Universidad de Málaga

Escuela Técnica Superior de
Ingeniería de Telecomunicación

TRABAJO FIN DE MÁSTER

Real-time INS/GNSS Integration of Low-cost IMU and Multi-element Antenna Array Receiver

Máster en Ingeniería de Telecomunicación

Pablo Mateos Ruiz

Málaga, 2021

E.T.S. de Ingeniería de Telecomunicación
Universidad de Málaga

Real-time INS/GNSS Integration of Low-cost IMU and Multi-element Antenna Array Receiver

Author: Pablo Mateos Ruiz

Supervisor: Carlos Camacho Peñalosa

Co-Supervisor: Elena Abdo Sánchez

Department: Ingeniería de Comunicaciones (IC)

Degree: Máster en Ingeniería de Telecomunicación

Keywords: GNSS, integration, position, attitude, estimation, Kalman filter, inertial sensors, direction of arrival, real-time, C++.

Abstract

Safety-critical applications such as autonomous cars and small UAVs have gained popularity over the last years. These require a good knowledge of their position and attitude at all times. Recent research has shown that multi-antenna GNSS receivers can provide an accurate attitude estimation in addition to the position. However, they are sensitive to interference and can not output a solution at high rates. Thus, these systems can benefit from the use of an inexpensive inertial measurement unit (IMU), as it provides measurements that are always available. In this work, an INS/GNSS integrated system has been implemented, which is able to take advantage of the multi-antenna receiver capabilities in order to obtain a robust and accurate vehicle attitude estimation.

The design of the system architecture is presented, along with the different algorithms used for the navigation solution computation. In addition, several mathematical models have been developed for the designed Kalman filter used in the integration. This filter integrates position in a loosely-coupled way while the attitude is integrated from the one computed by the GNSS receiver or by directly using the direction of arrival (DOA) of each visible satellite, significantly improving the attitude accuracy. Additional layers of protection have been included into the filter to prevent faulty masurements from corrupting the final solution.

The designed algorithms have been implemented in a C++ framework able to support both offline and real-time operation modes. The algorithms and code have been validated and tested using a small platform with a low-cost IMU and GALANT, the GNSS receiver with multi-antenna support developed at DLR, through a measurement campaign carried out with the DLR MessBus.

E.T.S. de Ingeniería de Telecomunicación
Universidad de Málaga

Integración INS/GNSS en Tiempo Real con IMU de Bajo Coste y Receptor con Sistema de Antenas Multi-elemento

Autor: Pablo Mateos Ruiz

Tutor: Carlos Camacho Peñalosa

Cotutora: Elena Abdo Sánchez

Departamento: Ingeniería de Comunicaciones (IC)

Titulación: Máster en Ingeniería de Telecomunicación

Palabras clave: GNSS, integración, posición, orientación, estimación, filtro de Kalman, sensores inerciales, dirección de llegada, tiempo real, C++.

Resumen

En los últimos años, algunas aplicaciones emergentes en los que la seguridad es esencial han ganado popularidad, como los coches autónomos o pequeños UAVs. Éstos requieren saber su posición y orientación de manera precisa en todo momento. Investigaciones recientes han demostrado que los receptores GNSS multiantena pueden proporcionar una estimación precisa de la orientación además de la posición, pero son sensibles a las interferencias y no pueden proporcionar una solución a frecuencias muy altas. Por ello, estos sistemas pueden beneficiarse del uso de una unidad de medición inercial (IMU) de bajo coste, ya que suministra mediciones que están siempre disponibles. En este trabajo se ha implementado un sistema integrado INS/GNSS que es capaz de aprovechar las capacidades del receptor multiantena para obtener una estimación robusta y precisa de la orientación del vehículo.

Se presenta el diseño de la arquitectura del sistema, junto con los diferentes algoritmos utilizados para el cálculo de la solución de navegación. Además, se han desarrollado varios modelos matemáticos para el filtro de Kalman que se ha diseñado y utilizado en la integración. Este filtro es capaz de integrar la orientación de GNSS a partir de la calculada por el receptor o utilizando directamente la dirección de llegada (DOA) de cada satélite visible, mejorando significativamente la precisión de la orientación estimada. Se han incluido capas adicionales de protección en el filtro para evitar que posibles mediciones erróneas corrompan la solución final.

Los algoritmos diseñados se han implementado en un entorno C++ capaz de operar tanto en tiempo real como en diferido. Los algoritmos y el código han sido validados y probados utilizando una pequeña plataforma hardware con una IMU de bajo coste y GALANT, un receptor GNSS con soporte multiantena desarrollado en el DLR, a través de una campaña de medidas realizada con el MessBus del DLR.

This master thesis has been developed at the Institute of Navigation and Communications of the German Aerospace Center (DLR) in Munich, under the supervision of Omar García Crespillo and Lothar Kurz.

Este trabajo fin de máster se ha desarrollado en el Instituto de Navegación y Comunicaciones del Centro Aeroespacial Alemán (DLR) en Munich, bajo la supervisión de Omar García Crespillo y Lothar Kurz.

Acknowledgements

I would like to thank Omar García for giving me the opportunity to develop my master thesis in the DLR under his supervision. Thanks are also given to my co-supervisor, Lothar Kurz, who has greatly helped me, especially during the laborious measurement campaign days. I extend my gratitude to my tutors from the University of Málaga, Elena and Carlos. They always have kind words for me while giving constructive suggestions.

I am also thankful to my workmates at DLR for the time they spent with me, enriching this experience. A special acknowledgement goes to Philipp Rudnik for his hard work improving the hardware interface that I have used to develop this work.

Thanks are also given to my friends from the University of Málaga. I sincerely appreciate their friendship and the significant assistance they constantly gave me during these years in the university. In particular, I am thankful to my colleagues Lorena, Paco and Carlos, who have shared this experience with me in Munich. These months abroad would certainly not have been the same without you.

Special thanks go to Isa for her everlasting patience, encouragement, understanding and caring support, which have kept me going through the more demanding days of the development of this thesis.

Finally, my deepest gratitude goes to my family members, in particular to my parents. They have always helped and guided me through all my decisions, and they continue to be an invaluable source of love and support in every situation. They have made it possible for me to be here today writing these lines.

Acronyms

DOA	Direction of Arrival
DCM	Direction Cosine Matrix
DCV	Direction Cosine Vector
DLR	Deutsches Zentrum für Luft- und Raumfahrt
ECEF	Earth-Centered Earth-Fixed
ECI	Earth-Centered Inertial
EKF	Extended Kalman Filter
ENU	East-Norht-Up
FRD	Forward-Right-Down
FSM	Finite-State Machine
GM	Gauss-Markov
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GST	Greenwich Sidereal Time
GT	Ground Truth
IDE	Integrated Development Environment
i.i.d.	independent and identically-distributed
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
KF	Kalman Filter
LS	Least Squares

NED	North-East-Down
NMEA	National Marine Electronics Association
PVT	Position, Velocity and Time
RMSE	Root-Mean-Square Error
UAV	Unmaned Aerial Vehicle
UTC	Coordinated Universal Time

Contents

Acknowledgements **vii**

Acronyms **ix**

1 Introduction **1**
1.1 Background . . . 1
1.2 Motivation . . . 2
1.3 Research Objectives . . . 3
1.4 Methodology . . . 3
1.5 Structure of the Thesis . . . 4

2 Inertial Navigation System Overview **5**
2.1 Coordinate Frames and Transformations . . . 5
2.2 INS Kinematics and Mechanization . . . 11
2.3 Inertial Sensor Errors . . . 14
2.4 INS Alignment . . . 16
2.4.1 Leveling . . . 16
2.4.2 Gyrocompassing . . . 17

3 INS/GNSS Integration **19**
3.1 Integration Benefits and Architectures . . . 19
3.2 Kalman Filter Theory . . . 21
3.2.1 Definitions and Algorithms . . . 21
3.2.2 Extended Kalman Filter . . . 24
3.3 Loosely-Coupled Integration . . . 25

4 Navigation Processor Design **33**
4.1 Operational Design and State Machine . . . 33
4.2 Propagation of Attitude Uncertainties . . . 37
4.3 Loosely Coupled System . . . 41
4.3.1 Inertial Sensor Error Models . . . 41
4.3.2 Kalman Filter System Model . . . 42
4.3.3 Kalman Filter Measurement Models . . . 44
4.4 DOA-aided Tight Attitude Integration . . . 46
4.4.1 DOA Handling and Advantages . . . 46

4.4.2 DOA Measurement Model 48
4.5 Fault and Spoofing Detection 49

5 Hardware Setup and Software Overview 51
5.1 Hardware Platform 51
5.2 Navchip IMU 52
5.3 DLR GALANT GNSS Receiver 54
5.4 Software Implementation 55
5.4.1 Picozed Launcher 55
5.4.2 INS/GNSS Controller 56

6 Results and Evaluation 61
6.1 Test Description and Ground Truth 61
6.2 Base INS/GNSS Integration Solution 62
6.2.1 Analysis of the Results 64
6.2.2 Position Fault Detection 68
6.3 Loose-Attitude KF Solution 69
6.3.1 Performance Improvements 69
6.3.2 Attitude Fault Detection 71
6.4 Tight-Attitude KF Solution 72
6.4.1 Performance Improvements 72
6.4.2 DOA Filtering and Fault Detection 72
6.5 Comparison between KF Configurations 77

7 Conclusions and Outlook 79
7.1 Achievements 79
7.2 Conclusions 80
7.3 Outlook 81

A KF Transmission Matrices 83

B Resumen 85
B.1 Motivación y Objetivos 85
B.2 Logros 86
B.3 Conclusiones 87
B.4 Líneas futuras 88

Bibliography 93

List of Figures

2.1 The ECI frame. 6
2.2 The ECEF and NED local frames. 7
2.3 The FRD body frame (Modified resource from Freepik). 8
2.4 Diagram of the mechanized strapdown algorithm in the n-frame. . . . 12

3.1 INS/GNSS integration operation example. 20
3.2 Kalman filter elements and interactions diagram (adapted from [1]). . 23
3.3 Flow diagram of the Kalman filter prediction/update scheme. 24
3.4 Loosely-coupled INS/GNSS integration system. 26

4.1 Navigation Processor FSM diagram. 34
4.2 Attitude covariance propagation process. 40
4.3 Elevation and Azimuth with respect to NED local and FRD body frames. 47

5.1 Diagram of the hardware platform connections. 52
5.2 Top view of the board with the IMU mounted. 53
5.3 Data flow diagram of the Picozed Launcher program. 56
5.4 Diagram of the INS/GNSS Controller modes. 59

6.1 MessBus with test equipment mounted on the roof. 62
6.2 Ground truth solution. 63
6.3 Base system coarse alignment performance. 64
6.4 Base system IMU biases EKF estimation. 65
6.5 Base system navigation solution. 66
6.6 Base system navigation solution error (Estimation minus GT). 67
6.7 Base EKF position fault detection. 68
6.8 Loose system coarse alignment performance. 69
6.9 Loose system navigation solution error (Estimation minus GT). . . . 70
6.10 Loose system EKF attitude fault detection. 71
6.11 Tight system navigation solution error (Estimation minus GT). . . . 73
6.12 DOAs computed from the Ephemeris by the GNSS receiver ($\mathbf{d}_j^n$). . . . 74
6.13 Comparison of DOA skyplots in antenna frame. 75
6.14 Comparison of DOA skyplots in GT local frame. 75
6.15 Fault detection with antenna-obtained DOAs in GT local frame. . . . 76

6.16 Antenna-obtained DOAs in GT local frame without discarded measurements. 77
6.17 Comparison of the root-mean-square error (RMSE) of the different EKF configurations. 78

List of Tables

3.1 Comparison of INS and multi-antenna GNSS characteristics. 20

4.1 Navigation Processor FSM states. 33

5.1 NavChip IMU characteristics assumed in the implemented system. . . 53
5.2 GNSS receiver statistics assumed in the implemented system. 55

Chapter 1

Introduction

The purpose of this chapter is to present the topic of this master thesis. First, some background on satellite and inertial navigation technologies is given, presenting their advantages and limitations. This basic presentation helps understanding the motivation that have led to the development of this work. Next, the objectives pursued by this work are pointed out, as well as the methodology followed. Finally, the thesis outline is provided.

1.1 Background

Since the deployment of the first Global Navigation Satellite Systems (GNSS), they have been increasingly used for several navigation and localization-based services, such as car and aircraft route planning, map creation or different positioning applications. GNSS importance has reached a point where it has become an essential part of most applications where position estimation is needed.

A GNSS is able to continuously provide accurate position and velocity information thanks to its satellite constellations. These satellites, distributed along different Earth orbits, broadcast signals with navigation messages containing information about timing parameters and the satellite position, known as Ephemeris. When signals from at least four satellites reach the GNSS user equipment, it can compute a position, velocity and time (PVT) solution by solving the associated geometrical problem.

However, GNSSs are sensitive to satellite visibility conditions and interference, which limits their use in applications where robust positioning is needed. Hence, these systems can benefit from the use of an inertial measurement unit (IMU), whose measurements are always available, as it is not sensitive to interference and it has high output rates.

Furthermore, advances in micro-electro-mechanical systems (MEMS) technologies have permitted the development of small and low-cost IMUs. These sensors have developed rapidly during the last decades, and are widely used in inertial navigation systems (INS) for land and air vehicles. However, in contrast to higher-grade systems, a low-cost INS can suffer from large position and attitude errors over the

long term, caused by the significant uncertainty in the sensor output.

In addition, it is challenging for these sensors to continuously provide a precise attitude determination due to the known observability problem of their biases and, particularly, of the vehicle heading determination.

1.2 Motivation

The last years have seen an increasing interest in several emerging safety-critical applications, such as autonomous cars or small unmanned aerial vehicles (UAVs). GNSSs are commonly used as the base for any of these applications. However, it is crucial for services related to safety to have an accurate and robust position and attitude estimation, even in the presence of interference.

From the basic information given about GNSS and INS, it is extracted that the integration of both systems is thus a practical approach for land and airborne applications where robust positioning is required. However, the accurate attitude determination continues to be a challenge, as GNSS itself does not provide an attitude solution, and an IMU only has observability of the vehicle heading in high dynamics conditions.

Some techniques about attitude estimation can be found in the state of the art. For example, there are vision-based techniques [2], but these require cameras and highly depend on lighting conditions. There are other techniques based on GNSS, such as the vehicle velocity alignment, in which the estimated heading is assumed to point in the direction of movement. However, this can only be used with some vehicles, like cars, and the motion constraints must be designed into the system [3], thus making it specific for the type of vehicle.

In this context, in the Institute of Navigation and Communications of the German Aerospace Center (DLR), the benefits of using multi-antenna GNSS receivers are being studied as a way to both estimate the vehicle attitude and limit possible interference through beamforming techniques [4, 5, 6]. The used technique takes advantage of the GNSS signal phase difference between the multiple elements of the antenna array. Through signal processing techniques, an estimation of the direction of arrival of each satellite signal can be obtained, which can be used for attitude determination. This technique obtains an attitude that is stable over time and does not drift. Furthermore, it allows for a small separation between antennas, it is theoretically usable in challenging environment as it only needs limited satellite visibility, and it has spoofing (intentional attacks which send fake GNSS signals) detection capabilities.

In this sense, research on multi-antenna GNSS receivers has been the subject of increasing interest. However, it has some difficulties associated. For example, multi-antenna equipment is not very common and they have been mostly used in high-grade solutions. Furthermore, these techniques are still in development and they have not been integrated with other systems yet.

Therefore, the use of this technique in combination with the INS continuous and high output rate solution aims to become a powerful approach to the robust and

precise attitude estimation problem for safety-critical applications with cost and space restrictions, and for which new integration algorithms are required.

1.3 Research Objectives

This work attempts to contribute to the research on accurate and robust attitude determination by combining the information from a multi-antenna GNSS receiver and an IMU. The main objectives of this thesis are:

- The theoretical design of the necessary mathematical models and algorithms to integrate the IMU and the multi-antenna GNSS receiver data.
- Development of software to perform the designed algorithms.
- Evaluation of the algorithms with real data using a low-cost IMU and GALANT (the DLR real-time GNSS receiver with multi-antenna support). Compare its performance to basic INS/GNSS integration without the aid of GNSS attitude.

1.4 Methodology

The development of this thesis has been carried out following a methodology that can be divided in different phases:

1. **Previous study and documentation:** This phase consists of collecting information about the typical algorithms used with inertial sensors and techniques for INS/GNSS integration, particularly Kalman filters. A review about the different attitude estimation techniques is carried out, focusing on the challenges of IMU self-alignment processes.
2. **GNSS receiver output data:** In this second stage, the multi-antenna GNSS receiver outputs are interpreted in order to derive which information is available for their use in the integration system, and how to extract the required data.
3. **Theoretical derivation:** This third phase consists of the development of the necessary Kalman filter extensions that allow the integration of GNSS attitude to the system. In particular, the use of the direction of arrival of each satellite directly into the filter required new mathematical models.
4. **Software implementation:** Once the design of the whole system is complete, it needs to be implemented in a framework that allows real-time operation. The navigation system is programmed in C++ to be executed in a small hardware platform. In addition, simple input interfaces are needed to make the system convenient to use with different possible sensors.

5. **Experimental evaluation:** Finally, the implemented system must be tested with real data. Thus, a measurements campaign is performed and results are obtained. These results are processed with different configurations and comparisons between them are shown, mainly focusing on the attitude estimation performance.

1.5 Structure of the Thesis

The rest of this thesis is organized in six chapters as follows: Chapter 2 introduces the basic theoretical background on inertial navigation systems, including the mechanization equations. In addition, descriptions on inertial sensor errors and some common alignment techniques are given. Chapter 3 develops the benefits about INS/GNSS integration, explaining the theory behind Kalman filter-based estimation. Furthermore, the loosely-coupled integration scheme commonly used for navigation is presented, along with its mathematical models. In Chapter 4, the actual system design is described, along with the extension of the loosely-coupled Kalman filter architecture. The derivation of the algorithms for improving the attitude estimation and fault detection tests are explained. Chapter 5 presents the used hardware platform with some of its characteristics. An overview of the software implementation of the real-time navigation system is also given. In Chapter 6, the evaluation of the system is provided through the results obtained from the measurements campaign. Finally, Chapter 7 summarizes the most important conclusions, emphasizing the achievements of this work and suggesting possible future work.

Chapter 2

Inertial Navigation System Overview

A navigation system determines the position, velocity and, potentially, attitude (orientation) of a body with respect to a known reference frame in a sustained, mechanized way. In particular, an Inertial Navigation System (INS) uses the measurements from an Inertial Measurement Unit (IMU) to compute the *navigation solution*. An IMU is usually composed of 3 accelerometers and 3 gyroscopes mounted in an orthogonal way, to account for each of the three-dimensional axes.

While gyroscopes measure *angular rates*, accelerometers measure *specific forces*, which are the accelerations due to all forces except for gravity. These IMU outputs are fed into the navigation processor, which integrates them through a set of mechanization equations. As accelerometers sense the reaction to gravity, these equations must incorporate a gravity model to compensate for this effect (i.e. to integrate only the accelerations due to the user body movements). This way, the combination of IMU and navigation processor forms the INS.

In this chapter, first a brief description is given about the coordinate frames usually used in navigation and transformations between them. Then, the inertial navigation equations are introduced, explaining how to mechanize them for discrete-time computation in the INS. After that, some typical systematic errors present in IMU measurements are described and how to correct them. Last, some notes about how initial INS alignment is carried out through typical processes such as *leveling* and *gyrocompassing*, detailing their advantages and limitations.

2.1 Coordinate Frames and Transformations

The description of the position, velocity and attitude of a body first requires defining with respect to which axes and point in space they need to be calculated. The combination of a set of three mutually perpendicular axes and an origin point comprises an orthogonal *coordinate frame*. Any navigation problem involves at least two of these: the **object frame**, which describes the body for which the position and orientation are desired, and the **reference frame**, describing the known body relative to which the solution is required. Usually, a third one called the **resolving frame** is also used to explicit in which axes the quantities are expressed.

Regarding this basic knowledge about frames, in this work, the notation $\mathbf{x}^{\gamma}_{\beta\alpha}$ represents a vector describing a kinematic property of frame α (body) with respect to frame β (reference), expressed in the frame γ (resolving) axes.

Earth-Centered Inertial Frame

The ECI frame, denoted by the symbol i, has its origin at the Earth's centre of mass. As it is an inertial frame, its axes do not rotate with the Earth, and they are described as follows:

- x^i is the direction from the Earth to the Sun at the vernal equinox.
- y^i points 90° ahead of the x-axes in the direction of the Earth's rotation.
- z^i points along the rotation axis of the Earth (North Pole).

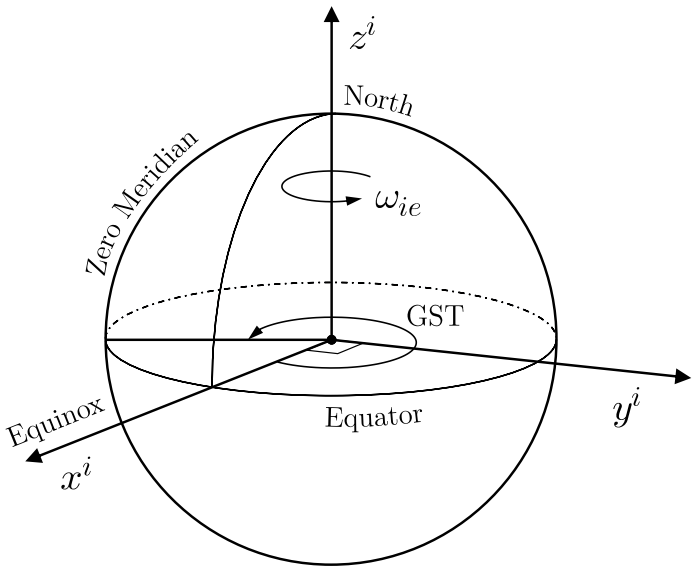


Figure 2.1: The ECI frame.

Earth-Centered Earth-Fixed Frame

The ECEF frame, denoted by the symbol e, is very similar to the ECI one. Its centre is the same, but its axes rotate along with the Earth, as if they were attached to it, with an angular rate $\omega_{ie} = 7.292\,115 \times 10^{-5}\,\text{rad/s}$ with respect to the i-frame. They are described as follows (Fig. 2.2):

- x^e points to the Conventional Zero Meridian (0° longitude).
- y^e towards the intersection of the equator with the 90° East meridian.
- z^e points along the rotation axis of the Earth (same as i-frame).

Local Navigation Frame

A local navigation frame, denoted by the symbol n, is typically used for representing vehicle velocity and attitude in near Earth's surface operation. Its origin is the object of the navigation solution itself, and its axes are aligned with the topographic directions. Although they can vary with the convention used, the most common in inertial applications is the North-East-Down (NED) orientation, which is the one used in this work and described as follows:

- x^n in the direction of increasing latitude from the object position (N).
- y^n points towards the east, to complete the right-handed set (E).
- z^n is normal to the surface of the reference ellipsoid pointing downwards (D).

