



Master in Aerospace Systems - Navigation and
Telecommunications

Ai-Chi, Chang

ASNAT 23

End of Studies Project Memoir

01/12/2025

"GNSS/IMU sensor fusion integration framework"

Deutsches Zentrum für Luft- und Raumfahrt (DLR)

Enterprise supervisor: Filippo Giacomo Rizzi

Academic supervisor: Jeremy Vezinet

2025

Contents

Acknowledgements	9
Résumé	11
Abstract	13
Reading Guide	15
1 Project Introduction	17
1.1 Deutsches Zentrum für Luft- und Raumfahrt (DLR)	17
1.2 Problem Statement	17
1.3 Request Works	18
1.4 Project Organizations	20
2 Methodology	21
2.1 General Background	21
2.1.1 GNSS	21
2.1.2 INS	21
2.2 State-of-the-art	21
2.3 Frame definition	23
2.3.1 Body frame	23
2.3.2 Local navigation frame	24
2.3.3 Global frame: Earth-Centered Earth-Fixed (ECEF)	25
2.3.4 Lever arm	26
2.4 IMU mechanization	26
2.4.1 Attitude update	26
2.4.2 Velocity update	28
2.4.3 Position update	29
2.5 Kalman filter overview	29
2.5.1 Classification of kalman filter	29
2.5.2 Mathematics behind the kalman filter	31
2.5.3 Observability of kalman filter	32
2.6 Allan deviation	33
3 Implementation	37
3.1 IMU characterization	37
3.1.1 Experiment Setup	37
3.1.2 Analysis of ZED-F9R and STIM300	38
3.2 Data manipulation	40
3.2.1 U-blox ZED-F9R	40
3.2.2 Online-SPP test	42
3.3 Error state kalman filter design (ESKF)	43
3.3.1 Nominal states and discretization	44
3.3.2 Error states and discretization	44
3.3.3 Measurement model and update	46
3.3.4 Error Correction and Reset	49
3.3.5 Initialization	49
3.3.6 Time synchronization	51

3.3.7	Summary and Flow Diagram	52
4	Results and Discussion	53
4.1	Static data	53
4.1.1	Case1: constant velocity model	53
4.1.2	Case2: IMU model + LC/TC	55
4.2	Dynamic data	56
4.2.1	Parameter tuning	57
4.2.2	Case3: PortOut, Quiet sea	58
4.2.3	Case4: OpenSea, Rough sea	60
4.2.4	Discussion of lever arm	62
4.2.5	Ionosphere, troposphere correction estimation	63
5	Conclusion and Future Works	65
	Bibliography	68

List of Figures

1.1	Challenging environment	19
1.2	Project plan	20
2.1	Integration system	22
2.2	z-axis is pointing upward	24
2.3	The NED frame	24
2.4	Different coupling scheme of Kalman filter [1]	30
2.5	Kalman Filter	31
2.6	Illustrate of τ	34
2.7	Allan Deviation plot from [2]	35
3.1	Comparison of u-blox F9R and STIM 300	37
3.2	Experimental setup for static IMU testing	38
3.3	Accumulated absolute position error without aiding	38
3.4	Allan deviation (ADEV) plots of F9R and STIM300	39
3.5	Decode Ephemeris	41
3.6	IMU data description	42
3.7	Decoding workflow	42
3.8	Ephemeris validation by comparing decoded broadcast parameters with those from a reference RINEX file.	42
3.9	Comparison between the implemented SPP solution and the PPP reference from CSRS-PPP.	43
3.10	Time synchronization	51
4.1	Constant Velocity model results ($Q = 0.01$)	54
4.2	Constant velocity LC innovation distribution	54
4.3	STIM 300 dataset	55
4.4	STIM 300 dataset	55
4.5	Dynamic dataset	56

List of Figures

4.7	Artificial GNSS outage	56
4.6	Dynamic IMU dataset	57
4.8	Results of TC with GNSS outage (Quite sea, PortOut)	59
4.9	Error plot for rough sea scenario	60
4.10	Results of TC (Rough sea, OpenSea)	61
4.11	Error plot for rough sea scenario	62
4.12	Discussion of lever arm	63
4.13	Results of TC w/ GNSS outage (rough sea, open-sea)	64

List of Tables

2.1	Typical noise processes and their slopes in Allan deviation plots	34
3.1	Estimated accelerometer and gyroscope biases of F9R and STIM300	39
3.2	Summary of u-blox F9R messages used in this work	40
4.1	Estimated accelerometer and gyroscope biases of F9R and STIM300	53
4.2	Initial State Covariance Parameters	58
4.3	Process Noise Parameters	58
4.4	RMS of LC, TC and SPP (Quite sea)	60
4.5	RMS of LC, TC and SPP (Rough sea)	62

Acknowledgements

Cheers to the days spent at DLR.

First, I would like to express my gratitude to Danial and Andrea, who made this internship possible. The process was tedious and sometimes tough, but without your effort, the story would have been totally different. I truly enjoyed my time at DLR and getting to know everyone there.

I am lucky enough to have received all the patience in the world. A huge thanks to my supervisor Filippo, who has been attentive and supportive for all the time. Always give me piratical advice and the intuition of an engineer directing me not deviate from the path too far so everything is well control under the timeline. The farewell gift is right on my taste. My heartfelt thanks to Michailas, Lucas, Shradha, and Shang Ping, with whom I spent most of my days. You made this internship more colorful, and I will miss all the serious traditions and kindness. The timekeeper, bicycle collector... everyone in the MSS group is truly amazing and unique.

Special thanks to my second supervisor Jeremy for constantly following up on my progress and for being extremely patient. Our discussions were always helpful, and I wish I could learn even more.

Lastly, thanks to my family, my wonderful roommate, and my friends in Toulouse, who accompanied me through some of the lowest moments in my life. Life is always better with you.

Vielen Dank 01.12.2025

Résumé

Les systèmes globaux de navigation par satellite (GNSS) sont largement utilisés pour le positionnement et la navigation, mais leurs performances peuvent être limitées par un faible taux de mise à jour ou par la perte de signal. À l'inverse, les centrales inertielles (IMU) fournissent des mesures à haute fréquence mais dérivent avec le temps. Dans cette thèse, un système de fusion GNSS/INS orienté temps réel est développé à partir des données binaires brutes du récepteur. Deux capteurs inertiels IMU intégrée du u-blox F9R et le STIM300 sont analysés afin d'évaluer l'influence de leur qualité sur les performances de navigation. Un filtre de Kalman à états derreur (ESKF), basé sur les quaternions, est implémenté pour assurer la fusion robuste des capteurs. Le cadre proposé inclut une chaîne complète d'extraction de données permettant de décoder directement les mesures GNSS et IMU sans recourir aux fichiers RINEX. Des essais réalisés sur des données réelles mettent en évidence le comportement du système dans différents scénarios, notamment lors de coupures GNSS temporaires, démontrant sa robustesse et son aptitude au traitement en temps réel.

Abstract

Global Navigation Satellite Systems (GNSS) have become increasingly widespread, with more satellites being launched and better signal availability than ever before. Due to their convenience and accessibility, GNSS technologies are now being used in a wide range of applications, among them, autonomous navigation for vehicles such as cars, ships, and drones. High accuracy and real-time positioning is crucial for these applications which are rapidly growing.

Despite advancements in satellite technologies, GNSS alone still faces inherent limitations: the update rate is relatively low, and signal blockage or multipath effects in urban environments can hinder continuous and reliable positioning. In contrast, inertial sensors are self-contained and provide high-frequency motion data, making them ideal for capturing rapid dynamics. However, they suffer from drift over time due to the accumulation of measurement errors.

To overcome these limitations, this thesis presents an online oriented GNSS and Inertial Navigation System (INS) integration system that directly processes GNSS and IMU raw measurements from the receiver. A quaternion-based Error State Kalman Filter (ESKF) is implemented to perform sensor fusion, combining the advantages of both systems while maintaining numerical stability.

The proposed framework includes a complete data extraction pipeline capable of decoding GNSS and IMU measurements directly from the receivers binary messages, enabling online processing without relying on RINEX files. Experimental validation is carried out using simulated datasets, and the performance is analyzed under various scenarios, including temporary GNSS outages, to assess the robustness of the developed system.

Keywords: GNSS/INS integration, data extraction, ESKF, IMU mechanization

Reading Guide

Given that Kalman Filter (KF) has been studied and developed for decades, and that many open-source implementations are already available, rebuilding the algorithm still remains a complex task. Finding an existing implementation that exactly matches the desired application is often difficult, as the relevant information is scattered across different sources. Therefore, one of the main objectives of this report is to provide a comprehensive understanding of the concept from an implementation-oriented perspective.

Chapter 1 introduces the company and its ongoing project, explaining how these relate to the subject of this thesis and align with the company's objectives. It also explains how the current challenges have led to the identification of certain needs. Based on this context, the chapter outlines the work to be carried out and the expected outcomes.

Chapter 2 presents the state of the art of KF-based techniques, reviewing their evolution in recent years and identifying current research interests related to GNSS/IMU integration. This section also highlights that the KF remains the foundation for most extended algorithms. It then introduces the fundamental coordinate frames, rotation representations, and the use of quaternions, followed by the mechanization equations of the IMU. Subsequently, the integration of GNSS and IMU using the KF is discussed. With respect to the on-line requirements of the system, different coupling strategies (loosely, tightly, and deeply coupled) are analyzed in terms of their respective advantages and limitations. Furthermore, the concept of Allan variance is introduced to evaluate the performance and noise characteristics of the sensors.

Chapter 3 focuses on the implementation of the system. Data extraction part presents how raw binary messages are directly decoded to efficiently obtain the necessary information. The decoding procedure was developed based on the u-blox F9R receiver. Although both GPS and Galileo data can be extracted, only GPS measurements are used in this work. Pseudocode illustrating the realization of a on-line KF is also provided. The Error-State Kalman Filter (ESKF) is then presented, combining and comparing formulations from Joan Sola and Paul Groves [3, 4]. The estimated state variables are defined, and the impact of defining attitude errors in global versus local frames on quaternion multiplication is analyzed. The lever-arm effect is also discussed.

Chapter 4, the results section starts with standalone IMU positioning, supported by Allan deviation analysis to better tune the initial parameters of the filter. It is then followed by the results of both loosely and tightly coupled integrations, using static real data (F9R and STIM 300) and simulated dynamic datasets. These experiments aim to verify whether the developed algorithm can tolerate IMU noise and maintain reliable performance when GNSS measurements are unavailable.

Finally, Chapter 5 wraps up the thesis by highlighting what has been done, the main results and ideas for future improvements.

Chapter 1

Project Introduction

This chapter sets the context of the internship project by highlighting their connection to the objectives of the present work. The main problem to be addressed is then presented, together with the motivation that led to the development of the project. Finally, the planning and timeline that structured the different phases of the internship are summarized.

1.1 Deutsches Zentrum für Luft- und Raumfahrt (DLR)

The German Aerospace Center (DLR) is Germany's national research institution for aerospace, energy, and transportation. Within DLR, the Institute of Communications and Navigation focuses on the development of reliable and secure communication and navigation technologies. The institute operates at two main locations, Oberpfaffenhofen (headquarters) and Neustrelitz, employing around 200 staff members.

Its research is structured around four main missions:

- Global Connectivity – Setting standards in communications for people and machines.
- Global Positioning – Taking accuracy, reliability, and robustness to the next level.
- Autonomy and Cooperation – Creating solutions for transportation and exploration.
- Cyber Security – Protecting society in the digital age through secure radio systems.

Nautical Systems Department, located in Neustrelitz, focuses on maritime applications of Positioning, Navigation, and Timing (PNT) systems. Its research activities are organized into three main working groups: Multi-Sensor-Systems, Maritime Services, and Traffic Systems.

The present work was carried out within the Multi-Sensor-Systems (MSS) working group, which specializes in developing and testing advanced multi-sensor navigation platforms for critical operations. The department operates a variety of experimental facilities, including a measuring vehicle, the research vessel AURORA, and multiple radar and PNT sensor testbeds, providing a comprehensive environment for developing and validating on-line navigation algorithms.

1.2 Problem Statement

Nowadays, global satellite navigation systems (GNSS) are the main source of position, navigation and timing information in multiple domains including maritime. Despite their world-wide avail-

1.3 Request Works

ability and high accuracy services, satellite-based navigation systems can be easily affected by natural impairments, e.g., by ionospheric scintillations, as well as unintentional and intentional man-made interference such as jamming and spoofing. To cope with these issues and provide PNT information to mariners over sea, an alternative terrestrial system known as R-Mode has been developed.

In parallel with the R-Mode initiative, complementary solutions are being investigated to ensure the availability of reliable navigation information when GNSS signals are partially or completely unavailable. One promising approach is the integration of GNSS with inertial navigation systems (INS) including IMU, which can bridge signal outages and enhance system robustness.

Alternatively, terrestrial navigation systems, such as R-Mode [5], can be used to overcome the unavailability of GNSS in regional areas like the Baltic Sea. The requirement of this work is to implement a GNSS/IMU sensor fusion algorithm which can be easily integrated with other sensors or combined with and R-Mode receiver to increase the robustness of overall position PNT system. The software implementation was performed in Python as it provides fast prototyping options with modular and scalable options. The candidate researched and implemented different types of algorithm which were then tested mainly with simulated data.

Reliable navigation requires continuous and accurate PNT information under all conditions. However, no single sensor can meet these requirements in every environment. Understanding the strengths and weaknesses of individual sensors highlights the need for their integration.

- **Global Navigation Satellite System (GNSS):** Provides accurate, drift-free position and velocity information under open-sky conditions. However, its performance degrades in environments with poor satellite visibility, multipath, or intentional interference. In other words, it is least performing when most needed, such as urban canyons or under severe weather conditions in Fig. 1.1.
- **Inertial Navigation System (INS):** INS operates as a self-contained system that provides continuous dynamic information regardless of external signals. It performs well in short-term motion tracking but suffers from drift and accumulated errors over time due to sensor biases and noise.
- **Integration:** By combining GNSS and INS, the complementary strengths of both systems can be exploited. GNSS provides absolute reference information to correct the long-term drift of the INS, while the INS bridges GNSS outages by offering short-term continuity and high-rate motion updates.

The fusion of multiple sensors commonly referred to as multi-sensor integration offers a robust solution for maintaining accurate and reliable navigation in all scenarios. Beyond navigation, this technique can be further applied to advanced domains such as 3D mapping, mobile robotics, and autonomous systems.

1.3 Request Works

The objective of this project is to develop a on-line GNSS/IMU integration system implemented in Python. For future use, the system should be designed to be modular and maintainable, allowing new functionalities, measurement models, or integration with external frameworks such as GNU Radio or ROS to be easily added. In order to achieve this, several key tasks and challenges were identified throughout the project:



Figure 1.1: Challenging environment

- **Data Handling:** Understand the receiver hardware and the types of data transmitted, identifying which messages are essential for navigation and integration purposes. This includes decoding raw binary streams and organizing them into synchronized, time-stamped data structures suitable for on-line and post processing.
- **Research and Algorithmic Development:** Study the fundamental principles of IMU mechanization and the mathematical foundations of the Kalman filter. Investigate different integration architectures such as loosely, tightly, and ultra-tightly coupled and evaluate their suitability under given system constraints and performance requirements.
- **Implementation and Evaluation:** Address key implementation aspects such as time synchronization, IMU noise modeling, and their impact on navigation performance. Design various test scenarios to demonstrate both the strengths and limitations of the developed method and identify potential issues in practical use.
- **Collaborative and Professional Development:** Interact with other teams within the department to explore potential research topics and data-sharing opportunities. Gain familiarity with modular software design, version control (Git), and collaborative development workflows. Additionally, improve technical communication skills, gain exposure to the DLR working culture, and participate in weekly group meeting and monthly department meeting.

Although the project is implementation-oriented, a clear understanding of the theoretical and mathematical aspects remains essential for anyone aiming to continue research in the same direction. A solid theoretical foundation is necessary for future developments.

Within the MSS group, several other ongoing projects address related challenges in positioning, navigation, and sensing. These include radar-based navigation for autonomous or inland-waterway vessels operating under all-weather conditions, R-Mode, LiDAR-SLAM, as well as studies on jamming and spoofing, interference detection, high-precision positioning, integrity monitoring.

Although all sensors are different, they share many common aspects. They aim to solve similar problems: where am I, what does the environment look like, how good is the performance, how is its reliability and accessibility? These common points create opportunities for exchanging ideas, sharing experimental results, and avoiding repeated mistakes. Mutual feedback and comparison across different approaches even fail experience were extremely valuable.

1.4 Project Organizations

At the beginning of the internship, approximately two months were dedicated to reviewing relevant literature and evaluating different integration methods. During this period, a first Python implementation was developed to perform SPP and to decode raw binary data from the u-blox receiver. After comparing different options, the Error-State Kalman Filter (ESKF) was selected as the main fusion framework.

The following month focused on deriving the mathematical formulation of the ESKF and developing a simplified Python prototype. In parallel, static datasets were collected to analyze the characteristics and performance of the IMU sensors. This analysis helped confirm the limitations of inertial navigation when used alone and motivated the integration with GNSS measurements.

Before the mid-internship report, the objective was to implement a loosely coupled GNSS/INS integration in Python. The overall project plan is shown in Fig. 1.2. Progress was presented and discussed weekly during group meetings with the team leader, and additional meetings with the supervisor were held at least once per week. A progress presentation from my side was given every two weeks, providing an opportunity to discuss questions and receive feedback. Both the first-month report and the mid-term internship report served to track progress and refine the project direction when necessary. The thesis is written in last two months, serving both as a record of the development process and as a revisions.

Item	JUN	JUL	AUG	SEP	OCT	NOV
Literature review						
Data handling						
Sensor characterization						
KF algorithm design						
Implementation + debug						
Thesis writing						

Figure 1.2: Project plan

Chapter 2

Methodology

2.1 General Background

2.1.1 GNSS

GNSS provide absolute PNT information by measuring the propagation time of radio signals transmitted by satellites. Modern GNSS constellations, such as GPS, Galileo, GLONASS, and BeiDou, broadcast signals that enable receivers to obtain pseudorange, carrier-phase, and Doppler observations. Modern GNSS constellations transmit signals on multiple frequencies. For example, GPS broadcasts the L1 (1575.42 MHz), L2 (1227.60 MHz) and L5 (1176.45 MHz) frequencies [6], while Galileo provides E1, E5a, E5b, and E6 [7]. Multi-frequency measurements enable the mitigation of ionospheric delay through linear combinations, improve robustness against interference, and enhance overall positioning accuracy.

2.1.2 INS

An INS determines the position, velocity, and attitude of a platform by integrating inertial measurements over time. Modern INS are typically strapdown systems, in which the inertial sensors are rigidly attached to the vehicle body, and the navigation states are obtained through numerical integration of the mechanization equations. The sensing unit used in a strapdown INS is the Inertial Measurement Unit (IMU), which normally consists of tri-axial accelerometers and gyroscopes. The accelerometers measure specific force and the gyroscopes measure angular velocity, providing the fundamental inputs required for the INS computation. Based on their bias stability, noise level, long-term drift, and overall measurement accuracy, IMUs are generally classified into navigation, tactical, industrial, and consumer-grade systems .

2.2 State-of-the-art

Over the past decades, GNSS/IMU integration systems have been widely used to achieve continuous, high-precision positioning. Improvement strategies can generally be classified into two main categories: methods that enhance the GNSS solution itself and methods that improve the inertial navigation subsystem. This section follows this structure to review the corresponding techniques, as illustrated in Fig. 2.1. A similar classification is adopted in [8], where studies are

grouped according to their target problems, such as IMU sensor errors, outliers and integrity violations, or periods of GNSS unavailability.

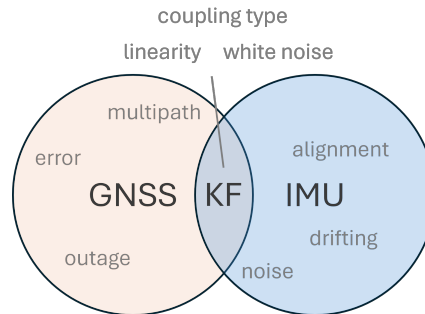


Figure 2.1: Integration system

GNSS-based improvements: Accurately modeling and estimating the different GNSS error sources can significantly improve positioning performance. GNSS errors arise from satellite orbit and clock inaccuracies, tropospheric and ionospheric delays, multipath, receiver noise, and carrier-phase ambiguities. Various strategies exist to mitigate these effects. Precise Point Positioning (PPP) incorporates high-precision satellite orbit, clock, and bias corrections [9]. In multi-antenna configurations, differencing enables cancellation of common errors, as in Real-Time Kinematic (RTK) positioning [10]. Dual-frequency observations allow ionosphere-free combinations, whereas multi-constellation processing and the use of diverse measurement types (code, carrier phase, Doppler) further enhance robustness and accuracy.

