsymbol{u}_k = \boldsymbol{u}_k^{\mathrm{n}} + \boldsymbol{u}_k^{\mathrm{err}} \tag{3.25}$$

where $\boldsymbol{z}_t^{\mathrm{n}}$ is the nominal state and $\boldsymbol{z}_t^{\mathrm{err}}$ is the error state, $\boldsymbol{u}_t^{\mathrm{n}}$ is the nominal input $\boldsymbol{u}_t^{\mathrm{err}}$ is the accelerometer and gyroscope bias. Assuming small $\boldsymbol{z}_t^{\mathrm{err}}$ and $\boldsymbol{u}_t^{\mathrm{err}}$ and applying taylor expansion to $c(\boldsymbol{z}_k, \boldsymbol{u}_k)$ leads us to:

$$\boldsymbol{z}_{k+1}^{\mathrm{n}} + \delta \boldsymbol{z}_{k+1}^{\mathrm{err}} \approx c(\boldsymbol{z}_k^{\mathrm{n}}, \boldsymbol{u}_k^{\mathrm{n}}) + \boldsymbol{C}_{1,k}\delta \boldsymbol{z}_k^{\mathrm{err}} + \boldsymbol{C}_{2,k}\delta \boldsymbol{u}_k^{\mathrm{err}} + \boldsymbol{w}_k' \tag{3.26}$$

where $\boldsymbol{C}_{1,k}$ and $\boldsymbol{C}_{2,k}$ are defined as:

$$\boldsymbol{C}_{1,k} = \left.\frac{\partial c(\boldsymbol{z}, \boldsymbol{u})}{\partial \boldsymbol{z}}\right|_{\boldsymbol{z}=\boldsymbol{z}_k^{\mathrm{n}}}, \quad \boldsymbol{C}_{2,k} = \left.\frac{\partial c(\boldsymbol{z}, \boldsymbol{u})}{\partial \boldsymbol{u}}\right|_{\boldsymbol{u}=\boldsymbol{u}_k^{\mathrm{n}}} \tag{3.27}$$

Let's assume we have a perfect prediction (no noise), the nominal state will be:

$$\boldsymbol{z}_{k+1}^{\mathrm{n}} = c(\boldsymbol{z}_k^{\mathrm{n}}, \boldsymbol{u}_k^{\mathrm{n}}) \tag{3.28}$$

Then the linear model for the **error** becomes:

$$\delta \boldsymbol{z}_{k+1}^{\mathrm{err}} = \boldsymbol{C}_{1,k}\delta \boldsymbol{z}_k^{\mathrm{err}} + \boldsymbol{C}_{2,k}\delta \boldsymbol{u}_k^{\mathrm{err}} + \boldsymbol{w}_k' \tag{3.29}$$

Observing that $\delta x_k = [\delta z_k, \delta u_k]$ it is evident that $C_{1,\mathrm{k}}$ and $C_{2,\mathrm{k}}$ corresponds to the top part of the ψ_k matrix, which is also known as the transition matrix. The ψ_k matrix is a 15 by 15 matrix, where the bottom 6 x 6 stores the IMU bias values. Since the model (3.29) is linear, we can apply the standard Kalman filter:

$$\begin{bmatrix} \delta \hat{\boldsymbol{z}}_{k+1}^{-} \\ \delta \hat{\boldsymbol{u}}_{k+1}^{-} \end{bmatrix} = \boldsymbol{\Psi}_k \begin{bmatrix} \delta \hat{\boldsymbol{z}}_k \\ \delta \hat{\boldsymbol{u}}_k \end{bmatrix} \tag{3.30}$$

$$\begin{bmatrix} \delta \hat{\boldsymbol{z}}_k \\ \delta \hat{\boldsymbol{u}}_k \end{bmatrix} = \begin{bmatrix} \delta \hat{\boldsymbol{z}}_k^{-} \\ \delta \hat{\boldsymbol{u}}_k^{-} \end{bmatrix} + \boldsymbol{K}_{f,k}\left(\boldsymbol{y}_k - \boldsymbol{H}_k \begin{bmatrix} \boldsymbol{z}_k^{\mathrm{n}} \\ \boldsymbol{u}_k^{\mathrm{n}} \end{bmatrix} - \boldsymbol{H}_k \begin{bmatrix} \delta \hat{\boldsymbol{z}}_k^{-} \\ \delta \hat{\boldsymbol{u}}_k^{-} \end{bmatrix}\right) \tag{3.31}$$

$$\boldsymbol{K}_{f,k} = \boldsymbol{P}_k^{-}\boldsymbol{H}_k^{T}\left(\boldsymbol{H}_k\boldsymbol{P}_k^{-}\boldsymbol{H}_k^{T} + \boldsymbol{R}_{d,k}\right)^{-1} \tag{3.32}$$

$$\boldsymbol{P}_k = \boldsymbol{P}_k^{-} - \boldsymbol{K}_{f,k}\boldsymbol{H}_k\boldsymbol{P}_k^{-} \tag{3.33}$$

$$\boldsymbol{P}_{k+1}^{-} = \boldsymbol{\Psi}_k\boldsymbol{P}_k\boldsymbol{\Psi}_k^{T} + \boldsymbol{Q}_{d,k} \tag{3.34}$$

Where:

- Variables with $(^-)$ represents the prediction values, so for example $\delta\hat{\boldsymbol{z}}_{k+1}^-$ represents the predicted error state at time k+1 and $\delta\hat{\boldsymbol{z}}_k$ represents the estimated error state at time k.
- $\boldsymbol{y_k}$ is the measurement vector
- $\boldsymbol{K}_{f,k}$ represents the Kalman filter gain.
- $\boldsymbol{P}_k^-$: Predicted error covariance matrix.
- $\boldsymbol{P}_k$: Updated error covariance matrix.
- $\boldsymbol{H}_k$: Measurement matrix.
- $\boldsymbol{R}_{d,k}$: Measurement noise covariance matrix.

In order to create an EKF, z nom is now added to both sides of equation (3.30) and replaced with the current estimate in all equations. The time and filter update for the estimates are provided below:

$$\hat{\boldsymbol{z}}_{k+1}^- = c(\hat{\boldsymbol{z}}_k, \hat{\boldsymbol{u}}_k) \tag{3.35}$$

$$\delta\hat{\boldsymbol{u}}_{k+1}^- = [\boldsymbol{\Psi}_k]_{10:15,10:15}\delta\hat{\boldsymbol{u}}_k \tag{3.36}$$

$$\begin{bmatrix} \delta\hat{\boldsymbol{z}}_k \\ \delta\hat{\boldsymbol{u}}_k \end{bmatrix} = \begin{bmatrix} \delta\hat{\boldsymbol{z}}_k^- \\ \delta\hat{\boldsymbol{u}}_k^- \end{bmatrix} + \boldsymbol{K}_{f,k}\left(\boldsymbol{y}_k - \boldsymbol{H}_k \begin{bmatrix} \hat{\boldsymbol{z}}_k^- \\ \hat{\boldsymbol{u}}_{\boldsymbol{k}}^- \end{bmatrix}\right) \tag{3.37}$$

Further analyzing (3.37), we see that $\hat{\boldsymbol{u}}_{\boldsymbol{k}}^-$ term (the IMU biases) is $\mathbf{0}_{6\times1}$ because the term $\boldsymbol{y_k}$ only contains information about the position. Similarly, $\delta\hat{\boldsymbol{z}}_k^-$ term is $\mathbf{0}_{9\times1}$ because this is the term is fed back to update the state, so this must be reset to zero at every update step [7]. The final equation then is:

$$\begin{bmatrix} \delta\hat{\boldsymbol{z}}_k \\ \delta\hat{\boldsymbol{u}}_k \end{bmatrix} = \begin{bmatrix} \mathbf{0}_{9\times1} \\ \delta\hat{\boldsymbol{u}}_k^- \end{bmatrix} + \boldsymbol{K}_{f,k}\left(\boldsymbol{y}_k - \boldsymbol{H}_k \begin{bmatrix} \hat{\boldsymbol{z}}_k^- \\ \mathbf{0}_{6\times1} \end{bmatrix}\right) \tag{3.38}$$

Now that we have defined our ESKF, we will explore how it was utilized to achieve sensor fusion, integrating multiple measurement sources as detailed in section 5.3.

Chapter 4

Overview of Localization Methods

4.1 Indoor Localization Techniques

Modern location-based (GNSS) service technologies are ideal for outdoor settings where there is wide space for satellite signals to easily reach the devices. Unfortunately for Indoor localization these services are of little to no use for us [12]. In an indoor environment, a GNSS device's power is significantly reduced because of signal attenuation caused by different building materials [6]. There are a few different ways in which we can have indoor localization such as Magnetic Based Navigation, Sound-Based Technologies, Radio Frequency (RF) Based Navigation and many more [13]. In this thesis we are more interested in RF Based Navigation and particularly Ultra-Wide Band (UWB) and the hybrid performance of UWB with INS.

4.1.1 Ultra-Wide Band (UWB)

In recent years, UWB technology has emerged as a radio communication technology that uses communication between the base station (also called anchors) and known position data to determine a tag's location based on the geometric relationship between the tag and the base station. In comparison to other technologies, UWB technology has the following advantages: good robustness, strong resistance to interference, high resistance to multi-path effect, and accurate positioning. However, compared to other positioning systems, UWB indoor positioning systems are more complicated and consequently more expensive to install [14].

To estimate the distance between the anchor and the tag, it is required to build a series of equations, based on the selected positioning model. There are different types of positioning models, most common being TOA, TDOA, RSSI and AOA [14].

TOA

Time of Arrival (TOA) is the most common positioning model in ranging, which uses which calculates the distance between the anchor and the tag using the wireless signal's time of flight. The TOA is further divided into two types: One-Way time-of-flight Arrival (OWR) and Two-Way time-of-flight Arrival (TWR).

One-Way time-of-flight Arrival works as shown in Figure 4.1. The clocks of the anchors and the tag must be synchronized. An anchor in the vicinity receives a signal that a tag transmits to its region at a specific moment, therefore the distance between the tag and the anchor is determined by:

$$\text{distance} = c \times (T_1 - T_0) \tag{4.1}$$

where c is the speed of light, T_1 is when the anchor receives the signal and T_0 is when the tag sends the signal.

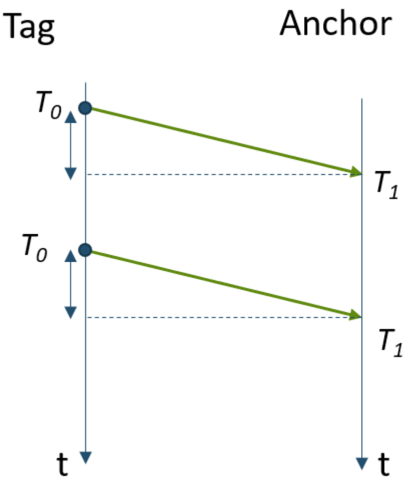


Figure 4.1: Operating Principle of OWR

Two-Way time-of-flight Arrival on the other hand is a ranging method based on the fact that the tag and the anchor communicates with each other. The anchor sends a signal at time T_0 and it arrives to the tag at T_a. The tag responds at T_b, which is intercepted by the anchor at T_1. The time to reply T_{reply} is:

$$T_{reply} = T_b - T_a \tag{4.2}$$

Then the distance is:

$$\text{distance} = \frac{c \times (T_1 - T_0 - T_{\text{reply}})}{2} \tag{4.3}$$

Where c is the speed of light. The divided by 2 term comes from the fact that its a round-trip time. This is further explained figuratively in Figure 4.2.

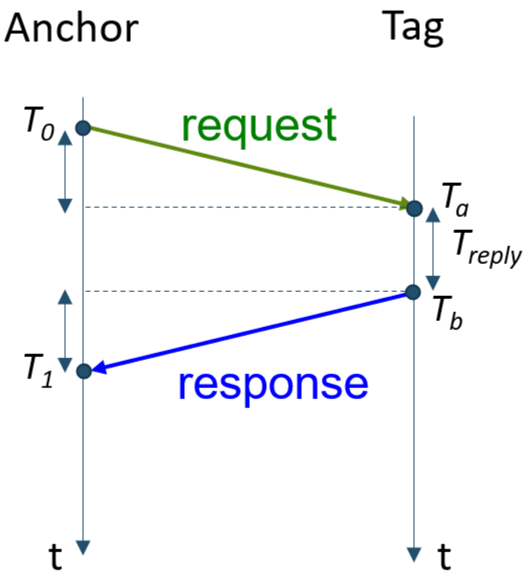


Figure 4.2: Operating Principle of TWR

Syncing the base station and tag clocks and determining the number of tags are two issues that are resolved by the TWR approach. This restricts how many tags can communicate simultaneously because the base station can only communicate with one tag at a time [14].

For 3D localisation, atleast 4 non coplanar anchor nodes are required [15]. If our anchor nodes have the following coordinates:

$$\begin{cases} N_1(x_1, y_1, z_1) \\ N_2(x_2, y_2, z_2) \\ N_3(x_3, y_3, z_3) \\ N_4(x_4, y_4, z_4) \end{cases} \tag{4.4}$$

The tag is located d_1, d_2, d_3, d_4 from the four anchor nodes. If the tag's three-dimensional coordinates are (x_0, y_0, z_0), we can obtain:

$$\begin{cases} (d_1)^2 = (x_1 - x_0)^2 + (y_1 - y_0)^2 + (z_1 - z_0)^2 \\ (d_2)^2 = (x_2 - x_0)^2 + (y_2 - y_0)^2 + (z_2 - z_0)^2 \\ (d_3)^2 = (x_3 - x_0)^2 + (y_3 - y_0)^2 + (z_3 - z_0)^2 \\ (d_4)^2 = (x_4 - x_0)^2 + (y_4 - y_0)^2 + (z_4 - z_0)^2 \end{cases} \tag{4.5}$$

The ranging data d_1, d_2, d_3, d_4 are provided by the UWB system. From there using the set of formulas (4.5), we can create a set of equations to find (x_0, y_0, z_0) [14].

TDOA

Time Difference Of Arrival (TDOA), like the time-of-arrival-based technique TOA, employs the airborne signal propagation time to accomplish ranging. TDOA must measure the distance difference between several anchors and the tag in order to determine the tag position, as opposed to measuring the distance between a single anchor and the tag like TOA. Only the tag must transmit data to the base station in order for the TDOA approach to work; neither signal synchronisation nor communication between the tag and the base station are necessary [14].

RSSI

The Received Signal Strength Indicator (RSSI) method is based on the fact that a signal becomes weaker with distance. Consequently, the power received by the tag and the signal's distance may be calculated using the following equation:

$$P_b(S) = \frac{P_i G_j G_{bz} \lambda^{\tau}}{(4\pi S)^{\tau}} \tag{4.6}$$

where λ is the signal wavelength, τ is the regional environmental path loss factor, G_j is the gain when the base station transmits the signal, G_{bz} is the signal gain when the tag receives the signal, P_b is the frequency of the signal received by the tag, P_i is the transmit power of the signal transmitted by the base station, and S is the distance transmitted.

Since the RSSI method is independent of time, it also does not require a high degree of synchronisation between the anchor and tag clocks. However, the environment has a significant impact on the signal's loss factor during propagation, and even small changes in the environment can have a significant impact on the results. As a result, the data error and positioning error are both larger [14].

AOA

The angle at which an incoming signal reaches a receiver in relation to a reference direction is known as the Angle of Arrival (AoA), and it is usually measured with an antenna or array of antennas. Techniques like phase difference, time difference, or amplitude fluctuation across many antennas are used to estimate AoA. The AoA approach is simple to use and also mitigates the issue of miss synchronized tag and base station clocks, but it comes at a comparatively high cost and necessitates the arrangement of several antenna matrices [14].

4.1.2 Hybrid UWB-IMU Localization

UWB has its own share of problems, which are multipath effects where UWB signals are reflected off floors, walls and objects, making it challenging to consistently provide reliable ranging results. Putting the receivers in a more open region or removing any barriers that might induce those inferences would be simple ways to get around multipath effects. However, in most interior situations, this alternative might not be feasible. UWB is also not able to provide us with euler angles, which is needed for pose estimation, a must-have for autonomous localization [16].

The IMU sensor provides us with acceleration and gyroscope data which is fed into the INS. The IMU is a popular sensor for robot pose estimation because of its compact size, quick update frequency, and excellent accuracy. Nevertheless, there will be cumulative errors because INS determines its position by integration. Eventually, after a lengthy operation, the position estimation will drift, hence the INS can not be used on its own. The integration of UWB and INS can yield its benefits. INS, on one hand, increases short term positioning accuracy. On the other hand, UWB can maintain long term position accuracy by suppressing the integral divergence of the INS [17].

In the following section, you will see the results of the HYBRID SIMULATOR, developed at LINKS Foundation.

Simulation Setup

The hybrid IMU-UWB simulator was implemented using Matlab. A virtual lab with UWB anchors was set up, and a path was generated. From the generated path, the following information was extracted:

1. Position data (x, y, z coordinates)
2. Acceleration data (Acc_x, Acc_y, Acc_z)
3. Gyroscope data (ω_x, ω_y, ω_z)

The ranging from each anchor was calculated based on the positional data, considering a specific UWB range. The generated path served as the ground truth. The real measurement was then modeled as:

$$M_{\text{real}} = M_{\text{true}} + \sigma \cdot \mathcal{N}(0,1) \tag{4.7}$$

where:

- $\boldsymbol{M_{\text{real}}}$ is the modeled value (ranging, acceleration and angular velocity).
- $\boldsymbol{M_{\text{true}}}$ is the ground truth generated with the help of Matlab
- $\boldsymbol{\sigma}$ is the standard deviation which differs in each case. In the case of finding the set of real rangings, we use the ranging standard deviation which is the spread of range (distance) measurements due to noise, interference, and environmental factors. For the acceleration. For the acceleration and the angular velocity, the σ represents the random noise in the environment the IMU experiences in real experiment faces.
- $\boldsymbol{\mathcal{N}}(0,1)$ generates a random number between 0 and 1 which represents the induced error, that follows a Gaussian distribution with mean value 0 and standard deviation of 1.