This system is also shown in Fig. 2.2, where L and λ refer to the geodetic latitude and longitude, respectively. It can be seen that the z-axis does not necessarily points towards the centre of the Earth, due to the ellipsoid shape of the globe.

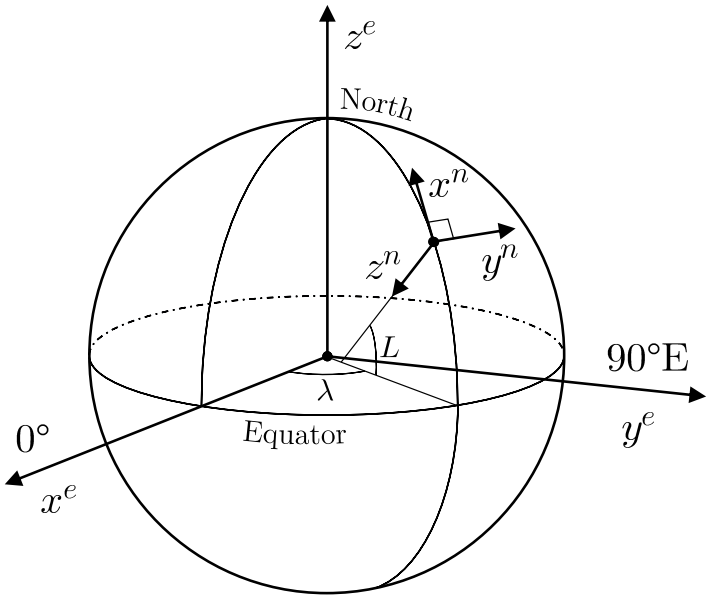


Figure 2.2: The ECEF and NED local frames.

Moreover, another typical convention is the East-North-Up (ENU). Both of these conventions are widely used in navigation and GNSS applications.

Body Frame

A body frame, denoted by the symbol b, describes the origin and orientation of the vehicle itself. Its centre and axes are fixed with respect to the body. Although there are also different conventions for this type of frame, in this work the used one is Forward-Right-Down (FRD):

- x^b is the usual direction of travel (Forward).
- y^b points to the right of the vehicle, to complete the right-handed set (Right).
- z^b is the down axis (Down).

An example of this frame within a drone is shown in Fig. 2.3. Regarding angular motion, these axes are also known as roll, pitch and yaw, respectively. These angles describe how the vehicle is rotated with respect to each axis in the initial position, with increasing values based on the right-handed corkscrew rule. They will be further explained in the following subsection.

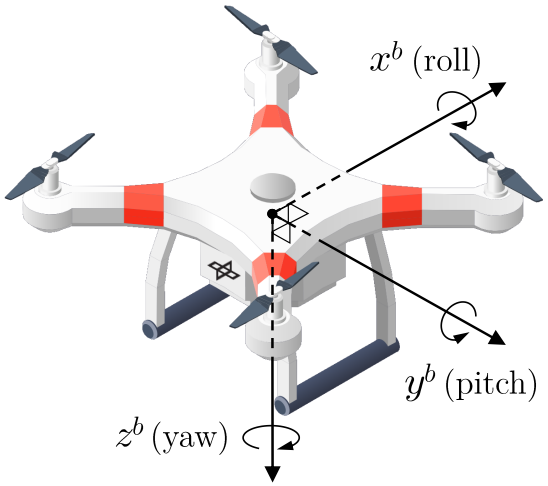


Figure 2.3: The FRD body frame (Modified resource from Freepik).

Coordinate Transformations

Once all the necessary frames have been described, it is important to know how to transform a kinematic property vector from one set of resolving axes to another. This can be carried out with different methods, such as Euler rotations and Direct Cosine Matrices (DCMs). The latter one is a 3x3 rotation (or transformation) matrix denoted by $\mathbf{C}_\alpha^\beta$ which transforms a vector from the α resolving frame to the β one:

$$\mathbf{x}^\beta = \mathbf{C}_\alpha^\beta \mathbf{x}^\alpha. \tag{2.1}$$

In addition, DCMs also have some important properties. A rotation can be reversed by transposing the matrix, and successive rotations are performed by simply using the multiplicative chain rule. Both of these properties are expressed as follows:

$$\mathbf{C}_\beta^\alpha = \left(\mathbf{C}_\alpha^\beta\right)^{\mathrm{T}}, \qquad \mathbf{C}_\alpha^\gamma = \mathbf{C}_\beta^\gamma \mathbf{C}_\alpha^\beta. \tag{2.2}$$

On the other hand, **Euler angles** are very useful to express attitude in terms of three successive rotations, namely roll (ϕ), pitch (θ) and yaw (ψ), each of them about one of the axis, as in Fig. 2.3. This method is usually used for representing the attitude from the local frame to the body frame, whose notation would be:

$$\mathbf{\Psi}_{nb} = \begin{bmatrix} \phi_{nb} \\ \theta_{nb} \\ \psi_{nb} \end{bmatrix}. \tag{2.3}$$

The order in which these rotations are applied is important, and is usually the following one, starting with the b-frame axes aligned with the n-frame ones: First, the yaw rotation is performed around the z-axis to bring the body to the desired azimuth, then the pitch rotation is carried out about the y-axis to reach the required elevation and, finally, the roll rotation, about the x-axis, takes the body to its final orientation.

For any conversion between two sets of resolving axes, there is a relation between the transformation matrix and the Euler angles associated. As mentioned before, this is usually used for expressing the attitude of the b-frame with respect to the n-frame, so the following matrix is already particularized for these:

$$\mathbf{C}_b^n = \begin{bmatrix} \mathrm{c}\theta\,\mathrm{c}\psi & -\mathrm{c}\phi\,\mathrm{s}\psi + \mathrm{s}\phi\,\mathrm{s}\theta\,\mathrm{c}\psi & \mathrm{s}\phi\,\mathrm{s}\psi + \mathrm{c}\phi\,\mathrm{s}\theta\,\mathrm{c}\psi \\ \mathrm{c}\theta\,\mathrm{s}\psi & \mathrm{c}\phi\,\mathrm{c}\psi + \mathrm{s}\phi\,\mathrm{s}\theta\,\mathrm{s}\psi & -\mathrm{s}\phi\,\mathrm{c}\psi + \mathrm{c}\phi\,\mathrm{s}\theta\,\mathrm{s}\psi \\ -\mathrm{s}\theta & \mathrm{s}\phi\,\mathrm{c}\theta & \mathrm{c}\phi\,\mathrm{c}\theta \end{bmatrix}, \tag{2.4}$$

where the subscripts of the angles have been ignored for simplicity, and "sin" and "cos" are abbreviated as "s" and "c", respectively. From this DCM, the Euler angles can also be extracted with the following expressions:

$$\phi_{nb} = \arctan_2(\mathbf{C}_{b\,3,2}^n, \mathbf{C}_{b\,3,3}^n) \tag{2.5}$$

$$\theta_{nb} = -\arcsin\big(\mathbf{C}_{b\,3,1}^n\big) \tag{2.6}$$

$$\psi_{nb} = \arctan_2(\mathbf{C}_{b\,2,1}^n, \mathbf{C}_{b\,1,1}^n) \tag{2.7}$$

with $\arctan_2(y, x)$ denoting a four-quadrant arctangent function.

When the Euler angles represent a small angular perturbation, the small angle approximation can be applied, and Eq. (2.4) becomes

$$\mathbf{C}_b^n \approx \begin{bmatrix} 1 & -\psi_{nb} & \theta_{nb} \\ \psi_{nb} & 1 & -\phi_{nb} \\ -\theta_{nb} & \phi_{nb} & 1 \end{bmatrix} = \mathbf{I}_3 + [\mathbf{\Psi}_{nb}\times], \tag{2.8}$$

where $[\mathbf{\Psi}_{nb}\times]$ denotes the skew-symmetric matrix of the Euler angles.

Moving on to specific transformations between frames used in this work, the rotation matrix between the e-frame and the n-frame is [1]:

$$\mathbf{C}_e^n = \begin{bmatrix} -\sin L_b \cos \lambda_b & -\sin L_b \sin \lambda_b & \cos L_b \\ -\sin \lambda_b & \cos \lambda_b & 0 \\ -\cos L_b \cos \lambda_b & -\cos L_b \sin \lambda_b & -\sin L_b \end{bmatrix}, \tag{2.9}$$

being L_b and λ_b the geodetic latitude and longitude of the local frame, respectively.

Besides transforming between resolving frames, it is also important in navigation to be able to change between position representations. There are two main ones: Cartesian and curvilinear. The first one describes the position of a body with respect to another one resolved in the axes of a third arbitrary frame:

$$\mathbf{r}_{\beta\alpha}^{\gamma} = \begin{bmatrix} x_{\beta\alpha}^{\gamma} & y_{\beta\alpha}^{\gamma} & z_{\beta\alpha}^{\gamma} \end{bmatrix}^{\mathrm{T}}. \tag{2.10}$$

On the other hand, the curvilinear representation describes a position in terms of geodetic latitude, longitude and height:

$$\mathbf{p}_b = \begin{bmatrix} L_b & \lambda_b & h_b \end{bmatrix}^{\mathrm{T}}. \tag{2.11}$$

This latter representation is very convenient as it offers a more familiar way of describing position using an ellipsoidal model of the Earth's surface. Indeed, this is how a position is usually represented in the n-frame. However, it is often more complex to operate with curvilinear positions at a mathematical level. Because of this, it is important to know how to convert it to the Cartesian representation.

There are different iterative [1] and closed-form methods [7] to carry out the conversion between the curvilinear position and the ECEF Cartesian one ($\mathbf{r}_{eb}^e$) in particular. However, there is a much simpler way of doing it when only small perturbations to the position apply, given by:

$$\boldsymbol{\delta}\mathbf{r}_{eb}^e \approx \mathbf{C}_n^e \mathbf{T}_p^{r(n)} \boldsymbol{\delta}\mathbf{p}_b, \qquad \boldsymbol{\delta}\mathbf{p}_b \approx \mathbf{T}_{r(n)}^p \mathbf{C}_e^n \boldsymbol{\delta}\mathbf{r}_{eb}^e \tag{2.12}$$

with

$$\mathbf{T}_p^{r(n)} = \begin{bmatrix} R_N + h_b & 0 & 0 \\ 0 & (R_E + h_b)\cos L_b & 0 \\ 0 & 0 & -1 \end{bmatrix}, \tag{2.13}$$

$$\mathbf{T}_{r(n)}^p = \begin{bmatrix} \dfrac{1}{R_N + h_b} & 0 & 0 \\ 0 & \dfrac{1}{(R_E + h_b)\cos L_b} & 0 \\ 0 & 0 & -1 \end{bmatrix}. \tag{2.14}$$

The meridian and transverse radii of curvature of the Earth are, respectively:

$$\begin{aligned} R_N(L) &= \frac{R_0\left(1-e^2\right)}{\left(1-e^2\sin^2 L\right)^{3/2}}, \\ R_E(L) &= \frac{R_0}{\sqrt{1-e^2\sin^2 L}}, \end{aligned} \tag{2.15}$$

being $R_0 = 6\,378\,137.0\,\text{m}$ the Equatorial radius of the Earth and $e = 0.08181919$ its eccentricity, based on the WGS-84 Ellipsoid [8].

2.2 INS Kinematics and Mechanization

The kinematics of a system refer to the mathematical description of its motion, as a function of its previously known position, velocity and attitude, and the applied accelerations and angular rates. The derivation of these inertial navigation equations, which is also known as the **strapdown computation**, depends on the chosen reference and resolving frames. In this work a local navigation frame approach is chosen as it already provides the navigation solution in an intuitive way for users operating on the Earth. Accordingly, the attitude is expressed with $\mathbf{C}_b^n$, the velocity is Earth-referenced in the n-frame axes:

$$\mathbf{v}_{eb}^n = \begin{bmatrix} v_N & v_E & v_D \end{bmatrix}^{\mathrm{T}}, \tag{2.16}$$

and position is expressed in the curvilinear form $\mathbf{p}_b$, given by Eq. (2.11).

On the other hand, **INS mechanization** refers to automatically converting the output of the IMU into the navigation solution (referenced to a certain frame) at discrete intervals. The strapdown equations can be divided into three parts: attitude, velocity and position updates. Due to the discrete and sequential nature of the mechanization, these must be performed in that exact order. The new velocity needs to be projected into the local frame with the already updated attitude, while the position update requires knowing the velocity at the current interval. This mechanized strapdown algorithm can be summarized as in the diagram in Fig. 2.4, which will be detailed in the following subsections.

Before presenting all the expressions, a brief note on notation: A skew-symmetric matrix is a square matrix whose transpose equals its negative. For a given rotation vector $\boldsymbol{\omega}$, its associated skew-symmetric matrix is defined as follows:

$$\boldsymbol{\Omega} = [\boldsymbol{\omega}\times] = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}. \tag{2.17}$$

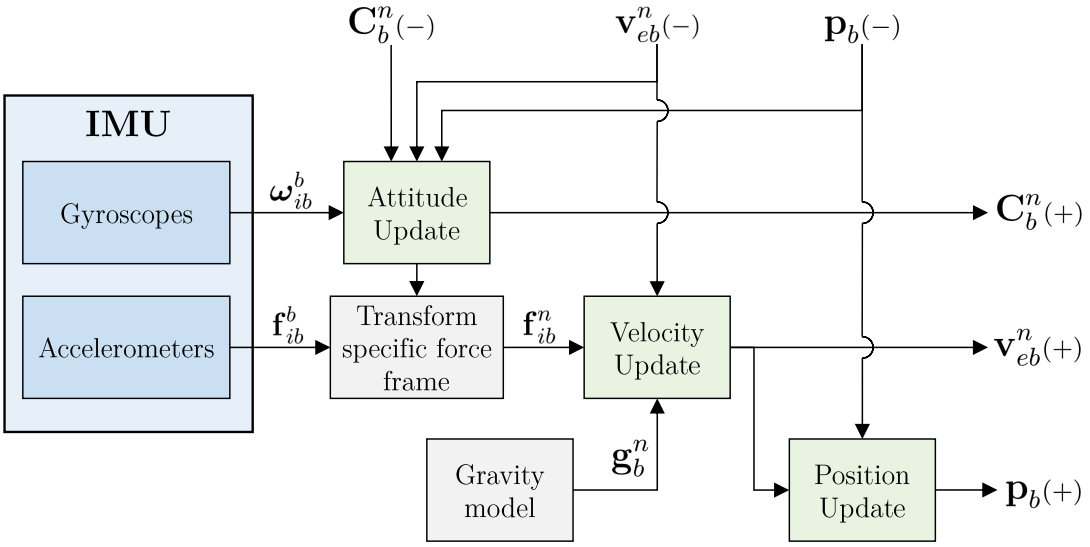


Figure 2.4: Diagram of the mechanized strapdown algorithm in the n-frame.

Attitude update

The **attitude** kinematics can be written as the following differential equation:

$$\begin{aligned}\dot{\mathbf{C}}_b^n &= \mathbf{C}_b^n \boldsymbol{\Omega}_{ib}^b - \boldsymbol{\Omega}_{in}^n \mathbf{C}_b^n \\ &= \mathbf{C}_b^n \boldsymbol{\Omega}_{ib}^b - \left(\boldsymbol{\Omega}_{ie}^n + \boldsymbol{\Omega}_{en}^n\right) \mathbf{C}_b^n,\end{aligned} \tag{2.18}$$

where the skew-symmetric matrices mean:

- $\boldsymbol{\Omega}_{ib}^b$ the angular rates measured by the gyroscopes;
- $\boldsymbol{\Omega}_{ie}^n$ the Earth rotation with respect to an inertial frame;
- $\boldsymbol{\Omega}_{en}^n$ the *transport rate*, which is the effect that arises from rotating the n-frame axis with respect to the e-frame.

The rotation vector of the e-frame with respect to the i-frame (Fig. 2.1) resolved in the former one is

$$\boldsymbol{\omega}_{ie}^e = \begin{bmatrix} 0 & 0 & \omega_{ie} \end{bmatrix}^{\mathrm{T}} \tag{2.19}$$

and, using Eq. (2.9), it can be resolved in the local frame:

$$\boldsymbol{\omega}_{ie}^n = \mathbf{C}_e^n \boldsymbol{\omega}_{ie}^e = \begin{bmatrix} \omega_{ie} \cos L_b & 0 & -\omega_{ie} \sin L_b \end{bmatrix}^{\mathrm{T}}. \tag{2.20}$$

On the other hand, $\boldsymbol{\omega}_{en}^n$ is given by [1]:

$$\boldsymbol{\omega}_{en}^n = \begin{bmatrix} \dot{\lambda}_b \cos L_b & -\dot{L}_b & -\dot{\lambda}_b \sin L_b \end{bmatrix}^{\mathrm{T}}. \tag{2.21}$$

However, it is more convenient to have this expression as a function of velocity and position. For this purpose, first the derivatives of the curvilinear position components need to be expressed that way [1], using Eq. (2.15):

$$\begin{aligned} \dot{L}_b &= \frac{v_N}{R_N + h_b}, \\ \dot{\lambda}_b &= \frac{v_E}{(R_E + h_b)\cos L_b}, \\ \dot{h}_b &= -v_D. \end{aligned} \tag{2.22}$$

Finally, substituting Eq. (2.22) in Eq. (2.21), the transport rate as function of position and velocity is:

$$\boldsymbol{\omega}_{en}^{n} = \begin{bmatrix} v_E / (R_E + h_b) \\ -v_N / (R_N + h_b) \\ -v_E \tan L_b / (R_E + h_b) \end{bmatrix}. \tag{2.23}$$

In order to mechanize this attitude kinematics expression, Eq. (2.18) needs to be integrated over the period of time between the current and previous IMU measurements. Given the discrete nature of the process, a first-order approximation can be obtained as:

$$\mathbf{C}_{b\,(+)}^{n} \approx \mathbf{C}_{b\,(-)}^{n} \left(\mathbf{I}_3 + \left[\boldsymbol{\omega}_{ib}^{b} \times \right] \Delta t \right) - (\boldsymbol{\Omega}_{ie(-)}^{n} + \boldsymbol{\Omega}_{en(-)}^{n}) \mathbf{C}_{b\,(-)}^{n} \Delta t, \tag{2.24}$$

where Δt is the time step, and "$_{(-)}$" and "$_{(+)}$" denote whether a value is obtained from the previous solution or the currently being computed, respectively.

However, this approximation lacks precision for most applications. A more precise formula can be used at the expense of a higher computational cost [9], making use of the DCM chain rule from Eq. (2.2):

$$\mathbf{C}_{b\,(+)}^{n} = \mathbf{C}_{n-}^{n+} \mathbf{C}_{b\,(-)}^{n} \mathbf{C}_{b+}^{b-} \tag{2.25}$$

where

$$\mathbf{C}_{b+}^{b-} = \mathbf{I}_3 + \frac{\sin(\|\boldsymbol{\alpha}\|)}{\|\boldsymbol{\alpha}\|} [\boldsymbol{\alpha}\times] + \frac{1 - \cos(\|\boldsymbol{\alpha}\|)}{\|\boldsymbol{\alpha}\|^2} [\boldsymbol{\alpha}\times]^2, \quad \text{with } \boldsymbol{\alpha} = \boldsymbol{\omega}_{ib}^{b} \Delta t \tag{2.26}$$

$$\mathbf{C}_{n-}^{n+} = \mathbf{I}_3 - \frac{\sin(\|\boldsymbol{\xi}\|)}{\|\boldsymbol{\xi}\|} [\boldsymbol{\xi}\times] + \frac{1 - \cos(\|\boldsymbol{\xi}\|)}{\|\boldsymbol{\xi}\|^2} [\boldsymbol{\xi}\times]^2, \quad \text{with } \boldsymbol{\xi} = \boldsymbol{\omega}_{in}^{n} \Delta t. \tag{2.27}$$

$\mathbf{C}_{b+}^{b-}$ accounts for the rotation of the body frame from the last integration step, while $\mathbf{C}_{n-}^{n+}$ does for the change of the local frame relative to non-rotating space.

Velocity update

The change in **velocity** is described by the expression below:

$$\dot{\mathbf{v}}_{eb}^{n} = \mathbf{C}_{b}^{n} \mathbf{f}_{ib}^{b} + \mathbf{g}_{b}^{n} - (2\boldsymbol{\Omega}_{ie}^{n} + \boldsymbol{\Omega}_{en}^{n}) \mathbf{v}_{eb}^{n}, \tag{2.28}$$

where

$\mathbf{f}_{ib}^{b}$ is the vector of specific forces measured by the accelerometers;

$\mathbf{g}_{b}^{n}$ is the vector of the gravity model in the local frame.

The last term in Eq. (2.28) compensates for the Coriolis effect, while $\mathbf{g}_{b}^{n}$ does for the specific force reactive to gravity. The values of the latter vary with height and position on the Earth. Furthermore, although its north and east components are not actually zero due to gravity anomalies, the local gravity vector can be approximated as having only the down component. Its value can be obtained through different models, like the following one [10]:

$$\begin{aligned} g_D(h) &= g_0 + h\left(-0.0000030877\left(1 - 1.39\times10^{-3}\sin^2 L\right)\right) \\ g_0 = g_D(0) &= 9.780318\left(1 + 5.3024\times10^{-3}\sin^2 L - 5.9\times10^{-6}\sin^2(2L)\right). \end{aligned} \tag{2.29}$$

In order to mechanize Eq. (2.28), the following discretized approximation can be applied:

$$\mathbf{v}_{eb(+)}^{n} \approx \mathbf{v}_{eb(-)}^{n} + \left[\mathbf{C}_{b}^{n}(+)\mathbf{f}_{ib}^{b} + \mathbf{g}_{b}^{n}(-) - (2\boldsymbol{\Omega}_{ie(-)}^{n} + \boldsymbol{\Omega}_{en(-)}^{n})\mathbf{v}_{eb(-)}^{n}\right]\Delta t. \tag{2.30}$$

Position update

Finally, using Eq. (2.14), the position changes following this kinematic expression:

$$\dot{\mathbf{p}}_b = \mathbf{T}_{r(n)}^{p}\mathbf{v}_{eb}^{n}, \tag{2.31}$$

which can be decomposed into the components already shown in Eq. (2.22).

In order to mechanize this last expression, the new curvilinear position components must be calculated using these approximations in the following order [1]:

$$\begin{aligned} h_{b(+)} &= h_{b(-)} - \frac{\Delta t}{2}\left(v_{D(-)} + v_{D(+)}\right), \\ L_{b(+)} &= L_{b(-)} + \frac{\Delta t}{2}\left(\frac{v_{N(-)}}{R_{N(-)} + h_{b(-)}} + \frac{v_{N(+)}}{R_{N(+)} + h_{b(+)}}\right), \\ \lambda_{b(+)} &= \lambda_{b(-)} + \frac{\Delta t}{2}\left(\frac{v_{E(-)}}{(R_{E(-)} + h_{b(-)})\cos L_{b(-)}} + \frac{v_{E(+)}}{(R_{E(+)} + h_{b(+)})\cos L_{b(+)}}\right). \end{aligned} \tag{2.32}$$

After these expressions are computed, the strapdown algorithm would be complete, as seen in Fig. 2.4.