IMU-based improvements: From the inertial navigation perspective [11], calibration and preprocessing steps play an important role prior to integration. Different sensor grades have been used to compare the performance. For instance, a tactical-grade IMU was used to demonstrate high accuracy in integrated navigation [12]. As inertial sensors became more affordable, considerable research focused on extending these techniques to low-cost MEMS devices [13], making inertial navigation accessible in consumer applications such as smartphones [14]. Moreover, accurate INS information can improve the reliability of carrier-phase ambiguity resolution in GNSS processing [15].

Integration: The following paragraph reviews methods used for GNSS/IMU fusion systems. From the algorithmic perspective, the Error-State Kalman Filter (ESKF) [3, 4] is widely adopted as an extension of the classical Extended Kalman Filter (EKF). Alternative formulations such as the Unscented Kalman Filter (UKF) [16] have been proposed to better handle nonlinearities.

Another major design choice concerns the integration strategy between GNSS and IMU. Different coupling schemes loosely coupled, tightly coupled, and ultra-tight aim to exploit measurement availability at different levels and compensate for the limitations of individual sensors [17]. Sequential Kalman filtering techniques [18] have also been introduced to reduce computational cost in tightly coupled architectures.

Beyond traditional filtering, learning-based approaches have been explored to maintain navigation continuity during GNSS outages by capturing nonlinear motion patterns. More recently, the Factor Graph Optimization (FGO) framework [19] has gained attention, formulating sensor fusion as a graph-based optimization problem over a sliding window and enabling joint estimation across multiple epochs.

Various multi-sensor integration approaches have also been proposed to improve positioning robustness in challenging environments by combining GNSS with other sensors such as IMUs, cameras, and SLAM. Different coupling and filtering strategies can be selected based on computational efficiency and the intended application. Overall, these studies provide valuable insights and guidance on choosing suitable integration methods under different operational constraints.

To address our objectives, we adopt a tightly focused subset of these methods that is most suitable for future online processing on low-cost hardware. A quaternion-based Error-State Kalman Filter (ESKF) is implemented due to its numerical stability and its widespread use in GNSS/IMU fusion. This choice is motivated by its balance of accuracy and computational efficiency. Despite the concepts introduced above, the following sections presents the concepts that are essential for our project.

2.3 Frame definition

Before presenting any sensor fusion or state estimation algorithm, it is crucial to define the reference frames used to describe motion and perform calculations.

In this section, we introduce the coordinate frames used in this work and explain how they relate to our implementation. In depth details can be found in [3, 12].

2.3.1 Body frame

This frame describes how the object moves without specifying where it is on Earth. Its origin is fixed to the sensor, and the frame moves together with the object. It is denoted by the superscript b . The body frame is typically defined according to the right-hand rule. For most sensors or vehicles, the axes are defined as:

- x -axis: along the forward (or along-track) direction of motion,
- y -axis: pointing to the right side of the object,
- z -axis: pointing downward or upward, in the direction of gravity.

In some cases, depending on how the IMU is installed, the y -axis may point to the left and the z -axis upward instead. Both definitions are valid as long as they form a right-handed coordinate system. Fig. 2.2 use u-blox F9R for example, the z -axis is pointing upward from the top of the module.

The accelerometer measures the *specific force* \mathbf{f} , which represents the non-gravitational acceleration acting on the sensor:

$$\mathbf{f} = \mathbf{a} - \mathbf{g} \quad (2.1)$$

where \mathbf{a} is the sensors linear acceleration and \mathbf{g} is the local gravity vector.

When the device is static on the ground, $\mathbf{a} = 0$, and the measured specific force equals $-\mathbf{g}$. If the accelerometer reading on the z -axis is approximately -9.81 m/s^2 , it indicates that the z -axis is pointing downward, meaning the frame follows the forward-right-down definition. Conversely, if the measured value is around $+9.81 \text{ m/s}^2$, the z -axis points upward, corresponding to a forward-left-up configuration. In the following discussion, the forward-right-down definition is adopted.

2.3 Frame definition

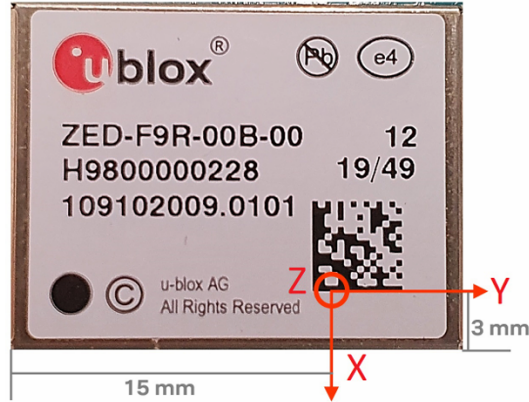


Figure 2.2: z-axis is pointing upward

2.3.2 Local navigation frame

A local coordinate system fixed to the Earth's surface, used to describe position, velocity, and attitude in directions that make sense to humans such as north, east, and down. The most common choice is the North-East-Down (NED) frame as in Fig. 2.3, where:

- The x -axis points to geographic north,
- The y -axis points to east,
- The z -axis points downward, toward the center of the Earth.

Another common option is the East-North-Up (ENU) frame.

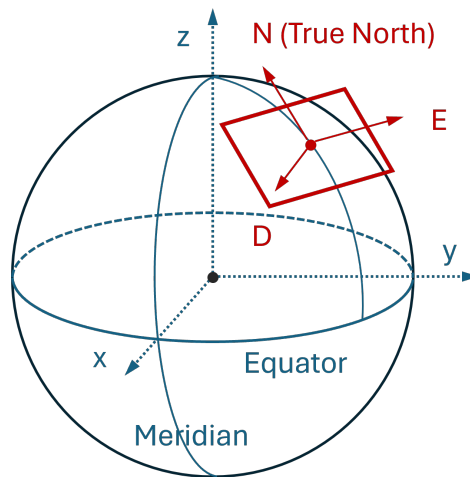


Figure 2.3: The NED frame

This frame does not provide global positioning information (like latitude and longitude), but it gives local directional context. For example, indicating which way is north or east relative to the initial location. It is typically defined at a reference point, forming a tangent plane to the Earth's surface, and it moves with the user.

For example, if the forward-right-down body frame is initially aligned with the NED local navigation frame, the rotation from body to navigation frame \mathbf{C}_b^n is identity matrix, and the

gravity vector in the navigation frame is approximately:

$$\mathbf{g}_n \approx \begin{bmatrix} 0 \\ 0 \\ 9.81 \end{bmatrix} \text{ m/s}^2$$

while the specific force measured by the accelerometer (in the body frame) should be:

$$\mathbf{f}_b \approx \begin{bmatrix} 0 \\ 0 \\ -9.81 \end{bmatrix} \text{ m/s}^2$$

Together, these satisfy the expected static condition:

$$\mathbf{C}_b^n \mathbf{f}_b + \mathbf{g}_n = \mathbf{0}$$

Furthermore, when motion occurs in a small area and over a short duration (such as in land vehicles, handheld devices, or drones), it is common to neglect the effects of earth rotation i.e. Coriolis force, and centrifugal acceleration [12].

2.3.3 Global frame: Earth-Centered Earth-Fixed (ECEF)

The ECEF frame is a global Cartesian coordinate system that rotates with the Earth. It is denoted by the superscript e . Its origin is located at the Earth's center of mass, and its axes are fixed relative to the Earth's surface meaning the frame rotates along with the planet.

From our perspective on the ground, it is ideal for describing the motion of objects relative to the Earth, such as satellites, GNSS receivers, or aircraft. Specifically:

- The x -axis points toward the intersection of the equator and the prime meridian,
- The y -axis points east along the equator (90° from the x -axis),
- The z -axis points toward the North Pole.

In high-precision applications, the effects of the earth rotation must be taken into account. However, for lower-grade IMUs or in simplified implementations, these effects can often be ignored, as their impact is small compared to the overall sensor noise level.

By using the static case we can also make sure that we have all the coordinates consistent:

$$\mathbf{C}_b^e \mathbf{f}_b + \mathbf{C}_n^e \mathbf{g}_n = \mathbf{0}$$

Although ECEF coordinates are useful for describing global positions, they are not very intuitive for interpreting local motion. Therefore, GNSS outputs are often converted into geodetic coordinates latitude, longitude, and height (LLA/LLH). Once the receiver's latitude (φ) and longitude (λ) are known, the conversion between the ECEF frame and the local navigation frame can be computed by using the following rotation matrices.

For the NED frame:

$$\mathbf{C}_{\text{ned}}^e = \begin{bmatrix} -\sin \varphi \cos \lambda & -\sin \varphi \sin \lambda & \cos \varphi \\ -\sin \lambda & \cos \lambda & 0 \\ -\cos \varphi \cos \lambda & -\cos \varphi \sin \lambda & -\sin \varphi \end{bmatrix} \quad (2.2)$$

2.4 IMU mechanization

For the ENU frame:

$$\mathbf{C}_{\text{enu}}^e = \begin{bmatrix} -\sin \lambda & \cos \lambda & 0 \\ -\sin \varphi \cos \lambda & -\sin \varphi \sin \lambda & \cos \varphi \\ \cos \varphi \cos \lambda & \cos \varphi \sin \lambda & \sin \varphi \end{bmatrix} \quad (2.3)$$

Note that \mathbf{C}_b^e , the rotation from body frame to ECEF, must be estimated (e.g., via ESKF or another estimation filter). Thus the rotation of the object in the local navigation frame \mathbf{C}_b^n can be computed from:

$$\mathbf{C}_b^e = \mathbf{C}_n^e \mathbf{C}_b^n$$

Finally, in our ESKF algorithm, the IMU measurements are provided in the body frame. To incorporate GNSS data, we work in the ECEF frame, which is the natural coordinate system for satellite positioning. The final estimated states (position, velocity, and attitude) are typically analyzed in a local navigation frame such as NED or ENU, and visualized on maps using geodetic coordinates LLA/LLH. In next section, we will explain how body-frame measurements are integrated and transformed into the ECEF frame for fusion with GNSS observations.

2.3.4 Lever arm

When the IMU and GNSS antenna are mounted at different locations on the platform, the pseudorange and carrier phase measurements originate from the antenna, while the motion dynamics are observed at the IMU. The lever arm correction must be applied, since the spatial offset between them introduces a discrepancy that must be compensated.

This offset, is typically expressed in the body frame and denoted as \mathbf{l}_b and represents the vector from the IMU to the GNSS antenna. The lever arm affects the measurement model, particularly the design of the observation matrix \mathbf{H} in Kalman filter-based estimators, which will be explained in details in later sections.

2.4 IMU mechanization

IMUs play a crucial role in integrated navigation systems by providing measurements of angular rate and specific force, which can be integrated to obtain attitude, velocity, and position. In practice, however, the accuracy of this mechanization strongly depends on the quality of the IMU and its calibration. The accelerometers and gyroscopes are affected by bias, scale factor errors, random noise, and possible misalignment between sensor axes, while the initial attitude can only be roughly estimated. Therefore, proper calibration and sensor configuration are required before the usage. Understanding these error characteristics is essential before integrating the IMU into the navigation filter. Practical recommendations for the experimental setup will be elaborated in later chapter.

In this section, the mechanization equations of an IMU are first introduced, followed by the characterization of two IMUs, the STIM300 and the internal IMU of a ublox F9R GNSS receiver.

2.4.1 Attitude update

The first step of the mechanization is the attitude update, since the accelerometer outputs are expressed in the body frame. To obtain a physically meaningful trajectory—that is, motion with respect to a fixed reference point on Earth—the specific force must be transformed into a

navigation frame. Then, the acceleration can be correctly projected onto the three axes of the navigation frame.

Attitude is the orientation of the body frame with respect to a chosen reference frame. The rotation matrix provides a mathematical representation of this attitude by specifying how each body axis is oriented relative to the reference frame. In this thesis, the rotation matrix is denoted by \mathbf{C} in order to avoid confusion with the symbol \mathbf{R} , which is later used for the measurement covariance matrix.

The relationship between the rotation matrix and the angular rate can be written in the time-derivative form, showing that a small change in attitude is obtained by pre-multiplying the current attitude by the instantaneous angular rate ω_{eb}^b , which represents the body rotation rate with respect to the Earth frame and is parameterized in the body frame.

$$\dot{\mathbf{C}}_b^e = \mathbf{C}_b^e [\omega_{eb}^b]_{\times} = \mathbf{C}_b^e [\omega_{ib}^b]_{\times} - [\omega_{ie}^e]_{\times} \mathbf{C}_b^e, \quad (2.4)$$

Here, ω_{ib}^b is the angular rate measured by the gyroscope, and ω_{ie}^e denotes the Earth rotation rate, with $\omega_{ie} \approx 7.292115 \times 10^{-5}$ rad/s. The skew-symmetric operator $[\cdot]_{\times}$ is defined as

$$\omega = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}, \quad [\omega]_{\times} = \Omega = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}, \quad \omega_{ie}^e = \begin{bmatrix} 0 \\ 0 \\ \omega_{ie} \end{bmatrix}. \quad (2.5)$$

Equivalently, the continuous quaternion kinematics (using the right-multiplicative convention) are given by

$$\dot{\mathbf{q}} = \frac{1}{2} \mathbf{q} \otimes \omega \quad \text{or} \quad \dot{\mathbf{q}} = \frac{1}{2} [\mathbf{q}]_R \omega, \quad (2.6)$$

where the right quaternion product matrix is defined as

$$[\mathbf{q}]_R = \begin{bmatrix} q_w & -q_x & -q_y & -q_z \\ q_x & q_w & q_z & -q_y \\ q_y & -q_z & q_w & q_x \\ q_z & q_y & -q_x & q_w \end{bmatrix}. \quad (2.7)$$

The angular rate used here is corrected for the Earth's rotation as follows:

$$\omega = \omega_{eb}^b = \omega_{ib}^b - \mathbf{C}_e^b \omega_{ie}^e. \quad (2.8)$$

Next, when it comes to the implementation, we need the discrete form. It can be derived by assuming that the angular rate remains constant and integrating the continuous differential equation over a short time interval Δt . Both the direction cosine matrix (DCM) and the quaternion representation can be used to perform the attitude update, and each can be interpreted as a single rotation about one instantaneous axis.

First, integrating the rotation matrix form in (2.4) gives

$$\mathbf{C}_b^e(t + \Delta t) = \mathbf{C}_b^e(t) \exp([\omega_{ib}^b \Delta t]_{\times}) - \left[\exp([\omega_{ie}^e \Delta t]_{\times}) - \mathbf{I} \right] \mathbf{C}_b^e(t). \quad (2.9)$$

Let $\phi = \omega \Delta t$, applying the small angle approximation by truncating the exponential at first order

$$\exp([\phi]_{\times}) = (\mathbf{I} + [\phi]_{\times}) \quad (2.10)$$

And assuming that the IMU angular-rate measurement is constant over the integration interval and $a[\mathbf{x}]_{\times} = [a\mathbf{x}]_{\times}$, the update simplifies to

$$\mathbf{C}_b^e(k+1) \approx \underbrace{\mathbf{C}_b^e(k)(\mathbf{I} + [\omega_{ib}^b]_{\times} \Delta t)}_{\text{body rotation}} - \underbrace{[\omega_{ie}^e]_{\times} \mathbf{C}_b^e(k) \Delta t}_{\text{Earth rotation}}. \quad (2.11)$$

Equivalently, in quaternion form,

$$\begin{aligned} \mathbf{q}_{k+1} &= \mathbf{q}_k \otimes \delta \mathbf{q} \\ \delta \mathbf{q} &= \mathbf{q} \{ \omega \Delta t \} = \begin{bmatrix} \cos\left(\frac{\theta_k}{2}\right) \\ \mathbf{u} \sin\left(\frac{\theta_k}{2}\right) \end{bmatrix}. \end{aligned} \quad (2.12)$$

In our implementation, the quaternion-based update is adopted, as it does not rely on the small-angle assumption required by the DCM update and is less prone to numerical drift that gradually violates the orthogonality constraint of the rotation matrix.

It is important to stay consistent with the multiplication convention when updating quaternions. Because the change in orientation is a physical process, it is always represented by right multiplication [4]. In the later Kalman filter chapter, we will see that whether right or left multiplication is applied depends on how the error is defined.

2.4.2 Velocity update

The measurement we got from IMU is the specific force \mathbf{f}_{ib}^b . The gravity (including gravitational and centrifugal accelerations) \mathbf{g}^e needs to be compensated, and the Coriolis term also needs to be considered. So, the continuous form of the acceleration is

$$\dot{\mathbf{v}}_{eb}^e = \mathbf{a}_{eb}^e = \mathbf{f}_{ib}^e + \mathbf{g}^e - \underbrace{2[\omega_{ie}^e]_{\times} \cdot \mathbf{v}_{eb}^e}_{\text{Coriolis}} \quad (2.13)$$

Since

$$\mathbf{f}_{ib}^e = \mathbf{C}_b^e(t) \mathbf{f}_{ib}^b \quad (2.14)$$

$$= \frac{1}{2}(\mathbf{C}_b^e(k+1) + \mathbf{C}_b^e(k)) \mathbf{f}_{ib}^b \quad (2.15)$$

$$= \mathbf{C}_b^e(k) \left(\mathbf{f}_{ib}^b + \frac{1}{2}[\Delta\theta]_{\times} \mathbf{f}_{ib}^b \right) \quad (2.16)$$

with first order approximation

$$\mathbf{C}_b^e(k+1) \approx \mathbf{C}_b^e(k)(\mathbf{I} + [\Delta\theta]_{\times}) \quad (2.17)$$

Thus the implementable form in the ECEF frame is

$$\mathbf{v}_{k+1} = \mathbf{v}_k + \mathbf{a}_{eb}^e \Delta t. \quad (2.18)$$

2.4.3 Position update

The continuous form of the position is

$$\dot{\mathbf{r}}_{eb}^e = \mathbf{v}_{eb}^e. \quad (2.19)$$

The implementable form in the ECEF frame is

$$\mathbf{r}_{k+1} = \mathbf{r}_k + \frac{1}{2}(\mathbf{v}_k + \mathbf{v}_{k+1})\Delta t. \quad (2.20)$$

Notably, due to the presence of bias, both the specific force and the angular rate need to be corrected by subtracting their estimated biases. All equations above will be used in the Kalman filter.

2.5 Kalman filter overview

The Kalman filter is a widely used approach for solving sensor fusion problems. It provides a statistical framework to continuously estimate the system state by balancing two sources of information: predictions based on a dynamic model and corrections from noisy sensor measurements. This balance is achieved by assigning confidence weights to each source depending on its estimated uncertainty. Such an approach is especially useful in real-world navigation applications, where GNSS signals may be temporarily lost, sensors may drift, or unexpected disturbances can occur. By adaptively blending model and measurement information, the Kalman filter helps maintain robust and accurate state estimates even under challenging conditions.

2.5.1 Classification of kalman filter

Depending on the system dynamics, the standard Kalman Filter can be applied to linear systems, while the Extended Kalman Filter (EKF) is typically employed for nonlinear systems by linearizing the measurement and process models around the current state estimate. Another alternative is the Unscented Kalman Filter (UKF). Instead of linearizing with Jacobians, the UKF propagates a set of deterministically chosen sigma points through the nonlinear system. These sigma points approximate the state distribution, allowing the UKF to capture the mean and covariance more accurately than a first-order linearization, which can improve estimation performance in highly nonlinear problems such as high-dynamic vehicle maneuvers or spacecraft attitude estimation, though at the cost of increased computational effort [20].

In the context of GNSS/IMU integration, the EKF can be classified into three main architectures: loosely coupled, tightly coupled, and ultra-tightly coupled.

Each architecture differs in how the GNSS measurements are integrated with the IMU data:

- **Loosely coupled (LC):** As illustrated in Fig. 2.4(a), the GNSS receiver provides standalone position and velocity estimates. These are then used as direct measurements in the EKF to correct the IMU-propagated states. This architecture does not combine raw sensor data; rather, it fuses two independent navigation solutions at the filter level.
- *Measurement input:* GNSS position and velocity.