After we have our real data, we will apply the kalman filters discussed above. The EKF is used to estimate our UWB position (input ranging parameters, output positional data). The ESKF is used to determine the hybrid positional data and the euler angles.

Simulation Results

In this section we will see the results of three simulation carried out by letting randn decide different set of data for us. Below are the UWB solutions:

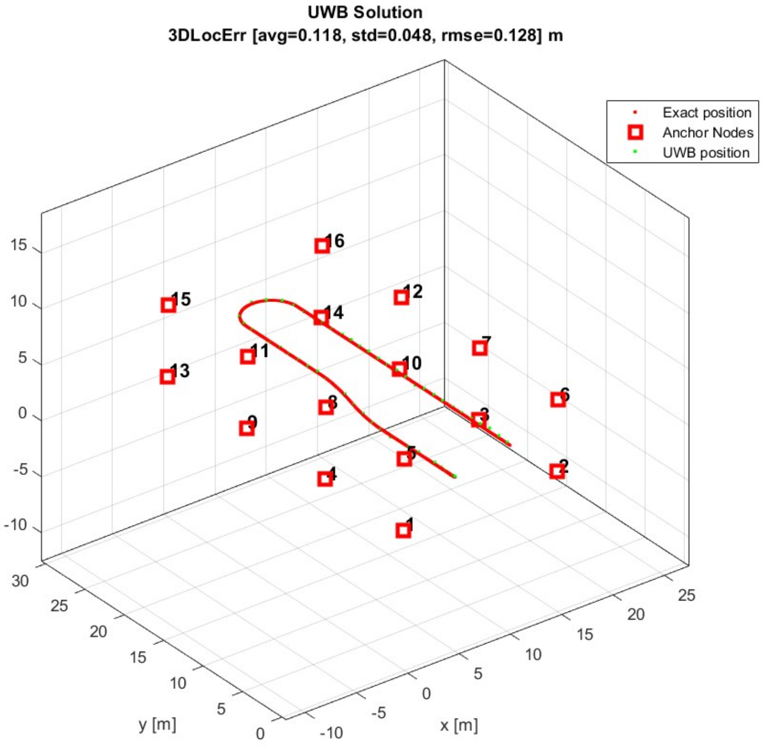


Figure 4.3: UWB solution

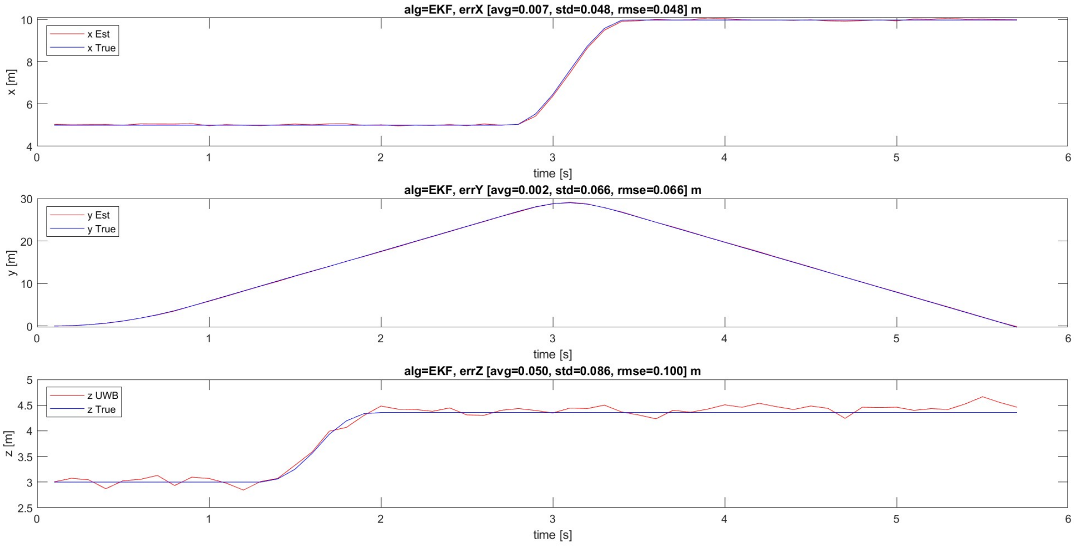


Figure 4.4: UWB Positioning (X, Y, Z)

Now let us take a look at the hybrid solution:

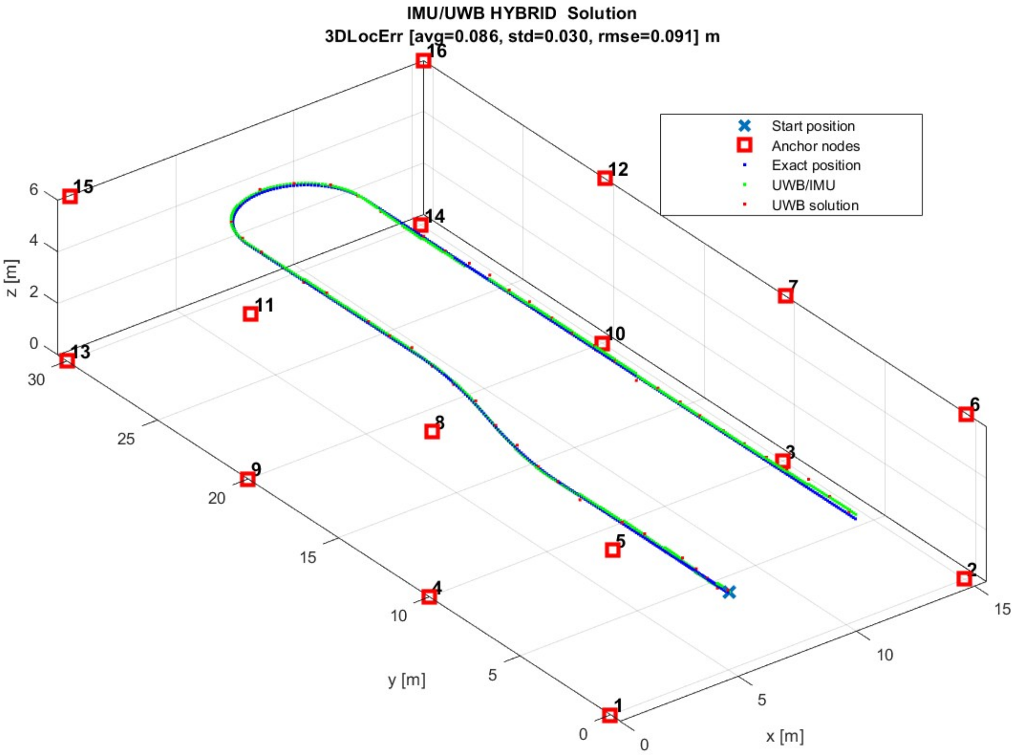


Figure 4.5: Hybrid Solution

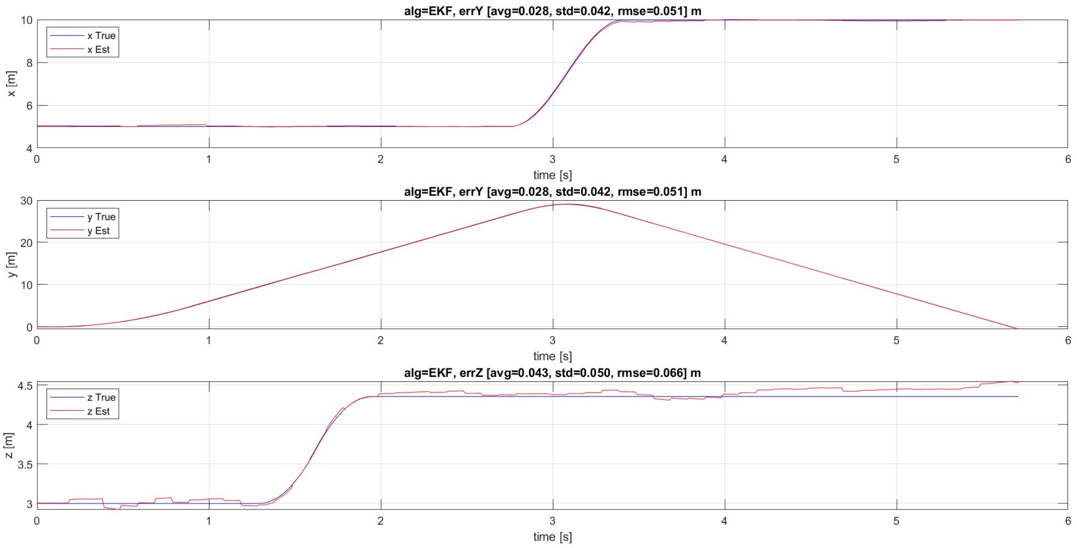


Figure 4.6: Hybrid Solution Positioning (X, Y, Z)

And now finally lets take a look at the euler angle estimation:

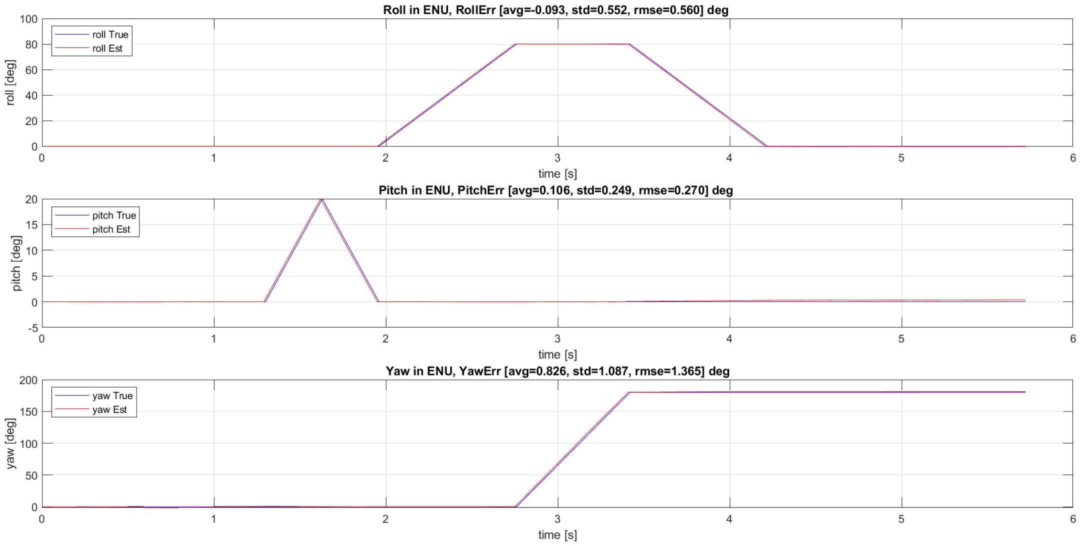


Figure 4.7: Euler Angles

The above results are of simulation 1. The results of the other two simulations and simulation 1 are present in the table below.

Sim No.	**UWB 3D Positional Error (m)**	**Hybrid 3D Positional Error (m)**	**% Improvement**	**Err Roll (°)**	**Err Pitch (°)**	**Err Yaw (°)**
1	0.128	0.091	28.91	0.560	0.270	1.365
2	0.112	0.085	24.11	0.559	0.269	1.349
3	0.159	0.115	27.67	0.515	0.258	1.434

Table 4.1: Comparison of UWB and Hybrid 3D Positional Errors and Euler Angle Errors

The results clearly show that, the implementation of hybrid solution is worth the cost because we have almost 25% improvement in our solution from UWB only solution. In indoor environments this is a lot and in-fact, with this we also get euler angle estimation, which is also extremely accurate.

4.2 Outdoor Localization Techniques

Among outside localisation techniques, Global Navigation Satellite Systems (GNSS) are the most widely utilized. These kinds of systems are made up of groups of artificial satellites in high orbit that are geo-referenced and constantly send out timing and location information. Generally, the GNSS can have a positional localization error of 1-10 m or in some cases even more [18]. This unreliable range can be a issue because autonomous navigation requires very precise localization data. To improve the localization error, the most common solutions are GNSS paired with RTK [19], GNSS paired with IMU [8] or GNSS-RTK paired with IMU [20].

4.2.1 GNSS

A Global Navigation Satellite System (GNSS) consists of constellations of medium-Earth orbit satellites that provide global or regional positioning and timing services. Currently, there are four primary GNSS systems in use: Galileo (E.U.), Beidou (China), GLONASS (Russia), and GPS (U.S.). There is also the Indian NavIC which provides coverage for the Indian area and 1500 km beside it and Japanese QZSS in East Asia and Oceania region. The time-of-arrival (TOA) notion is used by the Global Navigation Satellite System to calculate the user's position. The receiver calculates the satellite-to-user distance by estimating the satellite signal's travel time and multiplying it by the speed of light. When enough measurements are available, the user can estimate the satellite's position through trilateration because the receiver knows its exact position thanks to the modulated data delivered by the signals. A satellite's theoretical visibility does not necessarily translate into its actual use for location determination; obstructions or poor signal quality could prevent the receiver from processing the information. Theoretically, when at least four satellites are visible, the user can locate the point in three-dimensional space that corresponds to its position by intersecting the four spheres that were created from the corresponding satellite-to-user distances, as illustrated in figure 4.8 [21].

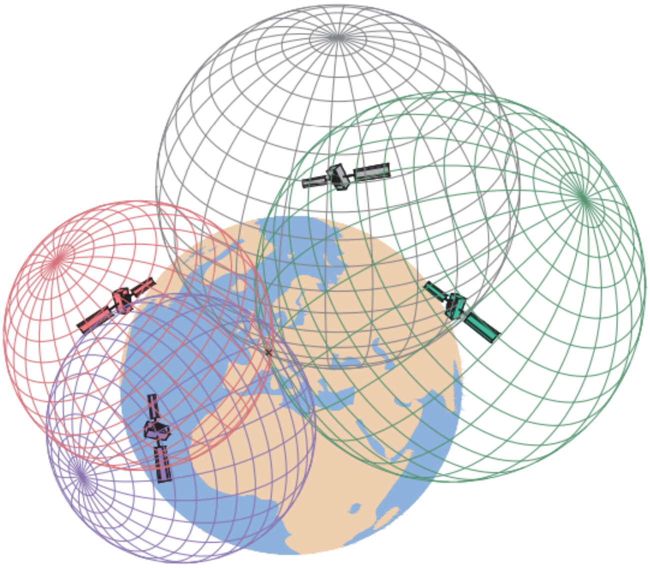


Figure 4.8: Four satellites for three-dimensional trilateration[1]

To achieve precise positioning, GNSS receivers use pseudoranges, which represent the estimated distance between a satellite and the receiver. These pseudoranges are derived from the difference between the satellite's broadcast time and the receiver's recorded time, multiplied by the speed of light. However, pseudoranges are affected by several error sources, including ionospheric and tropospheric delays, satellite clock drift, and receiver clock bias.

The representation of position in ECEF coordinate frame has less significance in navigation. Hence, the ellipsoidal representation (longitude, latitude and altitude) are more commonly use for coordinate representation. While working on outdoor localization for this thesis, the ENU coordinate frame was used to evaluate the results. The relation between the ECEF coordinates and the ellipsoidal representation is as follows [22]:

$$\mathbf{X} = (\mathbf{N} + \mathbf{h}) \cos\phi \cos\lambda \tag{4.8}$$

$$\mathbf{Y} = (\mathbf{N} + \mathbf{h}) \cos\phi \sin\lambda \tag{4.9}$$

$$\mathbf{Z} = \left(\frac{\mathbf{b^2}}{\mathbf{a^2}} \mathbf{N} + \mathbf{h} \right) \sin\phi \tag{4.10}$$

$$\mathbf{N} = \frac{\mathbf{a^2}}{\sqrt{\mathbf{a^2} \cos^{\mathbf{2}}\phi + \mathbf{b^2} \sin^{\mathbf{2}}\phi}} \tag{4.11}$$

where:

- $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ are coordinates in ECEF-frame.
- λ is longitude, ϕ is latitude, h is the height.
- $\mathbf{N}$ is the prime vertical radius of curvature.
- a and b are the semi-major and semi-minor axes of the ellipsoid

Now, the relation between the ECEF coordinate frame and the ENU coordinate frame is as follows:

$$\mathbf{R^{ENU}_{ECEF}} = \begin{bmatrix} -\sin\lambda & \cos\lambda & 0 \\ -\sin\phi\cos\lambda & -\sin\phi\sin\lambda & \cos\phi \\ \cos\phi\cos\lambda & \cos\phi\sin\lambda & \sin\phi \end{bmatrix} \tag{4.12}$$

$$\begin{bmatrix} x_{enu} \\ y_{enu} \\ z_{enu} \end{bmatrix} = \mathbf{R^{ENU}_{ECEF}} \begin{bmatrix} X - X_0 \\ Y - Y_0 \\ Z - Z_0 \end{bmatrix} \tag{4.13}$$

where:

- X_0, Y_0, Z_0 is the reference position which is a fixed point in ECEF (Earth-Centered Earth-Fixed) coordinates, usually taken as the starting point for the ENU frame.
- $\mathbf{R^{ENU}_{ECEF}}$ is the rotation matrix, used to convert points in ECEF frame to ENU frame.

GNSS has its own share of limitations in areas where there are obstacles between satellites and receivers. These can lead us to have multipath errors (when the receiver, receives multiple signals due to reflection). These cases can make it difficult for us to reach high levels of accuracy. In these case, we can apply GNSS integrated with Real Time Kinetic positioning (GNSS-RTK).

GNSS-RTK is a solution developed to mitigate errors caused by the earth's atmosphere which influences satellite-transmitted signals when GNSS is used for independent positioning. A real-time navigation device can attain precision down to the centimeter level by correcting for computation mistakes brought on by atmospheric signal interference. A base station is set up at known location. The base station continuously receives GNSS signals and then it calculates the difference between its known position and the GNSS based position, By doing this, it can identify the errors affecting GNSS signals. On the other hand, we have our vehicle, which also receives the GNSS signals, which gets corrected, by the base station. The corrections are transmitted over radio, internet or other communication methods. It is important to note that, the accuracy of RTK depends on how close we are to the base station and wether there is a clear line of sight (LOS) to the base station. If we are too far from the base station, then the improvement in RTK solution is not that much. Solutions coming from GNSS-RTK can be precise at centimeter level at times [19].