2.3 Inertial Sensor Errors

Inertial sensors, as any other type of sensing devices, exhibit a variety of errors which deteriorate the accuracy of the measurements. Error sources usually have four components: a fixed contribution, temperature-dependent, run-to-run and in-run variations. The first two can be calibrated in the laboratory and corrected

by the sensor processor itself — although this may not even be done with low-cost sensors —, but the last two need to be estimated during operation.

Among the usual error sources, the following can be outlined:

- **Bias:** Constant error independent of the measured magnitude, usually denoted by $\mathbf{b}$. They produce a non-zero output when it should actually be so. These biases can also drift during operation, which can potentially be estimated as well, as will be seen in Subsection 4.3.1.
- **Scale factor and Cross-coupling:** They are errors proportional to the measured magnitude. While scale factors (denoted by $\mathbf{s}$) are misadjustments in the gradient between input and output, cross-coupling errors (denoted by $\mathbf{m}$) arise from the misalignments of the sensor axes due to manufacturing imperfections, making them not completely orthogonal.
- **Non-linearity:** These errors make the scale factor vary with higher-order power series of the measured magnitude. As its shape is difficult to model exactly, it is usually expressed as the scale factor variation over the operative range.
- **Random noise:** This error source is unpredictable and arises from electrical noise, mechanical instabilities or any other source which may interfere with the measured property. It is denoted by $\mathbf{w}$, and some of its components are usually modelled as zero-mean white Gaussian random variables. **Quantization noise**, which is the error from rounding the output to the nearest possible digital value, is also often seen as another source of random noise.

Thus, the following expression shows how these errors contribute to the sensor output (denoted with a tilde "~"), which although particularized for specific forces, is the same for angular rates:

$$\tilde{\mathbf{f}}_{ib}^{b} = \mathbf{b}_a + (\mathbf{I}_3 + \mathbf{M}_a)\,\mathbf{f}_{ib}^{b} + \mathbf{w}_a. \tag{2.33}$$

$\mathbf{M}_a$ is a matrix grouping the scale factor and cross-coupling errors as:

$$\mathbf{M}_a = \begin{bmatrix} s_{a,x} & m_{a,xy} & m_{a,xz} \\ m_{a,yx} & s_{a,y} & m_{a,yz} \\ m_{a,zx} & m_{a,zy} & s_{a,z} \end{bmatrix}. \tag{2.34}$$

It is important to correct as much of these errors as possible, as the INS integrates the IMU outputs making the navigation solution drift with time. Specifically, accelerometer errors lead to velocity and position errors that grow proportionally to the integration time and its square, respectively, while gyroscope errors make the computed attitude stray from its true value.

Assuming that some of the systematic errors are estimated (denoted with a hat "^"), the following expression can be applied to correct them:

$$\hat{\mathbf{f}}_{ib}^{b} = \left(\mathbf{I}_3 + \hat{\mathbf{M}}_a\right)^{-1} \left(\tilde{\mathbf{f}}_{ib}^{b} - \hat{\mathbf{b}}_a\right) \tag{2.35}$$

How to model and estimate some of these errors is explained in Subsection 4.3.1.

2.4 INS Alignment

The strapdown algorithm integrates the IMU measurements to calculate the new navigation solution from the previous one. Thus, when the INS starts, it is important to provide it with an initial solution from where it can continue. Position and velocity must be provided by an external entity, like GNSS, another INS or the last known position if the vehicle is static. Attitude can also be provided by external systems, all of them with their advantages and constraints. Some of them are described:

- **Magnetic compass:** It can align the orientation based on the Earth's magnetic field, but needs to compensate for the surroundings that may have an effect on the magnetic field measurements.
- **Vision-based techniques:** Some type of vision sensor can be used with Horizon detection, vanishing points or stereo-vision based techniques [11]. They have the disadvantages of being highly dependent on visibility and lighting conditions and requiring high computational loads.
- **Vehicle velocity alignment:** Taking advantage of some motion constraints related to most land vehicles, the x-axis can be aligned with the direction of travel obtained by another positioning system like GNSS. However, these constraints must be incorporated in the system design and cannot be easily generalized for other vehicles such as aircraft or ships.
- **Array-antenna carrier-phase detection:** Attitude can be obtained measuring the phase difference between close antennas, which helps knowing the direction of arrival of satellites. This method uses GNSS signals, which are usually necessary for position anyway, but is dependent on satellite visibility and multi-antenna equipment must be used. This is the technique this work focuses on, and will be further described in following sections.

However, attitude can be also obtained from *self-alignment*, which uses the IMU measurements themselves through different processes such as leveling and gyrocompassing. Accurate self-alignment requires high-grade gyroscopes, and that is why external references are often necessary with low-cost IMUs. In any case, these two processes are explained in the following subsection.

2.4.1 Leveling

To carry out the IMU attitude initialization, a calibration process called **leveling** is used to partially estimate the attitude of the body frame. It takes advantage of the knowledge that the only force that an object senses while it is stationary is the one from gravity. This way, attitude can be obtained from the transformation matrix of the following expression:

$$\mathbf{f}_{ib}^{b} = \mathbf{C}_{n}^{b}\mathbf{g}_{b}^{n}(L_b, h_b) \tag{2.36}$$

where $\mathbf{g}_b^n$ represents the specific force due to gravity that the body senses resolved in the local frame. In the NED n-frame, this vector would be

$$\mathbf{g}_b^n(L_b, h_b) = \begin{bmatrix} \sim 0 \\ \sim 0 \\ -g(L_b, h_b) \end{bmatrix} \tag{2.37}$$

as the specific force due to gravity points in the opposite direction to it, i.e. up direction on Earth's surface.

Expanding Eq. (2.36) with the help of Eqs. (2.2), (2.4) and (2.37):

$$\begin{bmatrix} f_{ib,x}^b \\ f_{ib,y}^b \\ f_{ib,z}^b \end{bmatrix} = \begin{bmatrix} \sin\theta_{nb} \\ -\cos\theta_{nb}\sin\phi_{nb} \\ -\cos\theta_{nb}\cos\phi_{nb} \end{bmatrix} g(L_b, h_b), \tag{2.38}$$

from which the roll and pitch of the b-frame with respect to the n-frame can be obtained:

$$\theta_{nb} = \arctan\left(\frac{f_{ib,x}^b}{\sqrt{{f_{ib,y}^b}^2 + {f_{ib,z}^b}^2}}\right), \quad \phi_{nb} = \arctan_2\left(-f_{ib,y}^b, -f_{ib,z}^b\right) \tag{2.39}$$

This self-alignment process is usually applied over averaged IMU measurements, consequently improving the accuracy of the estimated roll and pitch angles over the time the vehicle is static.

2.4.2 Gyrocompassing

On the other hand, in order to estimate the yaw, a process called **gyrocompassing** can be used. It is based on the knowledge that the only rotation that a stationary INS senses is the Earth's one, which is in the z direction of the e-frame. The attitude may be obtained by solving the following direct gyrocompassing expression:

$$\boldsymbol{\omega}_{ib}^b = \mathbf{C}_n^b \mathbf{C}_e^n \begin{bmatrix} 0 \\ 0 \\ \omega_{ie} \end{bmatrix} \tag{2.40}$$

Substituting Eq. (2.9) in the previous equation and rearranging gives

$$\begin{bmatrix} \omega_{ie}\cos L_b \\ 0 \\ -\omega_{ie}\sin L_b \end{bmatrix} = \mathbf{C}_b^n \boldsymbol{\omega}_{ib}^b, \tag{2.41}$$

from which taking the second row, and knowing the roll and pitch already from leveling in Eq. (2.39), the yaw can be obtained using:

$$\begin{aligned} \psi_{nb} &= \arctan_2(\sin\psi_{nb}, \cos\psi_{nb}) \\ \sin\psi_{nb} &= -\omega^b_{ib,y}\cos\phi_{nb} + \omega^b_{ib,z}\sin\phi_{nb} \\ \cos\psi_{nb} &= \omega^b_{ib,x}\cos\theta_{nb} + \omega^b_{ib,y}\sin\phi_{nb}\sin\theta_{nb} + \omega^b_{ib,z}\cos\phi_{nb}\sin\theta_{nb} \end{aligned} \tag{2.42}$$

As with leveling, this process is also applied over averaged IMU measurements to lower their noise. However, gyrocompassing has a main drawback: The angular rate that arises from the Earth rotation is very small, and it usually makes gyrocompassing useless with low-cost IMUs. In order to circumvent this heading observability problem, this work is focused on the potential of the array-antenna phase detection technique for attitude estimation.

Chapter 3

INS/GNSS Integration

An INS can provide a navigation solution on its own as long as an starting point is provided, following the concept of "dead-reckoning". However, if no other sources of information about position and attitude are used, the solution error will grow indefinitely because of the drifting caused by the integration of the IMU errors, as already seen in the previous chapter. Thus, the importance of integrating different sources of information together. The Kalman filter is one of the most popular tools for doing it in this area.

In this chapter, the integration between an INS and GNSS is explained. In particular, some of its benefits are described, as well as the different strategies or architectures that can be used for it. In addition, a brief overview is provided about how Kalman filter-based estimation works and its particularization for non-linear discrete systems.

3.1 Integration Benefits and Architectures

Inertial navigation has a number of advantages: it provides a complete navigation solution at high rates, it is invulnerable to external interference and it exhibits low short-term noise. However, it requires initialization and its accuracy degrades over time. On the other side, although GNSS provides a lower rate and high short-term noise solution, it is stable in the long-term with errors of only a few meters in open-sky conditions. Furthermore, GNSS can also provide attitude in addition to position, but it is subject to interference. This can be achieved by combining information from multiple receivers or if a multi-element antenna with a supporting receiver is used

This complementarity in the characteristics of both systems makes their integration a desirable option in order to combine their individual advantages. In Table 3.1, a summary of INS and GNSS characteristics is shown.

A continuous, high-bandwidth, complete navigation solution with high short- and long-term accuracy can be achieved integrating both systems. GNSS prevents the INS from drifting while the INS helps filtering and bridging the GNSS solution. This behaviour is illustrated in Fig. 3.1, where the black dashed line describes the

	INS	GNSS
Measurements	$\mathbf{f}_{ib}^{b}$, $\boldsymbol{\omega}_{ib}^{b}$	Position, velocity, attitude
Meas. Availability	Self-contained	Satellites visibility
Output rate	High ($> 50\,\text{Hz}$)	Low ($< 10\,\text{Hz}$)
Interference-sensitive	No	Yes
Short-term noise	Low	High
Position accuracy	Degrades over time	Long-term stable

Table 3.1: Comparison of INS and multi-antenna GNSS characteristics.

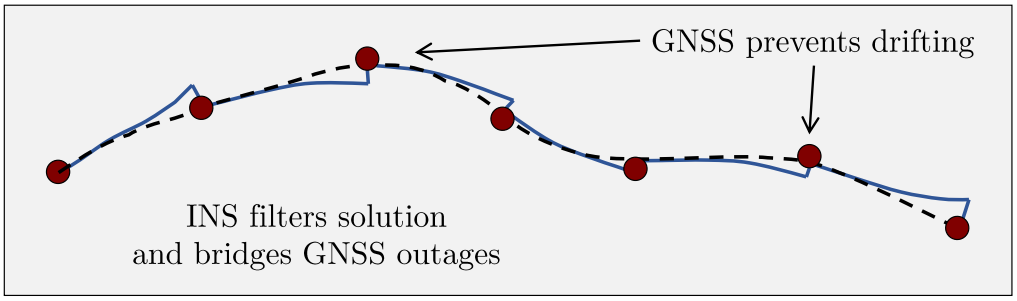


Figure 3.1: INS/GNSS integration operation example.

true path and the blue line is the path calculated by the INS, which is corrected by the GNSS positions, represented by the red dots.

In order to implement this integration scheme, several architectures can be used. All of them share the main characteristic: there is an integration algorithm that compares the inertial navigation solution with the GNSS outputs and, from it, estimates corrections to the final navigation solution.

The different architectures depend on three factors, which are briefly described along with some of their respective strategies:

GNSS measurements type: When the position and velocity already computed by the GNSS equipment are used as input for the integration algorithm, it is called a *loosely-coupled system*. If the integration algorithm uses the GNSS pseudo-ranges and pseudo-range-rates directly as input, it is called a *tightly-coupled system*. In addition, there also exist more deeply coupled systems which even use the GNSS signals in the estimation algorithm.
The integrated solution will be more accurate as the used GNSS measurements are closer to the signal domain, but at the expense of a higher complexity of the integration algorithm. Furthermore, tighter coupled systems need more access to the GNSS processor, which may not be always possible.

Corrections application: The integrated navigation solution is the result of the corrected INS one. However, the estimation algorithm can output the absolute solution (called a total-state integration) or only the error corrections

that need to be applied to the INS solution (called an error-state integration). Being the latter the most common, the type of implementation depends on which stage the corrections are applied at. If they are used to correct the integrated solution at the end, it is called an *open-loop* or *feed-forward* architecture. On the other hand, a *closed-loop* or *feedback* architecture uses the estimated position, velocity, attitude and sensors errors to correct the solution directly in the INS, thus reseting the errors after each correction. The latter is more advantageous for low-grade IMUs, as otherwise the large correction values would lead to linearization problems.

GNSS aiding: Although it is not strictly necessary, the INS solution may be used to help the GNSS equipment with the acquisition and tracking of satellite signals. INS position, velocity or even attitude can be used by the GNSS processor to be able to reduce acquisition times, limit the number of satellites to search or narrow the range of directions in which a multi-antenna receiver should expect a satellite signal.

Inertial systems that use low-cost IMUs tend to implement closed-loop architectures, either loosely-coupled or tightly-coupled. The integration algorithm commonly used is a Kalman filter, whose operation is explained in the following section. Moreover, loosely-coupled strategies are easier to implement and they can be more suited for real-time systems where power consumption and processing time are limited.

3.2 Kalman Filter Theory

A Kalman Filter (KF) is a recursive algorithm able to estimate several parameters of a system using noisy measurements and knowledge about the system properties. It has been widely used in the navigation field since its introduction in 1960 [12] due to its capability of maintaining optimal estimates for position, velocity and other navigation parameters using a weighted average of their new and previous values. It uses a *predict and update* scheme in order to propagate the states and its uncertainties. Furthermore, only new measurements need to be processed on each iteration, making it an efficient algorithm for real-time systems.

3.2.1 Definitions and Algorithms

Formally, a Kalman filter purpose is to estimate $\mathbf{x}(t)$ in a linear system, which is described by a given differential equation of the form

$$\dot{\mathbf{x}}(t) = \mathbf{F}(t)\mathbf{x}(t) + \mathbf{G}(t)\mathbf{w}(t), \tag{3.1}$$

from observations that are linear functions of $\mathbf{x}(t)$ as

$$\mathbf{z}(t) = \mathbf{H}(t)\mathbf{x}(t) + \mathbf{v}(t), \tag{3.2}$$

where $\mathbf{w}$ is the system noise vector, $\mathbf{G}$ is the system noise distribution matrix and $\mathbf{v}$ is the vector with the white noise sources of the measurements. $\mathbf{F}$ and $\mathbf{H}$ are described in the following paragraphs.

In order to implement this iterative algorithm, first of all, a KF is composed of 5 main elements:

States: Set of parameters describing the system and stored in the *state vector* $\mathbf{x}$ which the KF estimates along with their uncertainties, kept in the *error covariance matrix*, denoted by $\mathbf{P}$.

System model: Describes how the physical magnitudes represented by the states vary with time. The KF predicts the future states values based on the *system matrix*, denoted by $\mathbf{F}$, while their uncertainties are increased with each prediction based on the *system noise covariance matrix* $\mathbf{Q}$ (related to $\mathbf{w}$ in Eq. (3.1)), to account for errors in the system dynamical model or noise.

Measurements: They are sets of observations of the physical magnitudes which are functions of the states. These are input to the system as a *measurement vector* $\mathbf{z}$, from which the KF derives the information to estimate the states. They also have a *measurement noise covariance matrix* associated, denoted by $\mathbf{R}$, describing the statistics of the measurements noise (related to $\mathbf{v}$ in Eq. (3.2)).

Measurement model: Describes how the measurement vector projects into the true state vector. Denoted by $\mathbf{H}$, it can change at every update iteration depending on the available measurements.

KF algorithm: The mathematical expressions that use all the elements above to maintain optimal estimates of the state vector. It computes the *Kalman gain matrix* $\mathbf{K}$ to weight the correction of the states according to their current uncertainties in a minimum variance approach. The complete algorithm is described below.

As a summary, Fig. 3.2 shows the different described elements and how they interact with each other, with solid lines representing persistent data flows and dashed ones only for some applications.

In the navigation field, new data is usually available at discrete times, as for IMU or GNSS measurements. Thus, Kalman filters are commonly implemented with their discrete-time version. Basically, the system matrix $\mathbf{F}$ is discretized into the *transition matrix* $\mathbf{\Phi}$ with the following expression:

$$\mathbf{\Phi}_k = \exp(\mathbf{F}(t_k)\Delta t) = \sum_{r=0}^{\infty} \frac{(\mathbf{F}(t_k)\Delta t)^r}{r!} = \mathbf{I} + \mathbf{F}(t_k)\Delta t + \frac{1}{2}(\mathbf{F}(t_k)\Delta t)^2 + ... \quad (3.3)$$

Kalman filters are implemented as a two-stage algorithm. First, in the **prediction** stage the previous time estimated state vector (and its covariance) is propagated to the current time step based on the system model. As a result, an a priori estimation of the states can be obtained. The prediction step can be represented by

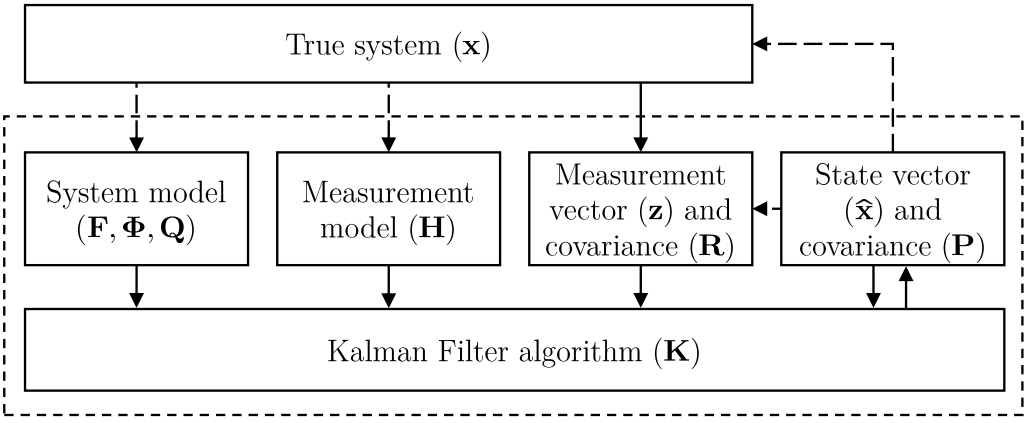


Figure 3.2: Kalman filter elements and interactions diagram (adapted from [1]).

the following equations [1]:

$$\hat{\mathbf{x}}_k^- = \mathbf{\Phi}_{k-1}\hat{\mathbf{x}}_{k-1}^+, \tag{3.4}$$

$$\mathbf{P}_k^- = \mathbf{\Phi}_{k-1}\mathbf{P}_{k-1}^+\mathbf{\Phi}_{k-1}^{\mathrm{T}} + \mathbf{G}_{k-1}\mathbf{Q}_{k-1}\mathbf{G}_{k-1}^{\mathrm{T}}, \tag{3.5}$$

where the subscripts represent the iteration number and the superscripts denote whether the states have been already corrected ($^+$) or not ($^-$).

The second stage is the **update** or correction phase, in which the state vector estimate and its error covariance incorporate the information from new measurements. In order to do that, first the Kalman gain, which weights the amount of new information which must be input to the system in order to obtain a minimum error variance, must be computed [1]:

$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^{\mathrm{T}}\left(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^{\mathrm{T}} + \mathbf{R}_k\right)^{-1}. \tag{3.6}$$

Once this matrix is obtained, the actual update can be carried out using these expressions:

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k\left(\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_k^-\right) = \hat{\mathbf{x}}_k^- + \mathbf{K}_k\boldsymbol{\delta}\mathbf{z}_k^-, \tag{3.7}$$

$$\mathbf{P}_k^+ = \left(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k\right)\mathbf{P}_k^-, \tag{3.8}$$

where $\boldsymbol{\delta}\mathbf{z}_k^-$, called the *measurement innovation*, is the difference between the actual and predicted observations, and represents how much the system estimates differ from the provided external information.

This two-stage scheme, summarized in Fig. 3.3, allows the use of different update rates for each type of measurement depending on its arrival frequency. Furthermore, several prediction steps can be carried out between two updates, thus propagating the system estimates at shorter time steps than the ones imposed by the measurements rate.

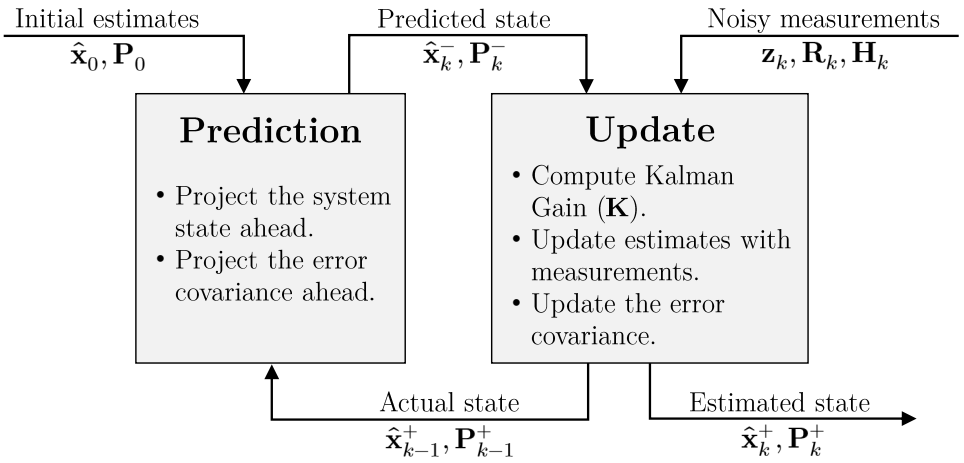


Figure 3.3: Flow diagram of the Kalman filter prediction/update scheme.

In order to initialize the Kalman filter, suitable initial estimates of the states and their uncertainties must be assigned based on the known information of the system. Thus, it is important to have, at least, a rough knowledge of the system properties and statistics for the KF to be able to correctly propagate them forward. In addition, a mathematically appropriate system model is essential for the KF to properly work.