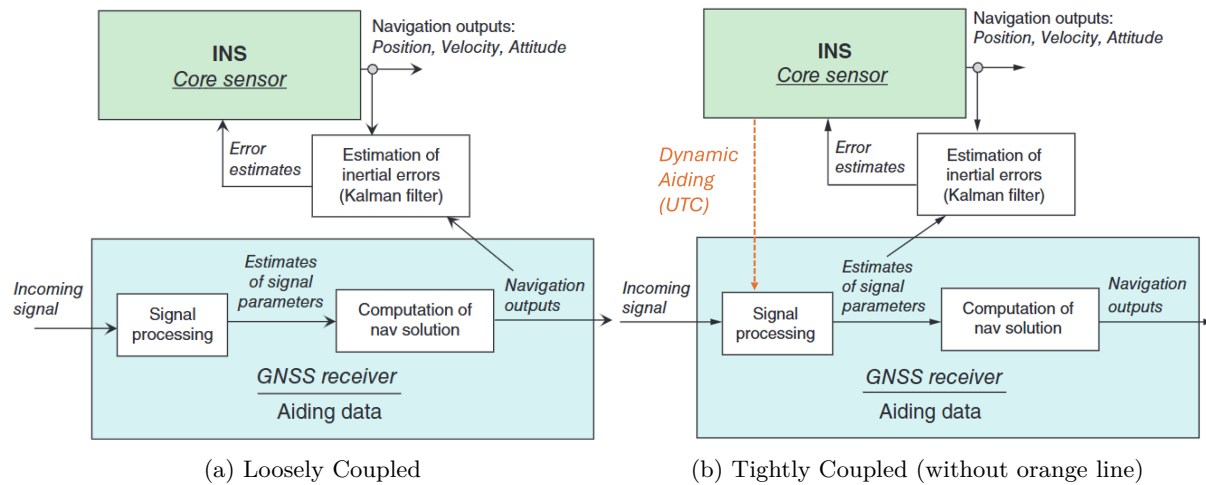


Figure 2.4: Different coupling scheme of Kalman filter [1]

- Advantages:** simple to implement; works with off-the-shelf GNSS receivers; low computational load.
- Disadvantages:** requires at least four visible satellites; performance degrades significantly in GNSS-challenged environments.
- Tightly coupled (TC):** In this scheme, shown in Fig. 2.4(b) without the orange line, raw GNSS measurements such as pseudorange and Doppler are directly incorporated into the EKF, together with the IMU data. This allows the filter to operate even when fewer than four satellites are visible, as long as enough raw observations exist.
 - Measurement input:** such as pseudorange, carrier phase, Doppler.
 - Advantages:** more robust in urban environments or under satellite blockage; allows measurement noise to be modeled individually for each satellite and signal type, improving estimation accuracy.
 - Disadvantages:** requires access to raw GNSS measurements; more complex to implement; increased computational demand.
- Ultra-tightly coupled (UTC):** The scheme is shown Fig. 2.4(b) with orange line. This architecture goes one step further by integrating IMU data directly into the GNSS tracking loops (rather than just the navigation filter). The IMU aids in predicting Doppler and code phase, improving robustness in degraded signal conditions and reducing signal reacquisition time.
 - Measurement input:** GNSS tracking loop outputs (code, carrier, Doppler), aided by IMU prediction.
 - Advantages:** significantly more resilient to signal outages or weak signals; enables faster reacquisition after loss.
 - Disadvantages:** requires low-level access to GNSS receiver internals; high implementation complexity; often unavailable in commercial hardware.

In practice, tightly coupled integration is often preferred for advanced navigation systems, especially in environments where GNSS signals are frequently blocked or degraded. However, loosely

coupled architectures remain widely used in commercial applications due to their simplicity and compatibility with off-the-shelf GNSS receivers.

2.5.2 Mathematics behind the kalman filter

The Kalman Filter is a minimum mean-square error estimator (MMSE). It assumes that the system can be represented by a linear model with additive Gaussian noise.

The core idea of the Kalman Filter is to dynamically determine the optimal weighting between the model prediction and the measurement, based on their statistical characteristics (the process and measurement noise covariances). By doing so, it produces an estimate that is statistically optimal in the sense of minimizing the mean squared estimation error. The overall procedure is as Fig. 2.5, and the following outlines its general procedure.

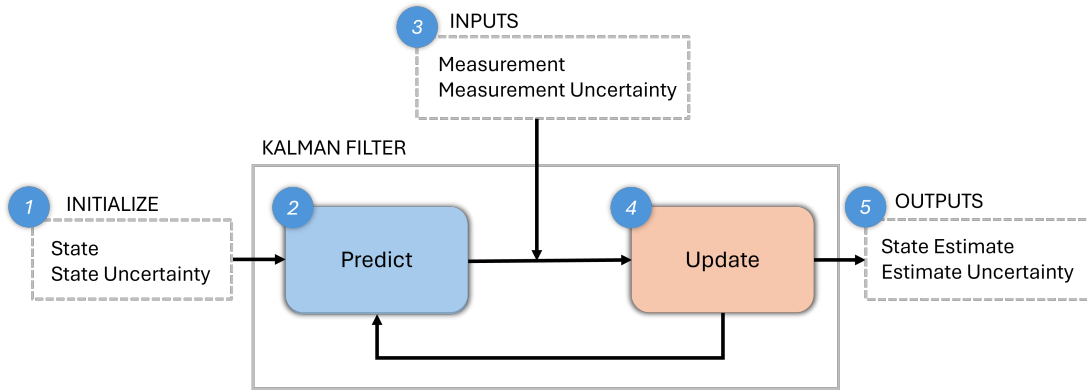


Figure 2.5: Kalman Filter

1. Initial value

The process starts with an initial guess of the state vector \mathbf{x}_0 and its associated covariance matrix \mathbf{P}_0 . These values represent our prior knowledge of the system and its uncertainty before any measurements are processed, in other words, how much we trust our initial values.

2. State prediction and covariance propagation

In this step, information from the current epoch k is used to predict the state at the next epoch $k + 1$. A discrete state-space model can be written as

$$\mathbf{x}_{k+1} = \mathbf{F} \mathbf{x}_k + \mathbf{G} \mathbf{u}_k + \mathbf{w}_k, \quad (2.21)$$

where \mathbf{F} is the state transition matrix, \mathbf{G} is the control-input matrix, \mathbf{u}_k represents any known control input, and \mathbf{w}_k denotes the process noise. If today we define our state to be the error state, then \mathbf{F} will be the error state transition matrix.

In this formulation, the process noise represents imperfections in the dynamic model, such as IMU bias drift or integration errors. In practice, different motion models can be used depending on the application. For example, a constant-velocity model can be adopted for ground vehicles, or a stationary model can be used when the platform is assumed static.

Since the model is not perfect, we must also account for its uncertainty by propagating the covariance matrix \mathbf{P} :

$$\mathbf{P}_{k+1} = \mathbf{F} \mathbf{P}_k \mathbf{F}^\top + \mathbf{Q}, \quad (2.22)$$

2.5 Kalman filter overview

It comes from the expected squared deviation $\mathbf{E}[(\mathbf{x} - \hat{\mathbf{x}})((\mathbf{x} - \hat{\mathbf{x}})^T]$ which represents the covariance between the true state \mathbf{x} and its estimate $\hat{\mathbf{x}}$. The process noise covariance matrix \mathbf{Q} characterizes the uncertainty of the system model [12]. A larger \mathbf{P} indicates higher uncertainty in the predicted state (we trust our model less), while a smaller \mathbf{P} indicates greater confidence in the model prediction.

3. Get measurement

The measurement model is formulated below, When new sensor data are available, such as GNSS position or pseudorange measurements, they are collected into a measurement vector \mathbf{y}_k and compared against the predicted measurement $h(\hat{\mathbf{x}}_k)$ from the model.

$$\mathbf{y}_{k+1} = \mathbf{H}\mathbf{x}_{k+1} + \mathbf{v}_{k+1} \quad (2.23)$$

4. Update state

The update step refines the predicted state using the newly received measurements. The Kalman gain \mathbf{K} is computed to balance the relative confidence between the prediction and the measurement:

$$\begin{aligned} \mathbf{S} &= \mathbf{H}\mathbf{P}_{k+1}\mathbf{H}^T + \mathbf{R}, \\ \mathbf{K} &= \mathbf{P}_{k+1}\mathbf{H}^T\mathbf{S}^{-1}. \end{aligned} \quad (2.24)$$

Here, \mathbf{R} is the measurement noise covariance matrix, which characterizes the expected uncertainty in the sensor measurements (e.g., GNSS pseudorange or velocity noise). The matrix \mathbf{S} is the innovation covariance, which represents the total expected uncertainty in the residual combining both the uncertainty in the predicted state (projected into the measurement space via \mathbf{H}) and the measurement noise. A larger \mathbf{R} indicates less trust in the measurement, leading to a smaller Kalman gain.

The state and covariance are then updated as

$$\hat{\mathbf{x}}_{k+1} = \hat{\mathbf{x}}_k + \underbrace{\mathbf{K}(\mathbf{y}_{k+1} - h(\hat{\mathbf{x}}_{k+1}))}_{\text{innovation}}, \quad (2.25)$$

$$\mathbf{P}_{k+1} = (\mathbf{I} - \mathbf{K}\mathbf{H})\mathbf{P}_{k+1}. \quad (2.26)$$

After the update, the covariance matrix \mathbf{P} typically decreases, indicating that the uncertainty of the state estimate has been reduced. This means the filter has become more confident in its estimate after incorporating the measurement information.

2.5.3 Observability of kalman filter

A simple example below can show the concept of observability of Kalman filter. States 1, 3, 5 can be observed, 2, 4 can not.

$$y_n = \mathbf{H}x_n + v_n = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} + v_n = \begin{bmatrix} x_1 \\ x_3 \\ x_5 \end{bmatrix} + v_n \quad (2.27)$$

In our GNSS/IMU integration system, most elements of the \mathbf{H} matrix are zeros, since only the pseudorange observations are used in the update step. If doppler measurements are included, the velocity states can also become observable. Apart from the effect of the measurement type, the motion of the platform also limits the system observability. For instance, with a single-antenna receiver, it is fundamentally impossible to directly observe the yaw angle. However, since the position is obtained by integrating the velocity, the errors caused by initial attitude misalignment or accelerometer biases may gradually propagate in the position error, as shown in Eq. (2.13), and more clearly in Eq. (3.4). During static or constant-speed straight-line motion, these errors remain unobservable until the platform changes its motion, such as during turns or acceleration, which provide additional information for the filter. Using a dual-antenna baseline enables direct observation of the attitude error, effectively resolving this limitation. Alternatively, magnetometer measurements can also help constrain the yaw angle through the known direction of the Earth's magnetic field, although their accuracy depends on the magnetic environment and sensor calibration.

2.6 Allan deviation

Allan deviation (ADEV) is a technique used to analyze errors in devices such as oscillators, accelerometers, and gyroscopes. Its concept is closely related to the power spectral density (PSD), as expressed in Eq. 2.28. While the PSD describes how noise is distributed over frequency, Allan deviation shows this information in the time domain, letting the user see how the noise behaves over different averaging times.

$$\sigma_y^2(\tau) = \int_0^\infty S_y(f) \cdot |H(f, \tau)|^2 df \quad (2.28)$$

The physical meaning of Allan variance is to quantify the sensors noise instability at a given averaging time τ . $\sigma(\tau)$ represents the standard deviation of the difference between consecutive τ -second averaged measurements, how differ they are, thereby revealing the dominant noise processes acting at different time scales.

In general, the main error sources in such devices include white noise, bias instability, and random walk. These components exhibit distinct signatures in the Allan deviation (ADEV) curve, which makes ADEV a widely used tool for identifying the noise characteristics of inertial sensors.

To apply this analysis in practice, the continuous-time Allan variance must be computed in its discrete form. In our work, we use the Python package `allantools`, which provides a direct implementation of the overlapping Allan deviation (OADEV), a more statistically efficient variant of ADEV.

```
allantools.oadev(data, rate, data_type='phase',  $\tau$ )
```

The key parameter τ in OADEV represents the averaging time (or observation interval) over which the signal is integrated before computing the variance. This discrete formulation is presented in Eq. (2.29) to give a more detailed understanding of the Allan deviation.

$$\sigma_{\text{OADEV}}^2(m\tau_0) = \frac{1}{2(m\tau_0)^2(N-2m)} \sum_{n=1}^{N-2m} (x_{n+2m} - 2x_{n+m} + x_n)^2 \quad (2.29)$$

2.6 Allan deviation

The concept is more clearly demonstrated by the discrete formulation in Eq. (2.29). where τ_0 denotes the sampling interval of the data (e.g., the IMU sampling period), and m is the number of samples within each averaging window. For each window, one variance value is computed, so in order to generate the ADEV curve, the algorithm need to generate a series of τ values with different window lengths. N is the total number of samples. An illustration is provided in Fig. 2.6. Since three sets of samples, x_n , x_{n+m} , and x_{n+2m} , are required for each computation, the maximum valid index is $N - 2m$. By varying τ over a suitable range, different noise characteristics can be revealed: at short averaging times, the influence of white noise dominates, whereas at longer averaging times, random walk effects become more significant.

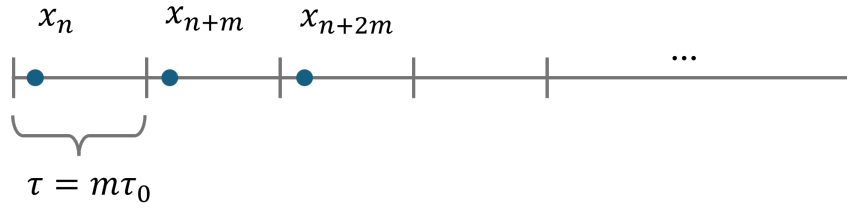


Figure 2.6: Illustrate of τ

Since the Allan variance requires multiple segments for computation, the minimal segment length is one sample. Therefore, the minimal averaging time is

$$\tau_{\min} = \frac{1}{f},$$

where f is the sampling frequency. On the other hand, at least two segments are required to compute the variance, so the maximal averaging time is

$$\tau_{\max} = \frac{T}{2},$$

where T is the total observation time.

Example: Assume we collect data for $T = 1$ second with sampling frequency $f = 100$ Hz, which gives $N = f \cdot T = 100$ samples in total.

- The minimal averaging time is $\tau_{\min} = 1/f = 0.01$ s, corresponding to one sample per segment.
- The maximal averaging time is $\tau_{\max} = T/2 = 0.5$ s, corresponding to $N/2 = 50$ samples per segment.

Thus, the smallest τ corresponds to one point per segment, while the largest τ corresponds to 50 points per segment. By plotting $\sigma_y(\tau)$ on a log-log scale, different noise processes can be identified by their characteristic slopes.

Table 2.1: Typical noise processes and their slopes in Allan deviation plots

Noise type	$\sigma_y(\tau)$	Slope of $\log \sigma_y(\tau)$	Typical source
White noise (angle/velocity)	N_0/τ	$-1/2$	Sensor measurement noise
Bias instability	$\sigma_0/0.664$	0	Temperature drift, electronics
Random walk	$K^2\tau/3$	$+1/2$	Accumulated bias or drift

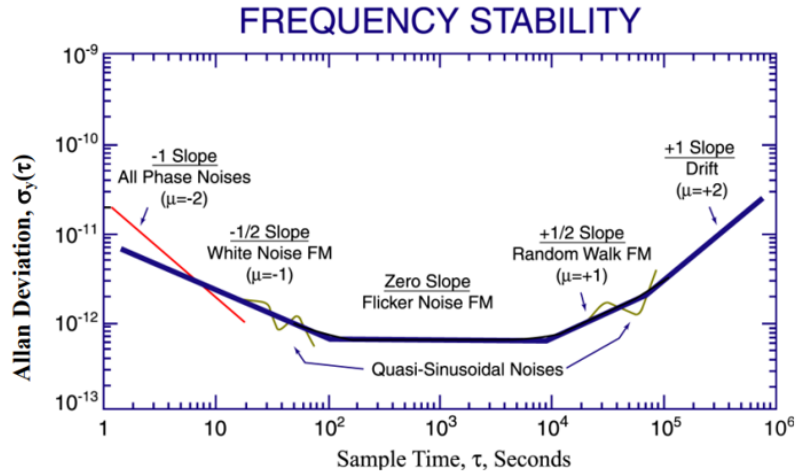


Figure 2.7: Allan Deviation plot from [2]

Using Fig. 2.7 as an example, the coefficient N_0 can be determined from the region where the ADEV curve exhibits a slope of $-1/2$. By selecting any point along this line and converting its value to the appropriate physical unit m/s^2 , the corresponding PSD level can be obtained.

2.6 Allan deviation

Chapter 3

Implementation

3.1 IMU characterization

Two inertial navigation sensors are used in this study: the IMU unit embedded in the u-blox ZED-F9R receiver and the high-grade STIM300 IMU. The aim of this subsection is to characterize and compare their performance, in order to obtain more accurate initial guesses and process noise settings for the Kalman filter, as well as to evaluate the potential for future calibration improvements. For low-cost IMU, the sensor is influenced by misalignment, cross-axis sensitivities, non accurate scaling and static/dynamic bias. The two main errors for acceleometers and gyroscopes are biases and scale errors. The information about two sensors are shown in Fig. 3.1.

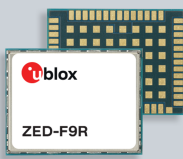

	F9R	STIM300
Item		
Hz	Max. 50 Hz	Max. 250 Hz
Level	Consumer grade	Tactical grade
Application	Automotive navigation	Aerospace, high-end navigation
Architecture	Integrated GNSS receiver + MEMS IMU + fusion engine	Pure IMU (3-axis accelerometers + 3-axis gyros + temperature compensation)

Figure 3.1: Comparison of u-blox F9R and STIM 300

3.1.1 Experiment Setup

Both sensors were placed stationary for 48 hours as shown in Fig. 3.2. The first 5 minutes of data were used to estimate the static bias, and were integrated to evaluate the unaided drift over time. The remaining data were used for ADEV analysis.

3.1 IMU characterization

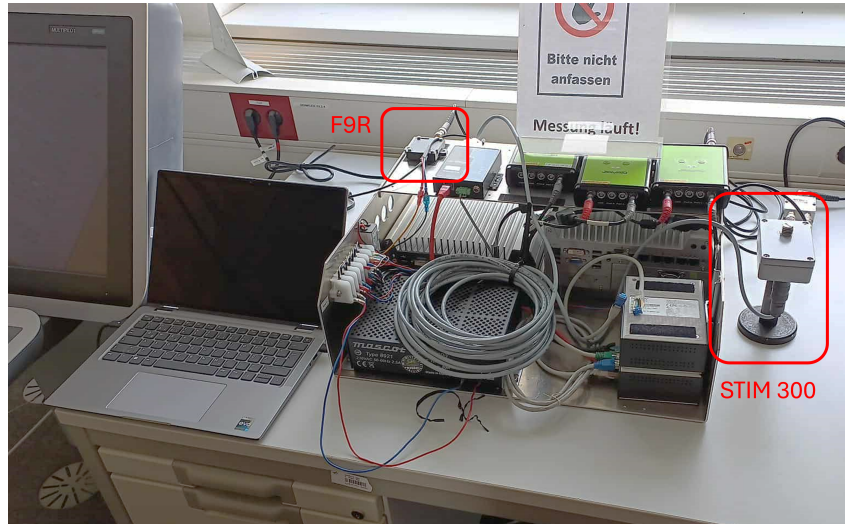


Figure 3.2: Experimental setup for static IMU testing

3.1.2 Analysis of ZED-F9R and STIM300

Without external aiding, the inertial solution drifts rapidly even when using a tactical-grade IMU such as the STIM300. The position error can reach up to approximately 2500 m after only 5 minutes of pure inertial integration, as shown in Fig. 3.3. The bias for each axis was computed by averaging the entire 5-minute static dataset. From Table. 3.1, it can be observed that the accelerometer biases of both sensors are of the same order for the x and y axes, whereas the z -axis bias in the F9R is significantly larger. Combined with the gyroscope bias, this introduces an error in the estimated rotation matrix, causing the z -axis error to be projected onto the horizontal axes. Consequently, the absolute position error accumulates even after static bias removal. These results highlight the importance of accurate attitude initialization and bias estimation in the navigation filter.