4.2.2 Hybrid GNSS-IMU Localization

GNSS on its own as previously discussed is not extremely precise, and it also does not have a very high frequency often maxing out at 50 Hz, which is not good enough for autonomous system. GNSS-RTK, without a proper LOS with base station, can also be a burden, given the extra relatively higher costs. This also means that if needed in a GNSS/GNSS-RTK denied environment (for example under a bridge, in a tunnel etc), we would have even worse positioning data. To address these limitations, integrating an IMU with GNSS/GNSS-RTK provides a more robust and reliable localization solution, ensuring continuous navigation even in GNSS/GNSS-RTK denied environments and also provide us with the exact orientation at all times, at a much higher refresh rate.

The system integration design is exactly the same as the case with UWB and IMU. The GNSS is used for long term estimation and the IMU is used for short term estimation with high frequency. Similar to indoor localization, we can achieve better results than standalone GNSS/GNSS-RTK when integrated with an IMU. There have been multiple studies, showing how effective GNSS/IMU solution can be [8], [7] (only talks about GPS/IMU localization). Similarly, GNSS-RTK paired with IMU showed great results [20].

Let us summarize the crucial findings in these papers:

- In [8], the author investigates the fusion of low cost IMU and GNSS with the help of EKF and ESKF. The GNSS signal was acquired at 1 Hz and the IMU acquisition frequency was 100 Hz. There were three different types of simulation trajectories set up- Oval, Serpentine and a Polygon. The author evaluated the positional errors along x, y and z (mentioned in the paper as lateral, longitudinal and vertical).

The general trend was ESKF fusion was better than EKF fusion. The GNSS only solution was as imagined, the worst of the lot. Looking at the OVAL based simulation, it is seen that we could expect the following percentages of improvement:

Filter	Lateral (%)	Longitudinal (%)	Vertical (%)
EKF	18.0	29.9	-17.4
ESKF	20.3	31.4	43.6

Table 4.2: Improvement Percentage of EKF and ESKF Over GNSS

- In [20], the author investigates the integration of GNSS-RTK and MEMS IMU with extended particle filter (EPF). The fusion was still done with the help of EKF, EPF was used after obtaining the gaussian approximates. The GNSS data was acquired at a frequency of 1 Hz and the IMU data acquisition frequency was 50 Hz. In this study, the tests were carried out in open sky and GNSS-denied environments (under bridge).

  The following table contains the 3D localization results achieved by the tests. The negative entry means decrease of performance in results. For reference, the percentage improvement of GNSS-RTK from GNSS was 98.75% in open sky condition and 23.87% under the bridge.

Method	Open Sky (%)	Under-Bridge (%)
GNSS-RTK/IMU EKF	-33.33	32.18
GNSS-RTK/IMU EPF	-33.33	64.97

Table 4.3: Improvement Percentage in Mean Localization Error Compared to GNSS-RTK

Chapter 5

Inertial Measurement Unit (IMU) for Localization

5.1 Selected IMU And Data Acquisition

For this thesis, the TDK ICM-42688-P IMU was used. It is a low-cost MEMS IMU. The ICM-42688-P chip contains an Accelerometer and a Gyroscope accessible via SPI and I^{2}C. Due to the limited capabilities of the MCU, only SPI 4 wires (full- duplex) is supported. An optional magnetometer could have also be connected but for the purpose of this thesis, we did not use any magnetometer. The IMU board is powered with a micro-USB cable [23].

The different protocols of communication can be accessed with different jumper connection, available on the IMU. During the activities carried out in this thesis, the SPI communication protocol was used, for which the jumper connection is as shown in Figure 5.1.

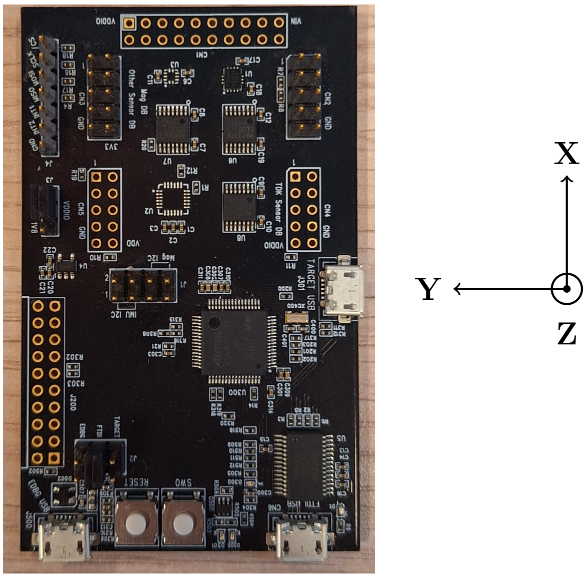


Figure 5.1: Illustration of IMU jumper connections and orientation axes.

The figure above also shows the default coordinate frame of the IMU (Z axis coming out of the plane of the page). This could also be changed by configuring the mounting matrix. In this study, the default coordinate frame was used without modifying the mounting matrix. The IMU can be configured with multiple frequencies (1 Hz, 100 Hz, 1000 Hz etc); for this thesis, a frequency of 200 Hz was used.

The IMU was designed for multiple uses like to detect steps, detect wake and sleep events and of course the measurement of acceleration and angular velocity. The C code for these different uses were provided. In this thesis, the C code given for the measurement of acceleration and angular velocity was modified slightly and used. The data coming from the USB Serial port of the IMU was logged using C++ code. The logging was done with the help of a Raspberry PI (Figure 5.3), eliminating the need to have a laptop physically mounted on the turtle bot/rover. The system architecture is further illustrated in the figure below (Figure 5.2).

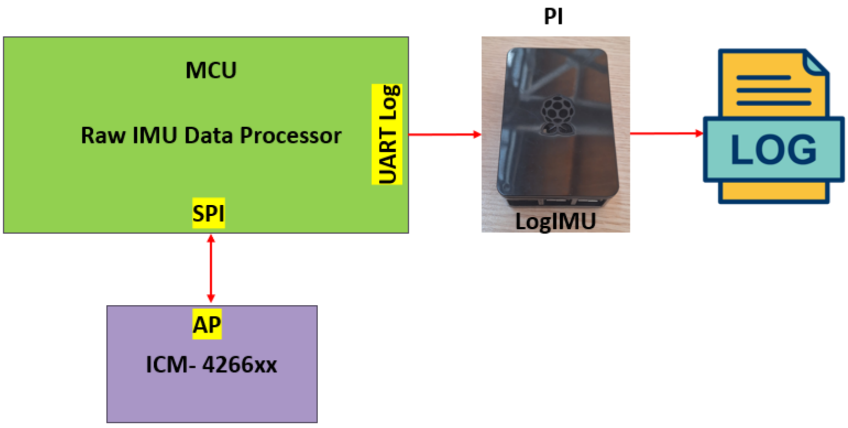


Figure 5.2: System Architecture

Figure 5.3: Raspberry PI 3rd generation

5.2 IMU Calibration And Functionality Check

For the IMU to function properly, it had to be calibrated, which was done only after the raw IMU data processor was loaded onto the MCU. To calibrate the gyroscope, hold the board motionless for about a second. Hold the board motionless on four distinct faces for about one second in order to calibrate the accelerometer. The calibration process is only needed when a new code is loaded onto the MCU.

To verify that the IMU was functioning well, a static test along with a gravity-based test was conducted.

Static test, as the name suggest was carried out by keeping the IMU motionless on a plane. The expected results of such a test and the measured results are in the table below:

	Expected	**Measured**
Average x-axis acceleration (acc_x)	0 m/s^2	-0.1526 m/s^2
Average y-axis acceleration (acc_y)	0 m/s^2	0.1809 m/s^2
Average z-axis acceleration (acc_z)	9.80665 m/s^2 [1]	9.7248 m/s^2
Average x-axis angular velocity (gyr_x)	0 rad/s	2.12×10^{-5} rad/s
Average y-axis angular velocity (gyr_y)	0 rad/s	7.47×10^{-6} rad/s
Average z-axis angular velocity (gyr_z)	0 rad/s	-1.78×10^{-5} rad/s

Table 5.1: Comparison of Expected and Measured IMU Data

We can see that acc_z is very close to its nominal value. The gyr_x, gyr_y, gyr_z are almost zero like expected. To deal with the acceleration bias discovered acc_x and acc_y, it was decided when carrying out the tests, to leave at least 1 minute for initial bias estimation. Then we can feed this initial bias in the ESKF and let the filter remove the bias.

Now we will take a look at the second set of experiments done, which was the gravity based test. In this test, the IMU was kept static on different position to measure the gravitational acceleration along the x, y, and z axes of the IMU (Figure 5.4 shows us on of the orientation). The results are summarized in the table below:

Test	**Expected Acceleration** (m/s^2)	**Measured Acceleration** (m/s^2)
T1	$(g_T, 0, 0)$	$(9.7836, 0.3376, 0.5425)$
T2	$(-g_T, 0, 0)$	$(-9.7459, -0.0848, 0.9654)$
T3	$(0, g_T, 0)$	$(0.1807, 9.7982, -0.0273)$
T4	$(0, -g_T, 0)$	$(-0.052, -9.7922, -0.3956)$
T5	$(0, 0, g_T)$	$(0.0810, 0.1060, 9.7271)$
T6	$(0, 0, -g_T)$	$(-0.1330, 0.0436, -9.8738)$

Table 5.2: Comparison of Expected and Measured Accelerations

---

[1] This value represents g_T which is the g measured in Turin.

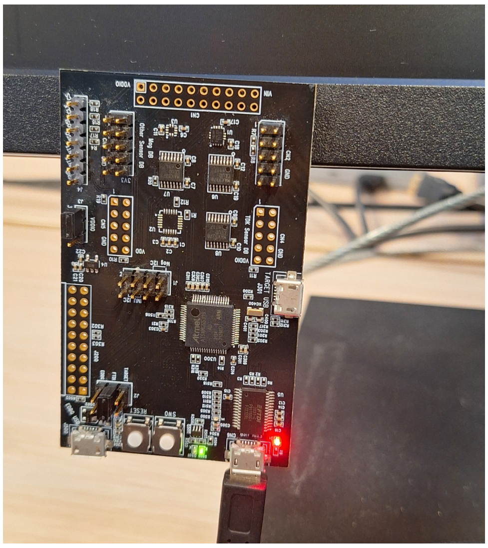


Figure 5.4: Positive X-axis gravitational acceleration measurement

5.3 IMU Data Integration In Localization

The IMU feeds data into the INS, which by itself is useless, as it determines the position by integration. Figure 5.5 shows us how the INS in our case works. It needs to be constantly reminded about the path to be followed. This is where we will use the UWB/GNSS measurements based on the type of localization (indoor/outdoor). Depending on the fusion strategy, these measurements can integrated in a "loosely" coupled or a "tightly" coupled manner.

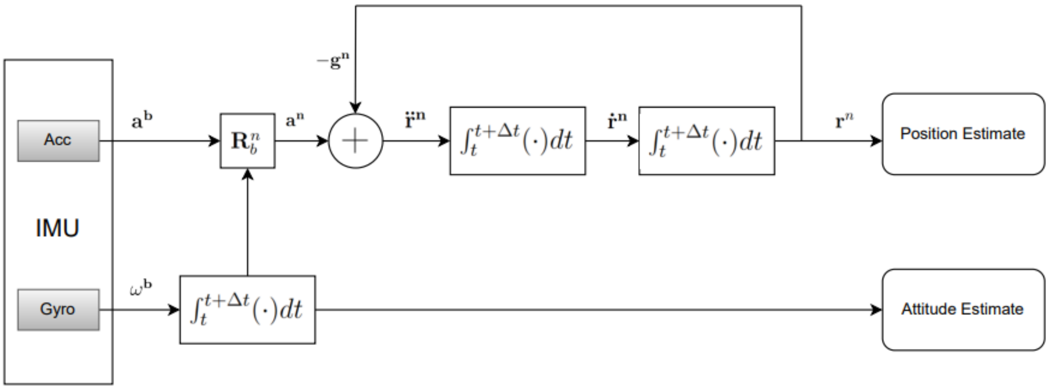


Figure 5.5: ENU INS

In the following section, we will take a look at two coupling solutions. To see how the equations are derived, please take a look at section 3.3 which talks in depth about the ESKF.

5.3.1 Loosely and Tightly Coupled Solutions

When the system is loosely connected (LC- Figure 5.7), it means that UWB/GNSS and IMU function separately and do not directly share data. This configuration makes advantage of the UWB/GNSS position estimates and does not use raw measurements such as pseudoranges or TOA rangings data which is used for tightly connected systems (Figure 5.7). For the purpose of this thesis, tightly coupling (TC) solution was only introduced in UWB/IMU sensor fusion. TC is useful especially in cases where fewer than 4 anchor nodes are available, as we can still have position estimation [24]. However, we did not face this issue when carrying out experiments for this thesis.

The use of the filter is essentially divided in two steps, irrespective of the algorithm selected:

1. Prediction- Happens when we do not have any UWB/GNSS data, we predict the state based only on the IMU (5.1)–(5.4)
2. Update- Happens whenever we have a UWB/GNSS update (5.5)–(5.9). The filter corrects the predicted state using measurement updates from UWB/GNSS. The update phase is then followed by a predict phase again(5.10)–(5.12) using the estimated values.

Loosely Coupled Solution

Prediction Phase

$$\hat{\boldsymbol{u}}_k^- = \tilde{\boldsymbol{u}}_k + \delta\hat{\boldsymbol{u}}_k^- \tag{5.1}$$

$$\hat{\boldsymbol{z}}_{k+1}^- = c(\hat{\boldsymbol{z}}_k^-, \hat{\boldsymbol{u}}_k^-) \tag{5.2}$$

$$\delta\hat{\boldsymbol{u}}_{k+1}^- = [\boldsymbol{\Psi}_k]_{10:15,10:15}\delta\hat{\boldsymbol{u}}_k^- \tag{5.3}$$

$$\boldsymbol{P}_{k+1}^- = \boldsymbol{\Psi}_k \boldsymbol{P}_k^- \boldsymbol{\Psi}_k^T + \boldsymbol{Q}_{d,k} \tag{5.4}$$

Update Phase

$$\boldsymbol{K}_{f,k} = \boldsymbol{P}_k^- \boldsymbol{H}_k^T (\boldsymbol{H}_k \boldsymbol{P}_k^- \boldsymbol{H}_k^T + \boldsymbol{R}_k)^{-1} \tag{5.5}$$

$$\delta\hat{\boldsymbol{z}}_k = \boldsymbol{K}_{f,k}\left(\boldsymbol{y}_k - \boldsymbol{H}_k \hat{\boldsymbol{z}}_k^-\right) \tag{5.6}$$

$$\hat{\boldsymbol{z}}_k = \hat{\boldsymbol{z}}_k^- + \delta\hat{\boldsymbol{z}}_k \tag{5.7}$$

$$\boldsymbol{P}_k = \boldsymbol{P}_k^- - \boldsymbol{K}_{f,k}\boldsymbol{H}_k\boldsymbol{P}_k^- \tag{5.8}$$

$$\hat{\boldsymbol{u}}_k^- = \tilde{\boldsymbol{u}}_k + \delta\hat{\boldsymbol{u}}_k \tag{5.9}$$

$$\hat{\boldsymbol{z}}_{k+1}^- = c(\hat{\boldsymbol{z}}_k, \hat{\boldsymbol{u}}_k) \tag{5.10}$$

$$\delta\hat{\boldsymbol{u}}_{k+1}^- = [\boldsymbol{\Psi}_k]_{10:15,10:15}\delta\hat{\boldsymbol{u}}_k \tag{5.11}$$

$$\boldsymbol{P}_{k+1}^- = \boldsymbol{\Psi}_k \boldsymbol{P}_k \boldsymbol{\Psi}_k^T + \boldsymbol{Q}_{d,k} \tag{5.12}$$

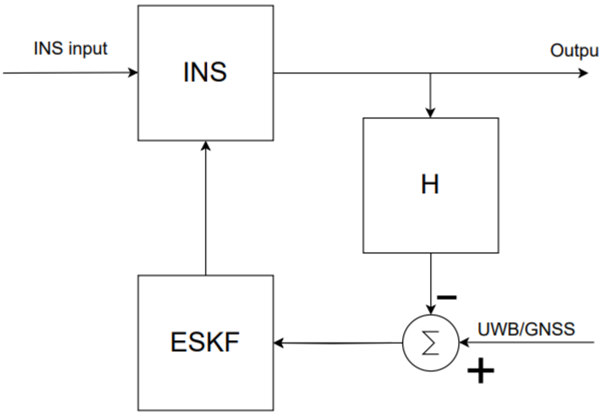


Figure 5.6: Loosely Coupled Algorithm [2]

Tightly Coupled Solution

Prediction Phase

Same as the set of equations (5.1)–(5.4)

Update Phase

$$\boldsymbol{r}_k^- = \|\hat{\boldsymbol{z}}_k^- - \boldsymbol{p}_{\text{anchor}}\| + v_k \tag{5.13}$$

$$\boldsymbol{H}_k = \frac{(\hat{\boldsymbol{z}}_k^- - \boldsymbol{p}_{\text{anchor}})^T}{\|\hat{\boldsymbol{z}}_k^- - \boldsymbol{p}_{\text{anchor}}\|} \tag{5.14}$$

$$\boldsymbol{K}_{f,k} = \boldsymbol{P}_k^- \boldsymbol{H}_k^T (\boldsymbol{H}_k \boldsymbol{P}_k^- \boldsymbol{H}_k^T + \boldsymbol{R}_k)^{-1} \tag{5.15}$$

$$\delta\hat{\boldsymbol{z}}_k = \boldsymbol{K}_{f,k}\left(\boldsymbol{y}_k - \boldsymbol{r}_k^-\right) \tag{5.16}$$

$$\hat{\boldsymbol{z}}_k = \hat{\boldsymbol{z}}_k^- + \delta\hat{\boldsymbol{z}}_k \tag{5.17}$$

$$\boldsymbol{P}_k = \boldsymbol{P}_k^- - \boldsymbol{K}_{f,k}\boldsymbol{H}_k\boldsymbol{P}_k^- \tag{5.18}$$

$$\hat{\boldsymbol{u}}_k^- = \tilde{\boldsymbol{u}}_k + \delta\hat{\boldsymbol{u}}_k^- \tag{5.19}$$

$$\hat{\boldsymbol{z}}_{k+1}^- = c(\hat{\boldsymbol{z}}_k^-, \hat{\boldsymbol{u}}_k^-) \tag{5.20}$$

$$\delta\hat{\boldsymbol{u}}_{k+1}^- = [\boldsymbol{\Psi}_k]_{10:15,10:15}\delta\hat{\boldsymbol{u}}_k^- \tag{5.21}$$

$$\boldsymbol{P}_{k+1}^- = \boldsymbol{\Psi}_k\boldsymbol{P}_k^-\boldsymbol{\Psi}_k^T + \boldsymbol{Q}_{d,k} \tag{5.22}$$