Moreover, a discrete-time Kalman filter assumes that the variation of the errors can be modelled as systematic (constant) errors or white noise sequences. Although the real errors may actually not fall into one of these categories exactly, they can be modelled as one of them with enough overbounding.

Lastly, as already mentioned in the last section, Kalman filters have two types of implementations regarding what the state vector represents: when the states estimate absolute properties of the system, such as position or velocity, it is called a *total-state* implementation, while an *error-state* implementation only estimates the errors made by the system, such as the ones in the INS position or velocity. Furthermore, when using the latter strategy, it is the KF state vector which is reset after each update (once the corrections are applied to the INS) in a closed-loop architecture.

3.2.2 Extended Kalman Filter

Kalman filters are designed for linear systems. However, most real applications are described by non-linear models and expressions, as is the case of INS/GNSS integration. Therefore, a non-linear extension called *Extended Kalman Filter* (EKF) can be used in these situations. This way, the system (**F**) and measurement (**H**) matrices can be substituted by non-linear functions of the state vector, converting

the model from Eqs. (3.1) and (3.2) into

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), t) + \mathbf{G}(t)\mathbf{w}(t), \tag{3.9}$$

$$\mathbf{z}(t) = \mathbf{h}(\mathbf{x}(t), t) + \mathbf{v}(t), \tag{3.10}$$

In order for the EKF to estimate the state vector, $\mathbf{f}$ and $\mathbf{h}$ need to be linearized first to obtain $\mathbf{F}$ and $\mathbf{H}$, respectively.

This way, the **prediction** step is similar as in the original KF from Eqs. (3.4) and (3.5), but applying the system matrix linearization at every iteration, obtaining its Jacobian:

$$\mathbf{F}_{k-1} = \left.\frac{\partial \mathbf{f}(\mathbf{x}, t_k)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1}^{+}} \tag{3.11}$$

Then, the linearized system matrix can be discretized as in Eq. (3.3). A first order approximation is accurate enough for applications that do not require very high precision or have limited processing time windows, like real-time systems:

$$\mathbf{\Phi}_k \approx \mathbf{I} + \mathbf{F}(t_k)\Delta t \tag{3.12}$$

Similarly, the **update** step can be computed with the same expressions as in the original KF in Eqs. (3.6)–(3.8), although the innovation vector can be computed with the actual non-linear $\mathbf{h}$ function:

$$\boldsymbol{\delta}\mathbf{z}_k^{-} = \mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_k^{-}, t_k) \tag{3.13}$$

For the other expressions, as in the prediction step, the measurement model needs to be linearized first:

$$\mathbf{H}_k = \left.\frac{\partial \mathbf{h}(\mathbf{x}, t_k)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_k^{-}} = \left.\frac{\partial \mathbf{z}(\mathbf{x}, t_k)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_k^{-}} \tag{3.14}$$

Although these linearizations allow the extension of the KF algorithm for a wide range of non-linear systems, it makes $\mathbf{P}$ and $\mathbf{K}$ to be functions of the state estimates. Consequently, if the true states are not very close to its estimates, it can cause stability problems that may lead the EKF to diverge. Thus, an EKF is very sensitive to the tuning of the $\mathbf{P}$ matrix initialization.

3.3 Loosely-Coupled Integration

Error-state implementations (where states represent the errors in the navigation solution) prevents severe linearization errors and, consequently, they are more suited for applications where low-grade IMUs are used. Furthermore, a loosely-coupled system is simpler to implement as long as position can be already obtained from the GNSS receiver. For these reasons, it is the base architecture for INS/GNSS integration, and its diagram is shown in Fig. 3.4.

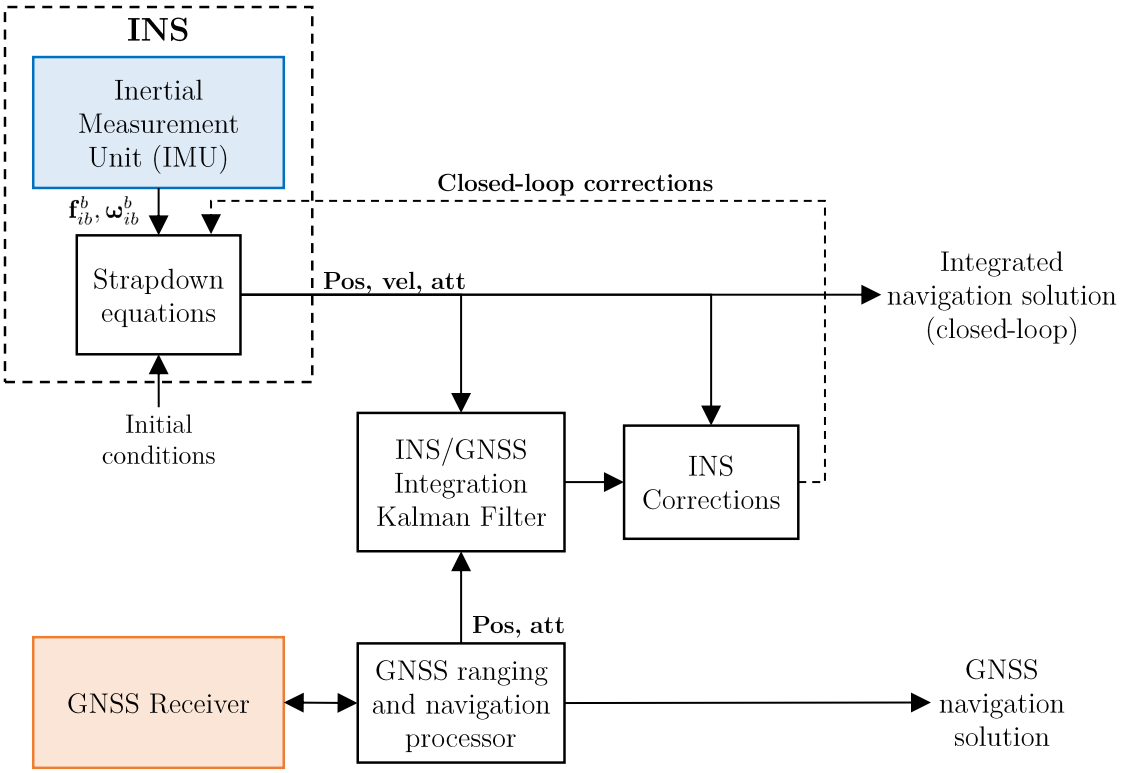


Figure 3.4: Loosely-coupled INS/GNSS integration system.

In this system, the INS solution drives the Kalman filter, predicting the following states. On the other hand, the GNSS solution, already computed by the GNSS receiver processor, is used to update the KF states, obtaining correction values. These are fed back to the INS, correcting the IMU biases and its inertial solution from drifting. This way, the integrated navigation solution is obtained.

In this section, the Kalman filter and the INS corrections blocks are detailed, as they contain the main functionality of the architecture. The basic INS/GNSS integration algorithm in a loosely-coupled architecture is an EKF with the following list of essential states (in this case, resolved in the local frame):

$$\mathbf{x}^n = \begin{bmatrix} \boldsymbol{\delta\Psi}_{nb}^n \\ \boldsymbol{\delta v}_{eb}^n \\ \boldsymbol{\delta p}_b \\ \boldsymbol{\delta b}_a \\ \boldsymbol{\delta b}_g \end{bmatrix}, \tag{3.15}$$

where the state vector components represent, in order, the errors in the INS-computed attitude with respect to NED (Eq. (2.3)), local velocity (Eq. (2.16)), curvilinear position (Eq. (2.11)), accelerometer biases and gyroscope biases.

This state vector can be augmented with additional IMU error states, such as

scale factors. The choice on which IMU errors to estimate depends on the particular implementation of the filter.

These errors are defined as the difference between the INS-computed quantity (denoted by "~") and its true value. Explicitly, the velocity and position states errors are:

$$\boldsymbol{\delta}\mathbf{v}_{eb}^{n} = \tilde{\mathbf{v}}_{eb}^{n} - \mathbf{v}_{eb}^{n} \tag{3.16}$$

$$\boldsymbol{\delta}\mathbf{p}_{b} = \tilde{\mathbf{p}}_{b} - \mathbf{p}_{b} \tag{3.17}$$

On the other hand, the attitude error must be defined through a product of transformation matrices as

$$\boldsymbol{\delta}\mathbf{C}_{b}^{n} = \tilde{\mathbf{C}}_{b}^{n}\mathbf{C}_{n}^{b}, \tag{3.18}$$

which, assuming that the small angle approximation from Eq. (2.8) can be used, the expression can be rearranged in terms of the actual attitude error state vector components as:

$$[\boldsymbol{\delta}\boldsymbol{\Psi}_{nb}^{n}\times] = \boldsymbol{\delta}\mathbf{C}_{b}^{n} - \mathbf{I}_{3}. \tag{3.19}$$

Last, the bias errors are the difference between the already corrected noisy IMU outputs and their true values. These expressions can be rearranged to be a function of the real and the measured biases. The derivation is done with the specific forces, where $\hat{\mathbf{f}}_{ib}^{b}$ is the specific force vector already corrected by the INS, and $\tilde{\mathbf{f}}_{ib}^{b}$ is the measurement as it is outputted by the IMU:

$$\begin{aligned}\boldsymbol{\delta}\mathbf{b}_{a} &= \hat{\mathbf{f}}_{ib}^{b} - \mathbf{f}_{ib}^{b} \\ &= \left(\tilde{\mathbf{f}}_{ib}^{b} - \tilde{\mathbf{b}}_{a}\right) - \mathbf{f}_{ib}^{b} = \left(\left(\mathbf{f}_{ib}^{b} + \mathbf{b}_{a}\right) - \tilde{\mathbf{b}}_{a}\right) - \mathbf{f}_{ib}^{b} \\ &= \mathbf{b}_{a} - \tilde{\mathbf{b}}_{a}\end{aligned} \tag{3.20}$$

$$\boldsymbol{\delta}\mathbf{b}_{g} = \mathbf{b}_{g} - \tilde{\mathbf{b}}_{g} \tag{3.21}$$

System Model

Once the state vector is defined, the system model which describes how these states vary with time is needed. In order to obtain the equations that describe the errors dynamics, it is necessary to calculate the time derivative of each of the errors in the state vector using Eqs. (3.16)–(3.19). This can be done using the kinematics equations from Section 2.2. In particular, Eq. (2.18) is used for the time derivative of the attitude error, Eq. (2.28) for the velocity error, and Eq. (2.31) for the position one.

On the other hand, the dynamics of the bias errors depend on how these are modelled. For this base system, the biases are assumed not to have a known time variation, thus giving null time derivatives.

Finally, the obtained expressions are functions of the states as [1]:

$$\boldsymbol{\delta}\dot{\boldsymbol{\Psi}}^n_{nb} = \mathbf{F}^n_{aa}\boldsymbol{\delta}\boldsymbol{\Psi}^n_{nb} + \mathbf{F}^n_{av}\boldsymbol{\delta}\mathbf{v}^n_{eb} + \mathbf{F}^n_{ap}\boldsymbol{\delta}\mathbf{p}_b + \mathbf{C}^n_b\boldsymbol{\delta}\mathbf{b}_g \tag{3.22}$$

$$\boldsymbol{\delta}\dot{\mathbf{v}}^n_{eb} = \mathbf{F}^n_{va}\boldsymbol{\delta}\boldsymbol{\Psi}^n_{nb} + \mathbf{F}^n_{vv}\boldsymbol{\delta}\mathbf{v}^n_{eb} + \mathbf{F}^n_{vp}\boldsymbol{\delta}\mathbf{p}_b + \mathbf{C}^n_b\boldsymbol{\delta}\mathbf{b}_a \tag{3.23}$$

$$\boldsymbol{\delta}\dot{\mathbf{p}}_b = \mathbf{F}^n_{pv}\boldsymbol{\delta}\mathbf{v}^n_{eb} + \mathbf{F}^n_{pp}\boldsymbol{\delta}\mathbf{p}_b \tag{3.24}$$

where the $\mathbf{F}^n_{ij}$ matrices contain the functions describing how the values of the j-th error state components propagate into the dynamics of the i-th components. The contents of these matrices can be found in the Appendix A.

The way these expressions are arranged makes it straightforward to obtain the complete system matrix using Eq. (3.11). In addition, a basic assumption of Kalman filters is that the system model is a linear function of the states, i.e. to compute the above equations, the values of the errors are the state vector estimates (denoted by "$\hat{}$"). Accordingly, the system matrix is as follows:

$$\mathbf{F}^n_{\text{INS}} = \begin{bmatrix} \mathbf{F}^n_{aa} & \mathbf{F}^n_{av} & \mathbf{F}^n_{ap} & \mathbf{0}_3 & \hat{\mathbf{C}}^n_b \\ \mathbf{F}^n_{va} & \mathbf{F}^n_{vv} & \mathbf{F}^n_{vp} & \hat{\mathbf{C}}^n_b & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{F}^n_{pv} & \mathbf{F}^n_{pp} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix} \tag{3.25}$$

Equation (3.12) is used to obtain the discrete-time transition matrix:

$$\boldsymbol{\Phi}^n_{\text{INS}} \approx \begin{bmatrix} \mathbf{I}_3 + \mathbf{F}^n_{aa}\Delta t & \mathbf{F}^n_{av}\Delta t & \mathbf{F}^n_{ap}\Delta t & \mathbf{0}_3 & \hat{\mathbf{C}}^n_b\Delta t \\ \mathbf{F}^n_{va}\Delta t & \mathbf{I}_3 + \mathbf{F}^n_{vv}\Delta t & \mathbf{F}^n_{vp}\Delta t & \hat{\mathbf{C}}^n_b\Delta t & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{F}^n_{pv}\Delta t & \mathbf{I}_3 + \mathbf{F}^n_{pp}\Delta t & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \tag{3.26}$$

This matrix needs to be effectively computed on every iteration at the linearization point given by the current state vector estimates.

The system noise vector is composed of the accelerometers and gyroscopes noises:

$$\mathbf{w} = \begin{bmatrix} \mathbf{w}_a & \mathbf{w}_g & \mathbf{w}_{b,a} & \mathbf{w}_{b,g} \end{bmatrix}^{\mathrm{T}}, \tag{3.27}$$

where the different noise components arise from:

- $\mathbf{w}_a$ the accelerometers random noise;
- $\mathbf{w}_g$ the gyroscopes random noise;
- $\mathbf{w}_{b,a}$ the accelerometers bias variation noise;
- $\mathbf{w}_{b,g}$ the gyroscopes bias variation noise.

From these noise sources, and assuming that all accelerometers and all gyroscopes have equal noise characteristics, the associated system noise covariance matrix would be:

$$\mathbf{Q} = \begin{bmatrix} \mathbf{I}_3\sigma_a^2 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3\sigma_g^2 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3\sigma_{b,a}^2 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3\sigma_{b,g}^2 \end{bmatrix} \Delta t \tag{3.28}$$

where the variance of each of the errors is represented by its σ^2. To adequately obtain their values, the used IMU errors must be modelled. Last, these uncertainties are distributed to the different states through the following system noise distribution matrix:

$$\mathbf{G} = \begin{bmatrix} \mathbf{0}_3 & \hat{\mathbf{C}}_b^n & \mathbf{0}_3 & \mathbf{0}_3 \\ \hat{\mathbf{C}}_b^n & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \tag{3.29}$$

Measurement Models

The measurements for the filter in this loosely-coupled architecture are the differences between the GNSS and the INS position and velocity solutions. However, their body origins are usually not the same, i.e. the IMU and the GNSS antenna are not in the same position or even orientation. Because of this, the GNSS solution must be transposed into the IMU position before computing their differences.

Let "b" represent the IMU body frame and "a" the GNSS antenna one. The position of the second with respect to the first resolved in the IMU body axes, denoted by $\mathbf{l}_{ba}^b$, is called the *lever arm*, while the orientation of the GNSS antenna with respect to the IMU is represented by $\mathbf{C}_b^a$, which is called the *boresight*. Assuming that these are well known, and denoting the GNSS antenna curvilinear position by $\mathbf{p}_a$, its velocity by $\mathbf{v}_{ea}^n$ and its attitude with respect to the n-frame by $\mathbf{C}_a^n$, the solution transposition can be carried out following these expressions [1], using Eqs. (2.14) and (2.20):

$$\mathbf{p}_b^{(a)} = \mathbf{p}_a - \mathbf{T}_{r(n)}^p \mathbf{C}_b^n \mathbf{l}_{ba}^b, \tag{3.30}$$

$$\mathbf{v}_{eb}^{n\,(a)} = \mathbf{v}_{ea}^n - \mathbf{C}_b^n \left(\boldsymbol{\omega}_{ib}^b \times \mathbf{l}_{ba}^b \right) + \boldsymbol{\Omega}_{ie}^n \mathbf{C}_b^n \mathbf{l}_{ba}^b, \tag{3.31}$$

$$\mathbf{C}_b^{n(a)} = \mathbf{C}_a^n \mathbf{C}_b^a. \tag{3.32}$$

With the GNSS solution already transposed to the IMU frame, their KF updates can be obtained. The basic GNSS-related updates for a loosely-coupled system are the position and velocity updates. The attitude update is more unusual, as it needs

a GNSS receiver capable of obtaining it. The measurement innovations (equal to the measurement vector in a closed-loop architecture, as can be extracted from Eq. (3.13)) for the GNSS position and velocity are:

$$\boldsymbol{\delta}\mathbf{z}_k^n = \begin{bmatrix} \hat{\mathbf{p}}_b^{\,(a)} - \hat{\mathbf{p}}_b \\ \hat{\mathbf{v}}_{eb}^{n\,(a)} - \hat{\mathbf{v}}_{eb}^n \end{bmatrix}_k \tag{3.33}$$

In order to obtain their measurement models, Eq. (3.14) needs to be applied. From Eqs. (3.30) and (3.31), it can be seen that the lever arm introduces a coupling between the position/velocity states and other states like the attitude errors or the gyroscope biases. However, these terms are weak if the lever arm is not very big, thus making possible to obtain the following simplified measurement models:

$$\mathbf{H}_k^n = \begin{bmatrix} \mathbf{0}_3 & \mathbf{0}_3 & -\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & -\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix}_k \tag{3.34}$$

Associated to these updates, the uncertainties in GNSS position and velocity are inserted in the measurement noise covariance matrix as follows:

$$\mathbf{R}_k^n = \begin{bmatrix} \boldsymbol{\Sigma}_\mathbf{p} & \mathbf{0}_3 \\ \mathbf{0}_3 & \boldsymbol{\Sigma}_\mathbf{v} \end{bmatrix}_k \tag{3.35}$$

with:

$$\boldsymbol{\Sigma}_\mathbf{p} = \mathrm{diag}\left(\begin{bmatrix} \sigma_L^2 & \sigma_\lambda^2 & \sigma_h^2 \end{bmatrix}\right) \tag{3.36}$$

$$\boldsymbol{\Sigma}_\mathbf{v} = \mathrm{diag}\left(\begin{bmatrix} \sigma_{v_N}^2 & \sigma_{v_E}^2 & \sigma_{v_D}^2 \end{bmatrix}\right) \tag{3.37}$$

where the values of the uncertainties depend on the accuracy of the GNSS solution.

These updates can be applied together or separately, i.e. it is possible to introduce position updates without velocity ones and vice versa, depending on their availability.

INS Corrections

The KF block functionality from Fig. 3.4 has been described. Now, finally the equations related to the INS corrections block can be derived. When KF updates are applied, the state vector is estimated with the new and previous information in the system. These estimates are fed back to the INS in order to correct its navigation solution (Fig. 3.1). These corrections are computed from the error definitions in Eqs. (3.16)–(3.21) by assuming that the "true" values are the KF estimates. This

way, rearranging, the correction expressions are obtained:

$$\begin{aligned}
\hat{\mathbf{C}}_b^n &= \left(\mathbf{I}_3 - [\boldsymbol{\delta\Psi}_{nb}^n\times]\right)\tilde{\mathbf{C}}_b^n \\
\hat{\mathbf{v}}_{eb}^n &= \tilde{\mathbf{v}}_{eb}^n - \boldsymbol{\delta}\mathbf{v}_{eb}^n \\
\hat{\mathbf{p}}_b &= \tilde{\mathbf{p}}_b - \boldsymbol{\delta}\mathbf{p}_b \\
\hat{\mathbf{b}}_a &= \tilde{\mathbf{b}}_a + \boldsymbol{\delta}\mathbf{b}_a \\
\hat{\mathbf{b}}_g &= \tilde{\mathbf{b}}_g + \boldsymbol{\delta}\mathbf{b}_g
\end{aligned} \tag{3.38}$$

Once these equations are computed, the state vector is reset as the INS has already applied the corrections to its calculated outputs, therefore ending the current iteration processing and producing the integrated navigation solution.

Chapter 4

Navigation Processor Design

The navigation processor is the system block which outputs the complete navigation solution when fed with the IMU measurements and GNSS data. This block is required to implement not only the whole strapdown and integration Kalman filter algorithms, but also an entire context-aware framework able to automatically perform a correct system initialization, the initial alignment, adequate updates depending on the arriving data, etc.

In the following sections, a description of the designed system is given. First, it is presented from a high-level operational perspective. Then, the different system parts are described in more depth. First, the propagation of uncertainties during the initial alignment process is detailed. Second, it is shown the particular design of the Kalman filter adapted for a multi-antenna GNSS receiver. In addition, the developed measurement models for tighter KF attitude updates are explained. Finally, it is described the operation of some KF fault tests that help preventing the solution from corrupting.

4.1 Operational Design and State Machine

The Navigation Processor is designed as a Finite-State Machine (FSM) whose different states depend on the vehicle dynamics and the uncertainty of the navigation solution. Accordingly, four different navigation states can be defined:

Uncertainty \ **Dynamics**	**Static (S)**	**Dynamic (D)**
High (H)	Coarse Alignment	Degraded
Low (L)	Fine Alignment	In-motion

Table 4.1: Navigation Processor FSM states.

These states determine the functionality of the system, while the possible transitions between states are determined at the IMU measurements arrival rate.