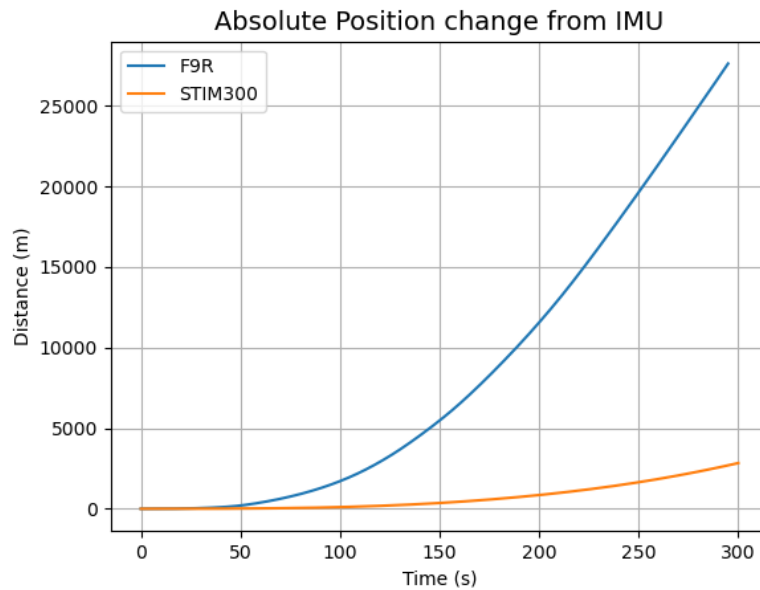
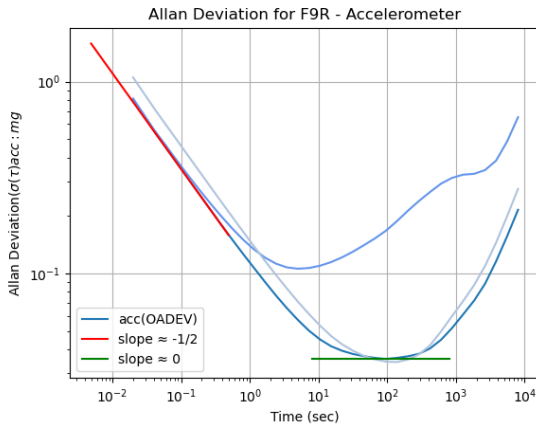


Figure 3.3: Accumulated absolute position error without aiding

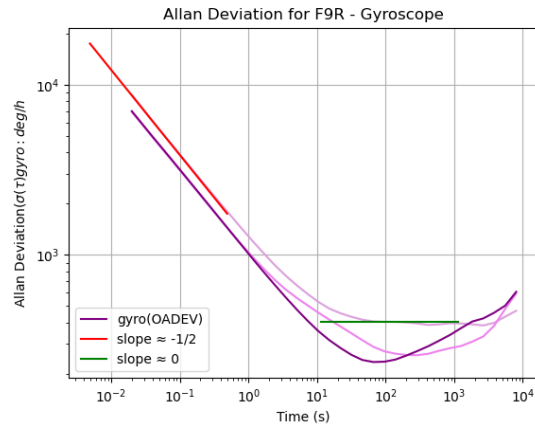
Table 3.1: Estimated accelerometer and gyroscope biases of F9R and STIM300

	Sensor	x	y	z
<i>Accelerometer bias (m/s^2)</i>	F9R	-0.1488	0.2808	0.4591
	STIM	-0.2672	0.1260	-0.0204
<i>Gyroscope bias ($^\circ/s$)</i>	F9R	0.1134	0.1484	0.1101
	STIM	-0.0002	0.0003	-0.0001

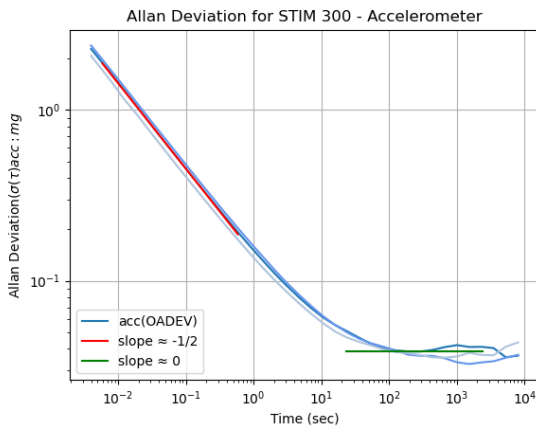
The Allan deviation plots of both sensors are shown in Fig. 3.4. The blue curves in (a) and (c) correspond to the accelerometer axes, while the purple curves correspond to the gyroscope axes. For the STIM300, the estimated noise characteristics closely match those reported in the manufacturer's datasheet, confirming its stability and low noise. In contrast, the detailed IMU specifications of the F9R are not publicly provided, making this analysis particularly useful for evaluating its actual performance. The ADEV results also serve as the basis for tuning the process noise parameters of the Kalman filter. Such as the white noise



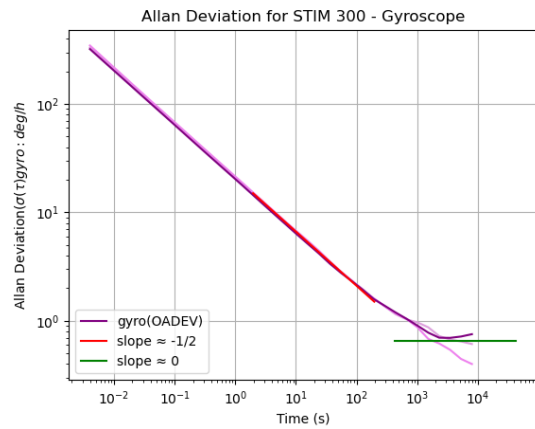
(a) F9R accelerometer



(b) F9R gyroscope



(c) STIM300 accelerometer



(d) STIM300 gyroscope

Figure 3.4: Allan deviation (ADEV) plots of F9R and STIM300

3.2 Data manipulation

To achieve on-line processing capability, the system reads binary messages directly from the receiver rather than using RINEX files. RINEX is a standardized text format used for storing and exchanging raw GNSS observation data and broadcast navigation messages. These binary messages are decoded and used as inputs for the following algorithms.

In this work, PyUBX2, an open-source Python library designed to communicate with u-blox GNSS receivers via serial or file interfaces, serves as the primary tool to decode the binary data stream and efficiently access raw GNSS and IMU measurements. It allows users to stream, parse, and log UBX, NMEA, and RTCM messages in real time. The library provides a convenient API to identify message types, extract payload fields, and even configure the receiver (e.g., enabling or disabling specific messages).

3.2.1 U-blox ZED-F9R

The u-blox F9R receiver is taken as an example in this section. Table. 3.2 introduces the three main messages used in our integration system and highlights several practical considerations and decoding details encountered when handling the binary stream from the receiver. It should be noted that the output data may vary depending on the receiver configuration, which can be adjusted using the u-center software or directly the PyUBX library.

Table 3.2: Summary of u-blox F9R messages used in this work

Message Type	Main Contents	Description / Usage in This Work
RXM-RAWX	constellation ID (<code>gnssId</code>), satellite ID (<code>svId</code>), signal ID (<code>sigId</code>), C/N0, GPS week number, receiver time of week (<code>ToW</code>), pseudorange, carrier Phase, doppler	Provides the primary GNSS raw measurements used for positioning. Includes precise timing information that serves as reference for IMUGNSS synchronization.
RXM-SFRBX	navigation message, raw ephemeris and clock data	Contains the broadcast navigation message from each satellite.
ESF-MEAS	Raw IMU measurements (data field, data type), Sensor time tags	Although F9R can internally run its own sensor fusion filter and output estimated navigation states, this study focuses on developing a custom integration algorithm using raw IMU measurements only. Therefore, the built-in Kalman filter output will not be discussed further.

- RXM-RAWX message:

TOW and the GPS week can be used to compute GPS time and converted to UTC (until now is 18 s later then GPS time). Since IMU data only provides a local time tick rather than an absolute timestamp, we used this GPS time as the reference.

We assume that the first IMU tick corresponds approximately to the first epoch of the RXM-RAWX data. By comparing the first IMU tick value and the corresponding GPS time from the first RAWX message, a fixed offset can be obtained. This offset is then used to convert

all subsequent IMU ticks into GPS time (and consequently UTC). Using this approach, both GNSS and IMU data streams can be accurately synchronized in time, allowing the system to extract data aligned to the desired UTC epoch for further integration and filtering.

- RXM-SFRBX message:

Under the GPS architecture, each satellite transmits navigation data divided into five subframes, and it takes approximately 30 seconds to receive a complete frame. The decoding process follows the IS-GPS-200 specification. To ensure consistency, it is important to verify that all subframes belong to the same ephemeris set. This can be checked using the Issue of Data Ephemeris (IODE) number. Once a change in IODE is detected, a new ephemeris is updated. Normally, the IODE changes every hour for GPS, unless there is packet loss or a correction broadcast.

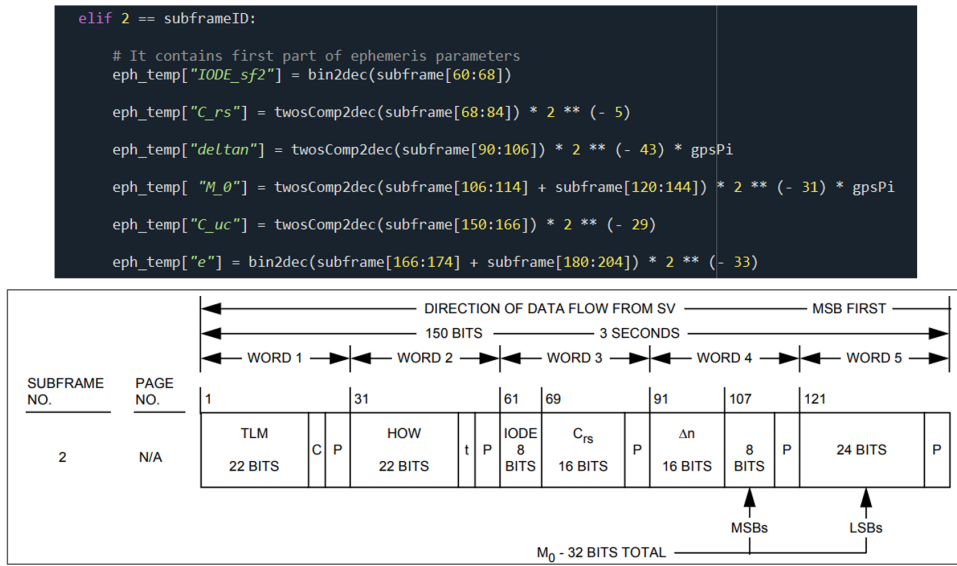


Figure 3.5: Decode Ephemeris

- ESF-MEAS message:

This message contains the raw IMU readings (accelerometer and gyroscope). The scaling factors and data format are provided in the u-blox ZED-F9R - Integration manual [21] as in Fig. 3.6. Care must be taken to correctly interpret the signed integer values and scaling coefficients to avoid overflow or misinterpretation errors. This message does not contain an absolute timestamp that can be directly used to synchronize with GNSS data. Instead, it provides a time tick from the internal oscillator, which must be referenced to the GNSS measurement time to achieve proper time alignment, as described in the integration manual.

3.2 Data manipulation

Type	Description	Unit	Format of the 24 data bits
8	rear-left wheel ticks		Bits 0-22: unsigned tick value. Bit 23: direction indicator (0=forward, 1=backward)
9	rear-right wheel ticks		Bits 0-22: unsigned tick value. Bit 23: direction indicator (0=forward, 1=backward)
10	single tick (speed tick)		Bits 0-22: unsigned tick value. Bit 23: direction indicator (0=forward, 1=backward)
11	speed	m/s * 1e-3	signed
12	gyroscope temperature	deg Celsius * 1e-2	signed
13	y-axis gyroscope angular rate	deg/s * 2 ⁻¹²	signed
14	x-axis gyroscope angular rate	deg/s * 2 ⁻¹²	signed
15	reserved		
16	x-axis accelerometer-specific force	m/s ² * 2 ⁻¹⁰	signed
17	y-axis accelerometer-specific force	m/s ² * 2 ⁻¹⁰	signed
18	z-axis accelerometer-specific force	m/s ² * 2 ⁻¹⁰	signed
19...63	reserved		

Table 11: Definition of data types

Figure 3.6: IMU data description

In summary, the whole decoding process can be illustrated in Fig. 3.7.

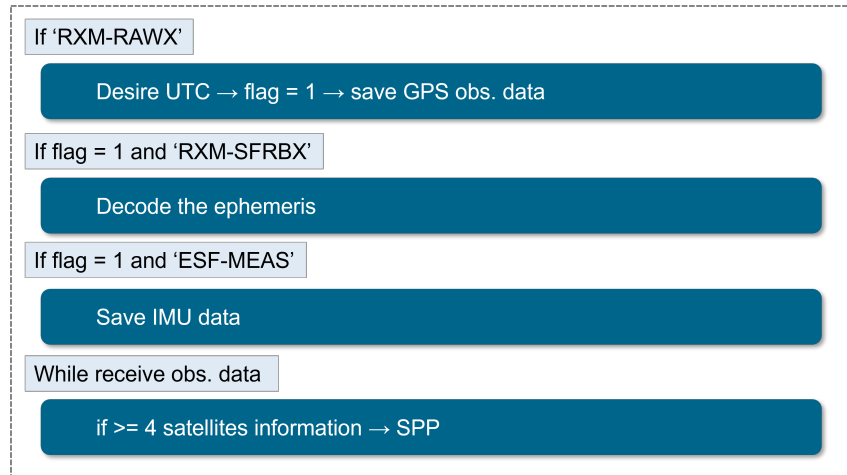


Figure 3.7: Decoding workflow

3.2.2 Online-SPP test

The correctness of the decoded ephemeris was verified by comparing the extracted parameters with those from a reference RINEX navigation file, as illustrated in Fig. 3.8. Once validated, these ephemeris parameters were used to compute SPP solution.

	Index	sqrtA	e	omega	omega_0	M_0	I_0	svId	time
0		-4.45652e-10	-1.00072e-16	-5.4956e-15	-2.4758e-14	-3.8014e-13	7.99361e-15	1	t1
1		-3.25599e-10	-2.00361e-16	-3.05977e-13	3.4861e-13	-4.4964e-13	7.77156e-15	3	t1
2		4.02906e-10	3.00107e-16	-3.45501e-13	-1.59872e-13	3.51941e-13	1.28786e-14	4	t1
3		-3.79259e-10	-4.996e-16	-2.64233e-14	-1.32117e-14	-3.26406e-13	-2.20934e-14	6	t1
4		4.28372e-10	9.99201e-16	-2.92211e-13	-2.02505e-13	-2.62901e-13	-4.51861e-14	7	t1
5		6.36646e-11	3.00107e-16	-3.30624e-13	-3.8014e-13	3.53717e-13	-4.69624e-14	9	t1

Figure 3.8: Ephemeris validation by comparing decoded broadcast parameters with those from a reference RINEX file.

An on-line SPP algorithm was implemented in Python. For each run, the average processing time is approximately 0.1 s. The resulting SPP trajectory is shown in Fig. 3.9, together with a Precise Point Positioning (PPP) solution obtained from the online CSRS-PPP service provided by Natural Resources Canada [22]. This PPP result serves as an external benchmark to assess the accuracy of the on-line SPP implementation and supports its subsequent integration with the IMU.

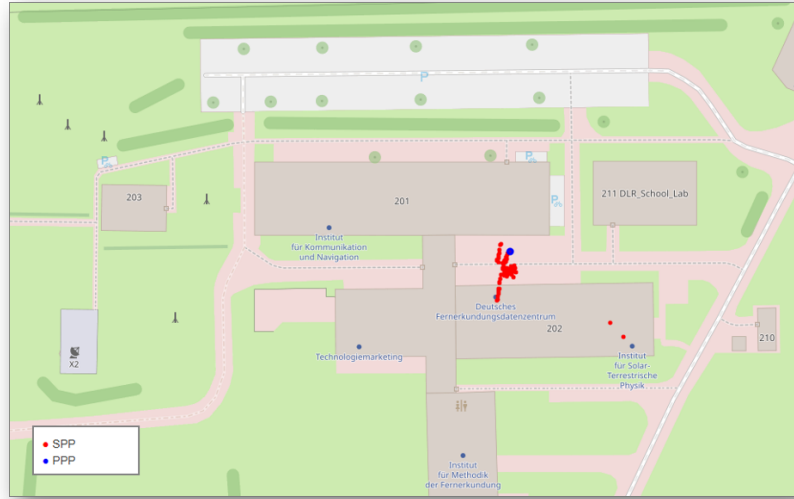


Figure 3.9: Comparison between the implemented SPP solution and the PPP reference from CSRS-PPP.

Some tips for coding: A data format which follows a structure consistent with that of RINEX-based processing is preferred.

3.3 Error state kalman filter design (ESKF)

While the classical KF, EKF, and UKF provide the general framework for sensor fusion, directly estimating the full navigation states with an EKF can lead to numerical issues, since very large values (position, velocity) and very small values (sensor biases) are combined in the same state vector, and orientation is often represented by Euler angles, which are prone to singularities.

To overcome these limitations, the ESKF reformulates the estimation problem by maintaining a nominal trajectory and estimating only the small deviations (errors) from it. This separation allows the filter to operate on small, linearized quantities, which improves numerical stability and robustness.

Another important feature of the ESKF is that it works with quaternions to represent attitude. Compared to Euler angles, quaternions are smoother and do not suffer from singularities such as gimbal lock [4]. Within the ESKF, attitude updates are expressed as small rotation vectors applied multiplicatively to the nominal quaternion, which ensures that attitude corrections remain consistent on the unit quaternion manifold.

In this section, we first introduce the nominal navigation equations and their discrete-time formulation, followed by the derivation of the error-state dynamics and their discretization. The subsequent sections explain the measurement update, the error correction and reset procedures,

3.3 Error state kalman filter design (ESKF)

and finally present an overall algorithmic summary with pseudo-code to illustrate the complete ESKF loop.

3.3.1 Nominal states and discretization

The relation between the true state and the nominal state is defined as

$$\mathbf{x}_{true} = \mathbf{x}_{nominal} + \delta\mathbf{x}$$

Nominal states are propagated and updated at each time step according to the equations in Eq. 3.1. The propagation of the attitude \mathbf{q} , velocity \mathbf{v} , and position \mathbf{p} follows the same formulation as described in the IMU mechanization chapter. The accelerometer bias \mathbf{b}_a and gyroscope bias \mathbf{b}_g are modeled as random-walk processes, whose slow drift is introduced through the process noise in the error-state model. For the LC configuration, these 15 states form the entire state vector. For the TC configuration, the receiver clock bias \mathbf{c}_o and clock drift \mathbf{c}_d are also estimated.

$$\begin{aligned} \mathbf{p} &\leftarrow \mathbf{p} + \frac{1}{2}(\mathbf{v}_{old} + \mathbf{v}_{new}) \cdot \Delta t, \\ \mathbf{v} &\leftarrow \mathbf{v} + \mathbf{a}_{eb}^e \cdot \Delta t, \\ \mathbf{q} &\leftarrow \mathbf{q} \otimes \delta\mathbf{q}, \\ \mathbf{b}_a &\leftarrow \mathbf{b}_a, \\ \mathbf{b}_g &\leftarrow \mathbf{b}_g, \\ \mathbf{c}_o &\leftarrow \mathbf{c}_o + \mathbf{c}_d \cdot \Delta t, \\ \mathbf{c}_d &\leftarrow \mathbf{c}_d. \end{aligned} \tag{3.1}$$

The position and velocity are defined in the global ECEF frame, while the quaternion represents the estimated attitude. In our implementation, it describes the orientation of the body frame with respect to the global frame.

3.3.2 Error states and discretization

The concept of the error state is to describe how small perturbations in the system evolve and affect the nominal states. These differential equations are the basis for deriving the continuous-time system matrix \mathbf{F} and its discrete equivalent Φ . Similar to the nominal model, the error-state equations are first derived in the continuous-time domain and then discretized using suitable assumptions or first-order approximations.