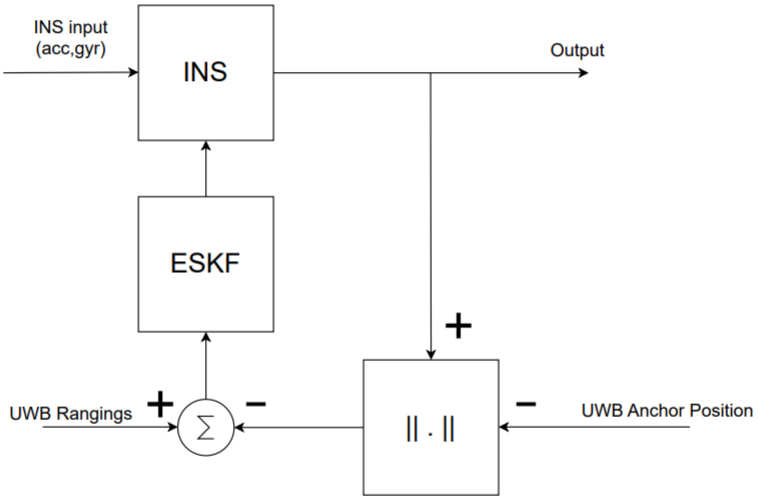


Figure 5.7: Tightly Coupled Algorithm

Chapter 6

Indoor Localization Solutions

6.1 Setup for Indoor Localization Experiments

6.1.1 Ground Truth (VICON)

When carrying out the indoor localization, the VICON system, was used as ground truth. The VICON system is made up of 6 cameras (one of them is shown in Fig 6.2b), covering the entire robotics lab's cage. The VICON has configurable frequencies (100 Hz, 200 Hz, 300 Hz) and uses infrared cameras which can provide very precise positional data (0.1 mm) and was logged using the help of a customized python script. The VICON system identifies the moving object with the help of markers placed on top of the moving object. The markers can be seen in Figure 6.5. The laboratory view from the VICON system with the identified object, which in this case is a turtle bot can be seen in Figure 6.1. The VICON camera the view from the VICON camera is presented in Figure 6.2.

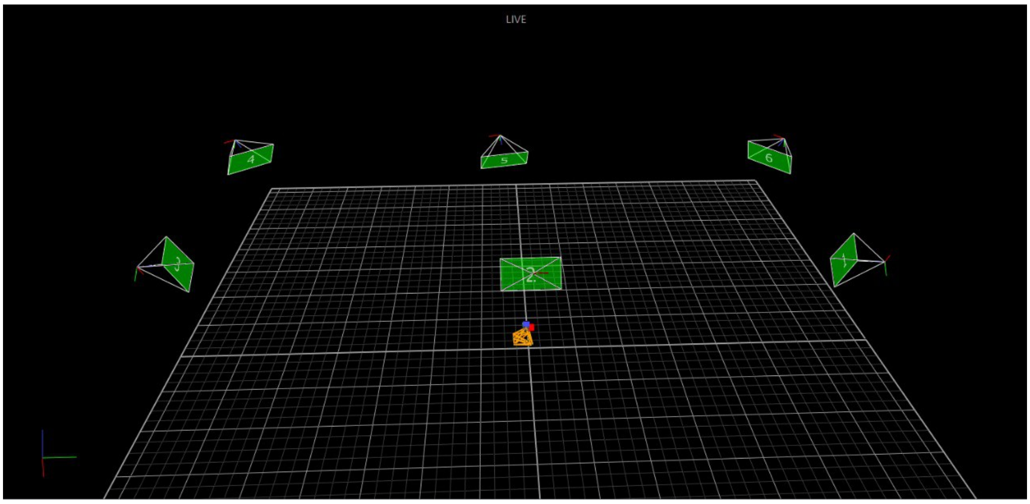
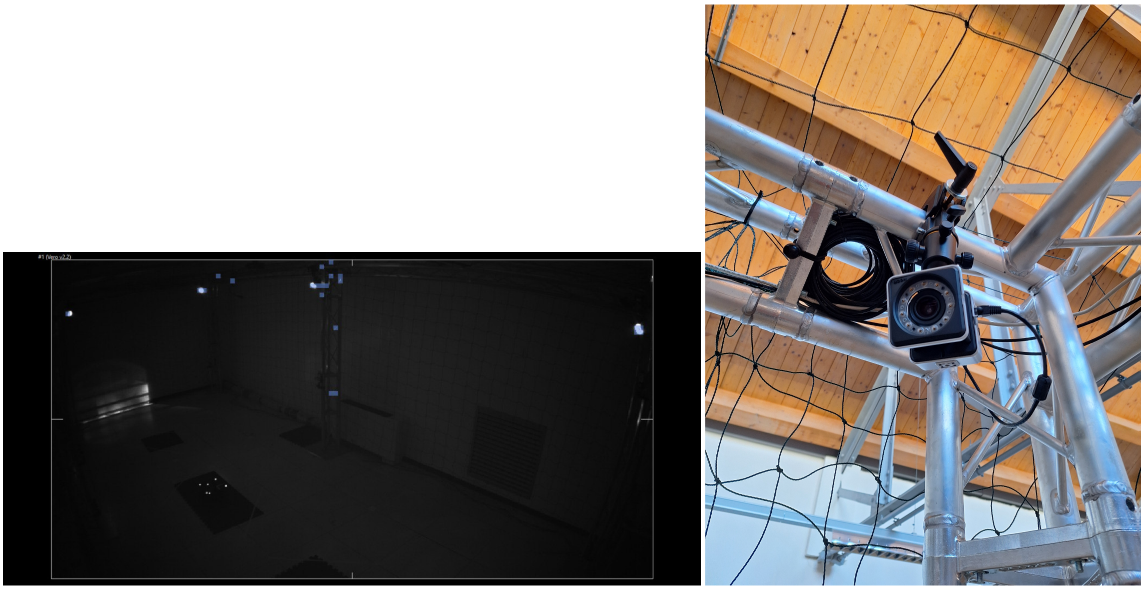


Figure 6.1: The robotics lab as viewed from VICON with the turtle bot [3]



(a) VICON Camera View (b) VICON Camera

Figure 6.2: VICON camera

6.1.2 Lab and Hardware Setup

The experimental setup can be divided in two parts, lab setup and hardware setup. The robotics lab at LINKS Foundation is shown in Figure 6.4. The area of the lab where the test was carried out is roughly 12 m^2. It consisted of the previously discussed VICON system and the UWB system. The UWB system consists of five anchor nodes, one of which is shown in Figure 6.3. Each anchor is equipped with a UWB module (DWM1000), a MCU (QN9090), and a power management unit. For positioning, we used the TOA-TWR model. The IMU is also mounted on the Turtle Bot with its frame aligned with the vehicle frame to eliminate the need for an additional rotation matrix. On the turtle bot, the tag which is required for UWB communication is also visible. The tag consists consists of the same UWB module and MCU as the anchor and a Bluetooth LE.

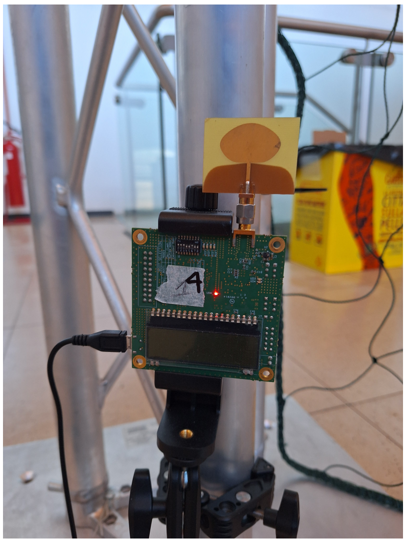


Figure 6.3: UWB Anchor

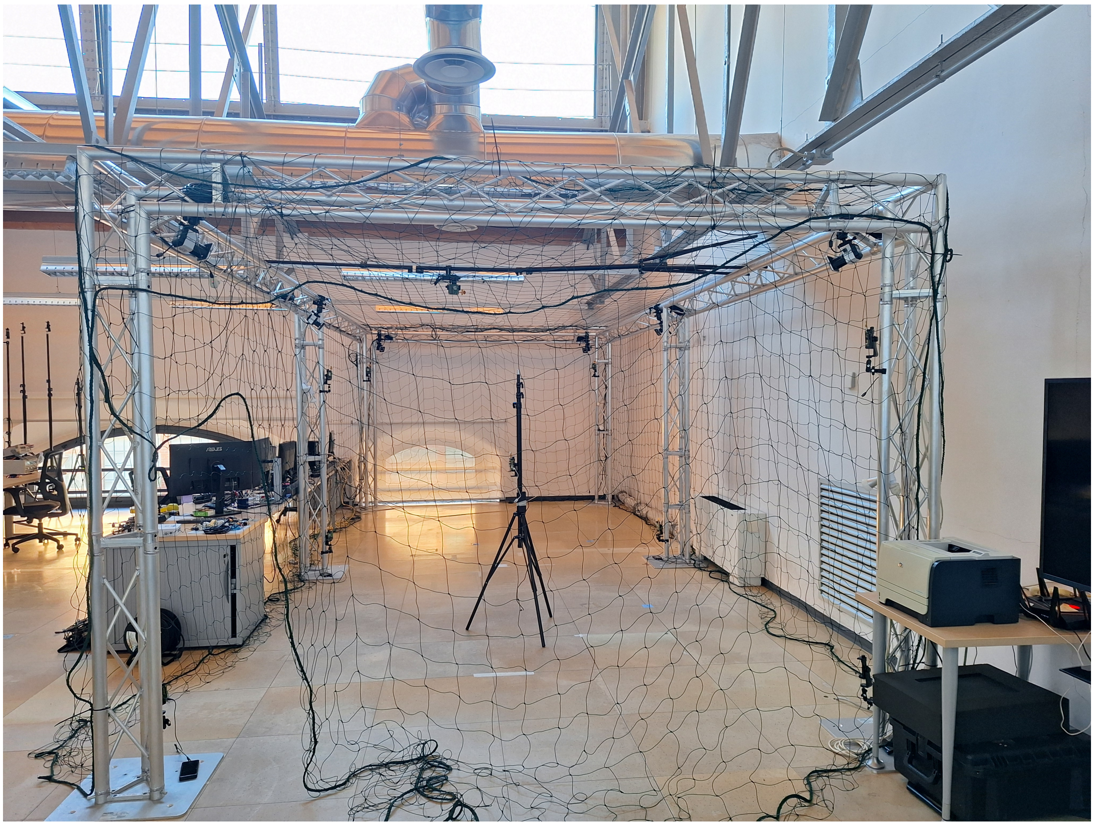
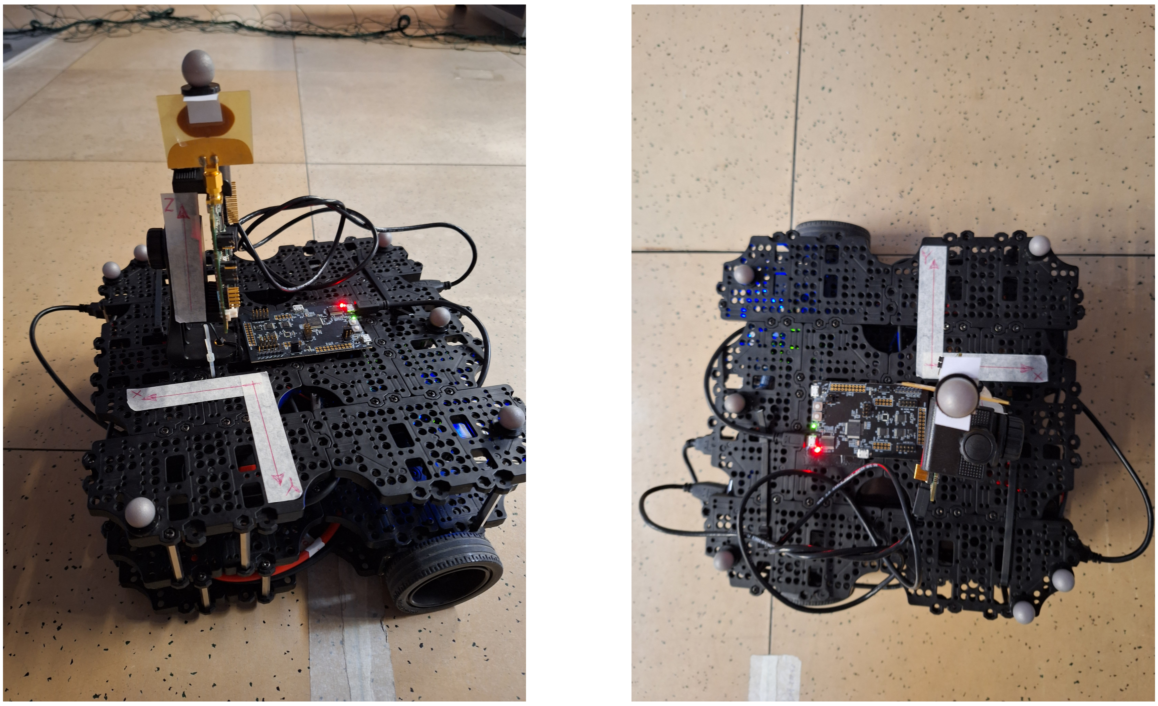


Figure 6.4: The Robotics Lab



(a) Front View

(b) Top View

Figure 6.5: Turtle Bot Setup

6.1.3 Test Cases and Execution

The tests were carried out by moving the turtle bot by hand inside the cage. This was because the UAVs available at LINKS foundation could not have all the hardware mounted on it at the same time (IMU, Raspberry Pi, power bank, UWB tag).

Two test scenarios were made with the UWB configuration to better estimate the z-axis position. The scenarios S1 and S2 are further shown in Figure 6.6 and 6.7. The difference between the two are the height of anchor 5. It must be mentioned that anchor 5 for each scenario (ID 9 for S1 and ID 8 for S2), were different in terms of hardware, but for simplicity have been labeled as anchor 5.

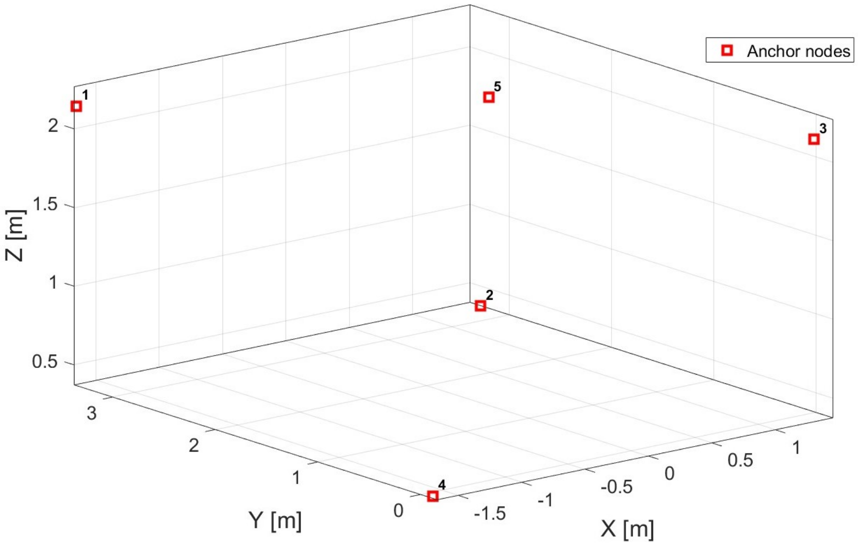


Figure 6.6: Scenario 1

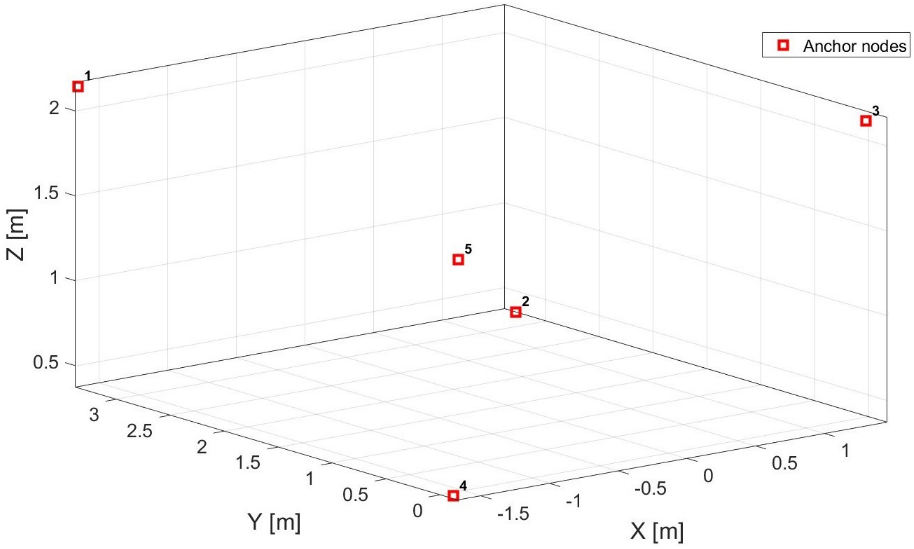


Figure 6.7: Scenario 2

6.2 UWB-Only Positioning

In this section, we will take a look at indoor localization solutions using only UWB. The UWB data was acquired at a frequency of 10 Hz. Some initial tests were carried out with the UWB, comparing the ranging solution it provided against the one computed with the help of VICON. The table below shows us the average error, standard deviation of the error and the root mean square error (RMSE) in ranging estimation.