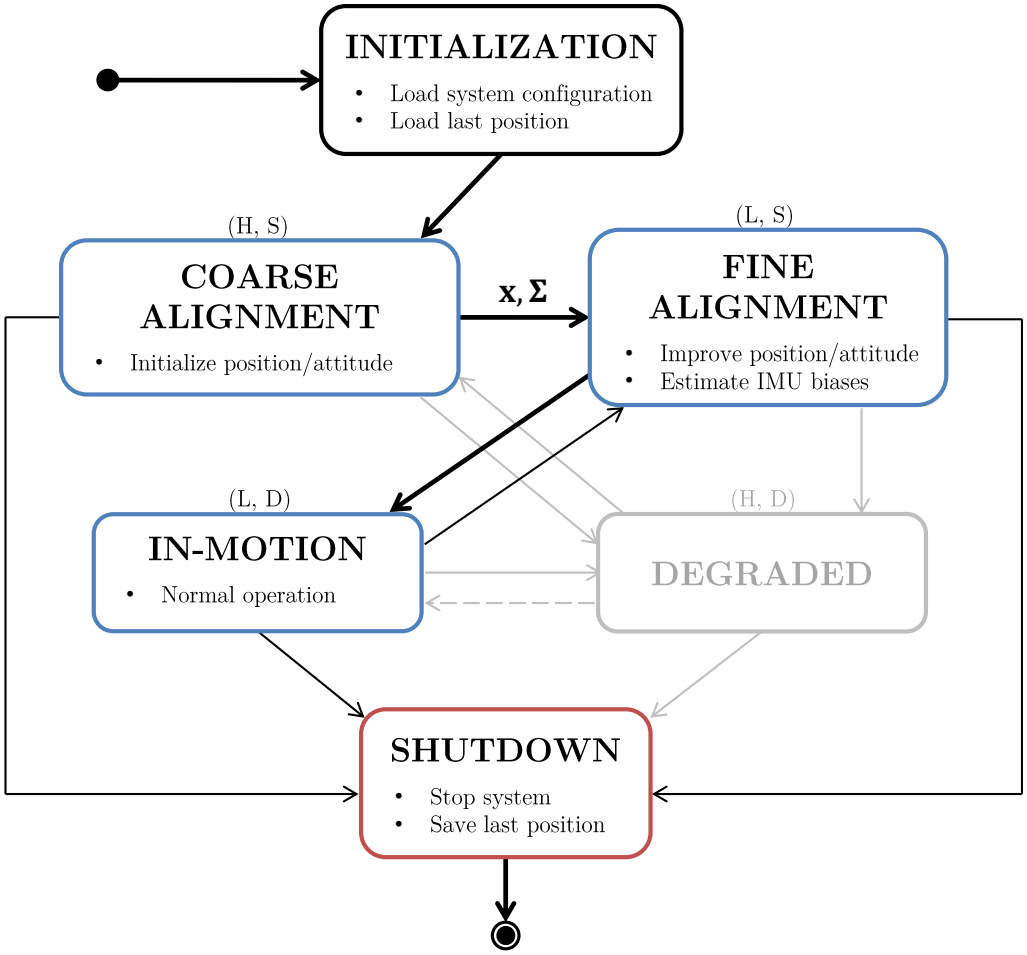


Figure 4.1: Navigation Processor FSM diagram.

In addition, the complete FSM also needs two more states in order to handle the start/stop of the system and get it ready for operation: An **Initialization** state in order to load all the system configuration and start from the last saved position, and a **Shutdown** state to correctly stop the system and save the last estimated position.

This way, the state machine diagram would be as is shown in Fig. 4.1. An explanation about the purpose and functionality associated to each of the navigation-related FSM states is given below.

Coarse Alignment State

When the system is initialized, and before running the Kalman filter, it is necessary to obtain a starting point from which the INS algorithms can propagate the navigation solution. However, the attitude of the vehicle is unknown at the beginning. Furthermore, the position loaded from the Initialization state may be wrong if the vehicle has moved from the last run. Because of this, the purpose of this state is to obtain a coarse estimation of both the attitude and position of the vehicle.

In order to obtain a rough estimate of the attitude, two processes are used. First, *Leveling* is always applied, which uses the gravity force to initialize the roll and pitch angles, as computed in Eq. (2.39). The IMU only senses the gravity force as the vehicle is static in this stage. Furthermore, IMU measurements are averaged to decrease the noise of the estimated roll and pitch angles.

A second process is used for the heading angle determination. However, as already mentioned in Section 2.4, estimating the yaw with low-grade IMUs is practically impossible, as the gyroscope biases are usually bigger than the Earth rotation angular rate. Thus, from Eq. (2.42), *gyrocompassing* would not have observability of the Earth rotation to be able to correctly estimate the heading angle. Furthermore, this process would take several minutes to obtain a stable value.

Fortunately, the multi-antenna GNSS receiver can help with the heading initialization. Although noisy and sensitive to interference, GNSS-acquired heading is much more accurate and faster than a low-cost IMU self-alignment, assuming relatively good satellite visibility conditions. When available, the navigation processor uses the GNSS attitude to obtain a rough estimate of the heading, taking into account the possible lever arm.

On the other hand, the system must also obtain the uncertainties associated to the computed attitude for the next FSM state to know which uncertainties initialize the KF error covariance matrix with. In the case of GNSS attitude, its variance must be given by the GNSS receiver (or modelled), but for leveling (and gyrocompassing if it is the case) the variances of the accelerometers and gyroscopes must be propagated to attitude ones. The mathematical derivation of the variance propagation equations that this coarse alignment state implements is explained in the Section 4.2.

Once a rough estimate of the position and attitude is obtained, and its uncertainties comply with the configured thresholds, the solution uncertainty can be assumed to be low, and the FSM would change to the Fine Alignment state, passing the computed coarse solution to it.

Fine Alignment State

After a coarse alignment has already been carried out, the system can start the strapdown and integration algorithms. In this state, the fact that the vehicle is static is exploited to correct the solution drift by making use of two types of static KF updates called *Zero Velocity Update* (ZVU) and *Zero Angular Rate Update* (ZARU). These help estimating the IMU biases relatively quickly, as they are the only source for change in the position, velocity or attitude estimation when the vehicle is still.

As this is the first state in which the Kalman filer starts operating, it needs to be initialized. As discussed on Section 3.2, it is highly important to feed the KF with properly modelled first state estimates and associated covariances that cover for the initial uncertainty of the states. This is possible thanks to the obtained data from the previous FSM state, where coarse estimates of the attitude and its uncertainties have been computed ($\mathbf{x}$ and $\mathbf{\Sigma}$ in Fig. 4.1). Moreover, position uncertainty is initialized with the particular GNSS receiver statistics, velocity is known to be zero (as these

FSM states only apply to static operation), and the different bias uncertainties can be taken from the IMU datasheet or obtained model.

After the KF is started, a new prediction step is carried out whenever a new IMU measurement arrives, while the corresponding update step is executed when any measurement is available, thus propagating and correction the inertial solution. GNSS-related updates are applied when new GNSS measurements arrive, while Zero updates are applied at fixed time intervals, instead of at the IMU rate, in order to avoid overfitting.

In-Motion State

When a new IMU measurement exceeds the given acceleration or rotation threshold, it is considered that the vehicle is moving, and the FSM moves to this state.

This In-motion state is very similar to the Fine Alignment one, in the sense that they operate the same KF, applying the prediction/update scheme as usual. The difference lies in the types of updates that they can carry out, as Zero updates can evidently not be applied in this state. In particular, the possible ones (which can also be executed in Fine Alignment) are GNSS position and velocity (explained in Section 3.3) and updates related to GNSS attitude, either loose or tight, as will be explained in Section 4.4.

In the case that new IMU measurements do not exceed any of the dynamics threshold for a given number of consecutive epochs, it is assumed that the vehicle is still, and the FSM would go back the Fine Alignment state.

Degraded State

Although the normal state-changing behaviour of the FSM is the one described by bold arrows in Fig. 4.1, a degraded operation state can also be considered for when the vehicle is moving without the system complying with a maximum solution uncertainty. For example, the FSM would change to this state if the vehicle starts moving during the Coarse Alignment or before the Fine Alignment already has a rough estimate of the biases. In addition, the system could change from In-motion to Degraded if too few updates are carried out and the integrated solution uncertainty gets too big. Potentially, a change in the opposite direction could also happen if enough updates are able to correct the solution while in movement.

This state simply performs the strapdown algorithm while waiting for the vehicle to be static again, in order to be able to restart the navigation solution returning to the Coarse Alignment state.

In the following sections it is given a more detailed explanation about the particular implementation of the processes carried out by the different FSM states.

4.2 Propagation of Attitude Uncertainties

The Coarse Alignment stage obtains a rough estimate of the vehicle attitude and its uncertainty. In order for the estimation to become better with time, IMU measurements are averaged over time, thus reducing their noise. This way, instead of performing the self-alignment algorithms with only a single epoch, these are fed with the average of the IMU outputs over a time window.

On the other hand, in order to obtain the Euler angles uncertainties, the variances from the averaged IMU measurements need to be propagated to the attitude domain through the corresponding mathematical expressions of leveling and gyrocompassing (when used).

First, a simplified model of the specific forces (the same would apply to the angular rates) is assumed, where the measurements are composed of the true magnitude (sub- and superscripts related to the used frames are omitted for simplicity) and a random vector with normal distribution as

$$\tilde{\mathbf{f}} = \mathbf{f} + \boldsymbol{\Delta}\mathbf{f}, \qquad \boldsymbol{\Delta}\mathbf{f} \sim N(\boldsymbol{\mu}_{\mathbf{f}}, \boldsymbol{\Sigma}_{\mathbf{f}}). \tag{4.1}$$

with

$$\boldsymbol{\mu}_{\mathbf{f}} = \begin{bmatrix} \mu_{\mathrm{f},x} & \mu_{\mathrm{f},y} & \mu_{\mathrm{f},z} \end{bmatrix}^{\mathrm{T}}, \quad \boldsymbol{\Sigma}_{\mathbf{f}} = \begin{bmatrix} \sigma^2_{\mathrm{f},x} & 0 & 0 \\ 0 & \sigma^2_{\mathrm{f},y} & 0 \\ 0 & 0 & \sigma^2_{\mathrm{f},z} \end{bmatrix}, \tag{4.2}$$

where $\boldsymbol{\mu}_{\mathbf{f}}$ is the mean vector $\boldsymbol{\Sigma}_{\mathbf{f}}$ is the covariance matrix of the multivariate normal distribution.

These measurements are averaged over a window of length M:

$$\bar{\mathbf{f}}_k = \sum_{j=k-M}^{k} \tilde{\mathbf{f}}_j, \tag{4.3}$$

resulting in the averaged vector following the statistical model below, assuming that the real specific force value is static:

$$\bar{\mathbf{f}} = \mathbf{f} + \overline{\boldsymbol{\Delta}\mathbf{f}}, \qquad \overline{\boldsymbol{\Delta}\mathbf{f}} \sim N(\boldsymbol{\mu}_{\bar{\mathbf{f}}}, \boldsymbol{\Sigma}_{\bar{\mathbf{f}}}). \tag{4.4}$$

Assuming that the measurements are independent and identically distributed (i.i.d.), the new mean vector and covariance matrix can be obtained as follows:

$$\boldsymbol{\mu}_{\bar{\mathbf{f}}} = \frac{1}{M} \sum_{i=k-M}^{k} \boldsymbol{\mu}_{\mathbf{f}i} = \boldsymbol{\mu}_{\mathbf{f}}, \tag{4.5}$$

$$\boldsymbol{\Sigma}_{\bar{\mathbf{f}}} = \frac{1}{M^2} \sum_{i=k-M}^{k} \boldsymbol{\Sigma}_{\mathbf{f}i} = \frac{1}{M} \boldsymbol{\Sigma}_{\mathbf{f}}. \tag{4.6}$$

As the self-alignment processes are used, the attitude is a function of the IMU outputs as $\boldsymbol{\Psi}_{nb} = \mathbf{g}(\bar{\mathbf{f}}, \bar{\boldsymbol{\omega}})$. The attitude statistics are computed based on the average

measurements in Eq. (4.4). The mean value is propagated using the expressions in Eqs. (2.39) and (2.42), and the covariance is propagated by using Taylor series expansions of first order.

In order to obtain the roll variance, and assuming that its true value is fixed, the series expansion linearized around its mean would be:

$$\phi\left(\bar{\mathbf{f}}\right) \approx \phi(\bar{\mathbf{f}}_0) + \left[\frac{\partial\phi(\bar{\mathbf{f}})}{\partial\overline{\Delta \mathrm{f}}_x}\left(\overline{\Delta \mathrm{f}}_x - \mu_{\mathrm{f},x}\right) + \frac{\partial\phi(\bar{\mathbf{f}})}{\partial\overline{\Delta \mathrm{f}}_y}\left(\overline{\Delta \mathrm{f}}_y - \mu_{\mathrm{f},y}\right) + \frac{\partial\phi(\bar{\mathbf{f}})}{\partial\overline{\Delta \mathrm{f}}_z}\left(\overline{\Delta \mathrm{f}}_z - \mu_{\mathrm{f},z}\right)\right]\Bigg|_{\bar{\mathbf{f}}=\bar{\mathbf{f}}_0}$$

$$\bar{\mathbf{f}}_0 = \begin{bmatrix}\mathrm{f}_x + \mu_{\mathrm{f},x} & \mathrm{f}_y + \mu_{\mathrm{f},y} & \mathrm{f}_z + \mu_{\mathrm{f},z}\end{bmatrix}^{\mathrm{T}} \tag{4.7}$$

Rearranging this expression, squaring it and applying the expectation operator, the variance expression can be obtained:

$$\sigma_\phi^2 = S_{\phi,\bar{\mathrm{f}}_x}^2\sigma_{\bar{\mathrm{f}}_x}^2 + S_{\phi,\bar{\mathrm{f}}_y}^2\sigma_{\bar{\mathrm{f}}_y}^2 + S_{\phi,\bar{\mathrm{f}}_z}^2\sigma_{\bar{\mathrm{f}}_z}^2, \tag{4.8}$$

where it has been assumed that the cross-elements of the specific forces covariance matrix are zero, as the IMU outputs are considered i.i.d.. S is the sensitivity of a function to a variable through the following expression, with the first term as the example:

$$S_{\phi,\bar{\mathrm{f}}_x} = \left.\frac{\partial\phi(\bar{\mathbf{f}})}{\partial\overline{\Delta \mathrm{f}}_x}\right|_{\bar{\mathbf{f}}=\bar{\mathbf{f}}_0} \tag{4.9}$$

Equation (4.8) can be obtained for each of the Euler angle variances, as well as their covariance matrix cross-elements if, instead of squaring the series expansion, it is multiplied by another Euler angle's one. It has to be taken into account that the self-aligned heading (yaw) is a function of not only the specific forces but also the angular rates; accordingly, its expressions also include the terms related to the latter. This way, the covariance matrix of the complete attitude and heading $\mathbf{\Sigma_\Psi}$ can be written as:

$$\mathbf{\Sigma_\Psi} = \begin{bmatrix}\sigma_\phi^2 & \mathrm{cov}(\phi,\theta) & \mathrm{cov}(\phi,\psi)\\ \mathrm{cov}(\theta,\phi) & \sigma_\theta^2 & \mathrm{cov}(\theta,\psi)\\ \mathrm{cov}(\psi,\phi) & \mathrm{cov}(\psi,\theta) & \sigma_\psi^2\end{bmatrix} = \mathbf{H}_\mathbf{\Psi}^{\mathrm{T}}\mathbf{\Sigma}_{\bar{\mathbf{f}},\bar{\boldsymbol{\omega}}}\mathbf{H}_\mathbf{\Psi} \tag{4.10}$$

where

$$\mathbf{H}_\mathbf{\Psi} = \begin{bmatrix}S_{\phi,\bar{\mathrm{f}}_x} & S_{\phi,\bar{\mathrm{f}}_y} & S_{\phi,\bar{\mathrm{f}}_z} & 0 & 0 & 0\\ S_{\theta,\bar{\mathrm{f}}_x} & S_{\theta,\bar{\mathrm{f}}_y} & S_{\theta,\bar{\mathrm{f}}_z} & 0 & 0 & 0\\ S_{\psi,\bar{\mathrm{f}}_x} & S_{\psi,\bar{\mathrm{f}}_y} & S_{\psi,\bar{\mathrm{f}}_z} & S_{\psi,\bar{\omega}_x} & S_{\psi,\bar{\omega}_y} & S_{\psi,\bar{\omega}_z}\end{bmatrix}^{\mathrm{T}} \tag{4.11}$$

and, from Eq. (4.6):

$$\mathbf{\Sigma}_{\bar{\mathbf{f}},\bar{\boldsymbol{\omega}}} = \begin{bmatrix}\mathbf{\Sigma}_{\bar{\mathbf{f}}} & \mathbf{0}_3\\ \mathbf{0}_3 & \mathbf{\Sigma}_{\bar{\boldsymbol{\omega}}}\end{bmatrix} = \frac{1}{M}\begin{bmatrix}\mathbf{\Sigma}_\mathbf{f} & \mathbf{0}_3\\ \mathbf{0}_3 & \mathbf{\Sigma}_{\boldsymbol{\omega}}\end{bmatrix}. \tag{4.12}$$

Computing Eq. (4.9) for Eqs. (2.39) and (2.42), the sensitivities of the roll and pitch angles with respect to the specific force components are:

$$\begin{aligned} S_{\phi,\bar{\mathrm{f}}_x} &= 0, & S_{\theta,\bar{\mathrm{f}}_x} &= \frac{\sqrt{\bar{\mathrm{f}}_y^2+\bar{\mathrm{f}}_z^2}}{\|\bar{\mathbf{f}}\|^2}, \\ S_{\phi,\bar{\mathrm{f}}_y} &= \frac{\bar{\mathrm{f}}_z}{\bar{\mathrm{f}}_y^2+\bar{\mathrm{f}}_z^2}, & S_{\theta,\bar{\mathrm{f}}_y} &= \frac{-\bar{\mathrm{f}}_x\,\bar{\mathrm{f}}_y}{\|\bar{\mathbf{f}}\|^2\sqrt{\bar{\mathrm{f}}_y^2+\bar{\mathrm{f}}_z^2}}, \\ S_{\phi,\bar{\mathrm{f}}_z} &= \frac{-\bar{\mathrm{f}}_y}{\bar{\mathrm{f}}_y^2+\bar{\mathrm{f}}_z^2}, & S_{\theta,\bar{\mathrm{f}}_z} &= \frac{-\bar{\mathrm{f}}_x\,\bar{\mathrm{f}}_z}{\|\bar{\mathbf{f}}\|^2\sqrt{\bar{\mathrm{f}}_y^2+\bar{\mathrm{f}}_z^2}}. \end{aligned} \tag{4.13}$$

On the other hand, the yaw depends on the specific force components through the leveling-computed roll and pitch. Using the chain rule, the specific-force-related sensitivities can be obtained. For example, the x component would be:

$$S_{\psi,\bar{\mathrm{f}}_x} = \left.\frac{\partial\psi(\bar{\mathbf{f}},\bar{\boldsymbol{\omega}})}{\partial\overline{\Delta\mathrm{f}}_x}\right|_{\substack{\bar{\mathbf{f}}=\bar{\mathbf{f}}_0\\ \bar{\boldsymbol{\omega}}=\bar{\boldsymbol{\omega}}_0}} = \frac{\partial\psi}{\partial\phi}\frac{\partial\phi}{\partial\overline{\Delta\mathrm{f}}_x}+\frac{\partial\psi}{\partial\theta}\frac{\partial\theta}{\partial\overline{\Delta\mathrm{f}}_x} = S_{\psi,\phi}S_{\phi,\bar{\mathrm{f}}_x}+S_{\psi,\theta}S_{\theta,\bar{\mathrm{f}}_x}. \tag{4.14}$$

This dependency on the roll and pitch sensitivities allows a more compact arrangement for Eq. (4.11): Let $\mathbf{H}_l$ be the matrix of sensitivities related to the Euler angles computed from the leveling process:

$$\mathbf{H}_l = \begin{bmatrix} S_{\phi,\bar{\mathrm{f}}_x} & S_{\phi,\bar{\mathrm{f}}_y} & S_{\phi,\bar{\mathrm{f}}_z} \\ S_{\theta,\bar{\mathrm{f}}_x} & S_{\theta,\bar{\mathrm{f}}_y} & S_{\theta,\bar{\mathrm{f}}_z} \end{bmatrix}^{\mathrm{T}}. \tag{4.15}$$

From Eq. (4.14), it can be derived that the vector of yaw sensitivities with respect to the specific force components can then be written as:

$$\mathbf{h}_{\psi,\mathrm{f}} = \begin{bmatrix} S_{\psi,\bar{\mathrm{f}}_x} \\ S_{\psi,\bar{\mathrm{f}}_y} \\ S_{\psi,\bar{\mathrm{f}}_z} \end{bmatrix} = \begin{bmatrix} S_{\phi,\bar{\mathrm{f}}_x} & S_{\theta,\bar{\mathrm{f}}_x} \\ S_{\phi,\bar{\mathrm{f}}_y} & S_{\theta,\bar{\mathrm{f}}_y} \\ S_{\phi,\bar{\mathrm{f}}_z} & S_{\theta,\bar{\mathrm{f}}_z} \end{bmatrix} \begin{bmatrix} S_{\psi,\phi} \\ S_{\psi,\theta} \end{bmatrix} = \mathbf{H}_l\mathbf{h}_{\psi,l}, \tag{4.16}$$

where $\mathbf{h}_{\psi,l}$ is the vector of yaw sensitivities with respect to the leveling-related roll and pitch angles. Finally, Eq. (4.11) can be rewritten as:

$$\mathbf{H}_{\boldsymbol{\Psi}} = \begin{bmatrix} \mathbf{H}_l & \mathbf{H}_l\mathbf{h}_{\psi,l} \\ \mathbf{0} & \mathbf{h}_{\psi,\omega} \end{bmatrix} \tag{4.17}$$

where

$$\mathbf{h}_{\psi,\omega} = \begin{bmatrix} S_{\psi,\bar{\omega}_x} & S_{\psi,\bar{\omega}_y} & S_{\psi,\bar{\omega}_z} \end{bmatrix}^{\mathrm{T}}. \tag{4.18}$$

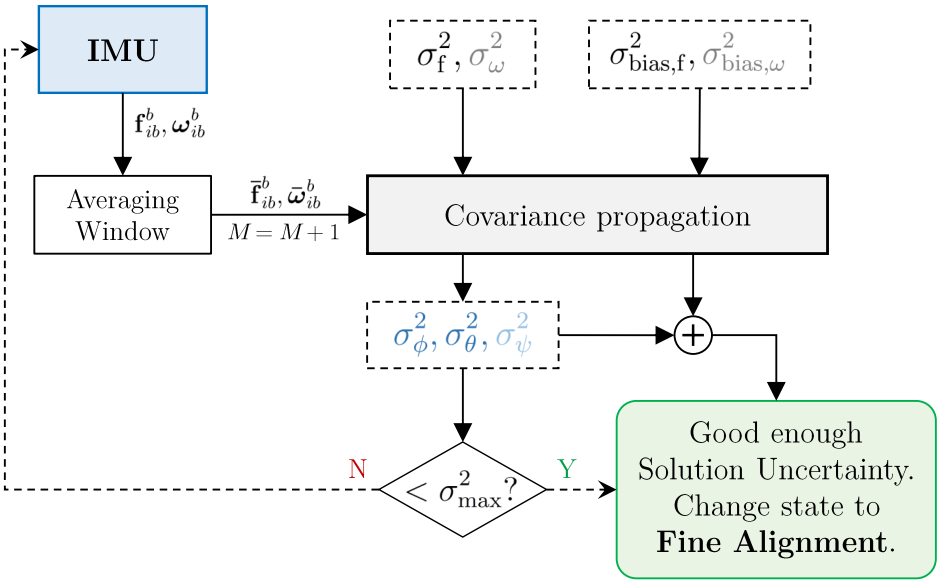


Figure 4.2: Attitude covariance propagation process.