We estimate a total of 17 error states for the tightly couples implementation, including position, velocity, attitude, accelerometer bias, gyroscope bias, clock offset, and clock drift for GPS, 15 error states for loosely coupled without clock estimation. The state vector is given by:

$$\delta\mathbf{x} = [\delta\mathbf{p}_e, \delta\mathbf{v}_e, \delta\boldsymbol{\theta}_b, \delta\mathbf{b}_a, \delta\mathbf{b}_g, \delta\mathbf{c}_o, \delta\mathbf{c}_d]^T. \tag{3.2}$$

or equivalently, when the attitude error is defined in the global frame,

$$\delta\mathbf{x} = [\delta\mathbf{p}_e, \delta\mathbf{v}_e, \delta\boldsymbol{\theta}_e, \delta\mathbf{b}_a, \delta\mathbf{b}_g, \delta\mathbf{c}_o, \delta\mathbf{c}_d]^T. \tag{3.3}$$

A key point, as discussed in [4], is that since the nominal kinematics do not explicitly involve errors, the error angle can be defined either in the **body frame** or in the **global frame**. The

only difference lies in the structure of the Jacobian matrix and the method used to inject the attitude correction after the update. Below, we first present the continuous-time error equations when the attitude error is defined in the body frame, following the formulation in Chapter 5 of [4].

$$\begin{aligned}
 \delta \dot{\mathbf{p}}_e &= \delta \mathbf{v}_e, \\
 \delta \dot{\mathbf{v}}_e &= -\hat{\mathbf{C}}_b^e [\hat{\mathbf{f}}_{ib}^b]_{\times} \delta \boldsymbol{\theta}_b - \hat{\mathbf{C}}_b^e \delta \mathbf{b}_a - \hat{\mathbf{C}}_b^e \mathbf{a}_n, \\
 \delta \dot{\boldsymbol{\theta}}_b &= -[\boldsymbol{\omega}_{eb}^b]_{\times} \delta \boldsymbol{\theta}_b - \delta \mathbf{b}_g - \boldsymbol{\omega}_n, \\
 \delta \dot{\mathbf{b}}_a &= \mathbf{a}_w, \\
 \delta \dot{\mathbf{b}}_g &= \boldsymbol{\omega}_w, \\
 \delta \dot{\mathbf{c}}_o &= \mathbf{c}_d, \\
 \delta \dot{\mathbf{c}}_d &= 0.
 \end{aligned} \tag{3.4}$$

If the attitude error is instead defined in the **global frame**, all other equations remain the same except for those related to velocity and attitude:

$$\begin{aligned}
 \delta \dot{\mathbf{v}}_e &= -[\hat{\mathbf{C}}_b^e \hat{\mathbf{f}}_{ib}^b]_{\times} \delta \boldsymbol{\theta}_e - \hat{\mathbf{C}}_b^e \delta \mathbf{b}_a - \hat{\mathbf{C}}_b^e \mathbf{a}_n, \\
 \delta \dot{\boldsymbol{\theta}}_e &= -\hat{\mathbf{C}}_b^e \delta \mathbf{b}_g - \hat{\mathbf{C}}_b^e \boldsymbol{\omega}_n.
 \end{aligned} \tag{3.5}$$

The corresponding discrete-time form for the body-frame-defined error is:

$$\begin{aligned}
 \delta \mathbf{p}_e &\leftarrow \delta \mathbf{p}_e + \delta \mathbf{v}_e \Delta t, \\
 \delta \mathbf{v}_e &\leftarrow \delta \mathbf{v}_e + (-\hat{\mathbf{C}}_b^e [\hat{\mathbf{f}}_{ib}^b]_{\times} \delta \boldsymbol{\theta}_b - \hat{\mathbf{C}}_b^e \delta \mathbf{b}_a) \Delta t + \mathbf{v}_i, \\
 \delta \boldsymbol{\theta}_b &\leftarrow \mathbf{C}^T \{\boldsymbol{\omega}_{eb}^b \Delta t\} \delta \boldsymbol{\theta}_b - \delta \mathbf{b}_g \Delta t + \boldsymbol{\theta}_i, \\
 \delta \mathbf{b}_a &\leftarrow \delta \mathbf{b}_a + \mathbf{a}_i, \\
 \delta \mathbf{b}_g &\leftarrow \delta \mathbf{b}_g + \boldsymbol{\omega}_i, \\
 \delta \mathbf{c}_o &\leftarrow \delta \mathbf{c}_o + \mathbf{c}_d \Delta t, \\
 \delta \mathbf{c}_d &\leftarrow \delta \mathbf{c}_d.
 \end{aligned} \tag{3.6}$$

Similarly, the discrete form for the global-frame-defined error is given by:

$$\begin{aligned}
 \delta \mathbf{v}_e &\leftarrow \delta \mathbf{v}_e + (-[\hat{\mathbf{C}}_b^e \hat{\mathbf{f}}_{ib}^b]_{\times} \delta \boldsymbol{\theta}_e - \hat{\mathbf{C}}_b^e \delta \mathbf{b}_a) \Delta t + \mathbf{v}_i, \\
 \delta \boldsymbol{\theta}_e &\leftarrow \delta \boldsymbol{\theta}_e - \hat{\mathbf{C}}_b^e \delta \mathbf{b}_g \Delta t + \boldsymbol{\theta}_i.
 \end{aligned} \tag{3.7}$$

Error-state equation Eq. (3.6) can be linearized to form the transition matrix \mathbf{F}_x by taking partial derivatives with respect to each term of the state vector. The discrete transition matrix is as follows:

$$\mathbf{F}_d = \left. \frac{\partial f}{\partial \delta \mathbf{x}} \right|_{\mathbf{x}, \mathbf{u}_m} = \begin{bmatrix} \mathbf{I} & \mathbf{I} \Delta t & 0 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{I} & -\hat{\mathbf{C}}_b^e [\hat{\mathbf{f}}_{ib}^b]_{\times} \Delta t & -\hat{\mathbf{C}}_b^e \Delta t & 0 & 0 & 0 \\ 0 & 0 & \mathbf{C}^T \{\boldsymbol{\omega}_{eb}^b \Delta t\} & 0 & -\mathbf{I} \Delta t & 0 & 0 \\ 0 & 0 & 0 & \mathbf{I} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{I} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.8}$$

3.3 Error state kalman filter design (ESKF)

The Jacobian matrix \mathbf{F}_i with respect to the process noise is expressed as:

$$\mathbf{F}_i = \left. \frac{\partial f}{\partial \mathbf{i}} \right|_{\mathbf{x}, \mathbf{u}_m} = \begin{bmatrix} \frac{1}{2} \hat{\mathbf{C}}_b^e \Delta t^2 & 0 & 0 & 0 & 0 & 0 \\ \hat{\mathbf{C}}_b^e \Delta t & 0 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{I} \Delta t & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{I} \Delta t & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathbf{I} \Delta t & 0 & 0 \\ 0 & 0 & 0 & 0 & \Delta t & \frac{1}{2} \Delta t^2 \\ 0 & 0 & 0 & 0 & 0 & \Delta t \end{bmatrix}. \quad (3.9)$$

The position noise mainly originates from the velocity noise. Since the derivative of clock bias is clock drift, the noise is define in clock drift [23]. For short time intervals, this contribution can be ignored. The process noise covariance matrix \mathbf{Q}_i is given by:

$$\mathbf{Q}_i = \text{diag}(\mathbf{V}_i, \mathbf{\Theta}_i, \mathbf{A}_i, \mathbf{\Omega}_i, \sigma_{c_o}^2, \sigma_{c_d}^2), \quad \begin{aligned} \mathbf{V}_i &= \sigma_{\text{acc}}^2 \mathbf{I}, \\ \mathbf{\Theta}_i &= \sigma_{\text{gyro}}^2 \mathbf{I}, \\ \mathbf{A}_i &= \sigma_{b_a}^2 \mathbf{I}, \\ \mathbf{\Omega}_i &= \sigma_{b_g}^2 \mathbf{I}. \end{aligned} \quad (3.10)$$

The covariance matrix \mathbf{P} is then propagated as:

$$\mathbf{P} \leftarrow \mathbf{F}_x \mathbf{P} \mathbf{F}_x^T + \mathbf{F}_i \mathbf{Q}_i \mathbf{F}_i^T. \quad (3.11)$$

The key difference between the EKF and the ESKF is that the state-transition matrix \mathbf{F} is used only to propagate the error state, not the nominal state. As a result, \mathbf{F} appears solely in the error-state propagation and in the computation of the update step, whereas the nominal state is propagated using the full nonlinear mechanization equations.

3.3.3 Measurement model and update

In our implementation, the measurement model is first expressed in its nonlinear form as

$$\tilde{\mathbf{z}} = h(\mathbf{x}) + \mathbf{v}, \quad (3.12)$$

where $\tilde{\mathbf{z}}$ denotes the raw measurements and \mathbf{v} represents the measurement noise. The **residual (innovation)** is defined as the difference between the actual and the predicted measurement:

$$\mathbf{y} \triangleq \tilde{\mathbf{z}} - h(\hat{\mathbf{x}}) \approx \mathbf{H} \delta \mathbf{x} + \mathbf{v}. \quad (3.13)$$

Hence, \mathbf{y} is not the measurement itself but the difference between the measurement and its prediction. The observation Jacobian matrix \mathbf{H} is obtained by taking the first-order partial derivatives of the linearized measurement function $h(\mathbf{x})$ with respect to each element of the error-state vector $\delta \mathbf{x}$.

Loosely coupled update:

For the loosely coupled case, the PVT result is used as the measurement. The residual is defined as the difference between the GNSS position and the position estimated from the IMU mechanization:

$$\mathbf{y} = \tilde{\mathbf{p}}_{SPP} - \hat{\mathbf{p}}_{IMU} = (\mathbf{p}_{true} + \mathbf{n}) - (\hat{\mathbf{p}}_{true} - \delta \mathbf{p}) = \delta \mathbf{p} + \mathbf{n}. \quad (3.14)$$

Where \mathbf{n} is the noise in the measurement. Since the position measurement is directly related to the position states, the observation Jacobian is obtained by taking the derivative of the position with respect to $\delta \mathbf{p}$, resulting in an identity matrix:

$$\mathbf{H} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}. \quad (3.15)$$

Tightly coupled update:

For the tightly coupled case, the raw pseudorange observations are used. The nonlinear measurement equation for satellite i is

$$\tilde{\rho}_i = \rho_i(\mathbf{p}_u) + c(\delta t_r - \delta t_{s,i}) + I_i + T_i + \epsilon_i, \quad (3.16)$$

where $\rho_i(\mathbf{p}_u)$ is the geometric range between the satellite and the receiver position \mathbf{x}_u , and I_i and T_i denote the ionospheric and tropospheric delays.

To compute the predicted measurement $\hat{\rho}_i$, each term in (3.16) is evaluated using the current state estimate $\hat{\mathbf{p}}_u$. The ionospheric and tropospheric delays are obtained from their respective models [24]. The satellite clock bias $\delta t_{s,i}$ is computed from the broadcast ephemeris, whereas the receiver clock bias δt_r is part of the Kalman filter state. Thus, $\hat{\rho}_i$ includes all modeled effects consistent with the current filter estimate.

In particular, the predicted geometric range is computed from the IMU-propagated receiver position $\hat{\mathbf{p}}$ and the satellite position $\mathbf{p}_{\text{sat},i}$ as

$$\hat{\rho}_i(\hat{\mathbf{p}}_u) = \|\mathbf{p}_{\text{sat},i} - \hat{\mathbf{p}}\| = \sqrt{(x_{\text{sat},i} - \hat{x}_u)^2 + (y_{\text{sat},i} - \hat{y}_u)^2 + (z_{\text{sat},i} - \hat{z}_u)^2}. \quad (3.17)$$

The residual for each satellite is

$$y_i \triangleq \tilde{\rho}_i - \hat{\rho}_i \approx \mathbf{H}_i \delta \mathbf{x} + v_i. \quad (3.18)$$

It represents how much the measured range differs from the predicted one based on the current estimated state.

The linearization process of (3.17) is based on the partial derivatives of the measurement equation with respect to the error-state variables.

$$\mathbf{H}_i = \frac{\partial \tilde{\rho}_i}{\partial \delta \mathbf{x}} = \begin{bmatrix} -\frac{\partial \rho_i}{\partial \delta \mathbf{p}} & \frac{\partial \rho_i}{\partial \delta \mathbf{v}} & \frac{\partial \rho_i}{\partial \delta \boldsymbol{\theta}} & \frac{\partial \rho_i}{\partial \delta \mathbf{b}_a} & \frac{\partial \rho_i}{\partial \delta \mathbf{b}_g} & \frac{\partial \rho_i}{\partial \delta t_r} & \frac{\partial \rho_i}{\partial \delta t_s} \end{bmatrix}. \quad (3.19)$$

Among these terms, only a few are non-zero:

$$\frac{\partial \rho_i}{\partial \delta \mathbf{p}} = -\mathbf{u}_i^\top, \quad \frac{\partial \rho_i}{\partial \delta t_r} = c,$$

where \mathbf{u}_i is the line-of-sight (LOS) unit vector from the receiver to the satellite. The final Jacobian for satellite i can therefore be written as

$$\mathbf{H}_i = \begin{bmatrix} -\mathbf{u}_i^\top & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & c & 0 \end{bmatrix}. \quad (3.20)$$

If a lever arm between the IMU and the GNSS antenna is considered, the geometric distance becomes

$$\rho_i(\mathbf{p}_{\text{ant}}) = \|\mathbf{p}_{\text{sat},i} - \hat{\mathbf{p}}_{\text{ant}}\| = \|\mathbf{p}_{\text{sat},i} - (\hat{\mathbf{p}}_{\text{IMU}} + \hat{\mathbf{C}}_b^e \mathbf{l}_b)\|, \quad (3.21)$$

3.3 Error state kalman filter design (ESKF)

where \mathbf{l}_b is the lever arm vector expressed in the body frame, and $\hat{\mathbf{C}}_b^e$ is the rotation matrix from the body frame to the ECEF frame.

Because the antenna position now depends on the attitude, the corresponding terms in the observation Jacobian matrix \mathbf{H} are affected. The result depends on how the attitude error is defined (i.e., in the body frame or in the global frame).

Error defined in the body frame:

If the attitude error is defined in the body frame, the quaternion (or rotation matrix) update uses **right multiplication**. The rotation perturbation can be written as

$$\begin{aligned}\mathbf{C}_b^e &= \hat{\mathbf{C}}_b^e(\mathbf{I} + [\delta\boldsymbol{\theta}_b]_{\times}), \\ \mathbf{C}_b^e \mathbf{l}_b &= \hat{\mathbf{C}}_b^e(\mathbf{I} + [\delta\boldsymbol{\theta}_b]_{\times}) \mathbf{l}_b \\ &= \hat{\mathbf{C}}_b^e \mathbf{l}_b + \hat{\mathbf{C}}_b^e [\delta\boldsymbol{\theta}_b]_{\times} \mathbf{l}_b \\ &= \hat{\mathbf{C}}_b^e \mathbf{l}_b - \hat{\mathbf{C}}_b^e [\mathbf{l}_b]_{\times} \delta\boldsymbol{\theta}_b.\end{aligned}\tag{3.22}$$

Substitute $\hat{\mathbf{C}}_b^e$ in Eq.(3.21) and take the derivative with respect to $\delta\boldsymbol{\theta}_b$, the lever arm introduces an additional term in the observation Jacobian as follows:

$$\mathbf{H}_i = \begin{bmatrix} -\mathbf{u}_i^\top & \mathbf{u}_i^\top \hat{\mathbf{C}}_b^e [\mathbf{l}_b]_{\times} & \mathbf{0} & \mathbf{0} & \mathbf{0} & c & 0 \end{bmatrix},\tag{3.23}$$

where \mathbf{u}_i is the line-of-sight (LOS) unit vector from receiver to satellite.

Error defined in the global frame:

If the attitude error is defined in the global frame, the quaternion update uses **left multiplication**. In this case,

$$\begin{aligned}\mathbf{C}_b^e &= (\mathbf{I} + [\delta\boldsymbol{\theta}_e]_{\times}) \hat{\mathbf{C}}_b^e, \\ \mathbf{C}_b^e \mathbf{l}_b &= (\mathbf{I} + [\delta\boldsymbol{\theta}_e]_{\times}) \hat{\mathbf{C}}_b^e \mathbf{l}_b \\ &= \hat{\mathbf{C}}_b^e \mathbf{l}_b + [\delta\boldsymbol{\theta}_e]_{\times} \hat{\mathbf{C}}_b^e \mathbf{l}_b \\ &= \hat{\mathbf{C}}_b^e \mathbf{l}_b + [\delta\boldsymbol{\theta}_e]_{\times} \mathbf{l}_e \\ &= \hat{\mathbf{C}}_b^e \mathbf{l}_b - [\mathbf{l}_e]_{\times} \delta\boldsymbol{\theta}_e,\end{aligned}\tag{3.24}$$

where $\mathbf{l}_e = \hat{\mathbf{C}}_b^e \mathbf{l}_b$ is the lever arm vector expressed in the ECEF frame. Accordingly, the observation Jacobian becomes

$$\mathbf{H}_i = \begin{bmatrix} -\mathbf{u}_i^\top & \mathbf{u}_i^\top [\mathbf{l}_e]_{\times} & \mathbf{0} & \mathbf{0} & \mathbf{0} & c & 0 \end{bmatrix}.\tag{3.25}$$

With the observation matrix \mathbf{H} , state covariance matrix \mathbf{P} , measurement covariance matrix \mathbf{R} , and the innovation covariance matrix \mathbf{S} , the Kalman gain \mathbf{K} can be computed.

$$\begin{aligned}\mathbf{S} &= \mathbf{H}\mathbf{P}\mathbf{H}^T + \mathbf{R} \\ \mathbf{K} &= \mathbf{P}\mathbf{H}^T\mathbf{S}^{-1}\end{aligned}\tag{3.26}$$

The weighting elements of \mathbf{R} can be defined as a function of the satellite elevation angle [24]. Satellites with higher elevation are assigned smaller values of σ due to improved signal quality and reduced atmospheric effects

$$\sigma = \frac{1}{\sin(\text{elevation})^2}$$

other weighting method such as C/N_0 -based weighting can also be considered, where satellites with higher signal-to-noise ratio are given smaller variance.

3.3.4 Error Correction and Reset

The states errors $\delta \mathbf{x}$ can be estimated and then injected back into the nominal states.

$$\begin{aligned} \delta \hat{\mathbf{x}} &= \mathbf{K} \cdot \mathbf{y} \\ \mathbf{x} &\leftarrow \mathbf{x} \otimes \delta \hat{\mathbf{x}} \end{aligned} \quad (3.27)$$

Notably, when updating the orientation using quaternions, the definition of the attitude error must be handled carefully. A global-frame error requires left-multiplication, whereas a body-frame error requires right-multiplication.

$$\text{Global-frame error:} \quad \mathbf{q} \leftarrow \delta \mathbf{q} \otimes \mathbf{q}, \quad (3.28)$$

$$\text{Body-frame error:} \quad \mathbf{q} \leftarrow \mathbf{q} \otimes \delta \mathbf{q}. \quad (3.29)$$

As explain in [4] there's one more step to reset the covariance matrix. Since the reference coordinate of error states has already change, \mathbf{P} has to change accordingly. It can be perform as follows:

$$\begin{aligned} \mathbf{P} &\leftarrow \mathbf{G} \mathbf{P} \mathbf{G}^T \\ \mathbf{G} &= \begin{bmatrix} \mathbf{I}_6 & 0 & 0 \\ 0 & \mathbf{I} - [\frac{1}{2}\delta\boldsymbol{\theta}]_{\times} & 0 \\ 0 & 0 & \mathbf{I}_8 \end{bmatrix} \end{aligned} \quad (3.30)$$

It is important to note that, under the global-frame definition, the small-angle update must use $\mathbf{I} + [\frac{1}{2}\delta\boldsymbol{\theta}]_{\times}$.