	Anchor 1	**Anchor 2**	**Anchor 3**	**Anchor 4**	**Anchor 5**
Avg Error (m)	0.0616	0.0893	-0.0112	0.0681	-0.2008
Std Error (m)	0.0467	0.0637	0.0572	0.0839	0.1803
RMSE Error (m)	0.0773	0.1097	0.0583	0.1080	0.2698

Table 6.1: Scenario 1

	Anchor 1	**Anchor 2**	**Anchor 3**	**Anchor 4**	**Anchor 5**
Avg Error (m)	0.0533	0.0953	0.0261	0.0448	0.1400
Std Error (m)	0.0576	0.0968	0.0509	0.0979	0.1546
RMSE Error (m)	0.0785	0.1358	0.0572	0.1077	0.2085

Table 6.2: Scenario 2

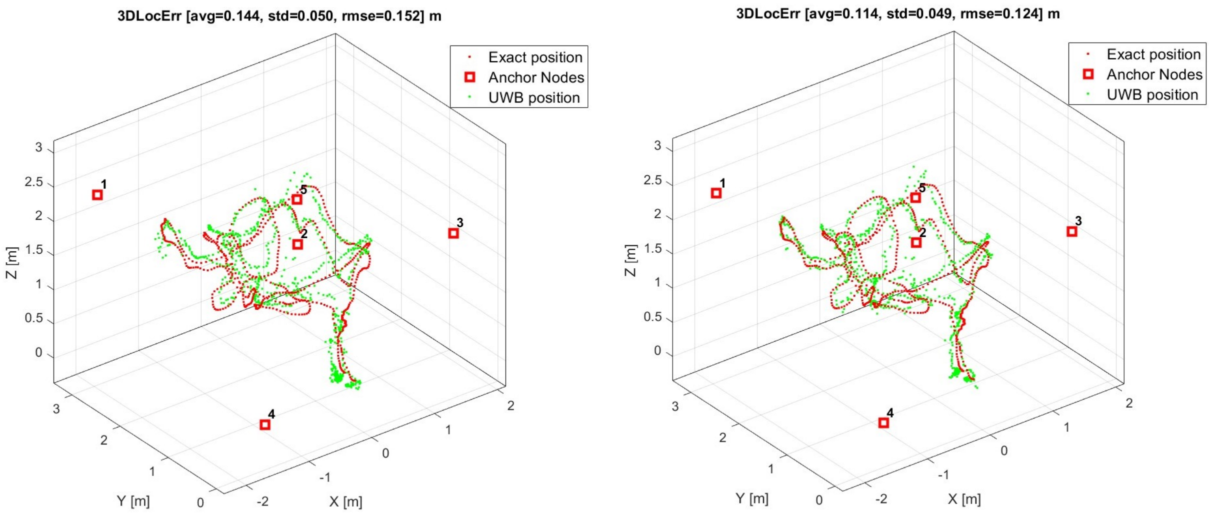
It was seen that the noise distribution of anchor 1 to anchor 4 were all different, within an acceptable range. Anchor 5 however in both scenarios, had a large average error and so all the ranging data was corrected. It is important to also note that anchor 1 to 4 (they were the same anchors) in both scenarios had more or less the same ranging error, meaning the estimated ranging error is deterministic.

To estimate the position with the UWB, the EKF described in section 3.2 was utilized. Only P model was used because it was sufficient since we did not evaluate the velocity and the movement speed was low. The matrix R_k from equation (3.20) which represents the noise covariance, is generally set to the same value for each anchor. In this thesis, R_k was also set to Std Error for the corresponding anchor, to see if we got better solution.

In this section, we will take a look at 2 sets of results, Sx-UWB(a) and Sx-UWB(b) where:

- x is the number of scenario, 1 or 2.
- a and b represent the selected noise covariance (a is the fixed noise covariance model, b is the anchor dependent noise covariance model)

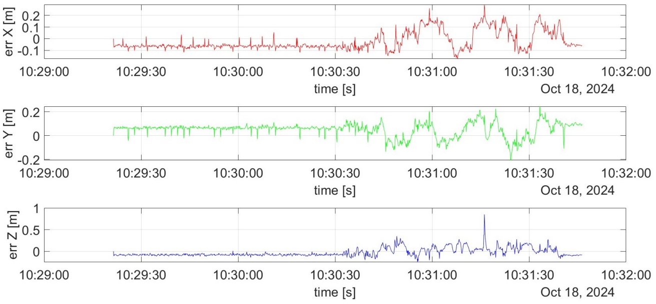
Scenario 1 (Figure 6.8) shows us that anchor dependent noise covariance model produces a better result (a difference of 2.8 cm in indoor positioning is quite substantial). This result is further validated by the result of test carried out in 2nd scenario shown in Figure 6.10. Figure 6.9 and 6.11 represents the positional error along x, y and z. The RMSE of the errors are mentioned in Table 6.3 and 6.4. Since the latter solutions produced better results, only they were used for the update phase of the loosely coupled solution in the hybrid UWB/IMU solution



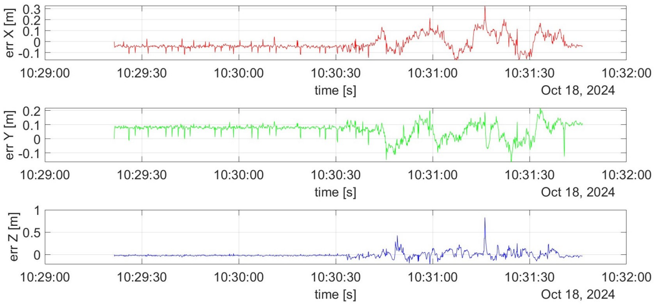
(a) Scenario 1-UWB(a)

(b) Scenario 1-UWB(b)

Figure 6.8: Comparison of trajectory between Scenario 1-UWB (a) and (b)



(a) Scenario 1-UWB(a) positional error

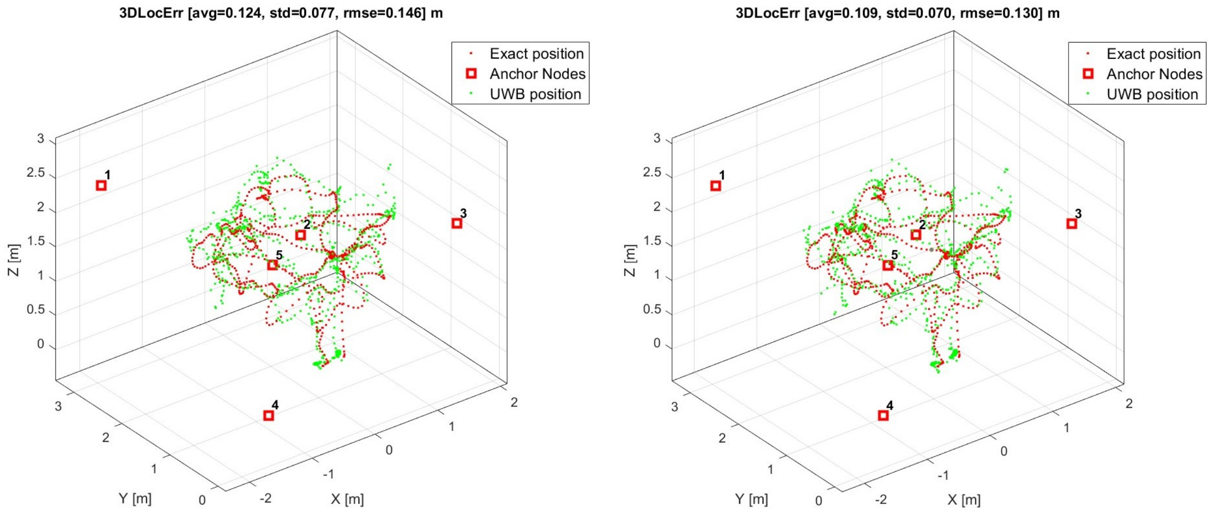


(b) Scenario 1-UWB(b) positional error

Figure 6.9: Comparison of positional error between Scenario 1-UWB (a) and (b)

Error	**S1-UWB(a)**	**S1-UWB(b)**
errX (m)	0.0836	0.0672
errY (m)	0.0775	0.0814
errZ (m)	0.1010	0.0654

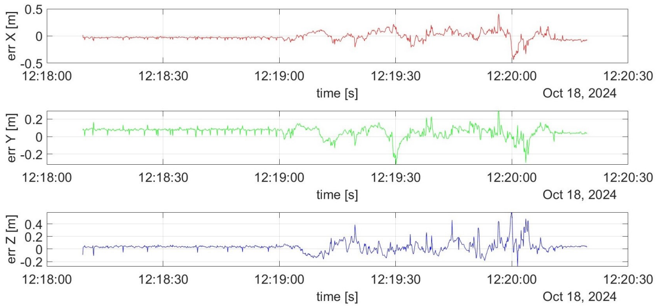
Table 6.3: RMSE of positional errors between Scenario 1-UWB (a) and (b)



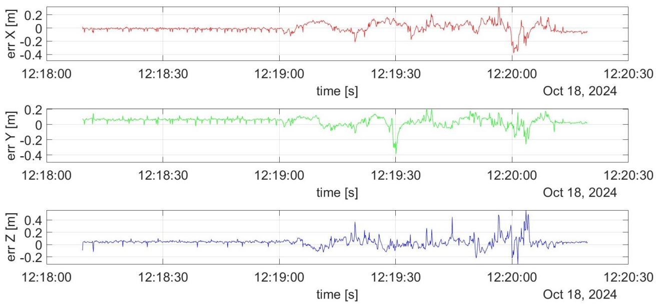
(a) Scenario 2-UWB(a)

(b) Scenario 2-UWB(b)

Figure 6.10: Comparison of trajectory between Scenario 2-UWB (a) and (b)



(a) Scenario 2-UWB(a) positional error



(b) Scenario 2-UWB(b) positional error

Figure 6.11: Comparison of positional error between Scenario 2-UWB (a) and (b)

Error	**S2-UWB(a)**	**S2-UWB(b)**
errX (m)	0.0791	0.0693
errY (m)	0.0819	0.0729
errZ (m)	0.0918	0.0820

Table 6.4: RMSE of positional errors between Scenario 2-UWB (a) and (b)

6.3 Hybrid Solution

6.3.1 Simulated UWB and Real IMU

To make sure the IMU was working before having the integration between real UWB data and real IMU data, we had an integration with simulated UWB data, which was normally distributed. There were two types of simulation carried out in only scenario 2 as it was enough to test wether the IMU was working as expected or not. The simulations and their results are explained below:

- Error on final UWB positioning- In this case, we discarded the real UWB positioning value and we took the ground truth acquired by the VICON system and added gaussian noise to it in this fashion:

$$\mathbf{UWB}_\mathrm{i} = \mathbf{GT_i} + \sigma \cdot \mathcal{N}(\mathbf{0{,}1}) \tag{6.1}$$

  where, GT is the ground truth, σ is the noise level and $\mathcal{N}(0{,}1)$ is a gaussian distribution. The results of this simulation is as follows:

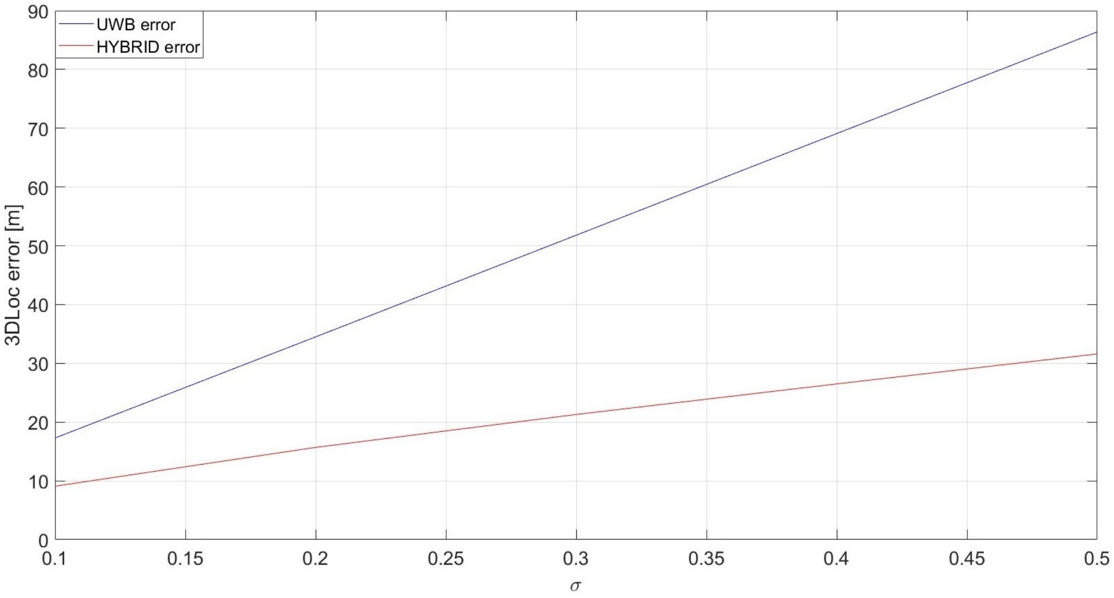


Figure 6.12: Noise on true position

- Error on UWB rangings- In this case, we discarded the ranging values, took the ground truth provided from the VICON system, calculated the true ranging and added gaussian noise to it before feeding it into the EKF. This can be formulated as follows:

$$r_{i,k} = \sqrt{(GT_{x,k} - A_{x,i})^2 + (GT_{y,k} - A_{y,i})^2 + (GT_{z,k} - A_{z,i})^2} + \sigma \cdot \mathcal{N}(0{,}1) \tag{6.2}$$

  where, $r_{i,k}$ is the ranging measurement from the i-th anchor at time k, $GT_{x,k}$ is the x-component of the ground truth at time k, and $A_{x,i}$ is the x-component of the i-th anchor coordinate. Similarly, $GT_{y,k}$, $GT_{z,k}$, $A_{y,i}$, and $A_{z,i}$ represent the respective components in the y and z directions. The result of this simulation is:

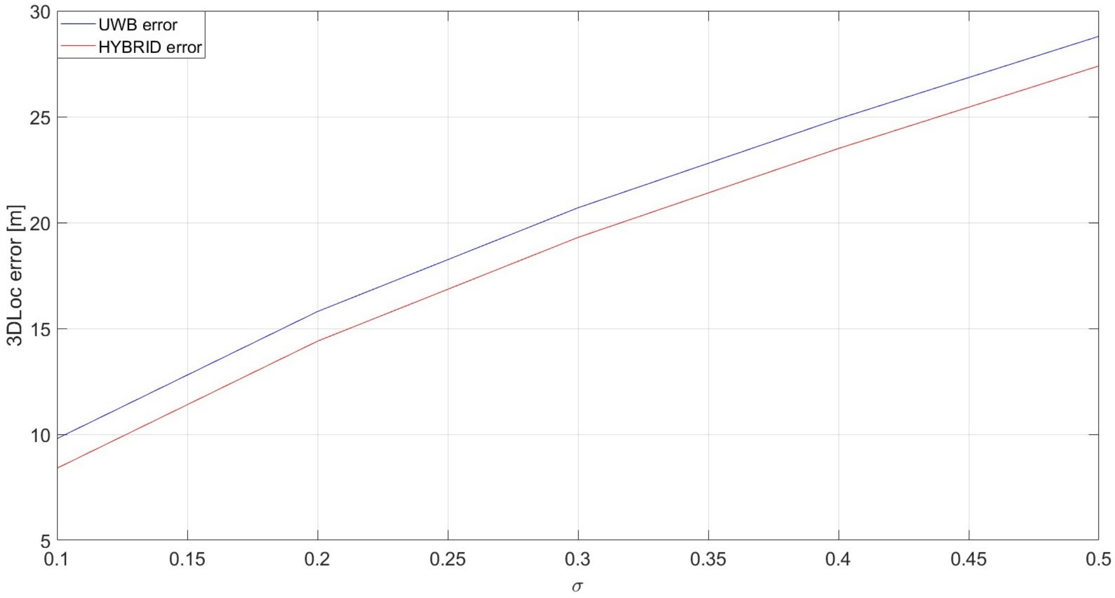


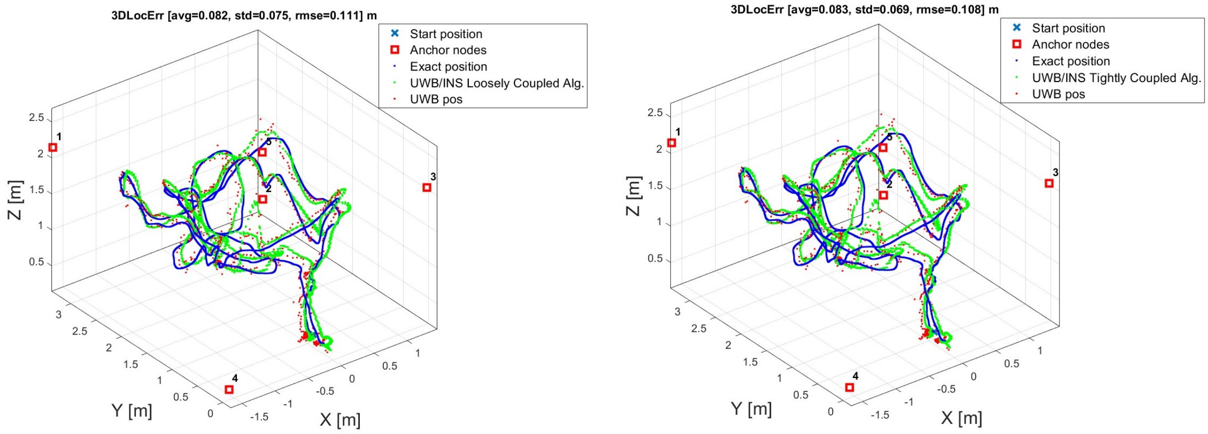
Figure 6.13: Noise on true rangings

It can be seen that when we add the error on the positioning (Figure 6.12), the hybrid solution tends to improve the result. As σ increases, the gap between the UWB localization error and the hybrid localization error increases. Now shifting our focus to the 2nd case (Figure 6.13), were we added error to the ranging, we see that the UWB errors are much lower, this is because we have a filtering process afterwards, and the EKF does a very good job removing the gaussian noise. Hence we do not see the large gap between UWB error and hybrid error, but we still see an improvement. These results prove that the sensor fusion was indeed working, and now we can finally take a look at the results obtained with real UWB and real IMU integration.