Last, the partial derivatives of Eq. (2.42) have the expressions below:

$$
\begin{aligned}
S_{\psi,\bar{\omega}_x} &= \frac{1}{A}\left[\cos\theta\left(\bar{\omega}_y\cos\phi - \bar{\omega}_z\sin\phi\right)\right], \\
S_{\psi,\bar{\omega}_y} &= \frac{1}{A}\left[-\bar{\omega}_x\cos\phi\cos\theta - \bar{\omega}_z\sin\theta\right], \\
S_{\psi,\bar{\omega}_z} &= \frac{1}{A}\left[\bar{\omega}_x\sin\phi\cos\theta + \bar{\omega}_y\sin\theta\right], \\
S_{\psi,\phi} &= \frac{1}{A}\left[\bar{\omega}_x\cos\theta\left(\bar{\omega}_y\sin\phi + \bar{\omega}_z\cos\phi\right) + \sin\theta\left(\bar{\omega}_y^2 + \bar{\omega}_z^2\right)\right], \\
S_{\psi,\theta} &= \frac{1}{A}\left[\left(\bar{\omega}_y\cos\phi - \bar{\omega}_z\sin\phi\right)\left(-\bar{\omega}_x\sin\theta + \cos\theta\left(\bar{\omega}_y\sin\phi + \bar{\omega}_z\cos\phi\right)\right)\right], \\
A &= \left(\bar{\omega}_x\cos\theta + \bar{\omega}_y\sin\phi\sin\theta + \bar{\omega}_z\cos\phi\sin\theta\right)^2 + \left(-\bar{\omega}_y\cos\phi + \bar{\omega}_z\sin\phi\right)^2.
\end{aligned}
\tag{4.19}
$$

These covariance propagation expressions are the ones used by the system in the Coarse Alignment state. Every time a new IMU measurement arrives, it is added to the averaging window in Eq. (4.3) and the self-alignment equations are applied. Afterwards, the covariance propagation is carried out using the currently available averaged specific forces and angular rates as the linearization points.

This process continues until the variances of the three Euler angles are below a threshold in the case that both leveling and gyrocompassing are being used, or only the roll and pitch ones if GNSS attitude is used. When the condition is met, the coarse alignment is complete. A summarized diagram of the covariance propagation process is shown in Fig. 4.2.

Unfortunately, the propagated covariances are calculated over the sensors biases,

i.e. averaging the measurements reduces the computed attitude noise, but they are calculated without subtracting the biases, as they are still unknown and do not decrease with the average window from Eq. (4.5). These produces a bigger uncertainty of the attitude, which needs to be taken into account.

This can be easily done assuming that the biases can also be modelled as random vectors with a given variance and zero mean. Accordingly, the same covariance propagation process can be applied to them using the variance of the sensor biases instead of their noises.

Finally, the complete Euler angles uncertainties, assuming that the biases and the noise random vectors are independent, would simply be the sum of the variances propagated from the measurement noise and the biases. These would be the values that are passed to the Fine Alignment stage.

4.3 Loosely Coupled System

The implemented integration architecture is a loosely-coupled KF strategy as it is simpler to implement and does not need access to the GNSS processor itself, as already discussed in the previous chapter. This KF runs during the Fine Alignment and In-motion stages of the Navigation Processor FSM. As an extension of the baseline system, a better model of the IMU errors is used and some additional types of KF updates are incorporated to the list of available ones. These modifications and extensions are detailed in the following subsections.

4.3.1 Inertial Sensor Error Models

Some IMU errors are difficult to estimate by the KF if they have low observability, i.e. if the random noise level eclipses the systematic error effect. In addition, larger KF state vectors need longer processing times to be computed, and these times can be limited in a real-time system. Accordingly, from the IMU errors already described in Section 2.3, the only ones that are estimated in this work are the biases, as they usually are the most significant in low-grade sensors.

These biases are usually modelled as the sum of a constant offset, or *turn-on bias*, and a time-variant component, or *bias drift*. The turn-on bias is constant over an entire run, although it can vary between different runs, while the bias drift is a contribution that keeps changing over the same run.

The turn-on contribution is effectively modelled as a random constant while the bias drift is typically modelled with a *first-order Gauss-Markov sequence* [13]. There are two main reasons for this: the driving errors of these sequences follow a Gaussian distribution, thus making it suitable for a KF, as mentioned in Section 3.2, and their statistical behaviour is similar enough to the one experimentally shown by the bias drift errors of inertial sensors. It is briefly described in the following paragraphs.

A Gauss-Markov (GM) process is a quantity that varies with time as a linear function of its previous values and white noise. In particular, a first-order GM process only depends on the last value and has an exponentially decaying correlation.

These processes are widely used as they can fairly describe many physical processes and they have a simple mathematical formulation [13].

A first-order GM sequence is the discrete-time equivalent of the respective process. Accordingly, using the bias drift already as the example, it is expressed in the following way:

$$b_k = e^{-\Delta t/\tau} b_{k-1} + w_k, \tag{4.20}$$

where τ is the correlation time of the sequence. The variance of the driving white noise sequence is described by:

$$\sigma_{w_k}^2 = \sigma_{bd}^2 \left(1 - e^{-2\Delta t/\tau}\right), \tag{4.21}$$

where σ_{bd}^2 is the variance of the bias drift.

This way, experimentally obtaining or fine adjusting τ and σ_{bd}^2, both previous expressions can be used directly in the KF transition and system noise covariance matrices.

4.3.2 Kalman Filter System Model

The system model used for this extension is very similar to the one presented in Section 3.3, but with two main modifications:

1. The IMU biases are modelled with first-order GM sequences, as aforementioned. The linear relation between two consecutive bias values given by Eq. (4.20), as it is already discretized, can be directly input into the discrete transition matrix $\mathbf{\Phi}$. In addition, as each bias is supposed independent from the others, and assuming that all accelerometers and gyroscopes have equal statistical characteristics, the transition submatrices related to the biases would be diagonal as [14]:

$$\mathbf{\Phi}_{b_a b_a} = \mathbf{I}_3 \left(e^{-\Delta t/\tau_a}\right), \qquad \mathbf{\Phi}_{b_g b_g} = \mathbf{I}_3 \left(e^{-\Delta t/\tau_g}\right) \tag{4.22}$$

2. The curvilinear position is internally converted to Cartesian coordinates. This is done in order to have better conditioned matrices for the KF calculations, as the curvilinear longitude and latitude in radians have very small values and they can cause numerical instabilities when calculating the Kalman gain from Eq. (3.6).

Therefore, the modified state vector would be:

$$\mathbf{x}^{nC} = \begin{bmatrix} \boldsymbol{\delta\Psi}_{nb}^n \\ \boldsymbol{\delta v}_{eb}^n \\ \boldsymbol{\delta r}_{eb}^n \\ \boldsymbol{\delta b}_a \\ \boldsymbol{\delta b}_g \end{bmatrix} \tag{4.23}$$

Accordingly, all the correction equations are as in Eq. (3.38) except for the position one. Knowing that the conversion between the curvilinear and Cartesian position is given by Eq. (2.12), the modified correction equation for the position would be:

$$\hat{\mathbf{p}}_b = \tilde{\mathbf{p}}_b - \hat{\mathbf{T}}^p_{r(n)} \boldsymbol{\delta}\mathbf{r}^n_{eb}. \tag{4.24}$$

Regarding the system model matrix, Eq. (3.25) needs its position-related components to be converted to Cartesian coordinates, which can be done as follows, [1]:

$$\mathbf{F}^{nC}_{\text{INS}} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \hat{\mathbf{T}}^{r(n)}_p & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \mathbf{F}^n_{\text{INS}} \begin{bmatrix} \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \hat{\mathbf{T}}^p_{r(n)} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \tag{4.25}$$

Expanding this expression, converting it to its discrete-time form (Eq. (3.12)) and adding the GM sequence submatrices related to the IMU biases in Eq. (4.22), the final transition matrix would be:

$$\boldsymbol{\Phi}^{nC}_{\text{INS}} \approx \begin{bmatrix} \mathbf{I}_3 + \mathbf{F}^n_{aa}\Delta t & \mathbf{F}^n_{av}\Delta t & \mathbf{F}^n_{ap}\hat{\mathbf{T}}^p_{r(n)}\Delta t & \mathbf{0}_3 & \hat{\mathbf{C}}^n_b\Delta t \\ \mathbf{F}^n_{va}\Delta t & \mathbf{I}_3 + \mathbf{F}^n_{vv}\Delta t & \mathbf{F}^n_{vp}\hat{\mathbf{T}}^p_{r(n)}\Delta t & \hat{\mathbf{C}}^n_b\Delta t & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3\Delta t & \mathbf{I}_3 + \left(\hat{\mathbf{T}}^{r(n)}_p \mathbf{F}^n_{pp} \hat{\mathbf{T}}^p_{r(n)}\right)\Delta t & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \boldsymbol{\Phi}_{b_a b_a} & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \boldsymbol{\Phi}_{b_g b_g} \end{bmatrix} \tag{4.26}$$

On the other hand, the system noise distribution matrix $\mathbf{G}$ would be the same as in Eq. (3.29) and its associated noise system noise covariance matrix would be as Eq. (3.28). Expanding the latter with the modelled bias drift noises from Eq. (4.21), it would result in:

$$\mathbf{Q} = \begin{bmatrix} \mathbf{I}_3\sigma_a^2 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3\sigma_g^2 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3\sigma^2_{bd,a}\left(1 - e^{-2\Delta t/\tau_a}\right) & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3\sigma^2_{bd,g}\left(1 - e^{-2\Delta t/\tau_g}\right) \end{bmatrix} \Delta t, \tag{4.27}$$

where $\sigma_{bd,a}$ and $\sigma_{bd,g}$ are the standard deviations of the accelerometer and gyroscope bias drift values, respectively.

Finally, the initialization of the $\mathbf{P}$ matrix would be done as follows:

$$\mathbf{P}_0 = \begin{bmatrix} \boldsymbol{\Sigma}_{\boldsymbol{\Psi}} & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \boldsymbol{\Sigma}_{\mathbf{p}} & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \boldsymbol{\Sigma}_{\mathbf{v}} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \boldsymbol{\Sigma}_{\mathbf{b,a}} & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \boldsymbol{\Sigma}_{\mathbf{b,g}} \end{bmatrix} \tag{4.28}$$

where

- $\boldsymbol{\Sigma}_{\boldsymbol{\Psi}}$ is the attitude covariance matrix obtained from Eq. (4.10) in the Coarse Alignment stage;
- $\boldsymbol{\Sigma}_{\mathbf{p}}$ is the position diagonal covariance matrix (in the NED frame to account for the Cartesian position modification) obtained from the Coarse Alignment stage;
- $\boldsymbol{\Sigma}_{\mathbf{v}}$ is the velocity covariance matrix (see Eq. (3.37)), whose values would be fairly low, depending on the confidence of the system on the vehicle being static;
- $\boldsymbol{\Sigma}_{\mathbf{b,a}}$ is the diagonal covariance matrix of the accelerometer turn-on biases;
- $\boldsymbol{\Sigma}_{\mathbf{b,g}}$ is the diagonal covariance matrix of the gyroscope turn-on biases.

The turn-on bias components are usually much bigger than their bias-drift ones. Consequently, as the states related to the biases include both the turn-on and the bias-drift contributions, their initial uncertainties can be assumed to be that of the former.

4.3.3 Kalman Filter Measurement Models

Aside from the typical GNSS updates already described in Section 3.3, some additional updates are available in this implemented system. As already mentioned in Section 4.1, usual Zero updates are carried out to prevent drifting during the Fine Alignment stage, when the vehicle is still. Furthermore, another GNSS update is implemented to take advantage of the attitude obtained by the multi-antenna GNSS receiver. These updates are described in the following paragraphs.

Zero Velocity Update

This update forces the INS velocity to be zero, preventing the position from drifting. Consequently, the KF corrects the errors that produce a non-zero velocity in static, i.e. the accelerometer biases.

The measurement innovation in the local frame is the difference between the (virtually) measured velocity and the INS-computed one:

$$\boldsymbol{\delta}\mathbf{z}^n_{\mathrm{ZV},k} = \mathbf{0}_3 - \hat{\mathbf{v}}^n_{eb,k}. \tag{4.29}$$

Following Eq. (3.14), the measurement model derivation from the previous equation is straightforward:

$$\mathbf{H}^n_{\mathrm{ZV},k} = \begin{bmatrix} \mathbf{0}_3 & -\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix}. \tag{4.30}$$

Zero Angular Rate Update

A static body can only sense the rotation of the Earth. Taking advantage of this fact, this update forces the angular rates to equal the Earth rotation components in the local frame. In practice, for low-cost IMUs, this is the same as forcing zero angular rates. This way, INS attitude drifting is prevented by correcting the errors that produce a non-zero angular rate, i.e. the gyroscope biases.

The measurement innovation is independent of the resolving frame. It would be the difference between the Earth rotation components and the INS-obtained angular rates (the gyroscope outputs with their estimated biases subtracted):

$$\boldsymbol{\delta}\mathbf{z}_{\mathrm{ZA},k} = \hat{\mathbf{C}}^b_n \boldsymbol{\omega}^n_{ie} - \hat{\boldsymbol{\omega}}^b_{ib,k} \approx -\hat{\boldsymbol{\omega}}^b_{ib,k}. \tag{4.31}$$

Accordingly, the approximated measurement model is:

$$\mathbf{H}_{\mathrm{ZA},k} = \begin{bmatrix} \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & -\mathbf{I}_3 \end{bmatrix}. \tag{4.32}$$

The measurement noise associated to each of both Zero updates depends on how much the system is trusted when it determines the vehicle is static.

GNSS Loose Attitude Update

In order to exploit the attitude-acquisition capabilities of the multi-antenna GNSS receiver, a KF update to incorporate it into the system is considered. It inputs the attitude already computed by the GNSS receiver into the integration system. In this sense, this update is explicitly called "loose" in order to differentiate it from another attitude-related update that is explained in the following section.

The mathematical derivation of this update is as follows. The attitude of the antenna received from the GNSS receiver is $\tilde{\mathbf{C}}^n_a$, where "a" denotes the antenna body frame. Assuming that the boresight between the antenna and the IMU frames, $\mathbf{C}^a_b$, is known, the antenna attitude can be transposed to the IMU body frame as in Eq. (3.32), obtaining the antenna-measured IMU attitude:

$$\tilde{\mathbf{C}}^n_b = \tilde{\mathbf{C}}^n_a \mathbf{C}^a_b \tag{4.33}$$

The INS attitude error can then be obtained with Eq. (3.18), where the GNSS-measured attitude is assumed to be the (noisy) true one. Rearranging this expression to directly input Eq. (4.33) in it, and using Eq. (3.19) for the small angle approximation, the following equation is obtained:

$$(\boldsymbol{\delta}\mathbf{C}^n_b)^{\mathrm{T}} = (\mathbf{I}_3 + [\boldsymbol{\delta}\boldsymbol{\Psi}^n_{nb}\times])^{\mathrm{T}} = \mathbf{I}_3 - [\boldsymbol{\delta}\boldsymbol{\Psi}^n_{nb}\times] = \tilde{\mathbf{C}}^n_a \mathbf{C}^a_b \hat{\mathbf{C}}^b_n, \tag{4.34}$$

where $\hat{\mathbf{C}}^b_n$ is the transpose of the currently estimated INS attitude.

From this expression, the measurement innovation can already be derived in its skew-symmetric form as:

$$[\boldsymbol{\delta}\mathbf{z}_k^n\times] = -[\boldsymbol{\delta}\boldsymbol{\Psi}_{nb}^n\times] = \tilde{\mathbf{C}}_a^n\mathbf{C}_b^a\hat{\mathbf{C}}_n^b - \mathbf{I}_3. \tag{4.35}$$

As the measurement innovation is already a function of the KF attitude error state, the measurement model is simply:

$$\mathbf{H}_k = \begin{bmatrix} -\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix}. \tag{4.36}$$

This update is applied whenever a new GNSS attitude measurement arrives. Its measurement noise covariance matrix associated would be:

$$\mathbf{R}_k = \begin{bmatrix} \sigma_\phi^2 & 0 & 0 \\ 0 & \sigma_\theta^2 & 0 \\ 0 & 0 & \sigma_\psi^2 \end{bmatrix}, \tag{4.37}$$

where the uncertainties of the roll, pitch and yaw depend on the GNSS receiver characteristics.

4.4 DOA-aided Tight Attitude Integration

The attitude computed by the GNSS receiver can be simply taken as input to integration filter as described in Subsection 4.3.3. However, the GNSS receiver computes that attitude from the Direction of Arrival (DOA) of the visible satellites, without more help than the GNSS solution itself. As the designed system already computes an inertial navigation solution, it makes sense using the raw DOAs directly inside the system to jointly compute an attitude estimation aided by both systems.

4.4.1 DOA Handling and Advantages

In order to obtain the attitude from the multi-antenna GNSS receiver, first the DOA for every incoming GNSS signal is obtained. This is achieved thanks to the fact that the observed carrier-signal phase differences between the elements of an antenna array depend on the direction of arrival of the signal [15]. Applying array signal processing techniques to compensate for the mutual coupling between the array elements, and with the help of direction finding methods [4], the DOAs can be estimated. These DOAs can be expressed as pairs of elevation and azimuth with respect to the antenna body frame, with 0° azimuth in the forward direction.

On the other hand, using the Ephemeris data, the GNSS receiver can also compute the expected DOA in the local frame for each satellite whose signal is decoded. In this case, these DOAs are also obtained as pairs of elevation and azimuth, but with respect to the local navigation frame (0° azimuth in the north direction). In addition, since the change in the receiver position is very small compared to the

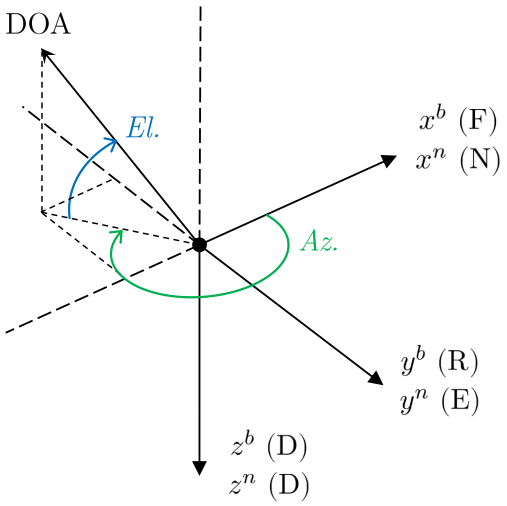


Figure 4.3: Elevation and Azimuth with respect to NED local and FRD body frames.

distance from the receiver to the satellite, a very rough position solution is needed to obtain accurate DOAs from the Ephemeris. The geometric interpretation of these values with respect to both the antenna frame and the local frame is shown in Fig. 4.3.

Elevation and azimuth pairs need to be transformed into Direction Cosine Vectors (DCVs) in order to facilitate mathematical operation with them. A DCV is a unitary vector representing a direction decomposed into its three axes contributions. As the frames used in this work, NED for local and FRD for body, follow the same axes convention, the conversion between elevation and azimuth, and its corresponding DCV, can be obtained as follows:

$$\mathbf{d} = \begin{bmatrix} \cos(El)\cos(Az) & cos(El)\sin(Az) & -\sin(Az) \end{bmatrix}^{\mathrm{T}}. \tag{4.38}$$

For a given satellite, let $\mathbf{d}^a$ be the DCV of the antenna-estimated DOA in its body frame, and $\mathbf{d}^n$ the DCV of the Ephemeris-based DOA in the local frame. There is a simple relation between both:

$$\mathbf{d}^n = \mathbf{C}_a^n \mathbf{d}^a = \mathbf{C}_b^n \mathbf{C}_a^b \mathbf{d}^a. \tag{4.39}$$

From this expression, assuming that the boresight between the antenna and the IMU is known, the IMU attitude can be obtained. Furthermore, this relation can be obtained for every available satellite. This is helpful in order to improve the attitude estimation, as the system of matrix equations is overdetermined.

Calculating the GNSS attitude from the DOAs by directly solving Eq. (4.39) inside the KF can have two main advantages:

- A tighter integration can potentially provide a better result since all the observables and IMU measurements information is directly combined in a single estimator, instead of in two steps [16].

- It enables the KF to detect faulty DOAs. Using the inertial solution, the DOAs that largely differ from the KF-estimated ones can be discarded. Furthermore, this faulty DOAs can be used to detect multipath or even spoofed signals, as these typically arrive from directions different from those of the true GNSS satellite signals.

In order to use the DOAs inside the KF, the corresponding update must be designed.