3.3.5 Initialization

In this thesis, a Least-Squares (LS) estimation is first used to obtain a position and receiver clock offset from the GNSS measurements. For attitude initialization, self-alignment is performed under the assumption that the IMU is stationary at the beginning of the experiment. In this condition, the accelerometers measure only the specific force that balances gravity, which satisfies

$$\mathbf{f}_{ib}^b = \mathbf{C}_n^b \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix}. \quad (3.31)$$

this specific-force vector aligns with the negative down direction of the local navigation frame near the Earth's surface. If a yaw-pitch-roll (Z-Y-X) rotation sequence is chosen, the rotation matrix from the body frame to the navigation frame can be written as

3.3 Error state kalman filter design (ESKF)

$$\mathbf{C}_b^n = \begin{bmatrix} c_\theta c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi & c_\phi s_\theta c_\psi + s_\phi s_\psi \\ c_\theta s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{bmatrix}, \quad (3.32)$$

where ϕ , θ , and ψ denote roll, pitch, and yaw angles, respectively, and $c(\cdot) = \cos(\cdot)$, $s(\cdot) = \sin(\cdot)$. Since only the third column of \mathbf{R}_n^b multiplies the gravity vector, the specific force measured in the body frame can be expressed as

$$\mathbf{f}_{ib}^b = \begin{bmatrix} gs_\theta \\ -gs_\phi c_\theta \\ -gc_\phi c_\theta \end{bmatrix}. \quad (3.33)$$

Hence, the roll and pitch angles below can be determined from the averaged accelerometer outputs. This procedure is commonly referred to as leveling.

$$\theta = \arcsin\left(-\frac{a_x}{g}\right), \quad \phi = \text{atan2}(a_y, -a_z). \quad (3.34)$$

The negative signs depend on the IMU axis convention (e.g., z pointing downward). These two angles provide an accurate estimate of the sensor orientation with respect to the local level plane, even for low-cost MEMS sensors. According to [3], we can also determine pitch and roll without knowing the gravity as below

$$\theta = \arctan\left(\frac{-f_{ib,x}^b}{\sqrt{(f_{ib,y}^b)^2 + (f_{ib,z}^b)^2}}\right), \quad \phi = \arctan 2\left(-f_{ib,y}^b, -f_{ib,z}^b\right) \quad (3.35)$$

To determine the yaw angle, the Earth's rotation rate vector is used as a natural reference when the IMU is static. The Earth's rotation in the local ENU frame is given by

$$\boldsymbol{\omega}_{ie}^n = \begin{bmatrix} 0 \\ \omega_{ie} \cos \varphi \\ \omega_{ie} \sin \varphi \end{bmatrix}, \quad (3.36)$$

where φ is the geodetic latitude. The first component is zero because there is no eastward component of the rotation axis; the second and third components correspond to the northward and upward projections of the Earth's spin vector.

The IMU measures the same rotation in its body frame as

$$\boldsymbol{\omega}_{ie}^b = \mathbf{C}_n^b \boldsymbol{\omega}_{ie}^n. \quad (3.37)$$

Given that the roll and pitch are already known, the body angular rate can be rotated back to the local level frame:

$$\boldsymbol{\omega}_h = \mathbf{C}_y(-\theta) \mathbf{C}_x(-\phi) \boldsymbol{\omega}_{ib}^b, \quad (3.38)$$

where $\boldsymbol{\omega}_h$ represents the angular rate expressed in a level (horizontal) frame. The projection of $\boldsymbol{\omega}_{ie}^n$ onto the horizontal plane is aligned with the geographic north, with magnitude $\omega_{ie} \cos \varphi$.

Therefore, the yaw (heading) angle can be estimated as

$$\psi = \text{atan2}(\omega_{h,E}, \omega_{h,N}), \quad (3.39)$$

which corresponds to the so-called gyrocompassing process [3]. If the IMU precision is not sufficient to resolve the small Earth rotation rate, or if the platform is moving, the initial yaw can alternatively be derived from the GNSS course-over-ground computed from consecutive position estimates.

Depending on the IMU body coordinate definition, either ENU or NED can be selected as the navigation frame. Once the navigation frame is defined, the corresponding rotation matrix \mathbf{C}_n^e is constructed. With the system dynamics and covariance propagation defined, the next step is to incorporate external measurements to correct the estimated states. This is achieved through the measurement model and update step of the Kalman filter. The formulation depends on whether a loosely or tightly coupled GNSS/INS architecture is used.

3.3.6 Time synchronization

In the u-blox F9R integration manual, the receiver local time is defined as a mapping from the internal 1-kHz oscillator clock to the GNSS time base. The receiver continuously refines this mapping as it acquires satellite timing information. Once two or more GNSS-referenced measurements are available, a linear relationship between the local clock ticks and the GNSS time can be established:

$$t_{GNSS} = a \cdot t_{tick} + b$$

where a represents the actual GNSS time interval per tick (capturing the oscillator frequency and drift), b is the GNSS time corresponding to the tick origin (i.e., the clock offset). Because the receivers estimate of GNSS time is not yet stable during startup, both a and b vary until the GNSS time lock is achieved.

For the STIM300 IMU, each measurement is time-tagged directly with the GPS seconds. Therefore, in our off-line implementation, time synchronization between the IMU and GNSS sensors is achieved simply by merging, sorting, and labeling all measurements according to their GPS timestamps. The overall procedure is illustrated in Fig. 3.10.

During the fusion step, an update is triggered whenever a newly received GNSS measurement occurs within 0.05 seconds of the most recent IMU propagation timestamp. This ensures that each GNSS update is applied at a sufficiently close epoch, minimizing interpolation error and maintaining temporal consistency between the two sensor streams.

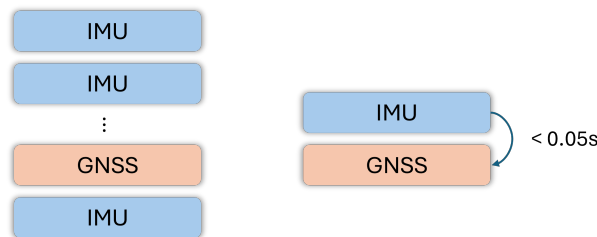


Figure 3.10: Time synchronization

3.3 Error state kalman filter design (ESKF)

3.3.7 Summary and Flow Diagram

Since practical implementation does not always match the methodology described in the literature, we use the pseudocode to summarize how the algorithm is executed, how the initial position is obtained, and which parameters are passed between functions during the update process.

Algorithm 1 Online GNSS/IMU ESKF Pipeline

```

1: Input: DataGNSS, DataIMU
2: Output: Estimated states  $(p, v, rpy, ab, wb, cb, cd)$ 
3: // Step 0: Build data objects
4: Build DataGNSS and DataIMU
5: // Step 1: Label timestamps
6: Label observation time and IMU time
7: // Step 2: Initialize ESKF
8: function INI_ESKF(acc, gyro)
9:    $ab, wb \leftarrow$  average first 100 samples (initialize sensor biases)
10:   $p_{ant}, cb \leftarrow$  initial position by LS ( $p_{ant}^{ini}$ , current_t)
11:   $rpy \leftarrow$  rough attitude estimation
12:  Build State object  $\rightarrow (p, v, rpy, ab, wb, cb, cd)$ 
13:  Build Sxx object  $\rightarrow (F_x, F_i, Q, P)$ 
14:  return  $p_{ant}, C_{b2e}$ 
15: end function
16: // Step 3: Main loop
17: for  $j = 1$  to  $\text{len}(\text{all\_label})$  do
18:   if  $\text{label}[j] = \text{IMU}$  then
19:     $p_{ant}, C_b^e \leftarrow \text{PREDICT}(\text{dt}, \text{state}, \text{DataIMU}, C_{b2e})$ 
20:   else if  $\text{label}[j] = \text{GNSS}$  then
21:     $(pr, \text{sat\_pos}, \text{ele}) \leftarrow \text{FIND\_PR\_SV\_POS}(p_{ant}, \text{current\_t}, \text{DataGNSS}, \text{angle\_mask})$ 
22:     $p_{ant}, C_{b2e} \leftarrow \text{UPDATE}(\text{State}, \text{Sxx}, pr, \text{sat\_pos}, \text{ele}, C_b^e)$ 
23:   end if
24: end for

```

Chapter 4

Results and Discussion

To evaluate our work, we mainly use two data sets. Data from STIM 300 and Spirent Simulator a software model which simulates IMU. The results are categorised into static and dynamic and listed in Table. 4.1.

Table 4.1: Estimated accelerometer 3and gyroscope biases of F9R and STIM300

Type	Source	Coupling	Description
<i>Static</i>	case1: N/A	N/A	Constant speed model
	case2: STIM300	LC/TC	250 Hz, 2 hr
<i>Dynamic</i>	case3: Spirent	LC	100Hz, 30min, quiet/rough sea
	case4: Spirent	TC	100Hz, 30min, quiet/rough sea

4.1 Static data

We began with a static case without using any IMU data, which serves as the simplest example for validating the overall procedure of the algorithm and for better understanding the effect of each tuning parameter. Afterwards, real IMU data from the STIM300 was used for comparison, allowing us to observe how the filter behaves when switching from a constant velocity model to the real mechanization.

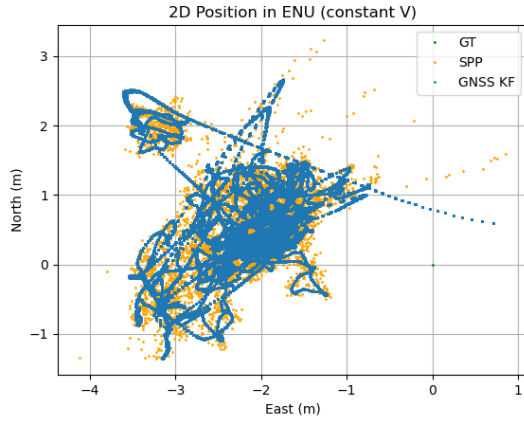
4.1.1 Case1: constant velocity model

In this example, only the six states of position and velocity are estimated. We investigate how the covariance matrix settings influence the estimation results and the behavior of the innovation. To make the setup comparable to the use of real IMU data, the update step is executed once every 50 prediction steps by directly using PVT solution from SPP. Thus, the time interval in the prediction step is $dt = 0.02$ s, and the input GNSS data is a 1 Hz dataset of two hours. The measurement covariance is set using $\sigma_q = 5$, meaning that the PVT position measurement is assumed to have a standard deviation of 5 meters.

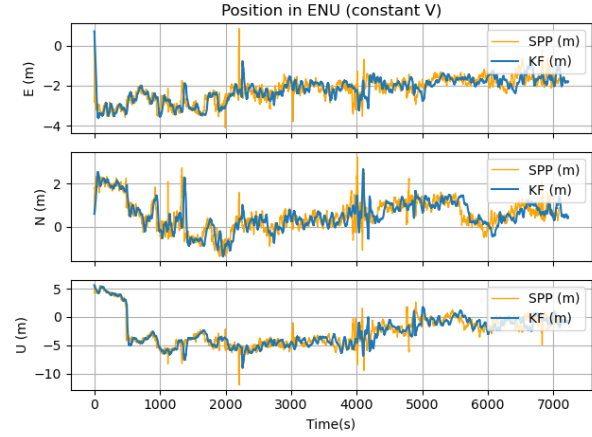
Figures 4.1(a) and (b) show the positioning results when the process-noise parameter σ_v in \mathbf{Q} is set to 0.1. In Fig. 4.1(a), the ground truth provided by the PPP Canadian service is used as the reference at the origin and is also set as the initial position. The estimated trajectory initially drifts to the right and is later pulled back by the during the update steps when PVT

4.1 Static data

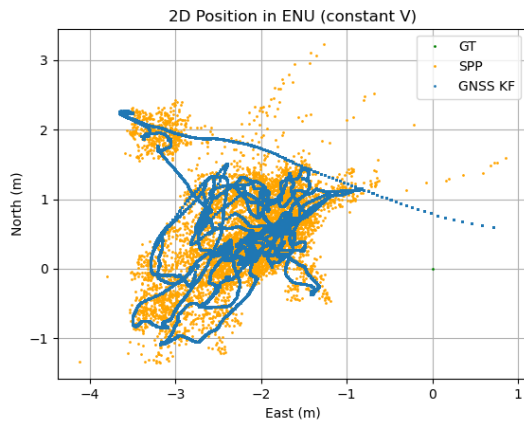
measurement is available. In (b), the estimated curves are smoother than the SPP solution and become even smoother when $\sigma_v = 0.01$, or when a larger measurement noise is assigned (i.e., when the measurement is trusted less), as illustrated in Figs. 4.1(c) and (d).



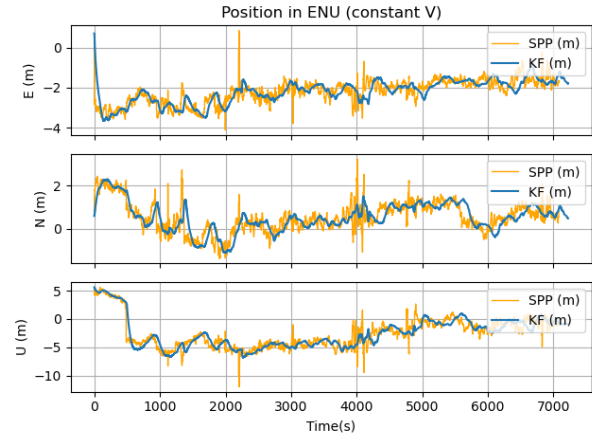
(a) 2D position estimation ($Q = 0.1$)



(b) Position estimation ($Q = 0.1$)



(c) 2D position estimation ($Q = 0.01$)



(d) Position estimation ($Q = 0.01$)

Figure 4.1: Constant Velocity model results ($Q = 0.01$)

The histogram of the position innovation in Fig. 4.2 follows a Gaussian distribution, which is consistent with the model assumptions.

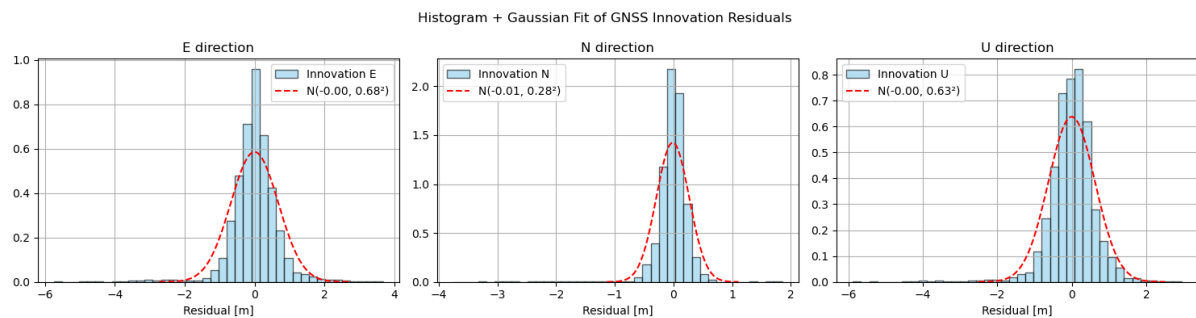


Figure 4.2: Constant velocity LC innovation distribution

4.1.2 Case2: IMU model + LC/TC

The static dataset used in this section is shown in Fig. 4.3. For the accelerometer, biases are present in both the along-track and cross-track directions. Since the sensor is known to be stationary, these biases are clearly visible, along with occasional spikes that may originate from sensor noise or artifacts. As for the gyroscope, the bias is relatively small, and its quality is generally better than that of the accelerometer, although it also exhibits noise and a few spikes.

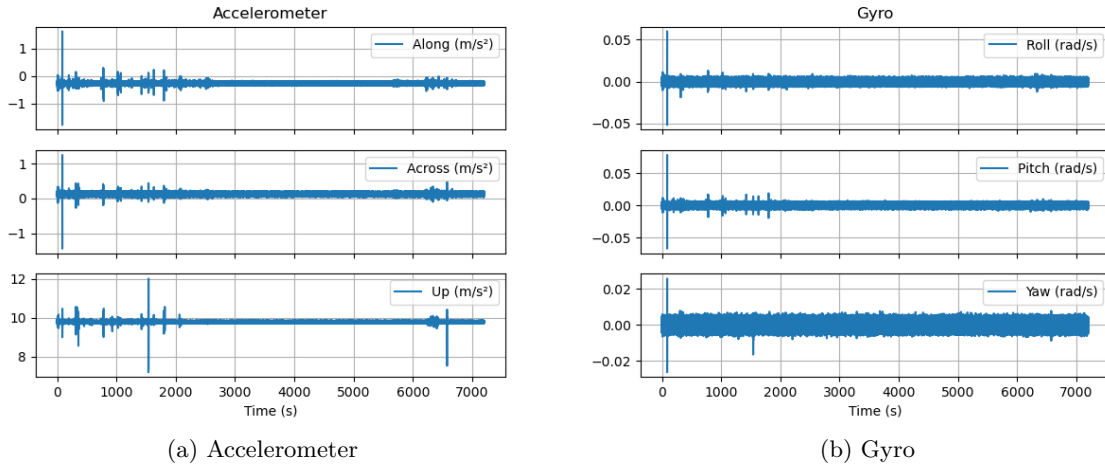


Figure 4.3: STIM 300 dataset

By using real sensor data and applying the IMU mechanization model, the positioning results are shown in Fig. 4.4. Both LC and TC approaches perform worse than SPP in this static scenario. This degradation is due to the continuous integration of accelerometer and gyroscope biases during the mechanization process. One effective method to mitigate this issue is the Zero-Velocity Update (ZUPT). When the platform is known to be static, enforcing a zero-velocity constraint helps reduce the impact of accelerometer bias on the velocity state and prevents drift accumulation. This strategy could be incorporated in future filter implementations.

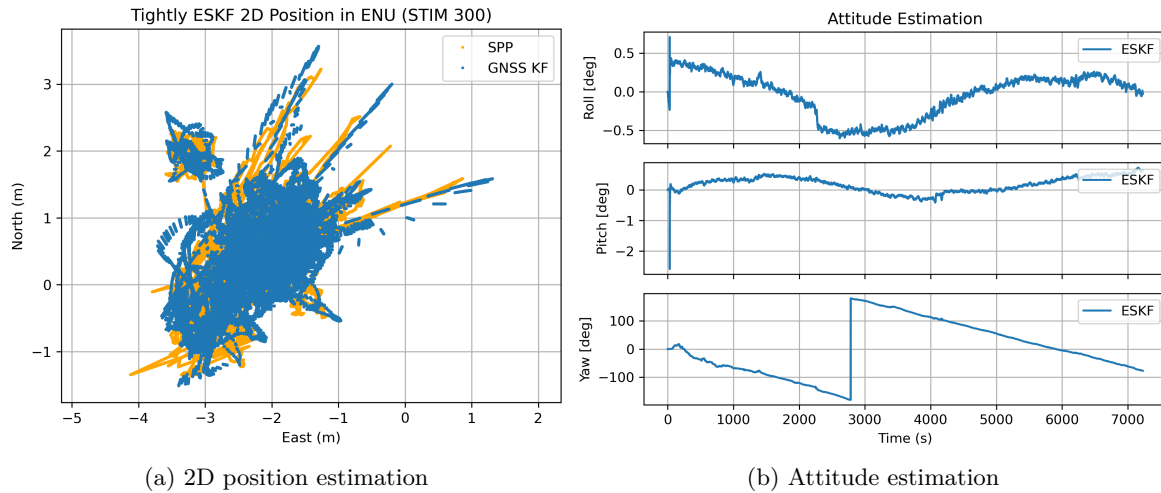


Figure 4.4: STIM 300 dataset

4.2 Dynamic data

4.2 Dynamic data

This section uses 30 minutes of simulated dynamic data to evaluate the performance of the GNSS/IMU integration under realistic maritime motion. Two scenarios are considered, quiet sea and rough sea, in order to examine how the filter behaves under different motion profiles and sensor noise levels. The overall trajectory is shown in Fig. 4.5.

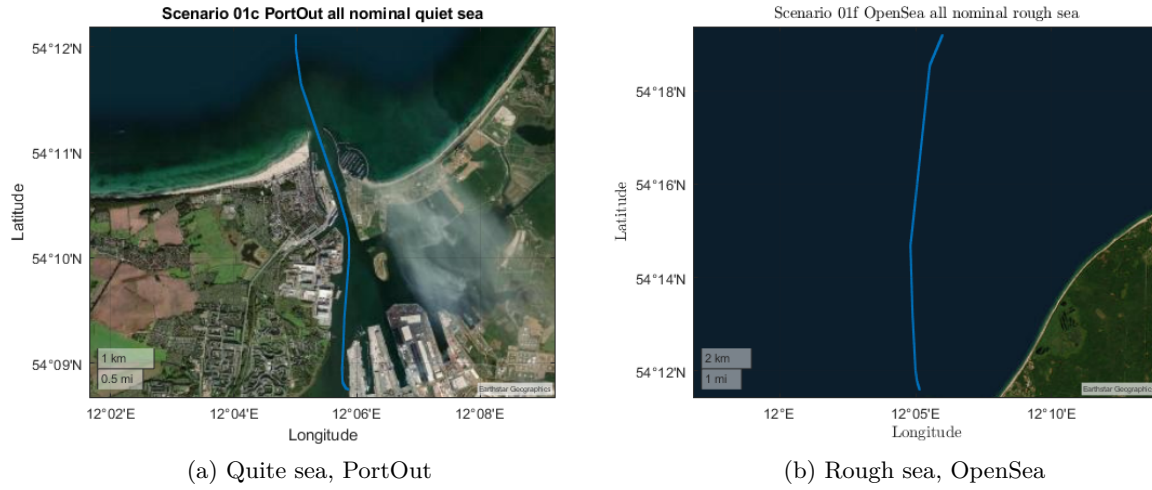


Figure 4.5: Dynamic dataset

A corresponding set of simulated IMU measurements for the same motion is presented in Fig. 4.6. The wave movement can be observed in roll and pitch, thus the fluctuation is larger in rough sea case. To further highlight the advantages of the LC and TC architectures, artificial GNSS outages of 30 s and 20 s are introduced in the middle of the dataset. As shown in Fig. 4.7, the input PVT measurements are removed during these intervals for the LC approach, while the pseudorange measurements are removed for the TC approach.