6.3.2 Real UWB and Real IMU

In this section, we will analyse the optimized hybrid solution developed with real UWB and IMU data. The section is divided into two subsection, showing us the hybrid solutions obtained in both scenario. In each subsection, two sets of solution are presented with LC and TC mechanism.

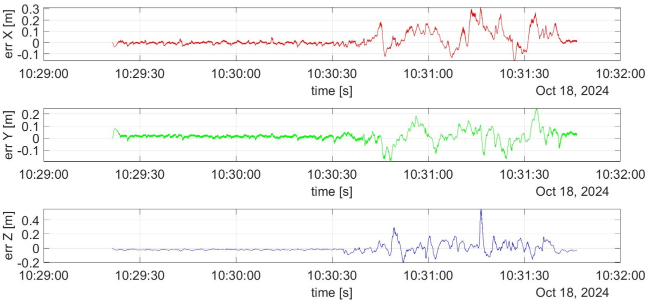
S1 Hybrid Solution



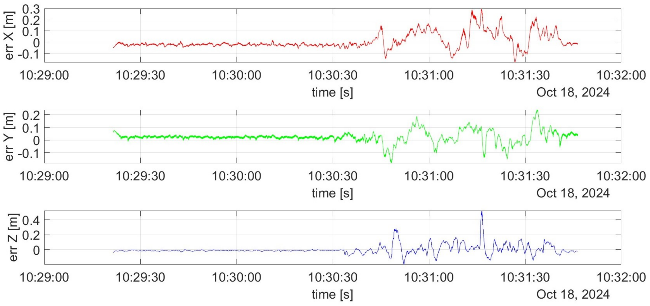
(a) S1-LC trajectory

(b) S1-TC trajectory

Figure 6.14: Comparison of trajectory between Scenario 1 LC and TC



(a) S1-LC positional error

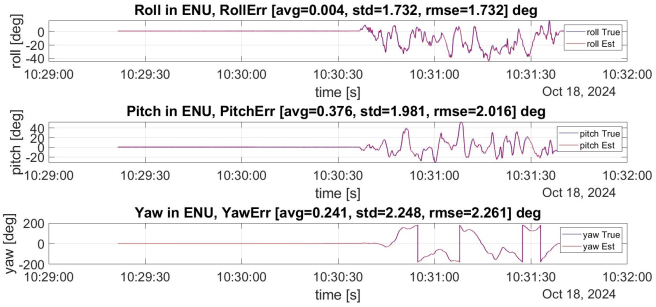


(b) S1-TC positional error

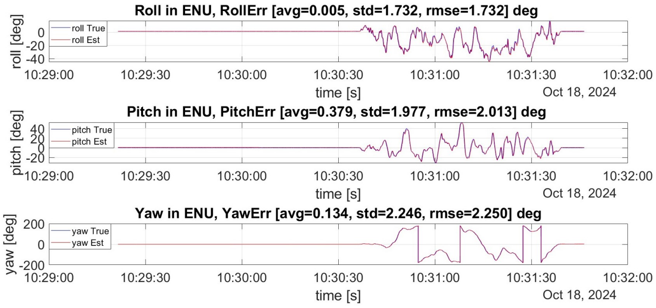
Figure 6.15: Comparison of positional error between Scenario 1 LC and TC

Error	S1-LC	S1-TC
errX (m)	0.07236	0.06892
errY (m)	0.05325	0.05452
errZ (m)	0.06519	0.06285

Table 6.5: RMSE of positional errors between Scenario 1 LC and TC



(a) S1-LC attitude estimation

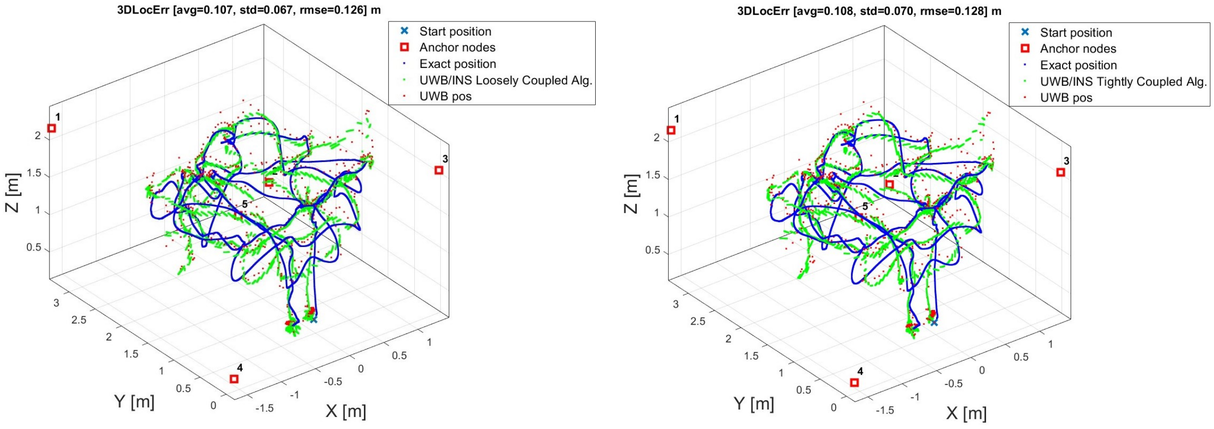


(b) S1-TC attitude estimation

Figure 6.16: Comparison of attitude estimation between Scenario 1 LC and TC

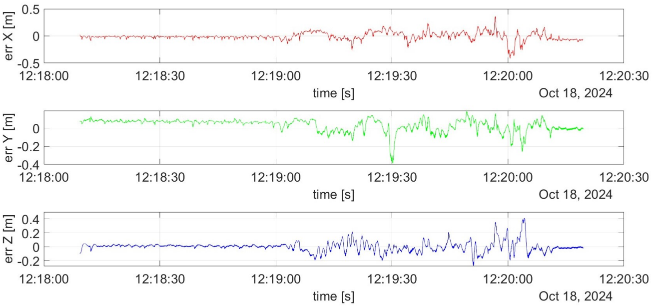
Figure 6.14 shows us the estimated trajectory (green), ground truth (blue), anchor nodes (red square) and UWB position estimation (red dots). From the plot, it is evident that the differences between the two mechanism are minimal, making it difficult to observe a clear distinction based solely on visual inspection. Hence, the RMSE is also mentioned at the top of picture. Figure 6.15 shows the positional errors and Table 6.5 reports the corresponding RMSE error. Finally, the attitude estimation are represented in Figure 6.16 (red), along with the true roll, pitch and yaw angles (blue).

S2 Hybrid Solutions

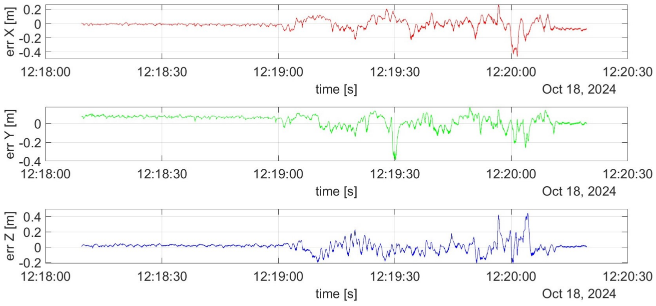


(a) S2-LC trajectory (b) S2-TC trajectory

Figure 6.17: Comparison of trajectory between Scenario 2 LC and TC



(a) S2-LC positional error

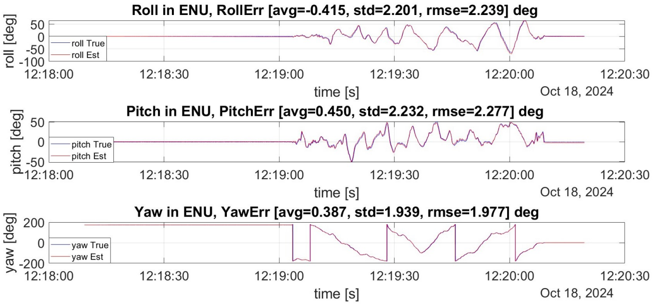


(b) S2-TC positional error

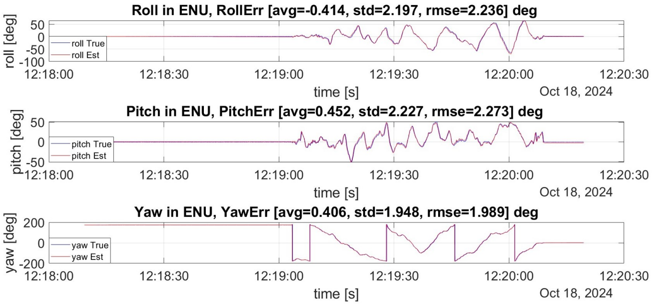
Figure 6.18: Comparison of positional error between Scenario 2 LC and TC

Error	S2-LC	S2-TC
errX (m)	0.07192	0.07554
errY (m)	0.07541	0.07442
errZ (m)	0.07120	0.07214

Table 6.6: RMSE of positional errors between Scenario 2 LC and TC



(a) S2-LC attitude estimation



(b) S2-TC attitude estimation

Figure 6.19: Comparison of attitude estimation between Scenario 2 LC and TC

The solutions in this case are also represented in the same way as described earlier for scenario 1 and so Figure 6.17 shows us the estimated trajectory (in green) along with ground truth (blue), anchor nodes (red square) and UWB position estimation (red dots). The RMSE is also mentioned at the top of picture. Figure 6.18 shows the positional errors and Table 6.6 reports the corresponding RMSE error. Finally, the real and estimated euler angles are represented in Figure 6.19

6.4 Result Discussion and Observations

Before carrying out the test, the hypothesis was that we would have a UWB 3D localization error around 20 cm, however the standalone UWB system, as seen in Section 6.2 already produces reasonably good indoor localization result. In fact S1 produces slightly better performance, and this could be due to the set up of the UWB anchors.

The article [25] shows us that with increasing spatial volume, the 3D localization error tends to improve as it is related to the DOP of the system. In the paper, the author dealt with 4 anchors and so to calculate the spatial volume of the area was relatively easier as it formed a tetrahedron. In our case, it is a bit more challenging since we have 5 anchors. In a simple system we would have to make 4 tetrahedrons and calculate the total volume as the sum of each of them. However, our system is quite complex and simply adding the volumes of individual tetrahedrons may lead to inaccuracies, as overlapping regions or gaps between tetrahedrons can result in incorrect total volume calculations. Thankfully though, with the help of Matlab function "convhull", we can calculate it directly.

Applying this in our case, we see that S1 has a spatial volume of 6.7563 m^3 and S2 has a spatial volume of 6.2476 m^3. This explains why, even though minor, we have a better 3D localization solution in S1.

Now, making our way to the hybrid solutions, looking at the simulations, we would have expected a 3D localization improvement of around 25% from UWB standalone case. However in real experiments that was not the case. One major reason for this was the UWB noise not being purely gaussian, and the EKF/ESKF generally have a more difficult time dealing with non-gaussian noise as it assumes all noise to be gaussian. The following table summarizes the percentage improvement in the hybrid solutions:

Method	**S1 (%)**	**S2 (%)**
LC	11.3	3.1
TC	12.9	1.5

Table 6.7: Percentage Improvement in Localization Accuracy for S1 and S2

S1 shows us decent results as we see a maximum of 12.9% improvement, which is about 50% of the expected result. Given the circumstances (very good UWB result means not a lot of room for improvement), this is a relatively good result.

However the results obtained with S2 were highly unusual. It was seen that when finding the optimum hybrid solution for S2, we could not trust heavily on IMU data (at least not as much as S1). Previous studies also showed that in UWB/IMU hybrid localization, the UWB solution tended to dominate the hybrid estimate [26]. This meant that if the UWB localization had errors, our hybrid solution could not improve them as expected. In fact, by taking a look at the XY plot of the S2 (Figure 6.20), it can be observed that the UWB estimate is off at the circled zone. One possible solution to this problem could be to not consider such UWB estimation for the UWB/IMU hybrid solution but this comes at a risk of divergences of INS especially if too many UWB solutions are filtered off.

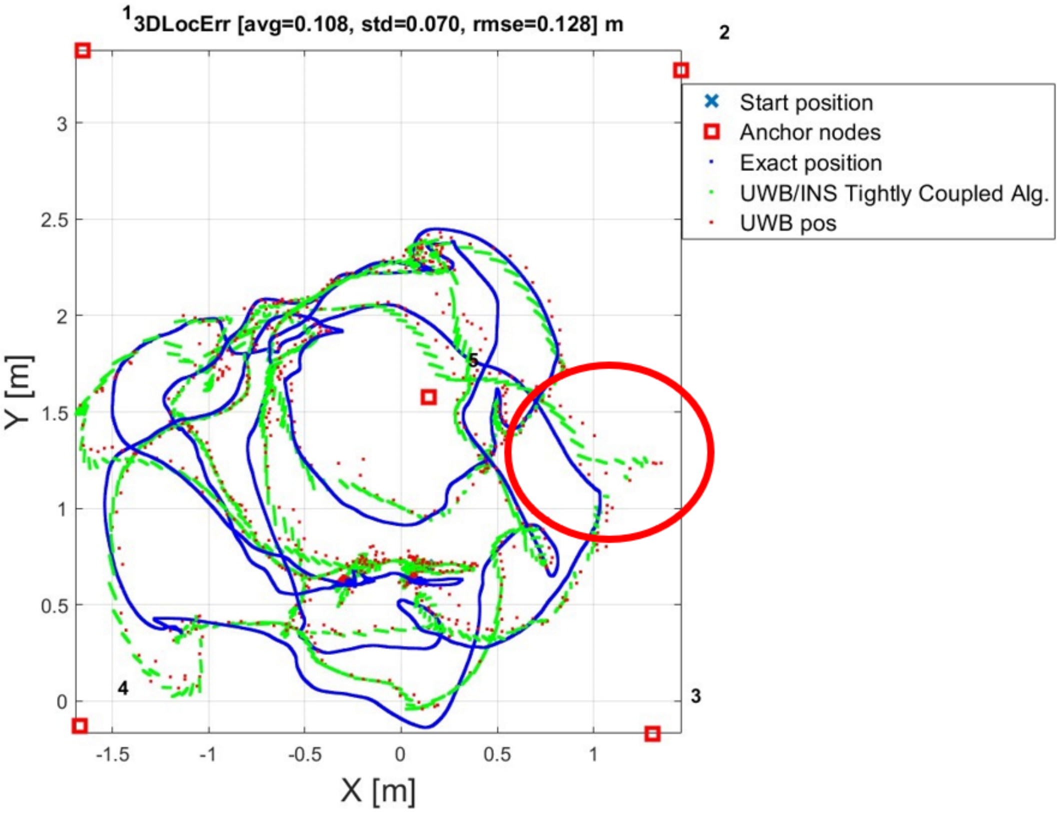


Figure 6.20: XY Plot S2 TC

The attitude estimation, in both scenarios were very good as in every case, roll (ϕ), pitch (θ) and yaw (ψ) errors were less than 5°, a minimum requirement for a stable autonomous system. The coupling mechanism (LC/TC), did not massively affect the euler angle estimation, as no stark differences were noticed between the two sets of solution, presumably because the gyroscope was very accurate as it could be seen with low bias in static situations.

These solutions are bound to change as the test environment changes, and so it would be recommended to go ahead and implement the UWB anchors as shown in S1, for a guaranteed better hybrid solution. In the current environment, both solutions are acceptable because:

- Very good attitude estimation.
- Hybrid localization solution at 200 Hz instead of 10 Hz in the case of UWB localization.
- A very good positioning estimation.

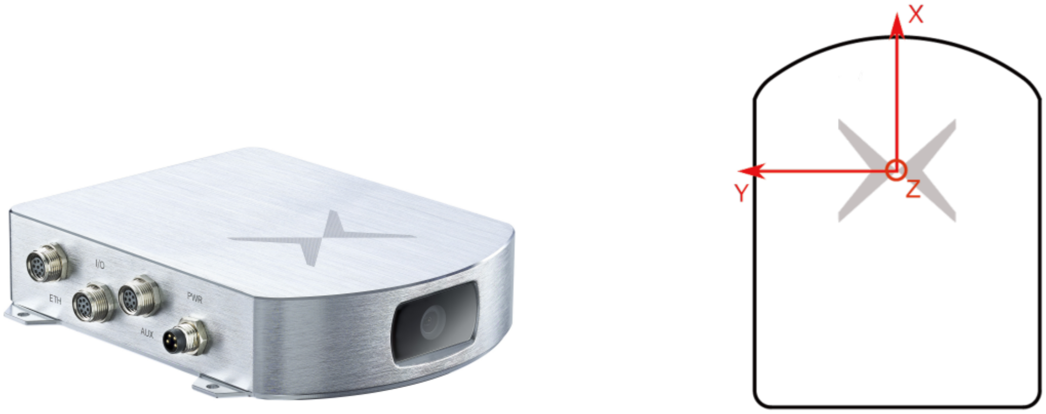
Chapter 7

Outdoor Localization Solutions

7.1 Setup for Outdoor Localization Experiments

7.1.1 Ground Truth (FIXPOSITION)

While carrying out the outdoor localization, Fixposition positioning sensor was used for GT. Fixposition a high-end sensor fusion solution that provides highly accurate real time pose information (3D position and orientation) in all scenarios, including GNSS degraded and denied environments. The system can also receive information from additional auxiliary sensors (such as wheel-speed) to increase performance whenever required. Fixposition can provide us with centimeter level accuracy [4].