4.4.2 DOA Measurement Model

The IMU attitude relates the Ephemeris-based and the antenna-estimated DOAs of a satellite as in Eq. (4.39). This way, subtracting both terms and rearranging, a new KF measurement innovation can be derived to obtain the INS attitude from the j-th satellite:

$$\boldsymbol{\delta}\mathbf{z}_{j,k} = \mathbf{d}_j^a - \mathbf{C}_b^a \hat{\mathbf{C}}_n^b \mathbf{d}_j^n. \tag{4.40}$$

Substituting Eq. (3.18) in Eq. (3.19) and rearranging, the INS-estimated attitude can be expressed as a function of the true attitude and the KF state vector:

$$\hat{\mathbf{C}}_n^b = \mathbf{C}_n^b \left(\mathbf{I}_3 - [\boldsymbol{\delta}\boldsymbol{\Psi}_{nb}^n .\times]\right) \tag{4.41}$$

The measurement innovation from Eq. (4.40) needs to be rearranged in order to be explicitly dependent on the KF states ($\boldsymbol{\delta}\boldsymbol{\Psi}_{nb}^n$ in this case) and obtain the measurement model directly. Thus, taking into account that the measured DOAs will have certain noise $\mathbf{N}$ associated, and using Eq. (4.41), the measurement innovation can be expressed as:

$$\begin{aligned}
\boldsymbol{\delta}\mathbf{z}_{j,k} &= \mathbf{C}_b^a \mathbf{C}_n^b \mathbf{d}_j^n + \mathbf{N} - \mathbf{C}_b^a \mathbf{C}_n^b \left(\mathbf{I}_3 - [\boldsymbol{\delta}\boldsymbol{\Psi}_{nb}^n \times]\right) \mathbf{d}_j^n \\
&= \mathbf{N} + \mathbf{C}_b^a \mathbf{C}_n^b \left[\boldsymbol{\delta}\boldsymbol{\Psi}_{nb}^n \times\right] \mathbf{d}_j^n \\
&= \mathbf{N} - \mathbf{C}_b^a \mathbf{C}_n^b \left[\mathbf{d}_j^n \times\right] \boldsymbol{\delta}\boldsymbol{\Psi}_{nb}^n.
\end{aligned} \tag{4.42}$$

This way, the measurement model for satellite j can be directly extracted:

$$\mathbf{H}_{j,k} = \begin{bmatrix} -\mathbf{C}_b^a \mathbf{C}_n^b \left[\mathbf{d}_j^n \times\right] & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix}. \tag{4.43}$$

Obtaining the measurement innovation and model for each of the available satellites from Eqs. (4.40) and (4.43), respectively, the matrix formulation of the complete KF update can be expressed as:

$$\boldsymbol{\delta}\mathbf{z}_k = \begin{bmatrix} \boldsymbol{\delta}\mathbf{z}_{1,k} \\ \boldsymbol{\delta}\mathbf{z}_{2,k} \\ \vdots \\ \boldsymbol{\delta}\mathbf{z}_{j,k} \end{bmatrix}, \qquad \mathbf{H}_k = \begin{bmatrix} \mathbf{H}_{1,k} \\ \mathbf{H}_{2,k} \\ \vdots \\ \mathbf{H}_{j,k} \end{bmatrix}. \tag{4.44}$$

Regarding the measurement noise associated to this update, the same approach followed in Section 4.2 can be used to obtain the uncertainties of the estimated DOAs. First, the antenna-estimated elevation and azimuth of a given satellite are modelled as their true values plus an error:

$$\widetilde{El} = El + \Delta El, \tag{4.45}$$

$$\widetilde{Az} = Az + \Delta Az. \tag{4.46}$$

The errors in elevation and azimuth are assumed to follow random variables independent from each other with a given variance and zero mean. Accordingly, these variances can be propagated to the DCV components through the expressions in Eq. (4.38). Using a first-order Taylor expansion around its mean values, the DCV covariance matrix can be approximated with a simple matrix expression:

$$\mathbf{\Sigma}_{\mathrm{DCV}} = \mathbf{J}_{\mathrm{DCV}}^{\mathrm{T}} \mathbf{\Sigma}_{El,Az} \mathbf{J}_{\mathrm{DCV}}, \quad \text{with } \mathbf{\Sigma}_{El,Az} = \begin{bmatrix} \sigma_{El}^2 & 0 \\ 0 & \sigma_{Az}^2 \end{bmatrix}. \tag{4.47}$$

The matrix of sensitivities is obtained as the Jacobian of the DCV components particularized at the true elevation and azimuth values:

$$\mathbf{J}_{\mathrm{DCV}} = \begin{bmatrix} S_{x,El} & S_{x,Az} \\ S_{y,El} & S_{y,Az} \\ S_{z,El} & S_{z,Az} \end{bmatrix}^{\mathrm{T}}, \tag{4.48}$$

where

$$\begin{aligned} S_{x,El} &= \left.\frac{\partial x}{\partial \Delta El}\right|_{\substack{\widetilde{El}=El \\ \widetilde{Az}=Az}} = -\sin(El)\cos(Az), & S_{x,Az} &= -\cos(El)\sin(Az), \\ S_{y,El} &= -\sin(El)\sin(Az), & S_{y,Az} &= \cos(El)\cos(Az), \\ S_{z,El} &= -\cos(El), & S_{z,Az} &= 0. \end{aligned} \tag{4.49}$$

In this case, the true elevation and azimuth values for the linearization point can be obtained from the Ephemeris DOAs rotated by the KF-estimated attitude. On the other hand, their standard deviations σ_{El} and σ_{Az} depend on the GNSS receiver characteristics.

4.5 Fault and Spoofing Detection

In order to detect possible faulty measurements that may corrupt the integrated solution, fault detection mechanisms are included in the KF for each GNSS-related update. Following the theory described in [17], normalized tests can be applied to the measurement innovations before executing the KF update itself.

Global snapshot tests detect sudden faults in the whole measurement innovation vector. Therefore, they are adequate for detecting faults in the GNSS loose updates, like the position or attitude ones. In these tests, the monitor test statistic q_k, called *Normalized Innovation Squared* (NIS), is calculated as the weighted norm of the current innovation vector:

$$q_k = \boldsymbol{\delta}\mathbf{z}_k^{\mathrm{T}}\mathbf{S}_k^{-1}\boldsymbol{\delta}\mathbf{z}_k, \tag{4.50}$$

where $\mathbf{S}_k$ is the covariance matrix associated with the innovation vector obtained using the predicted covariance matrix from Eq. (3.5):

$$\mathbf{S}_k = \mathbf{H}_k\mathbf{P}_k^{-}\mathbf{H}_k + \mathbf{R}_k. \tag{4.51}$$

Assuming that the measurement errors follow zero mean Gaussian distributions, q_k follows a central chi-squared distribution with three degrees of freedom, as every GNSS loose update (applied separately) is composed of a three-element measurement innovation vector (see Eqs. (3.33) and (4.35)).

In order to check whether the test was successful or not, a detection threshold T for each test must be defined. It can be obtained from the inverse cumulative chi-squared distribution for a given probability of false alarm P_{fa} [18]. Accordingly, when the fault detector checks that the following condition is true:

$$q_k > T, \tag{4.52}$$

it concludes that the given measurement innovation is faulty, and it excludes it from the update step.

The probabilities of false alarm are chosen to have very small values (usually $10^{-2} > P_{fa} > 10^{-6}$) in order to avoid discarding possible valid measurements.

On the other hand, partial tests are applied for each of the DOA vectors of the tight attitude updates. This way, instead of accepting or discarding a whole update with all its DOAs, only the ones that are detected as faulty are discarded, thus using the rest of the valid DOAs for the update. In order to apply these partial tests, the same approach as the NIS tests is used. However, instead of using the whole measurement innovation in Eq. (4.50) and corresponding measurement model and noise matrices in Eq. (4.51), only the subvectors and submatrices corresponding to each of the DOAs is used for each test. Accordingly, only the ones that pass the test are inserted into the final $\boldsymbol{\delta}\mathbf{z}_k$, $\mathbf{H}_k$ and $\mathbf{R}_k$ for the tight attitude update.

Consequently, this process allows for the detection of multipath or spoofed signals, which would be the ones corresponding to the DOAs that the KF has discarded. Even if there are several faulty DOAs, the attitude can be adequately computed from the correct DOAs taking advantage of the previous knowledge from the inertial solution. This fact makes the DOA partial tests a very helpful technique for safety-critical applications in which it is crucial to detect possible multipath or spoofed signals.

Chapter 5

Hardware Setup and Software Overview

Once all the algorithms and the complete system design have been described in the previous chapters, they need to be implemented in a system in order to test them. In particular, in this work a real-time system has been implemented to be able to test all the multi-antenna GNSS-related techniques in a real environment.

This chapter first gives a brief description of the used hardware platform. Some characteristics about the used IMU and GNSS receiver are detailed next. Finally, an overview of the software is given along with some related flow diagrams. This software implements the communication with the previous hardware components and all the designed algorithms.

5.1 Hardware Platform

The chosen board to implement the system is an extension of the commercially available PicoZed board from Avnet®. This board is a System-On-Module (SOM) based on the Xilinx Zynq®-7000 SoC. In particular, the model used in this work is a PicoZed 7030. It contains the common functions required to support the core of most SoC designs, and it is targeted to testing and measurement applications, among others. More information about this board can be obtained from its website [19].

This PicoZed board is mounted over an auxiliary in-house designed Printed Circuit Board (PCB). This board has four SMA connections for the antenna array elements, as well as an slot for the IMU, and Ethernet and USB ports for power and external connection. It is relevant to note that the design and set up of these boards is not the main focus of this work, and it was previously carried out by co-workers at DLR. A diagram summarizing the relevant hardware platform connections for this work is shown in Fig. 5.1.

The Zync SoC is composed of two main parts: The Programmable Logic (PL) part, which is a hardware-programmable FPGA, and the Processing System (PS), which is an ARM®-based processor in which software can be run. Both parts are

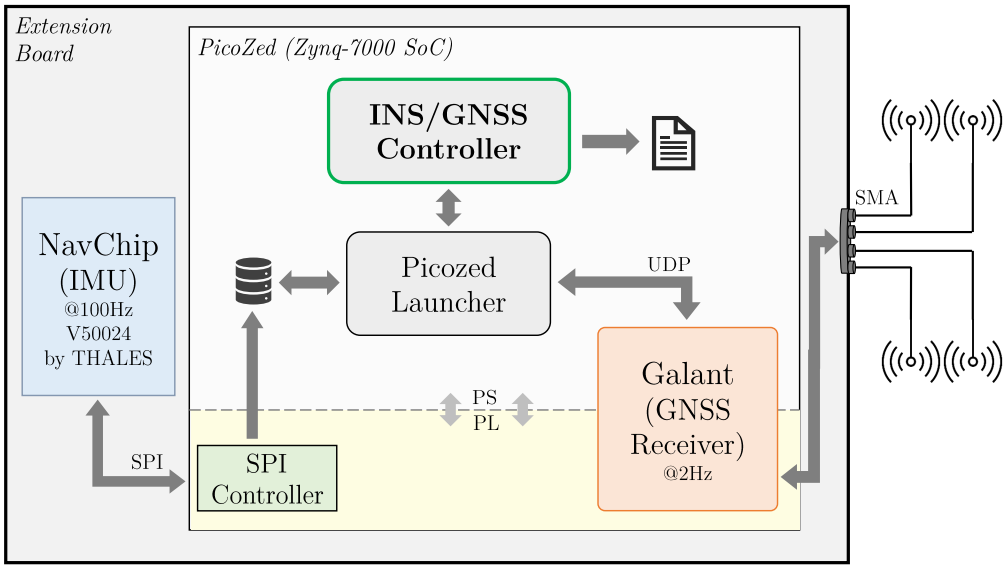


Figure 5.1: Diagram of the hardware platform connections.

interconnected and can communicate with each other. They are distinguished in the diagram in order to clarify in which part is programmed each component.

As can be seen in Fig. 5.1, the IMU communicates with the board through an SPI connection, and its data is stored in a memory location. On the other hand, **GALANT**, which is the GNSS receiver with multi-antenna support developed by DLR [20], is programmed in the PicoZed's Zynq FPGA. GALANT has direct connection with the four antenna channels.

The raw data from both the IMU and the GNSS receiver is collected by the **Picozed Launcher** software block. It gives the raw data the required format for the **INS/GNSS Controller**, which implements all the navigation algorithms and outputs the required processed data into several files.

In Fig. 5.2 it is shown how the described board physically looks like. In addition, the IMU has its sensing axes drawn over it. The SMA connections for the antennas are in the opposite part of the board.

5.2 Navchip IMU

In the test system, a MEMS-based IMU from the *NavChip* series by Thales Visionix is used. It is targeted for embedded applications where small size, low cost and low power consumption is required. It is factory-calibrated and temperature compensated, making the processing of its measurements more convenient. In addition, its output rate ranges from 100 to 200 Hz. The internal clock signal in charge of sampling the sensors values can be synchronized with an external clock, like PicoZed's FPGA one. Thus, the PicoZed is able to timestamp the IMU outputs in a convenient way. Last, the NavChip is very small in size ($12.5 \times 24.5 \times 5.4$ mm) and weights only 3 g, making it highly integrable in an embedded system, as seen in Fig.

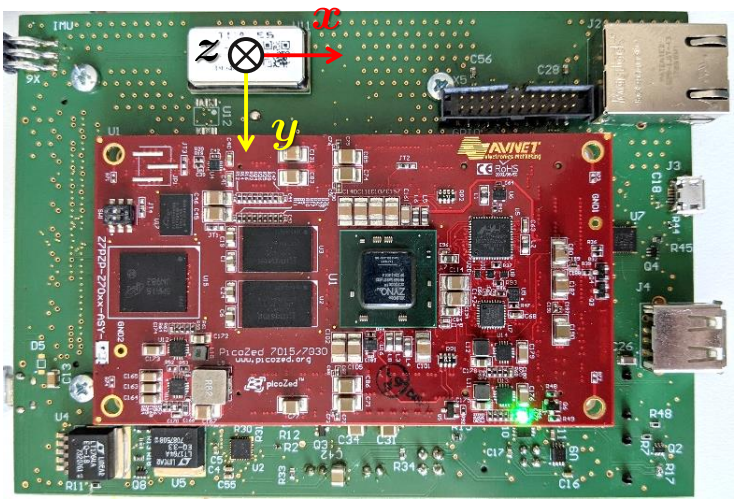


Figure 5.2: Top view of the board with the IMU mounted.

5.2. A complete list of its characteristics can be read from its datasheet [21].

The NavChip used in this work is configured to output measurements at a rate of 100 Hz. The relevant values assumed for the inertial navigation algorithms described in the previous chapters are summarized in Table 5.1. While some of the statistical values of the sensors are taken from the datasheet, the values related to the KF IMU model are extracted as a combination of the values from [22] and some fine-tuning.

	Accelerometers	**Gyroscopes**
Maximum absolute value	$120\,\mathrm{m/s^2}$	$20\,\mathrm{rad/s}$
Random walk	$0.05\,\mathrm{m/s/\sqrt{h}}$	$0.3\,^\circ/\sqrt{\mathrm{h}}$
Random noise std (σ)	$8.33 \times 10^{-3}\,\mathrm{m/s^2}$	$8.726 \times 10^{-4}\,\mathrm{rad/s}$
Turn-on bias std (σ_{tob})	$3 \times 10^{-2}\,\mathrm{m/s^2}$	$4 \times 10^{-3}\,\mathrm{rad/s}$
Bias drift std (σ_{bd})	$1.3 \times 10^{-5}\,\mathrm{m/s^2}$	$1.48 \times 10^{-6}\,\mathrm{rad/s}$
GM correlation time (τ)	25 s	600 s

Table 5.1: NavChip IMU characteristics assumed in the implemented system.

In particular, the Velocity Random Walk (VRW) and the Angle Random Walk (ARW) are given by the datasheet. From these, the random noise standard deviations of the accelerometers (σ_a) and the gyroscopes (σ_g) can be obtained [23]. On the other hand, the assumed GM correlation times are possible general values for low-cost MEMS sensors. A more dedicated error modelling would be necessary in order to better estimate these values.

5.3 DLR GALANT GNSS Receiver

GALANT is the real-time GNSS receiver with multi-antenna support developed at the DLR Oberpfaffenhofen site. It comprises RF front-ends for GPS/Galileo signal reception, baseband digital signal processing and compatibility with a 2×2 active antenna array. In addition, it is also able to estimate the DOA of satellite signals by solving the overdetermined system stated in Section 4.4 using suitable algorithms [5] [6]. More information about this receiver can be found in its web [24].

GALANT communicates via UDP as shown in Fig. 5.2. It sends messages following the NMEA-0813 interface standard, defined by the National Marine Electronics Association (NMEA). A brief introduction about NMEA and a summary its standard output messages can be found in [25]. The relevant messages for this work are:

GGA: It contains the calculated position as latitude, longitude and height, along with the associated timestamp for that measurement.

GSV: These messages contain the number of satellites in view, their IDs and their respective elevation and azimuth pairs in the local frame calculated from the GNSS Ephemeris data and the current user position.

DOA: These proprietary messages transmits the estimated DOAs for all the GNSS signals in track. In particular, it contains the number of estimated DOAs, their associated satellite IDs and their azimuth and elevation pairs in the antenna b-frame.

ATT: This proprietary message contains the attitude already computed by GALANT as the roll, pitch and yaw of the antenna respect to the NED n-frame.

In this work, the GNSS receiver is configured to output measurements at a rate of 2 Hz. However, the receiver can only output a solution when it is available. For example, position may be obtained when more than 4 satellites are in view, while attitude can be acquired when the internal algorithms find a suitable rotation from the estimated DOAs. Therefore, it must be taken into account that all the previous NMEA messages may not be sent at once every time.

On the other side, GALANT uses an "online" calibration method in order to perform the DOA estimation. This technique requires an additional reference signal in order to obtain the calibration coefficients for the antenna array [26].

Finally, some statistics have been obtained for the designed filter from some tests carried out with this receiver, which are detailed in Table 5.2.

These values are the ones used in the Kalman filter updates. Consequently, these variances are designed to overbound the true error distribution, taking into account the possible variations of the error distribution depending on the movement and the local topography.

Standard Deviations		
Position	**Attitude**	**DOAs**
$\sigma_N = 10\,\text{m}$	$\sigma_\phi = 5^\circ$	$\sigma_{El} = 10^\circ$
$\sigma_E = 10\,\text{m}$	$\sigma_\theta = 5^\circ$	$\sigma_{Az} = 15^\circ$
$\sigma_D = 15\,\text{m}$	$\sigma_\psi = 3^\circ$	

Table 5.2: GNSS receiver statistics assumed in the implemented system.

5.4 Software Implementation

The implemented software must comply with some real-time systems characteristics. In this sense, the filter has to process measurements faster than the rate at which they arrive. In addition, the software has to be concurrent in order to perform the different algorithms while accepting new incoming measurements.

Furthermore, another particularity of this real-time system arises from the uncertainties related to which type of new measurement can enter the system at any time. Specifically, the system must be prepared to accept IMU measurements or GNSS ones (which can contain position, velocity, attitude or DOAs, either separately or not). In addition, the possible time delays between measurements arriving from different sources have to be correctly handled in order to always process the data in chronological order.

All the aforementioned considerations along with the designed algorithms are implemented in two main software blocks. These have been developed in a C++ framework, and the Integrated Development Environment (IDE) used for this purpose was Visual Studio Code. The *Armadillo* C++ library [27] [28] has been used for the implementation of the required matrix equations.

The main program is the **INS/GNSS Controller**, where all the described algorithms in this work are implemented. On the other part, the second software block is the **Picozed Launcher**, which serves as a bridge between the INS/GNSS Controller and the IMU and GNSS receiver output interfaces. These two programs are more thoroughly described in the following subsections.

5.4.1 Picozed Launcher

This program serves as a bridge between the raw sensor outputs and the proper navigation state machine software, and it has several purposes:

1. Communicate with the IMU and the GNSS receiver.
2. Parse the outputs into suitable data structures.
3. Synchronize all the acquired data to the same time reference.
4. Send the data to the INS/GNSS Controller.

Thereby, this program is designed for the specific IMU and GNSS receiver that are

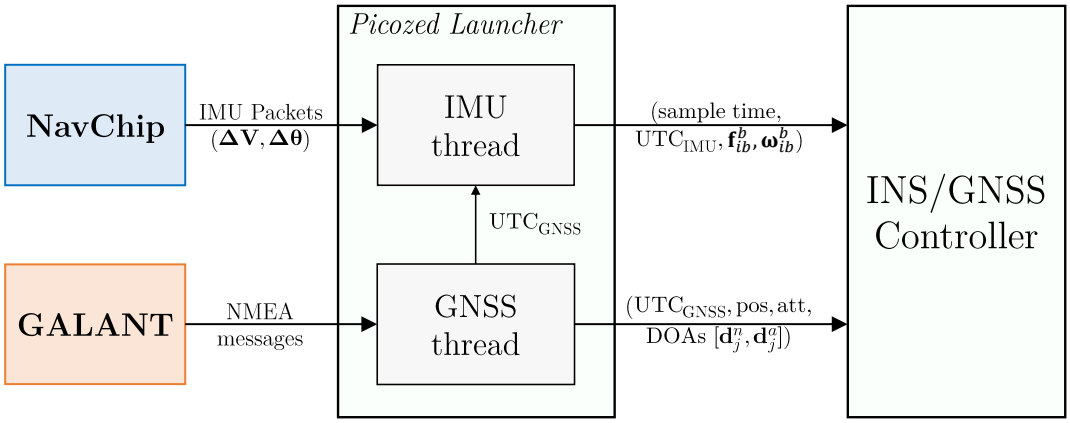


Figure 5.3: Data flow diagram of the Picozed Launcher program.

used in this work. This enables the INS/GNSS Controller to have a more generic high-level input interface, decoupled from the specific needs of this hardware setup.

In particular, the NavChip IMU outputs packets that are received and handled by the *IMU thread* of the Picozed Launcher. These packets contain the variation in velocity and angular rotation since the previous measurement, $\mathbf{\Delta V}$ and $\boldsymbol{\Delta\theta}$, respectively. The thread converts them to specific forces and angular rates dividing by the time step between the current and previous IMU packets. These values are the ones the INS/GNSS Controller expects.

On the other hand, all the GNSS-related processing is done in the *GNSS thread*. Depending on the type of GALANT NMEA message received, these are accordingly converted to vectors containing curvilinear position, antenna attitude, Ephemeris-based DOAs and antenna-estimated DOAs. In addition, GNSS timestamps are used to reference the IMU ones to UTC time. This process is shown in Fig. 5.3 with a data flow diagram.

5.4.2 INS/GNSS Controller

This program implements the INS/GNSS integration scheme designed in this work. It is programmed as a C++ class which implements the Navigation Processor State Machine from Section 4.1. This program controls the FSM and calls the different subclasses that implement the specific algorithms when necessary.

In order to be able to concurrently control the FSM, execute the algorithms and accept new input measurements, three threads run during operation:

Main thread: This is the top thread of the class. It allows calling the different input interface functions from outside the class (the Picozed Launcher in the real-time case).

Navigation Processor: This is the thread which implements the actual FSM, applying all the necessary INS and KF algorithms depending on the type of input measurement. For example, it executes the corresponding strapdown algorithms and KF prediction step when a new IMU measurement arrives.

State Monitor: This thread controls which FSM state needs to be applied at each moment. It continuously checks the conditions that trigger the changes between the Navigation Processor states. When one of these conditions is met, it changes to the corresponding state depending on the current one. For example, it would change the state to In-motion if, during Fine Alignment, it detects that the vehicle starts moving.

On the other hand, some of the main subclasses are described below:

INS: This class implements all the algorithms related to the strapdown computation (Section 2.2) and the self-alignment (Sections 2.4.1, 2.4.2 and 4.2). It also handles the IMU measurement corrections and stores the complete navigation solution. In addition, it implements some dynamics flags to know when the vehicle is moving, checking the norm of the IMU specific force and angular rate vectors.

EKFLooselyCoupled: This class implements the designed Kalman filter, with all the necessary functions to execute the prediction and update steps for each of the possible measurement models described in Sections 3.3, 4.3.3 and 4.4.2. In addition, the fault tests (Section 4.5) are integrated inside each corresponding update. It takes advantage of C++ inheritance capabilities to be a child class of a more generic KF class, which implements the common functions for every Kalman filter. This is done in order to ease possible future implementation of different KF designs.

imuData: This class serves as a container for each IMU measurement. It stores its sample time, its UTC timestamp, the 3-element specific forces vector and the angular rates one.

pvtaData: This class is a container for each GNSS measurement. In addition to its UTC timestamp, this class can store the curvilinear position, velocity and antenna attitude 3-element vectors.

doaDict: This class stores the Ephemeris-based and the antenna-estimated DOAs. It converts the elevation and azimuth pairs into their corresponding DCVs and computes the covariance propagation for each DOA as described in Subsection 4.4.2. In addition, it structures the valid DCVs in matrices already prepared for the tight-attitude KF updates.

All the configuration values for the system are stored in a *navUtils* header file. For example, the IMU and GNSS receiver statistics are stored in this file, along with the fault test thresholds, the KF P matrix initialization values, etc.