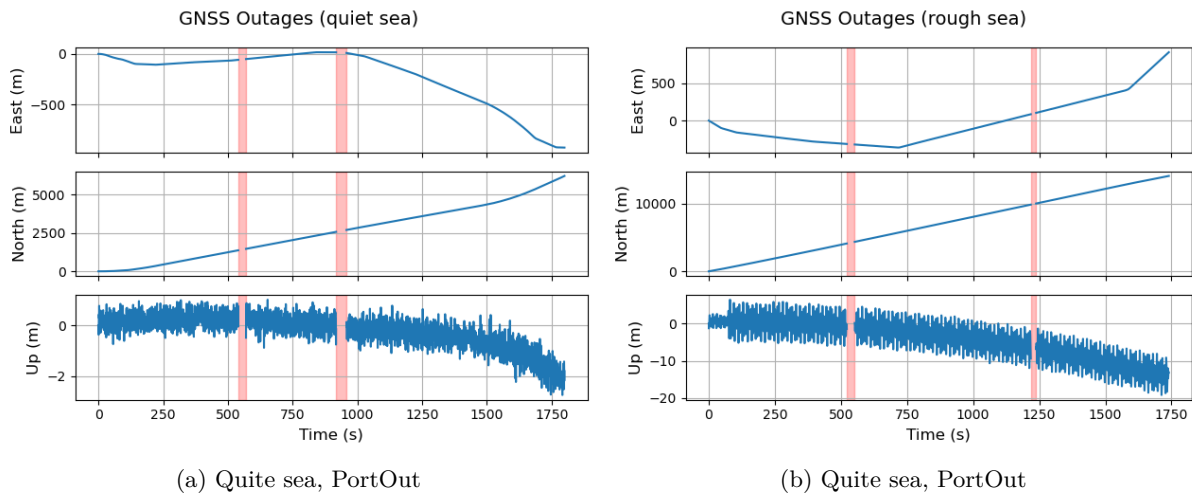
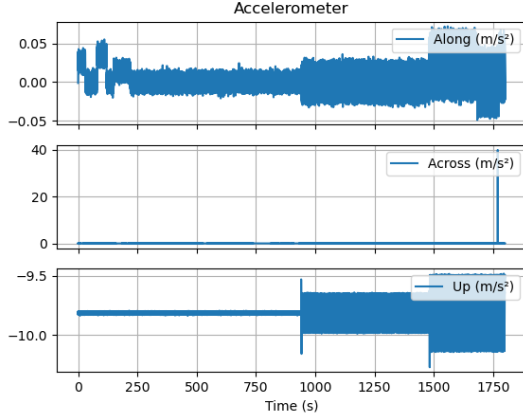
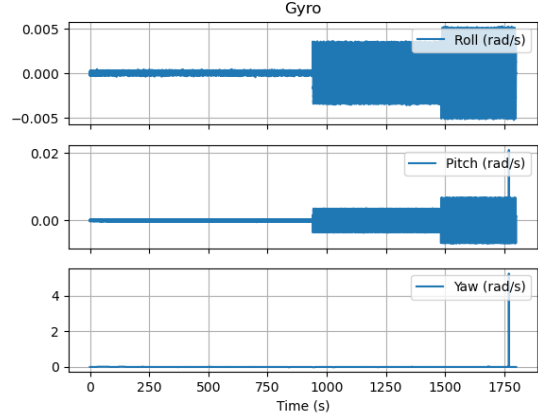


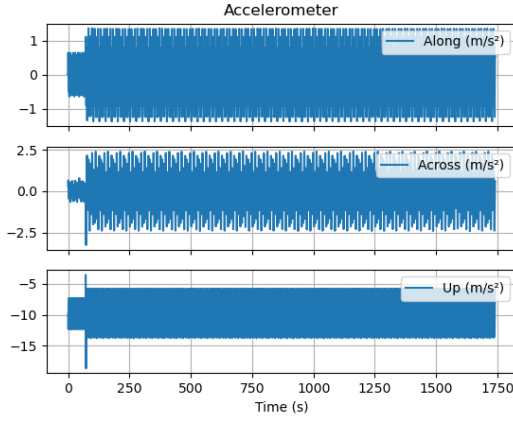
Figure 4.7: Artificial GNSS outage



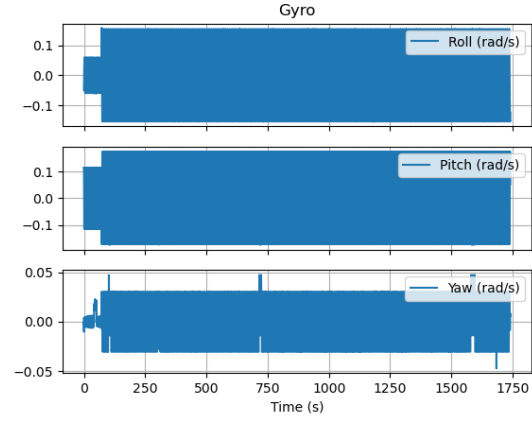
(a) Quite sea, PortOut



(b) Quite sea, PortOut



(c) Rough sea, OpenSea



(d) Rough sea, OpenSea

Figure 4.6: Dynamic IMU dataset

4.2.1 Parameter tuning

In [3], the continuous-time modeling of inertial sensor noise describes the white-noise components of the accelerometer and gyro measurements by their power spectral densities (PSDs). If σ_{ra} denotes the standard deviation of the sampled accelerometer noise, the corresponding continuous-time PSDs are given by

$$n_{ra}^2 = \sigma_{ra}^2 \tau_i,$$

where τ_i is the sampling interval of the IMU.

The initial state covariance and process noise for Spirent can be configured using the tables below. In the initial state covariance, a small value for position is given since we use least squares to initialise our position, providing a relatively accurate initial guess. The attitude can be given using a uniform distribution of the angle, and the other parameters are the result of trial and error. For the process noise covariance, the sensor noise was generated and added to the simulated data, normally the term is known.

Table 4.2: Initial State Covariance Parameters

State Component	Symbol / Value	Unit
Position (ECEF)	$\sigma_{p0} = [5, 5, 5]$	m
Velocity (ECEF)	$\sigma_{v0} = 1$	m/s
Attitude (RPY)	$\sigma_{q0} = 0.5$	rad
Accelerometer Bias	$\sigma_{a0} = 10^{-3}$	m/s ²
Gyroscope Bias	$\sigma_{g0} = 10^{-6}$	rad/s
Clock Bias	$\sigma_{co0} = 200$	m
Clock Drift	$\sigma_{cd0} = 1$	m

Table 4.3: Process Noise Parameters

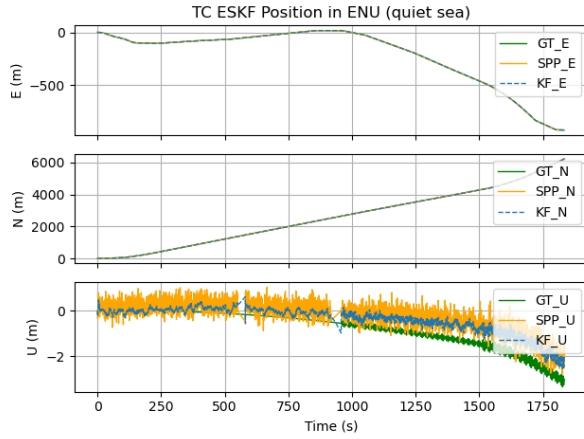
Noise Term	Symbol / Value	Unit
Accelerometer Noise	$\sigma_{acc} = 0.1$	(m/s ²)
Gyroscope Noise	$\sigma_{gyro} = 0.01$	(rad/s)
Accelerometer Bias RW	$\sigma_{ab} = 5 \times 5^{-2}$	m/s ² √s
Gyroscope Bias RW	$\sigma_{wb} = 1 \times 10^{-6}$	rad/s√s
Clock Bias Noise	$\sigma_{co} = 500$	m
Clock Drift Noise	$\sigma_{cd} = 100$	m

4.2.2 Case3: PortOut, Quiet sea

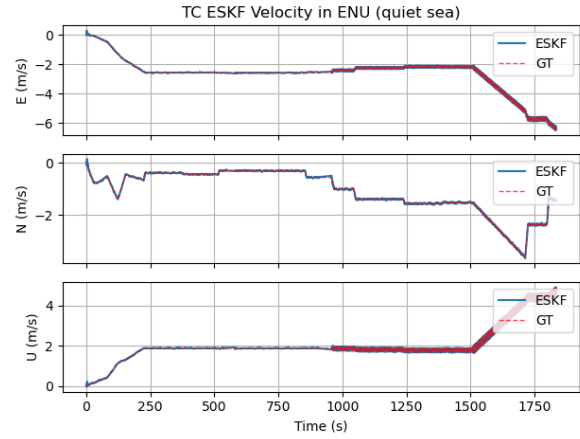
In the first simulated scenario Fig. 4.5(a), the vessel moves steadily forward on a calm surface, porting out so with slightly turning. Both LC and TC is are used for compare the performances and process time, but only the results of TC will be shown in the figure since the outcome is similar. The results of the position, velocity, attitude, clock bias, and clock drift estimation are shown in Fig. 4.8.

Compared to SPP, the positioning results from the tightly coupled ESKF are noticeably smoother. During the GNSS outage, the filter drifts less than 5 meters, while SPP cannot provide a solution at all. The attitude and velocity estimates are not significantly affected by the outage, and their behavior is similar to that observed in the LC case. It is obvious from Fig. 4.8(c) that the attitude estimation converges quickly, even when the initial guess is not close to the true value. However, a slight drift can be observed in the pitch angle, which is likely caused by imperfections in the gravity model. In our implementation, a fixed experimental gravity value was adopted due to the lack of detailed information about the true gravity model in the simulator, which may introduce small modeling errors.

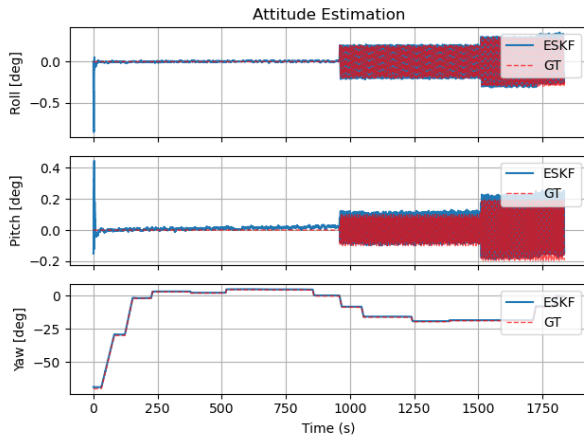
An important event occurs around 750 seconds as visible in Fig.4.8(e), where a sudden clock offset appears. Simply inflating the clock bias covariance is not practical, as it may destabilize the estimation of the remaining states. Therefore, a clock-offset detection mechanism is implemented. If the increment of the estimated clock bias within a single update exceeds a threshold, the system temporarily switches to an SPP-based correction to recompute the clock bias and reset its covariance. When such an event is detected, the filter intentionally becomes more pessimistic to avoid divergence.



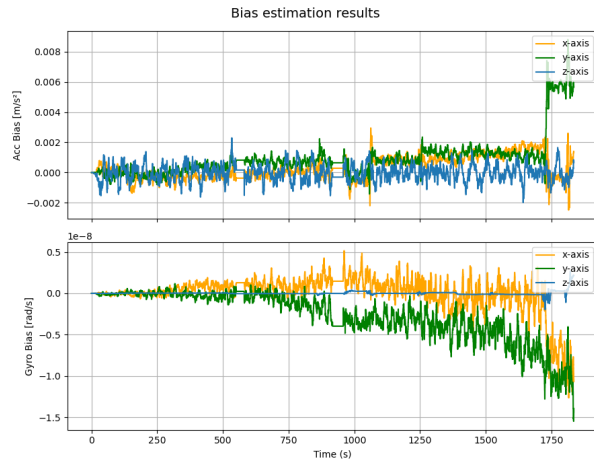
(a) Position estimation



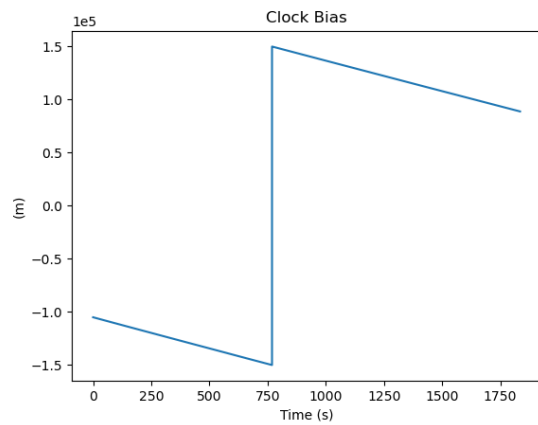
(b) Velocity estimation



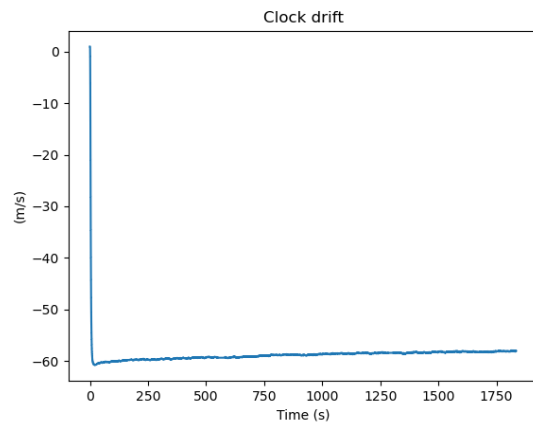
(c) Attitude estimation



(d) Bias estimation



(e) Clock bias estimation



(f) Clock drift estimation

Figure 4.8: Results of TC with GNSS outage (Quite sea, PortOut)

4.2 Dynamic data

4.2.2.1 Error analysis

The error bounds for the position and attitude are shown in Fig. 4.9. The position errors remain safely within the 3σ bounds, indicating that the estimates are reliable and the errors remain small. The drift during the GNSS outage is almost negligible. However, for the attitude error, a noticeable bias appears in the later part of the trajectory. Although the magnitude is small, the presence of a bias may lead to potential issues. Considering that the estimated profiles in Fig. 4.8(d) follow the general shape of the ground truth, a possible explanation is an incorrect gravity compensation or a bug in the implementation.

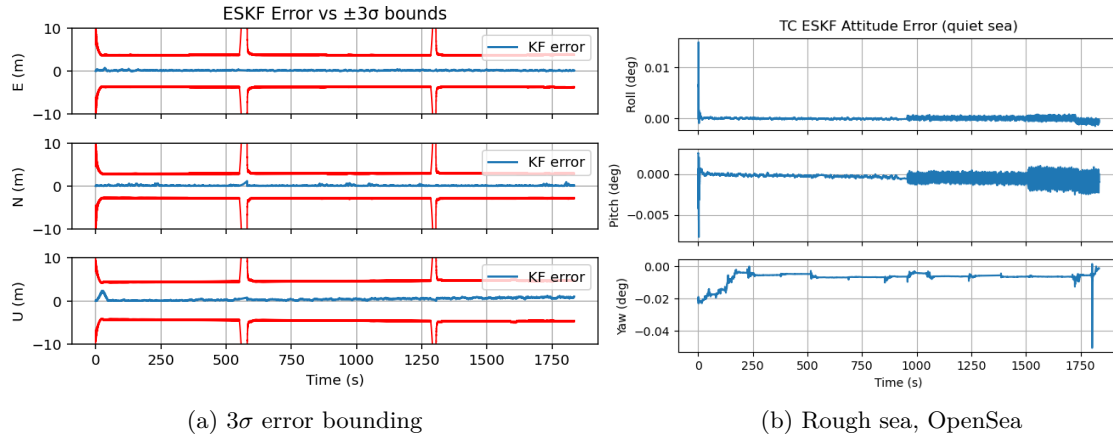


Figure 4.9: Error plot for rough sea scenario

Table 4.4 lists the statistics of the results. In the quiet sea case, TC achieves the best performance, while LC is naturally the fastest since it relies only on the PVT solution from the receiver, whereas TC performs all corrections beforehand. The SPP result is also included as a reference; however, it is not directly comparable because SPP fails to provide estimates during GNSS outages and also we need to consider that KF requires time to converge. In the last column, only approximate execution times are provided as reference values, since they are strongly affected by other simultaneous operations on the computer as well as the tuning parameters.

Table 4.4: RMS of LC, TC and SPP (Quite sea)

RMS (m)	E	N	U	2D	approx. time (s)
LC	0.16023933	0.15406612	0.60783823	0.22229	30s
TC	0.14868371	0.12115974	0.52023703	0.191798	80s
SPP	0.1898487	0.14742164	0.62780441	0.240365698	—

4.2.3 Case4: OpenSea, Rough sea

In the second scenario shown in Fig. 4.5(b), the vessel sails straight in open rough water, where stronger waves introduce noticeable disturbances into the IMU; therefore, a larger process noise is assigned during filter tuning. The estimation results for position, velocity, attitude, clock bias, and clock drift are shown in Fig. 4.10.

With GNSS outage, the filter is still able to track the larger fluctuations in the vessel's position and velocity. In this case, roll and pitch remain close to the ground truth, but the yaw angle

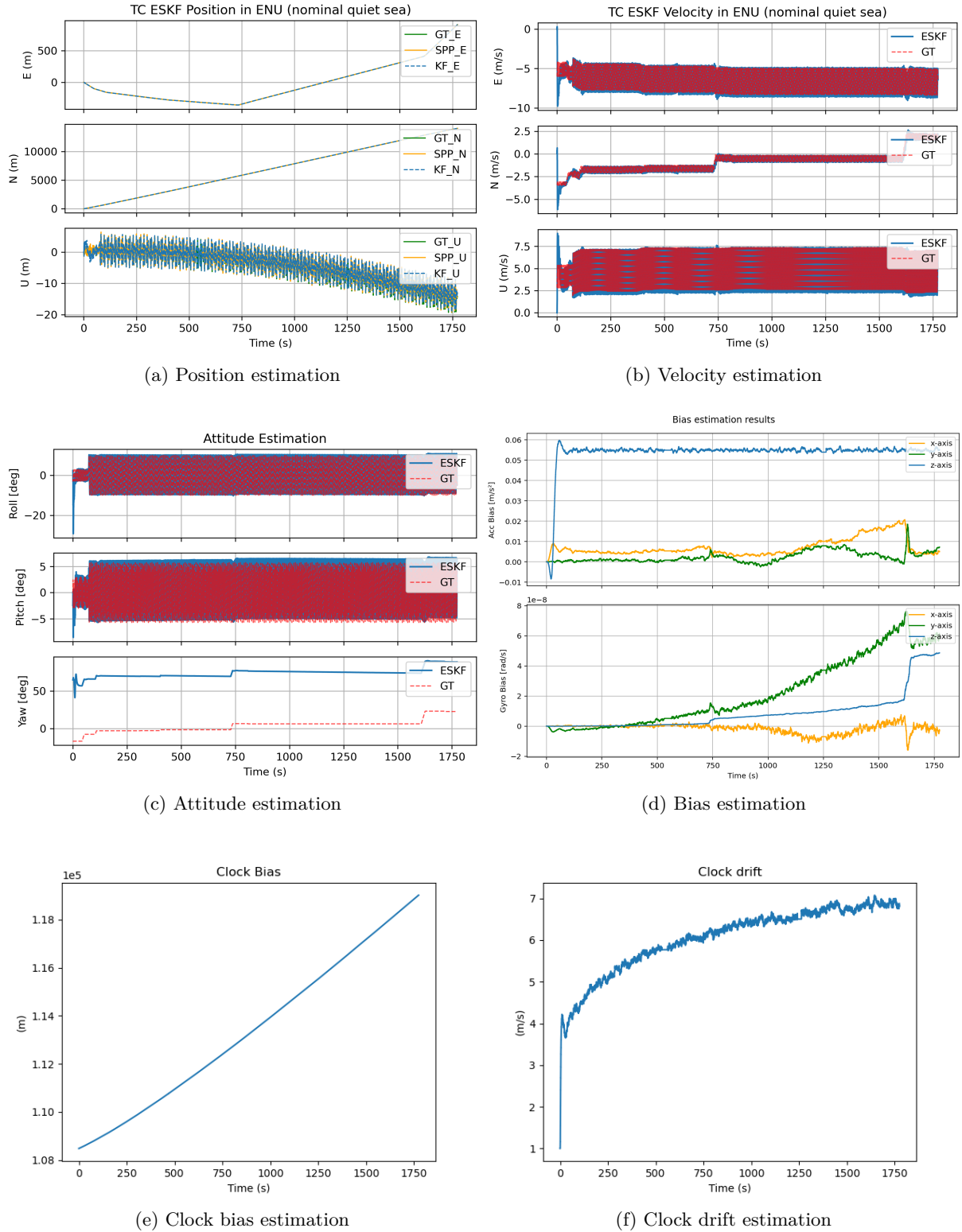


Figure 4.10: Results of TC (Rough sea, OpenSea)

retains a bias. Although this bias can be reduced by initializing the heading more accurately, the results still highlight the limited observability of the yaw angle.