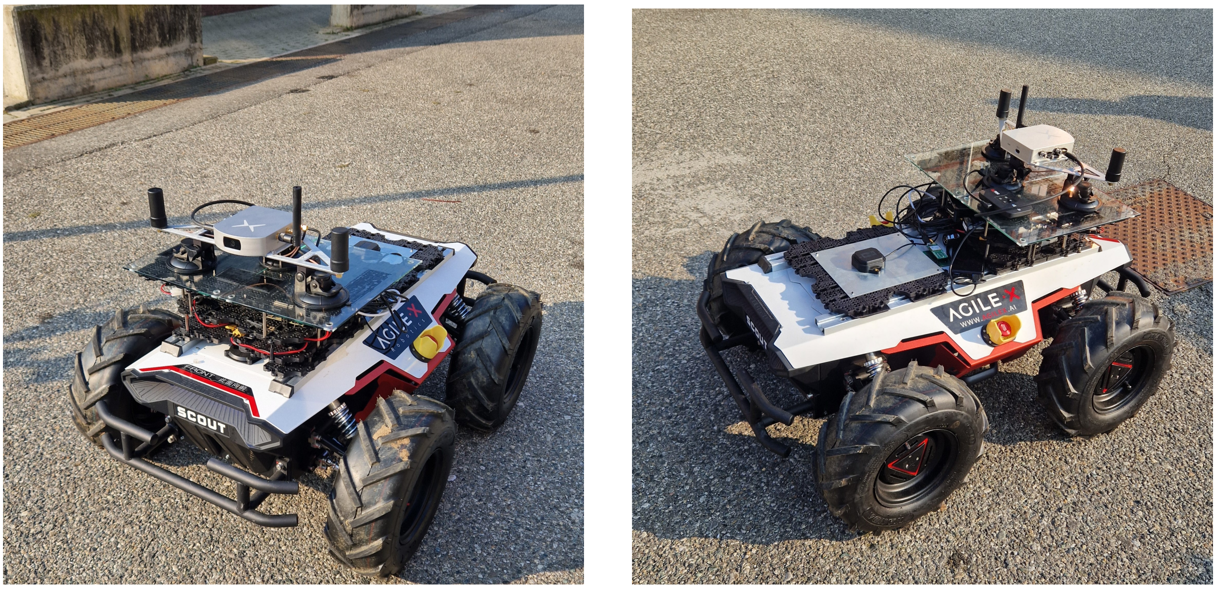
(a) Fixposition Module

(b) Fixposition Orientation

Figure 7.1: Fixposition [4]

7.1.2 Hardware Setup

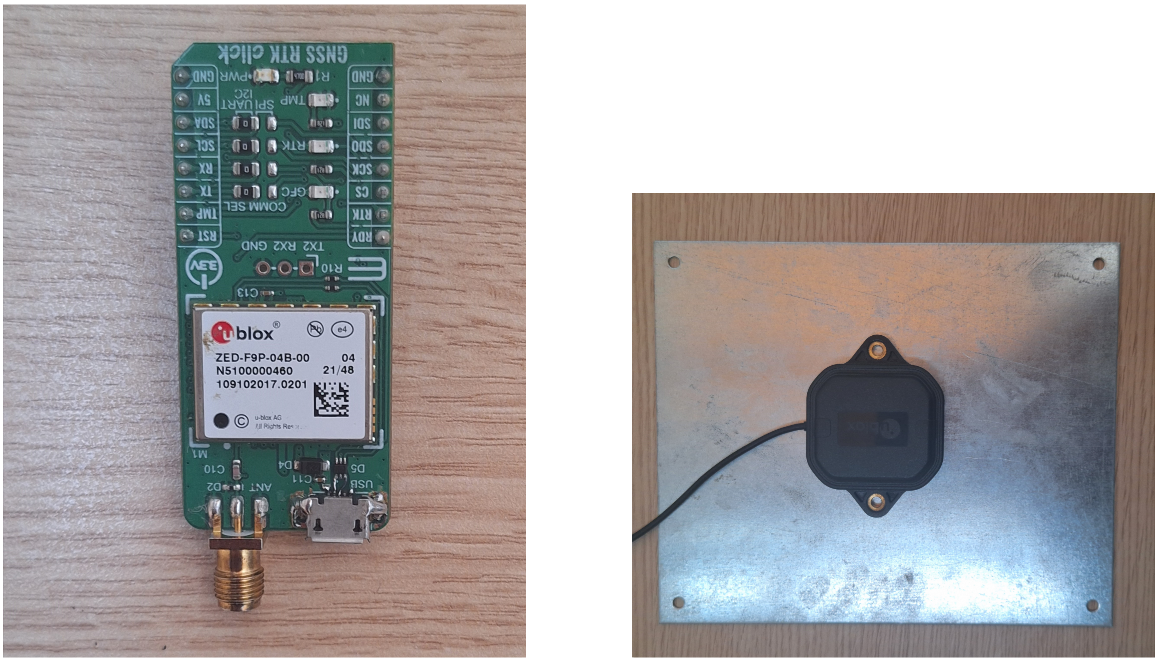
To carry out the test, the AgileX Scout 2.0 was used (Figure 7.2), on which three things there were mounted: IMU, U-blox module and antenna and Fixposition. The U-blox positioning set provides us with GNSS and GNSS-RTK data. It uses the antenna (Figure 7.3b) and the module (Figure 7.3a) and the raspberry pi described earlier (Figure 5.3) to the log the data. The IMU was mounted at the center of the rover. The Fixposition was powered with the help of a 7V battery. The PI was powered with the help of a power-bank, just like the case of indoor localization. It was important to note that the axis of Fixposition (Figure 7.1b) and IMU had to properly aligned. If the axes were misaligned, an extra rotation matrix would be needed to transform the coordinate frames, making the data processing more complicated.



(a) Front View

(b) Rear View

Figure 7.2: AgileX Rover



(a) U-blox module

(b) U-blox antenna

Figure 7.3: U-blox Positioning Set

7.1.3 Test Cases and Test Execution

The test cases for outdoor localization were relatively straightforward compared to indoor localization. Three main test scenarios were:

1. GNSS vs GNSS-RTK
2. GNSS vs GNSS/IMU
3. GNSS-RTK vs GNSS-RTK/IMU

Figure 7.4 shows the location where the test was carried out in semi-open-sky conditions. Semi-open-sky conditions represent conditions where we do not have clear and unobstructed view of the sky or in other words, urban areas. The red line is the path covered during the test which is approximately 500 m and the blue marker is LINKS Foundation for reference. The cross "X" represents the starting and the finishing point of the test. It is important to note that both GNSS and GNSS-RTK data was acquired at 10 Hz and at the same time using two U-blox module.

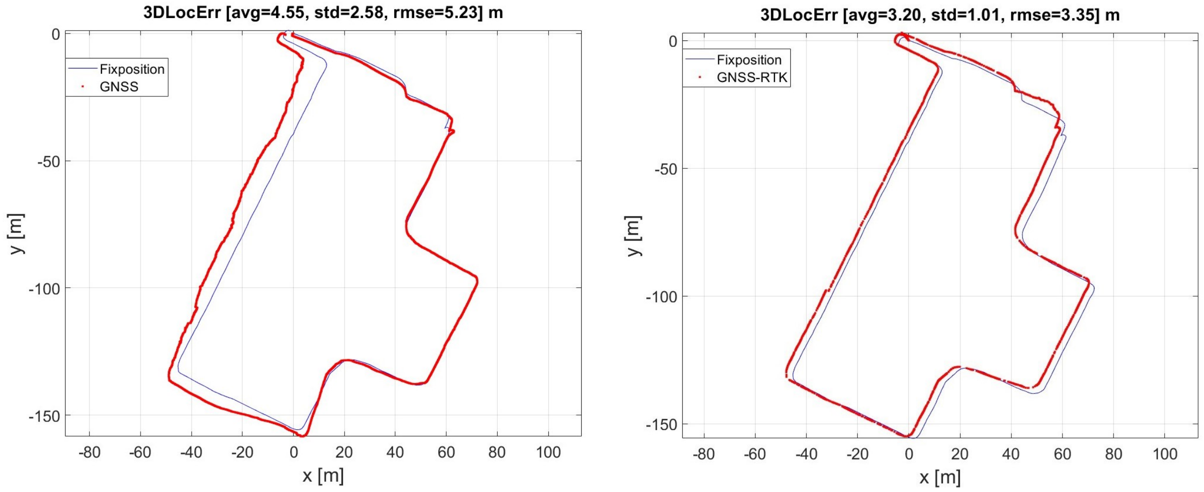
Figure 7.4: Outdoor test location

In this case, we unfortunately could not carry out a 3D experiment because of lack of equipments. The tests were carried out on a relatively flat surface with the beforementioned rover, which has a maximum speed of 1.5 m/s [27].

The figures in the following section are reported in 2D because to maintain clarity and since there was no significant movement in the z-direction. All solutions presented are in ENU frame with the origin being the starting point.

7.2 GNSS/GNSS-RTK Only Positioning

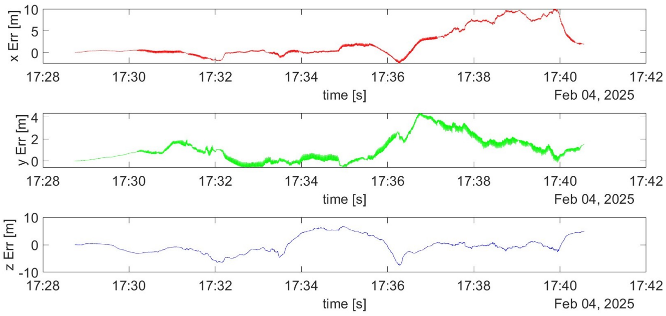
In this section, we will take a look at the GNSS/GNSS-RTK positioning solution. Since both the data for GNSS and GNSS-RTK were captured together , we can consider the following solution as a direct comparison. Figure 7.5a and 7.5b show us the trajectory for each case. Figure 7.6a and 7.6b show us the positional error along x, y and z.



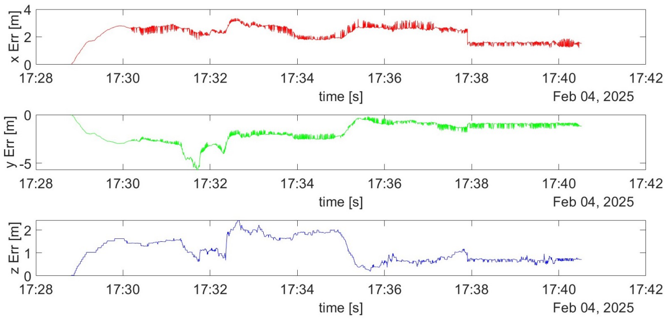
(a) GNSS

(b) GNSS-RTK

Figure 7.5: Comparison of GNSS and GNSS-RTK trajectory



(a) Positional errors for GNSS



(b) Positional errors for GNSS-RTK

Figure 7.6: Comparison of positional errors for GNSS and GNSS-RTK

Pos. Error	**GNSS**	**GNSS-RTK**
errX (m)	3.94	2.31
errY (m)	1.58	2.09
errZ (m)	3.06	1.24

Table 7.1: Comparison of positional errors for GNSS and GNSS-RTK

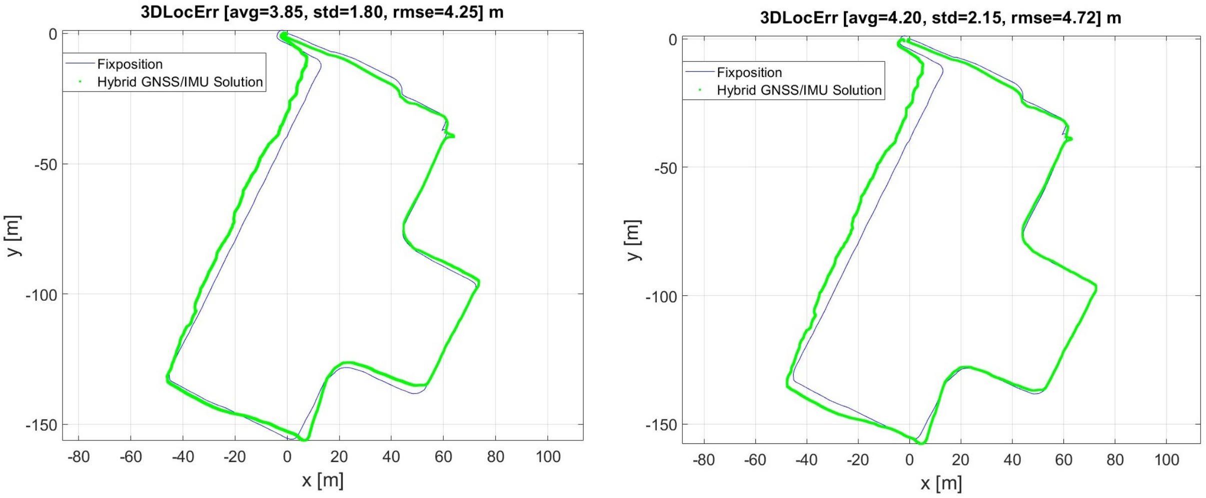
Looking at Figure 7.5 we can clearly see that GNSS-RTK provides us with better results, however not to centimeter level precision. Furthermore, significant data loss was observed during GNSS-RTK acquisition, which is also quite visible in it's trajectory in Figure 7.5b. Table 7.1 further reports the positional errors along x, y and z for both the experiments.

7.3 Hybrid Solution

7.3.1 GNSS/IMU Hybrid Solution

Now let us analyse the hybrid solutions. The best positional performance of the hybrid solution, did not give us the best attitude estimation. To have an eligible attitude estimation for autonomous navigation (errors less than 5°), we had to compromise on positional performance. Hence, the following section proposes two solutions:

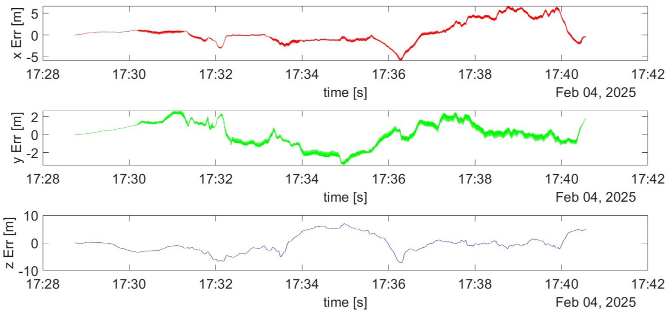
1. Case A: Best positional performance, can be utilized in cases were attitude estimation is not important (e.g. autonomous cars, agricultural vehicles).
2. Case B: Best tradeoff solution, between positional data estimation and attitude estimation (e.g. humanoid robots, UAVs).



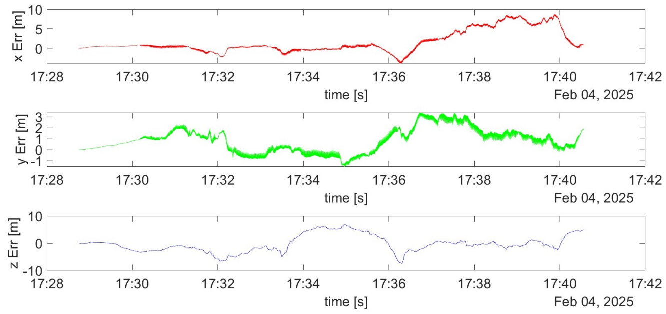
(a) Case A trajectory estimation

(b) Case B trajectory estimation

Figure 7.7: Comparison of the two developed solution

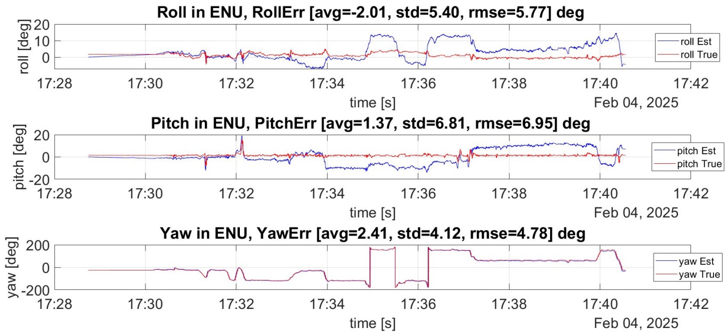


(a) Case A positional errors

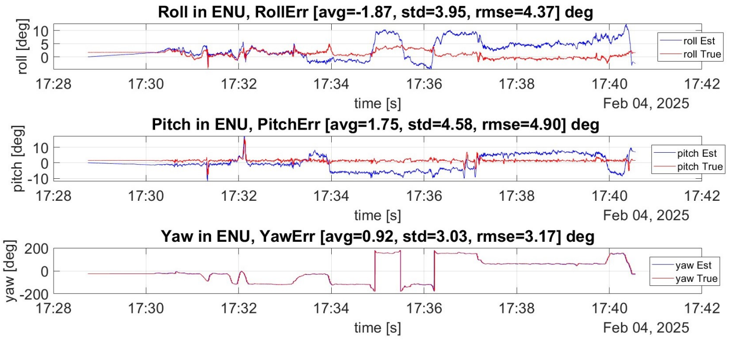


(b) Case B positional errors

Figure 7.8: Comparison of positional errors



(a) Case A attitude estimation



(b) Case B attitude estimation

Figure 7.9: Comparison of attitude estimation

Pos. Error	**Case A**	**Case B**
errX (m)	2.56	3.28
errY (m)	1.32	1.40
errZ (m)	3.14	3.09

Table 7.2: Comparison of positional errors for case A and B

Straightaway, looking at Figure 7.9, we can tell why we need to present two solutions. In case A, roll and pitch angle estimation error is what one would consider too high for autonomous navigation. However, if the euler angle estimation is not of importance as discussed earlier, it does produce a very good positional solution. Figure 7.7a and 7.7b show us the trajectory estimation of case A and case B. In green we have the estimation and in blue we have the ground truth provided by Fixposition. The RMSE of each case is mentioned on top of the figures. Figure 7.8a and 7.8b further illustrate the positional error.

7.3.2 GNSS-RTK/IMU Hybrid Solution

Finally, let's discuss the solution developed for the GNSS-RTK/IMU hybrid solution. As discussed previously, there was significant data loss (around 46%), and so incorporating the IMU data became a lot more challenging. So the focus shifted from having a better solution (like in the case of GNSS/IMU) to having a more consistent solution, which meant we would not see a high percentage of improvement. This is required considering the fact that GNSS-RTK theoretically should have arrived at 10 Hz but due to the loss of data, it was 5.32 Hz, which is not good enough for autonomous localization.

For simplicity, the GNSS-RTK/IMU hybrid solution is called RTK/IMU case. Figure 7.10 shows the estimated trajectory (in green) compared to the ground truth (in blue). As always, the RMSE is mentioned at the top of the figure. Figure 7.11 represents the positional errors along each axis, Table 7.3 summarizes the corresponding error. Finally, Figure 7.12 illustrates the estimated Euler angles (in blue) alongside the ground truth (in red) and the RMSE of the estimation.

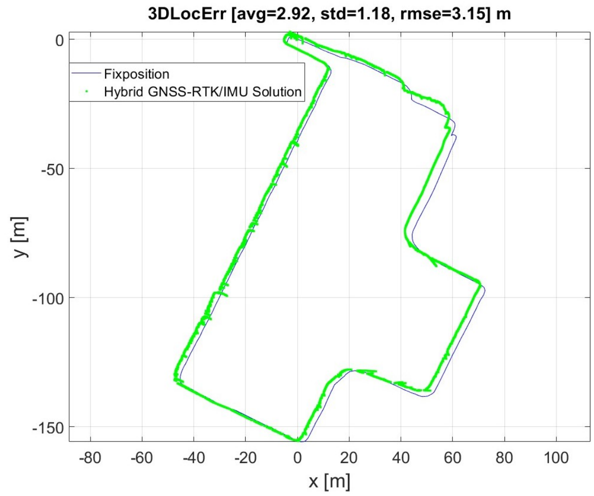


Figure 7.10: RTK/IMU trajectory estimation

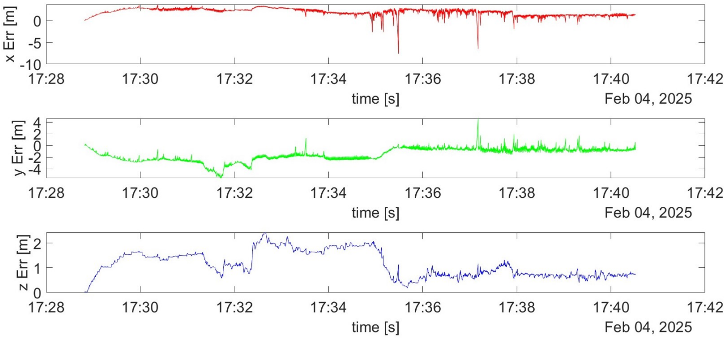


Figure 7.11: RTK/IMU positional errors

Pos. Error	**RTK/IMU**
errX (m)	2.14
errY (m)	1.95
errZ (m)	1.25

Table 7.3: Positional error RTK/IMU case

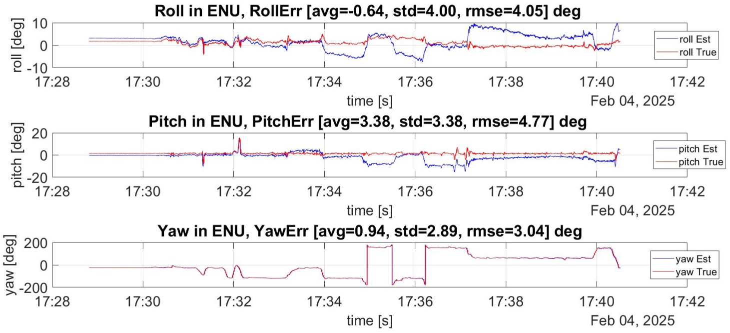


Figure 7.12: RTK/IMU attitude estimation

7.4 Result Discussion and Observations

First, let us compare the solutions obtained using GNSS and GNSS-RTK. GNSS has a RMSE error of 5.23 m GNSS-RTK has a RMSE error of 3.35 m, a 36% improvement, however, GNSS-RTK was not as precise as imagined (centimeter level accuracy) and to top it off it had a 46% loss of data. In fact, the GNSS-RTK solution closely resembles the solution of under-bridge scenario seen in [20]. Since the RTK base station is very close to the experiment location [28], the loss of data and the below par estimation could be due to some reasons like:

1. Multi-Path Interference- Urban area are prone to multi-path interferences due to buildings,vehicles or other surfaces before reaching the receiver. This results in delayed signal reception and correction by the RTK base station and incorrect position estimation
2. Satellite Visibility and Geometry- While in our test, satellite visibility was not an issue, satellite geometry was. Satellite geometry plays a crucial role in GNSS-RTK accuracy. Poor geometry, indicated by a high HDOP, occurs when satellites are clustered together. Ideally, satellites should be well distributed across the sky. In urban environments, buildings may block some satellites, reducing available signals and worsening positioning accuracy. This results in higher HDOP values which are not good for autonomous localization and subsequently degraded RTK performance. Figure 7.13 shows us HDOP distribution for the GNSS-RTK test.

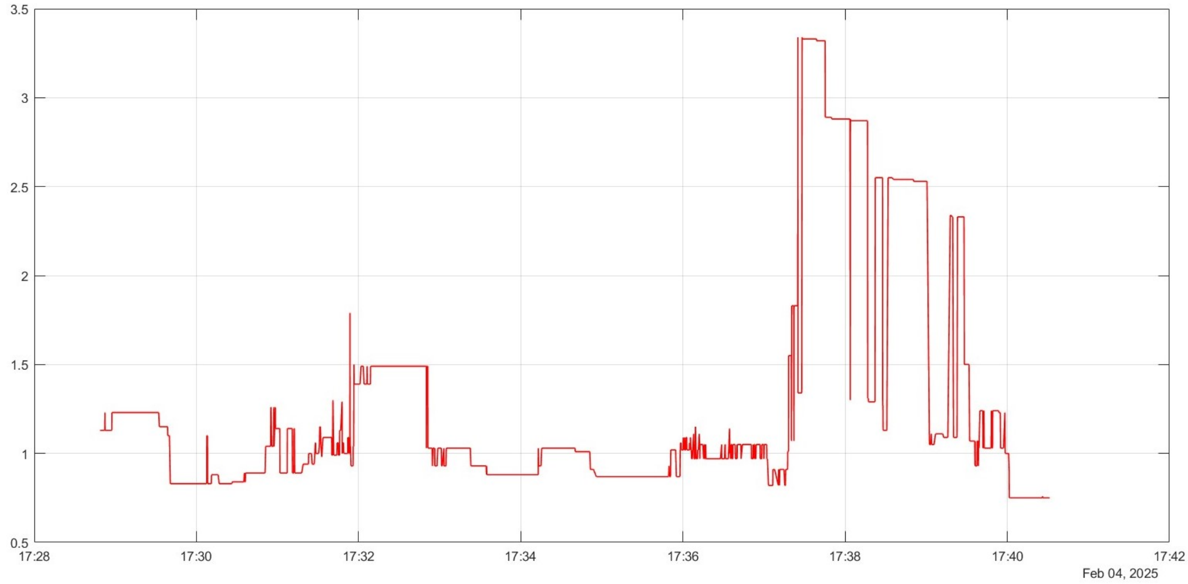


Figure 7.13: RTK HDOP

Now, turning our focus to GNSS/IMU hybrid solution, we saw that euler angle estimation was rather challenging, especially the roll and pitch estimation. In our experiments, there were no rotations about x or y and so the roll and pitch angles should have been relatively small, however as we improved positional estimation, our attitude estimation got worse. So for this reason there were two sets of solution adopted. A summary of the developed solution is described in Table 7.4.

Type	**RMSE (m)**	**Improvement (%)**
GNSS	5.23	-
Case A	4.25	**18.7%**
Case B	4.72	**9.8%**

Table 7.4: Improvement Over Baseline GNSS

When tuning the parameters for GNSS/IMU hybrid solution, it was seen that we could trust the IMU parameters a lot more compared to UWB/IMU hybrid solution in the ESKF and this resulted in a higher percentage of improvement in the solution.

Finally, analyzing the GNSS-RTK/IMU solution was the most challenging of the lot. Due to multiple zones in the trajectory having "gaps", the IMU predictions initially performed well but due to poor bias management, it deteriorated over time and whenever we had delays in receiving the GNSS-RTK update, estimated trajectory deviated significantly from the actual path.

To solve this issue, we could not heavily trust the IMU data in the ESKF, because whenever we did, those deviations only got worse. This meant that we would closely follow the GNSS-RTK solution, allowing us to have a modest 6% improvement from the original solution.

To investigate if the 2D solution leads to similar divergence issues, we obtain the following result:

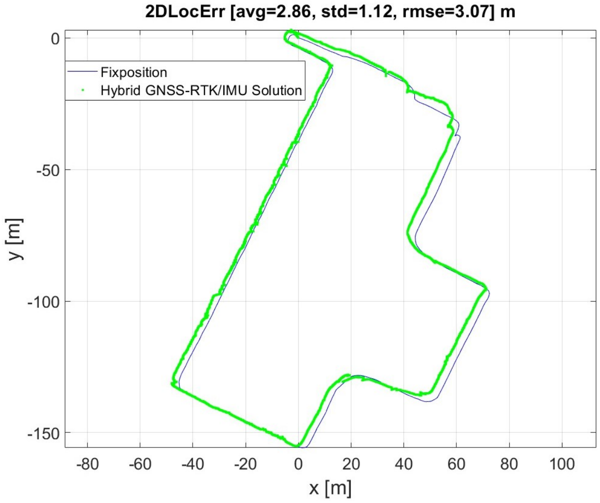


Figure 7.14: 2D RTK/IMU solution

The estimated yaw had an error of 5°. Visually, this solution is much better, however this comes at a cost of not estimating the altitude. This solution could be adopted if and only if the motive of our autonomous vehicle is to move on the ground and there is no significant variations in elevations.

Chapter 8

Conclusion and Further Work

8.1 Conclusion

In this thesis, we explored hybrid solutions integrating UWB/GNSS with an IMU to enhance the accuracy and reliability of AMR navigation in both indoor and outdoor settings. The proposed approach used the ESKF to fuse sensor data with UWB/GNSS, providing high-frequency position and attitude estimation, addressing the limitations of standalone UWB and GNSS solutions.

Through extensive real-world experiments conducted at LINKS Foundation's robotics lab and Politecnico Di Torino, the developed approach proved to be effective in ensuring reliable localization even in dynamic conditions. The results validated the advantages of hybrid localization, showing that the fusion of GNSS/UWB and IMU data provides a more reliable estimation of a robot's position and orientation.

Unfortunately, the hybrid approach did not demonstrate significant improvements over standalone localization methods, falling short of the improvements predicted in simulation. We also found that, unlike simulations, in real-world situations we face a lot more challenges like non-gaussian noise, IMU drift especially during extended gaps in updates from UWB/GNSS, sensor noise and synchronization issues in indoor localization arising from IMU data being collected on a Raspberry Pi while UWB data was processed on a separate PC.

This research contributes to the field of autonomous navigation with the help of hybrid localization techniques. Despite challenges faced during this thesis, this study provides insights into sensor fusion for autonomous robots, highlighting key areas for future improvement in hybrid localization, in the following section.

8.2 Further Work

Before implementing the real-time system, certain modifications to the system design are necessary to ensure a successful deployment.

- Model the IMU- The IMU has time varying biases, which needs to be taken into account when estimating our position and orientation. To deal with this, we can first model the IMU using the Allan variance analysis, more information can be found in [29]. Then we can expand the state of our ESKF from 9 to 15 elements, estimating the accelerometer and gyroscope biases.

- Using a smoothing filter- The ESKF used in this thesis assumes gaussian noise. While for simulations this is viable, in real life experiments it is not. To mitigate this issue, a smoothing like EPF could be used post ESKF stage. In [20] it can be seen that results improved when this was applied.
- Testing on an actual drone- While carrying out this thesis, unfortunately this was not possible due to limitation. However, it would be a good idea to test this because mounting UWB tags and sensors on flying platforms introduces unique challenges which are vibration management, optimal antenna placement during 3D maneuvers etc. In the case of outdoor localization, it becomes even more challenging with more objects to mount.

Bibliography

[1] François Peyret, Pierre-Yves Gilliéron, Laura Ruotsalainen, and Jesper Engdahl. *SaPPART White paper : Better use of Global Navigation Satellite Systems for safer and greener transport*. 2015.

[2] Isaac Skog and Peter Händel. A low-cost gps aided inertial navigation system for vehicle applications. In *2005 13th European Signal Processing Conference*, pages 1–4, Sep. 2005.

[3] Award winning motion capture systems. https://www.vicon.com/, April 2019. Accessed: 2025-3-20.

[4] Fixposition - enabling precise global positioning everywhere ™. https://www.fixposition.com/?srsltid=AfmBOooiXT-U22jn8UAqVkWOueTM6v7IzGKUv7TbhjQfEPOwYpCOZiS7. Accessed: 2025-3-20.

[5] Prabin Kumar Panigrahi and Sukant Kishoro Bisoy. Localization strategies for autonomous mobile robots: A review. *Journal of King Saud University - Computer and Information Sciences*, 34(8, Part B):6019–6039, 2022.

[6] Safar M Asaad and Halgurd S Maghdid. A comprehensive review of indoor/outdoor localization solutions in IoT era: Research challenges and future perspectives. *Comput. Netw.*, 212(109041):109041, July 2022.

[7] Isaac Skog. A low-cost GPS aided inertial navigation system for vehicular applications. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=7febe6014efd0c56368e9cb86ab57ba0491e4913. Accessed: 2025-1-21.

[8] Yuming Yin, Jinhong Zhang, Mengqi Guo, Xiaobin Ning, Yuan Wang, and Jianshan Lu. Sensor fusion of GNSS and IMU data for robust localization via smoothed error state kalman filter. *Sensors (Basel)*, 23(7), April 2023.

[9] Wenjie Zhu, Rongyong Zhao, Hao Zhang, Jianfeng Lu, Zhishu Zhang, Bingyu Wei, and Yuhang Fan. Improved indoor positioning model based on uwb/imu tight combination with double-loop cumulative error estimation. *Applied Sciences*, 13(18), 2023.

[10] Yan Pei, Swarnendu Biswas, Donald Fussell, and Keshav Pingali. An elementary introduction to kalman filtering. *Communications of the ACM*, 62, 10 2017.

[11] Mauricio Caceres, Francesco Sottile, and Maurizio Spirito. Adaptive location tracking by kalman filter in wireless sensor networks. *WiMob 2009 - 5th IEEE International Conference on Wireless and Mobile Computing Networking and Communication*, pages 123–128, 10 2009.

[12] Norziana Jamil, Jacentha Maniam, and Mohamad A Mohamed. Review of indoor localization techniques. *International Journal of Engineering and Technology(UAE)*, 7:201–204, 04 2018.

[13] Huthaifa Obeidat, Wafa Shuaieb, Omar Obeidat, and Raed Abd-Alhameed. A review of indoor localization techniques and wireless technologies. *Wirel. Pers. Commun.*, 119(1):289–327, July 2021.

[14] Junwei Qu. A review of uwb indoor positioning. *Journal of Physics: Conference Series*, 2669:012003, 12 2023.

[15] Mohamed Khalaf-Allah. Novel solutions to the three-anchor ToA-based three-dimensional positioning problem. *Sensors (Basel)*, 21(21):7325, November 2021.

[16] Leehter Yao, Yeong-Wei Andy Wu, Lei Yao, and Zhe Zheng Liao. An integrated imu and uwb sensor based indoor positioning system. In *2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–8, 2017.

[17] Weide You, Fanbiao Li, Liqing Liao, and Meili Huang. Data fusion of uwb and imu based on unscented kalman filter for indoor localization of quadrotor uav. *IEEE Access*, 8:64971–64981, 2020.

[18] Azin Moradbeikie, Ahmad Keshavarz, Habib Rostami, Sara Paiva, and Sérgio Ivan Lopes. GNSS-free outdoor localization techniques for resource-constrained IoT architectures: A literature review. *Appl. Sci. (Basel)*, 11(22):10793, November 2021.

[19] Hyun-Min Cho, Jin-Woo Park, Jung-Soo Lee, and Sang-Kyun Han. Assessment of the GNSS-RTK for application in precision forest operations. *Remote Sens. (Basel)*, 16(1):148, December 2023.

[20] Thanh Trung Duong. The integration of GNSS RTK and IMU with extended particle filter. *Journal of Hydro-meteorology*, 09(20):66–74, September 2024.

[21] Alessandro Moro. Outdoor localization of an assistive autonomous robot. Master's thesis, Politecnico di Torino, December 2021.

[22] `https://www.princeton.edu/~alaink/Orf467F07/GNSS.pdf`. Accessed: 2025-3-31.

[23] Icm-42688-p. `https://invensense.tdk.com/products/motion-tracking/6-axis/icm-42688-p/`, April 2020. Accessed: 2025-3-20.

[24] MathWorks. Ground Vehicle Pose Estimation for Tightly Coupled IMU/GNSS, 2025.

[25] Jungmin Cho and Byoungkil Lee. Optimal layout of four anchors to improve accuracy of Ultra-Wide band based indoor positioning. *Expert Syst. Appl.*, 255(124808):124808, December 2024.

[26] Jose Borras Sillero. Sensor fusion methods for indoor navigation using UWB radio aided INS/DR. `https://www.diva-portal.org/smash/get/diva2:600448/FULLTEXT01.pdf`. Accessed: 2025-3-21.

[27] SCOUT 2.0. `https://global.agilex.ai/products/scout-2-0`. Accessed: 2025-3-26.

[28] Stazione – SPIN3 GNSS. `https://www.spingnss.it/stazione/?id=TORI`. Accessed: 2025-3-26.

[29] Naser El-Sheimy, Haiying Hou, and Xiaoji Niu. Analysis and modeling of inertial sensors using allan variance. *Instrumentation and Measurement, IEEE Transactions on*, 57:140 – 149, 02 2008.