4.2 Dynamic data

4.2.3.1 Error analysis

From the plots in Fig. 4.11, the position errors are also well bounded within the 3σ limits. The attitude errors in roll and pitch show better performance compared to the quiet sea case; however, the error in the yaw angle is significantly larger. Together with the previous results, this confirms that the system does not have full observability for the yaw axis under single antenna configuration.

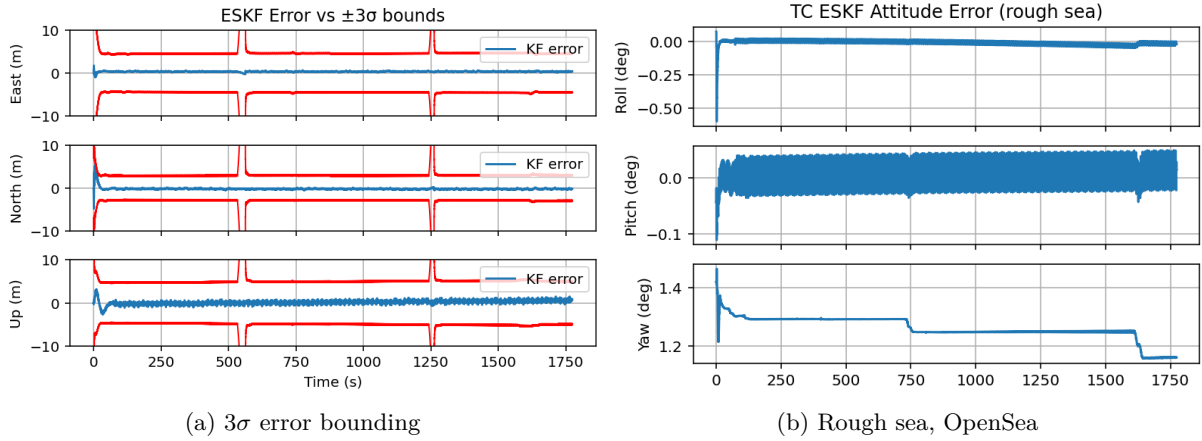


Figure 4.11: Error plot for rough sea scenario

Table 4.5 lists the statistics of the results. In the rough sea case, LC achieves better performance and also requires less processing time. We deduce that the open sea environment offers better line-of-sight conditions for the satellites, and therefore, in open-sea scenarios, the LC strategy is preferred.

Table 4.5: RMS of LC, TC and SPP (Rough sea)

RMS (m)	E	N	U	2D	approx. time (s)
LC	0.31100365	0.35606581	0.53236044	0.472764351	30s
TC	0.35117914	0.3463064	0.41630956	0.493208791	80s
SPP	0.31344906	0.27777494	0.68053783	0.418818852	—

4.2.4 Discussion of lever arm

The vessel is equipped with an INS located near the center of the boat and two GNSS antennas mounted on both sides. Only the left antenna is used in this thesis. Its distance from the IMU is -15.74 meters, which is long enough to have a significant impact on the results. Therefore, the effect of the lever arm is discussed in this subsection by using only LC.

The impact is particularly noticeable in the attitude estimation. In Fig. 4.12, the roll angle exhibits a drift over time in both cases when the lever arm is not included. This behavior can be explained by Eq. (3.23). When the lever arm is considered, the attitude becomes involved in the measurement matrix \mathbf{H} , making the system more observable and improving the estimation.

Meanwhile, with a single-antenna configuration, the estimation performance still heavily depends on the initial attitude and the vessels motion. Since there is no significant attitude changes for

the vessel in Open sea case, the filter lacks the information to separate different error sources, the gyroscope bias around the yaw axis remains almost unobservable. If the initial attitude is tuned more accurately, the overall performance can be improved.

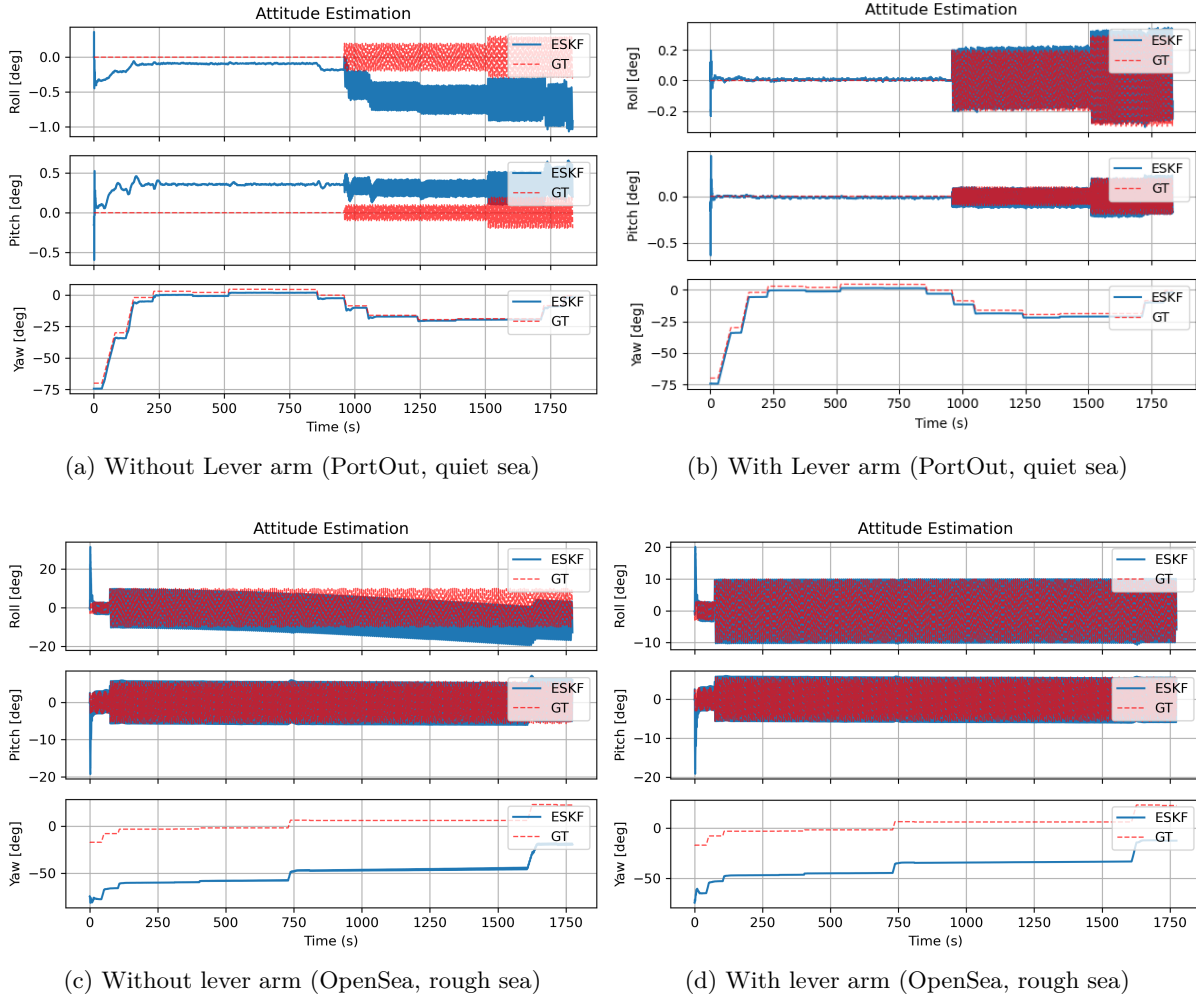


Figure 4.12: Discussion of lever arm

4.2.5 Ionosphere, troposphere correction estimation

Considering potential future applications, we would like to extend the filter to estimate additional error states. In this case, the satellite clock correction and Earth-rotation (Sagnac) compensation are handled inside the pseudorange model, while only the ionospheric and tropospheric delays are included in the error-state vector. The tropospheric delay is mapped to each satellite through an elevation-dependent mapping function. The purpose of this experiment is to compare the performance of this extended-state filter and to investigate whether the overall processing time can be reduced.

4.2 Dynamic data

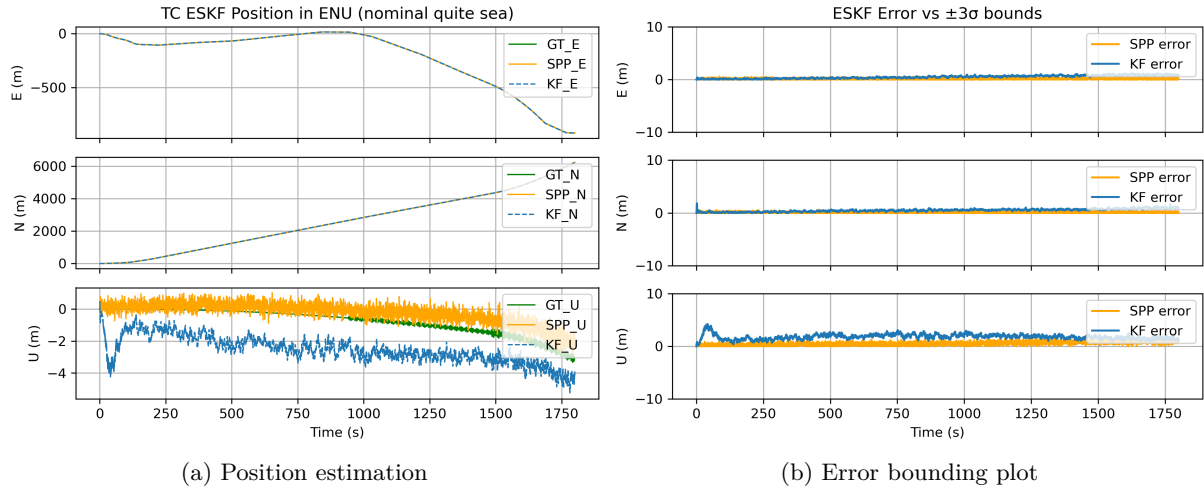


Figure 4.13: Results of TC w/ GNSS outage (rough sea, open-sea)

From the results in Fig. 4.13, the positioning error becomes larger, especially in the vertical direction, and the execution time has no significant improvement. Since the experiment was conducted in an open-sea environment, in principle multipath can be neglected. The main issue could be that the ionospheric and tropospheric delays are nearly constant in such conditions; therefore, if the filter estimates these delays incorrectly, the error is hardly observable and cannot be corrected. Based on these results, applying deterministic ionospheric and tropospheric models directly in the pseudorange correction remains the preferred approach in our case.

Chapter 5

Conclusion and Future Works

This thesis presented the development of a Python-based GNSS/IMU integration framework targeted for future on-line applications. A quaternion-based ESKF was implemented for both LC and TC configurations.

In the methodology section, the necessary theoretical foundations used to perform the integration system are introduced. The implementation section details the sensors used and the chosen algorithms, covering the entire process from data extraction to the final estimation results, and guiding the reader in implementing the algorithm in accordance with the theoretical framework.

In the results, the filter was first examined using static data to understand the influence of tuning parameters and process noise settings. Afterwards, dynamic simulated data were used to assess performance under various conditions, including satellite outages and noisy environments. The position estimation results were satisfactory, and both LC and TC architectures were able to bridge short GNSS outages using IMU mechanization. However, the attitude estimation exhibited noticeable biases, especially in yaw, revealing the inherent observability limitations of single-antenna systems under weak dynamics. These results indicate that certain states cannot be fully corrected without richer motion or additional measurements. The effect of the lever arm between the IMU and GNSS antenna was also discussed. The experiments showed that it might lead to wrong attitude prediction if the lever arm is not properly accounted for.

Overall, the LC architecture was computationally efficient and reliable under open-sky conditions with strong GNSS availability. To confirm the robustness of the TC approach, further tests under challenging conditions with degraded satellite geometry or limited measurements are recommended.

Some detail for the implementation. The value should be carefully pass from one to another. For example, when we consider the lever arm, if the IMU position is used instead of the antenna position to compute the satellite geometry and predict the pseudorange, the resulting error can easily reach several meters. other functionality also need to be consider such as the clock offset detection.

Reproducing issues during development is often useful for debugging. Throughout the implementation of the ESKF, several challenges were encountered:

- **Body-frame definition:** The sign convention of the IMU z-axis must be verified; an incorrect convention may lead to quaternion normalization errors.
- **Angle-error definition:** The attitude error must be consistently defined either in the body frame or the navigation frame; inconsistencies can also cause quaternion-related issues.

- **IMU frequency:** The sampling interval dt is not constant, and improper time scaling can lead to significant drift.
- **Time alignment:** GNSS observations are typically in UTC, which differs from GPS time by 18 seconds, and IMU, GNSS, and ground-truth data may use different time units.
- **Euler-angle convention:** The rotation from the body frame to the navigation frame is applied in the order yaw pitch roll.
- **Filter tuning:** All noise parameters must be physically reasonable, and unit consistency is essential.
- **Interpolation:** When comparing results, mismatched timestamps may lead to large but misleading differences.

There remain several directions for future work. A priority is to investigate the source of the attitude estimation bias, which may relate to the gravity model, reference-frame handling, or observability issues. Additional improvements include IMU calibration, the use of ZUPT, and extending the measurement model to incorporate Doppler, carrier phase, multi-constellation data, and multi-antenna configurations to enhance robustness and yaw observability. The system should also be validated with diverse real-world datasets and further assessed under limited satellite visibility. Developing a fully online processing chain, including real-time LC/TC ESKF implementations and integration within a GNU Radio framework, represents another promising direction. Finally, integrity monitoring and fault detection mechanisms, as well as the use of advanced correction services such as PPP or HAS-based PPP and optimization-based approaches, could significantly improve the reliability and overall accuracy of the system.

Bibliography

- [1] A. Soloviev, “Gnss-ins integration,” *Position, Navigation, and Timing Technologies in the 21st Century: Integrated Satellite Navigation, Sensor Systems, and Civil Applications, Volume 2*, p. 1447, 2020.
- [2] M. Weiss and K. Shenoi, “Tdev then and now,” in *Proceedings of the International Timing & Sync Forum (ITSF) 2015*, (Edinburgh, United Kingdom), 2015. Presentation, 22 October 2015.
- [3] P. D. Groves, *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*. Artech House, 2 ed., 2013.
- [4] J. Sola, “Quaternion kinematics for the error-state kalman filter,” *arXiv preprint arXiv:1711.02508*, 2017.
- [5] L. Grundhöfer, F. G. Rizzi, S. Gewies, M. Hoppe, J. Bäckstedt, M. Dziewicki, and G. Del Galdo, “Positioning with medium frequency r-mode,” *Navigation*, vol. 68, no. 4, pp. 829–841, 2021.
- [6] ESA&GSSC, “Gps signal plan navipedia.” https://gssc.esa.int/navipedia/index.php/GPS_Signal_Plan. Accessed: 2025-12-03.
- [7] ESA&GSSC, “Galileo signal plan navipedia.” https://gssc.esa.int/navipedia/index.php/Galileo_Signal_Plan. Accessed: 2025-12-03.
- [8] N. Boguspayev, D. Akhmedov, A. Raskaliyev, A. Kim, and A. Sukhenko, “A comprehensive review of gnss/ins integration techniques for land and air vehicle applications,” *Applied Sciences*, vol. 13, no. 8, p. 4819, 2023.
- [9] Z. Gao, H. Zhang, M. Ge, X. Niu, W. Shen, J. Wickert, and H. Schuh, “Tightly coupled integration of multi-gnss ppp and mems inertial measurement unit data,” *GPS solutions*, vol. 21, no. 2, pp. 377–391, 2017.
- [10] P. Teunissen and A. Khodabandeh, “Review and principles of ppp-rtk methods,” *Journal of Geodesy*, vol. 89, no. 3, pp. 217–240, 2015.
- [11] D. Titterton and J. L. Weston, *Strapdown inertial navigation technology*, vol. 17. IET, 2004.
- [12] M. G. Petovello, “Real-time integration of a tactical-grade imu and gps for high-accuracy positioning and navigation.,” 2003.
- [13] E.-H. Shin, “Estimation techniques for low-cost inertial navigation,” 2005.
- [14] V. Gikas and H. Perakis, “Rigorous performance evaluation of smartphone gnss/imu sensors for its applications,” *Sensors*, vol. 16, no. 8, p. 1240, 2016.

Bibliography

- [15] T. Li, H. Zhang, Z. Gao, Q. Chen, and X. Niu, “High-accuracy positioning in urban environments using single-frequency multi-gnss rtk/mems-imu integration,” *Remote sensing*, vol. 10, no. 2, p. 205, 2018.
- [16] P. Zhang, J. Gu, E. Milios, and P. Huynh, “Navigation with imu/gps/digital compass with unscented kalman filter,” in *IEEE International Conference Mechatronics and Automation*, 2005, vol. 3, pp. 1497–1502 Vol. 3, 2005.
- [17] G. Falco, M. Pini, and G. Marucco, “Loose and tight gnss/ins integrations: Comparison of performance assessed in real urban scenarios,” *Sensors*, vol. 17, no. 2, p. 255, 2017.
- [18] Y. Dong, D. Wang, L. Zhang, Q. Li, and J. Wu, “Tightly coupled gnss/ins integration with robust sequential kalman filter for accurate vehicular navigation,” *Sensors*, vol. 20, no. 2, p. 561, 2020.
- [19] W. Wen, T. Pfeifer, X. Bai, and L.-T. Hsu, “Factor graph optimization for gnss/ins integration: A comparison with the extended kalman filter,” *NAVIGATION: Journal of the Institute of Navigation*, vol. 68, no. 2, pp. 315–331, 2021.
- [20] E. A. Wan and R. Van Der Merwe, “The unscented kalman filter for nonlinear estimation,” in *Proceedings of the IEEE 2000 adaptive systems for signal processing, communications, and control symposium (Cat. No. 00EX373)*, pp. 153–158, Ieee, 2000.
- [21] u-blox AG, *ZED-F9R Integration Manual*. u-blox AG, Thalwil, Switzerland, 2023. Available online: https://content.u-blox.com/sites/default/files/ZED-F9R_Integrationmanual_UBX-20039643.pdf.
- [22] Canadian Geodetic Survey, Natural Resources Canada, “Precise point positioning (csrs-ppp).” Web site, <https://webapp.csrs-scrs.nrcan-rncan.gc.ca/geod/tools-outils/ppp.php>, 2025. Accessed: 2025-11-20.
- [23] J. R. Carpenter and C. N. D’Souza, “Navigation filter best practices,” tech. rep., 2025.
- [24] Z. Shen, D. Svehla, and T. Springer, *GNSS Data Processing: Volume IFundamentals and Algorithms*. European Space Agency (ESA), 2013.