Input Interface

The program offers a simple and generic input interface to be convenient and easy to use with any platform. The input functions are overloaded to accept a different

number of parameters, depending on their availability. Four distinct input functions can be highlighted:

new_imu_data: This function inputs a new IMU measurement into the INS/GNSS Controller. It accepts a sample time, a GNSS-referenced UTC timestamp, a 3-element specific forces array and a 3-element angular rates array. This new data will trigger a new strapdown update and its associated KF prediction step.

new_pvt_data: This function inputs a new GNSS PVT measurement into the system. It accepts its UTC timestamp and 3-element arrays with the receiver-computed curvilinear position and velocity, if available. Actually, as the GNSS receiver does not compute velocity in this setup, the input velocity vector is not currently used by the program. This data will trigger its corresponding KF position update.

new_pvta_data: It is an incremental version of the previous function, in which also a 3-element array with the GNSS attitude is accepted. This function will also trigger a KF loose attitude update.

new_gsv_data: Its name arises from the type of NMEA message that contains the data required by this function. It accepts the UTC timestamp associated with the measurements, a vector with the N IDs from the visible satellites, and two more vectors with the N elevations and azimuths computed from the Ephemeris. This function will update the Ephemeris-based DOAs stored in the *doaDict* object.

new_doa_data: Similar to the previous function, it accepts the same type of parameters but related to the antenna-estimated DOAs obtained from the corresponding proprietary NMEA messages. In this case, this function will update the antenna-estimated DOAs stored in the *doaDict* object and it will also trigger a new KF tight attitude update. This update will be carried out with all the DOAs that also have their corresponding Ephemeris-based data stored in the *doaDict* object (after succesfully passing their corresponding fault tests).

Before any of the actions triggered by these input functions can be executed, the INS/GNSS Controller must ensure these measurements are processed in chronological order. However, in real-time operation, it can not be assumed that this will automatically happen, as typically GNSS measurements arrive with a longer delay than IMU ones. Because of this, the Controller implements a **Measurements Time Alignment** block just after these input interface and before sending any measurement to the Navigation Processor.

First, each new input measurement is stored in its corresponding buffer, depending on its type. Then, this time alignment block first checks whether there are a given number of IMU measurements already stored in its buffer, to give some time for any possible GNSS measurement to arrive. Then, it outputs the measurement with the oldest UTC timestamps associated, sending it to the Navigation Processor.

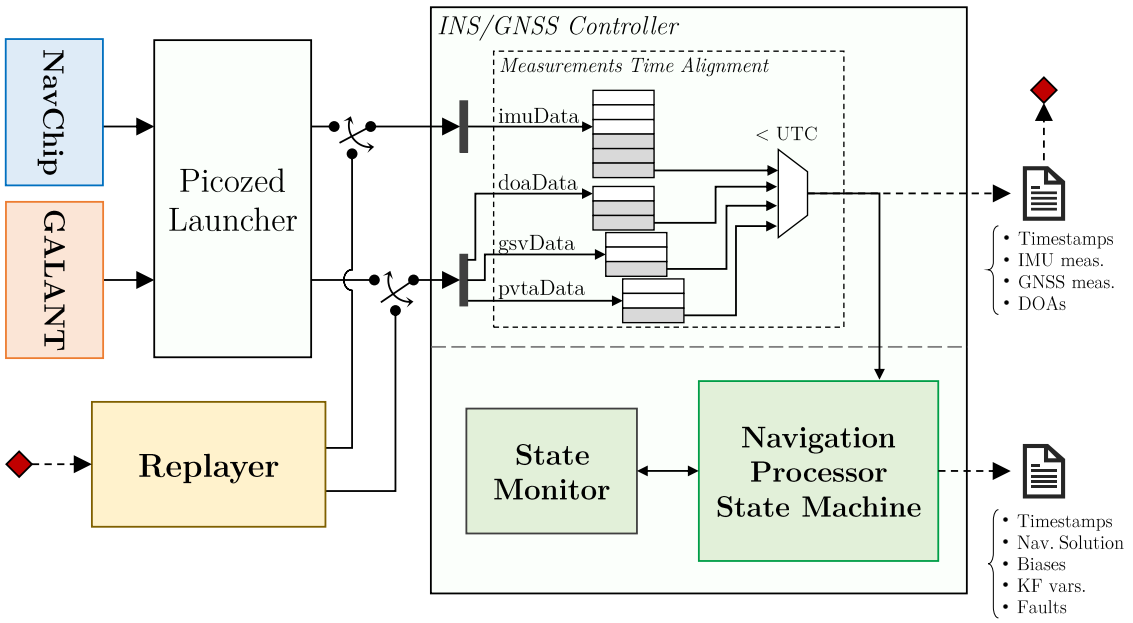


Figure 5.4: Diagram of the INS/GNSS Controller modes.

In Fig. 5.4 it is shown how this block works. The number of IMU measurements to store before one can be taken out can be configured, and it depends on the IMU rate and the GNSS receiver processing delay.

Input/Output Modes

The different operation modes of the program are summarized in the diagram in Fig. 5.4. These are explained in the following paragraphs.

The INS/GNSS Controller is able to log all the input data, the processed navigation data or both. Depending on which data is logged, the program can change its functionality. Accordingly, two different output modes can be distinguished:

Data logger: When only the input data logging is activated, the program assumes this is the functionality required. Thus, it works as a simple data logger, without processing any data. It logs all the data from the IMU and GNSS measurements, including DOAs.

Complete functionality: When any other logging option is chosen, the controller executes the Navigation Processor State Machine and its associated State Monitor to obtain the integrated navigation solution. If the output data logging option is activated, the program stores the estimated position, velocity and attitude, as well as the calculated IMU biases, some relevant KF variables and the results from the fault detection tests.

On the other hand, it is convenient to have a way to process data that is already stored, mainly for development purposes. This way, it is not necessary to use the real IMU and GNSS receiver each time a new modification needs to be tested.

Consequently, although this system is designed to operate in real-time, two input modes are implemented:

Real-time (Online): The INS/GNSS Controller original mode, in which the navigation solution can be obtained in real-time as new IMU and GNSS measurements arrive to the system. Consequently, this mode needs an operating IMU and GNSS receiver. In this test setup, the Picozed Launcher is responsible for driving the controller operation.

Replayer (Offline): When the input data of a previous run has been logged with the INS/GNSS Controller, it can be re-fed into the system through a replayer program. It simply reads the log files and sends the data to the controller through the corresponding input interface functions. This enables offline operation, being able to try different system designs or configurations for the same run. This gives more easily interpretable comparisons in order to test the system performance.

These input/output operation modes can be combined in the required way to obtain the desired functionality. This way, these different modes makes the implemented software very versatile.

Chapter 6

Results and Evaluation

The implemented system needs to be tested in real conditions to evaluate its performance. Accordingly, a measurement campaign was carried out in the DLR Oberpfaffenhofen site with the DLR MessBus.

In addition, several configurations of the INS/GNSS Controller are used to process the measurements to show the incremental benefits of using GNSS attitude loose and tight updates. In order to process the same set of measurements with different configurations, the multiple INS/GNSS Controller operation modes were exploited. First, the Data Logger output mode and the Real-time input mode were used to record the sensors data during the measurement campaign. Then, the Replayer was used with the Complete functionality INS/GNSS Controller mode to obtain the different configurations' navigation solution. All the recorded data is presented with MATLAB®.

This chapter first describes the setup used for the measurements and the ground truth, along with what is assumed to be the true solution. Then, three sections present the performance and improvements between the different used system configurations: 1) the base INS/GNSS integration, 2) the system with the KF loose attitude updates, and 3) the configuration with the DOA-aided tight updates. The corresponding fault tests for each type of KF update are assessed in several subsections associated with the system configuration in which the tests are first implemented. This way, the GNSS-only solution is directly shown in these subsections with the fault tests already applied on it. Last, a summarised comparison between the different configurations is given.

6.1 Test Description and Ground Truth

The test was carried out with the equipment mounted on the roof of the DLR MessBus, as is shown in the right photo in Fig. 6.1. In the left photo it is seen the PicoZed board inside a metallic case in order to limit the possible interferences. It is connected through four SMA cables to the used circularly-polarised 4-element array patch antenna.

The ground truth (GT) is obtained from the INS/GNSS integration of a NovA-

Figure 6.1: MessBus with test equipment mounted on the roof.

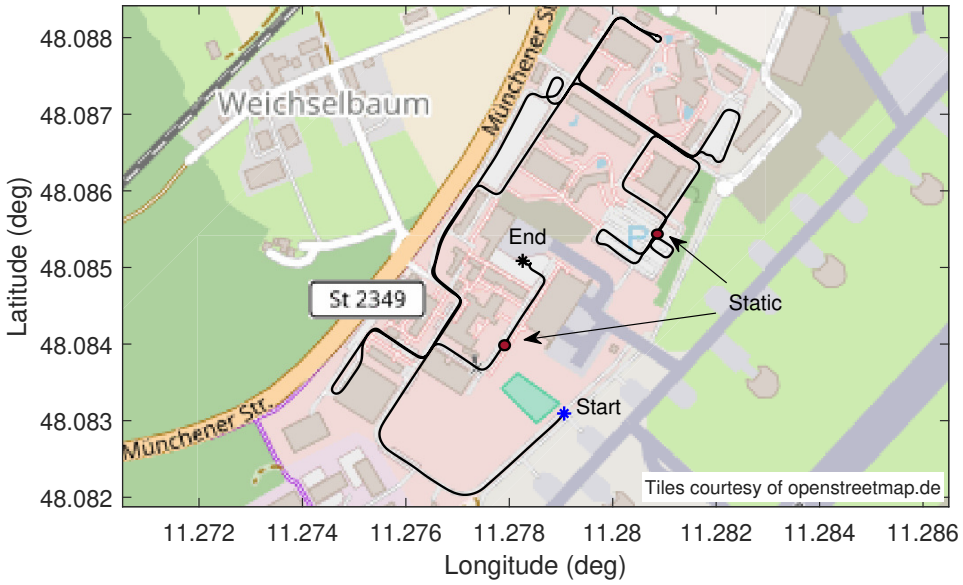
tel GNSS receiver and MEMS IMU included in the same package. The data from this device is post-processed by NovAtel *Inertial Explorer*, a professional navigation software which applies multiple integration filtering techniques, thus giving the navigation solution shown in Fig. 6.2.

Figure 6.2a shows the path taken for the test run. The MessBus started in the middle of an open-sky area, with more than three minutes of static recording for the several systems calibration. Then, the MessBus drove throughout the DLR Oberpfaffenhofen campus, making two intermediate stops where the red dots indicate. In addition, some sections of the route passed between buildings, thus suffering from poor satellite visibility. These variability of conditions are convenient to see how the implemented algorithms perform.

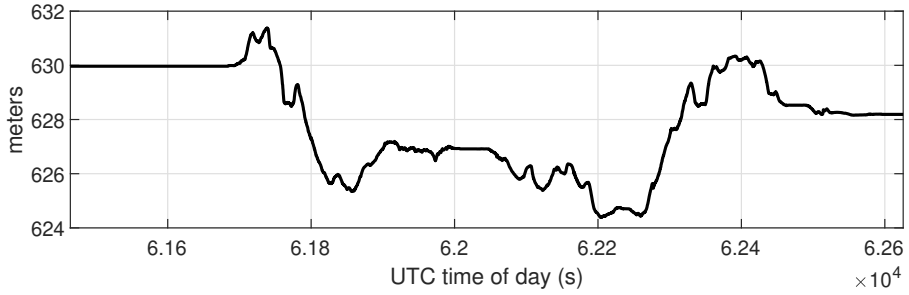
Moreover, as the measurements were taken with a car, the roll and pitch are always close to 0° (Fig. 6.2c) and the yaw relates to the direction of movement of the vehicle.

6.2 Base INS/GNSS Integration Solution

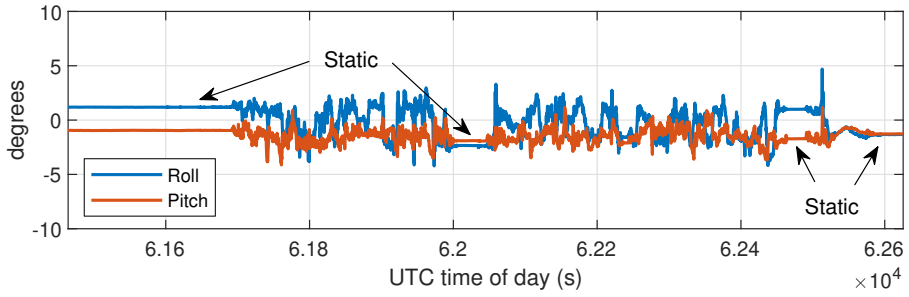
The first configuration used to process the measurements is the one without any GNSS attitude updates. In addition, the correct behaviour of the implemented algorithms can be checked, as well as the operation of the Kalman filter and the appropriate changes between the Navigation Processor FSM states.



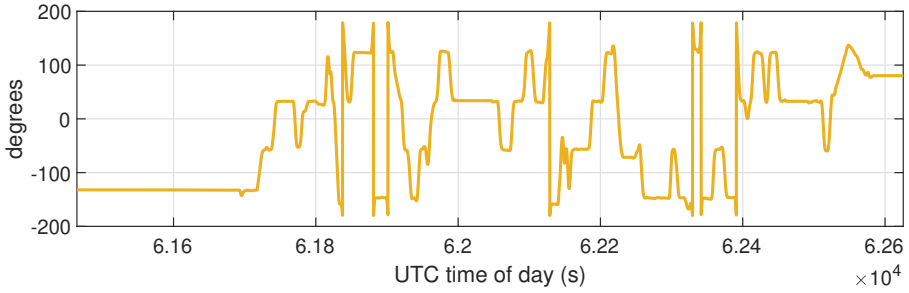
(a) Curvilinear coordinates.



(b) Height.

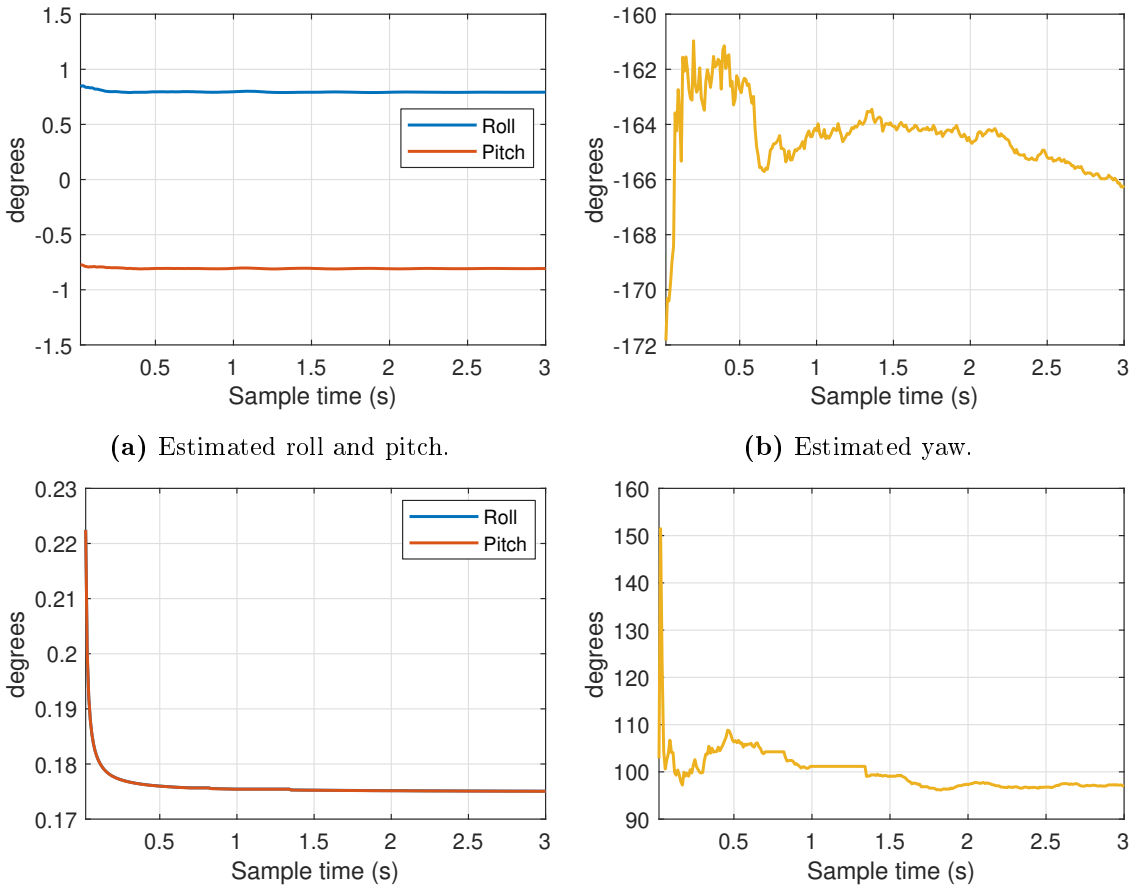


(c) Roll and pitch.



(d) Yaw.

Figure 6.2: Ground truth solution.



(a) Estimated roll and pitch.

(b) Estimated yaw.

(c) Roll and pitch standard deviations.

(d) Yaw standard deviation.

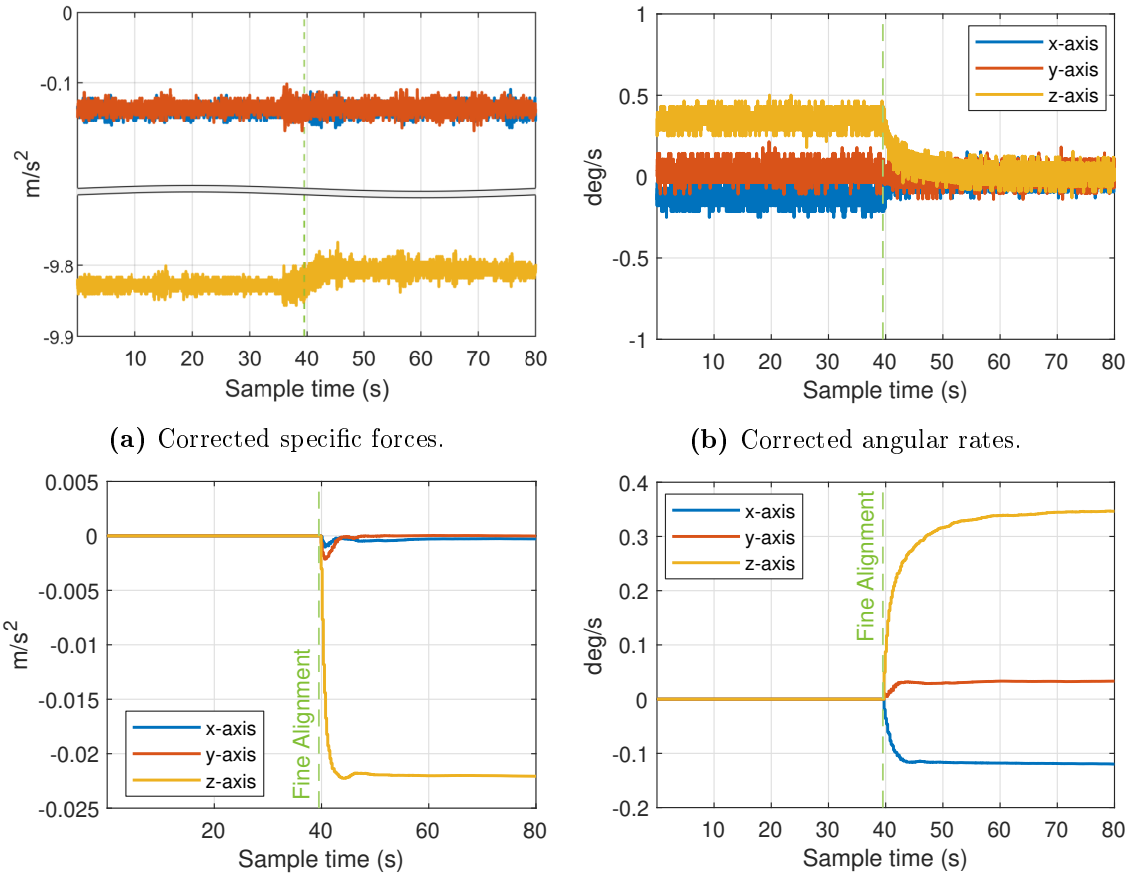
Figure 6.3: Base system coarse alignment performance.

6.2.1 Analysis of the Results

First of all, the performance of the Coarse Alignment is assessed. With this configuration, the roll and pitch are computed by the leveling process, while the yaw is obtained by gyrocompassing, as no attitude aid is provided by any external source. Hence, the covariance propagation is carried out for both processes, as described in Section 4.2. The first three seconds of this stage are shown in Fig. 6.3, where the attitude values are in the top figures and their propagated standard deviations in the bottom ones.

The roll and pitch quickly reach stable values, as the accelerometers have good observability of the gravity-related specific force. However, the low-cost gyroscopes struggle to properly detect the small angular rate associated to the Earth rotation. Thus, gyrocompassing is much slower averaging the noise of the yaw. This is clearly seen in the figures, where the uncertainties of the roll and the pitch monotonically decrease, converging in less than three seconds. On the other hand, the yaw standard deviation has more difficulty to converge.

The attitude uncertainties converge to the propagated IMU biases standard de-



(a) Corrected specific forces.

(b) Corrected angular rates.

(c) Estimated accelerometer biases.

(d) Estimated gyroscope biases.

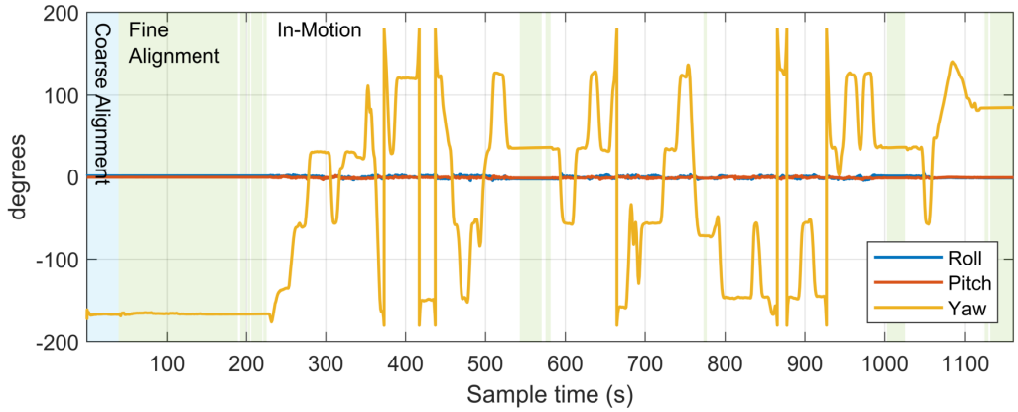
Figure 6.4: Base system IMU biases EKF estimation.

viations, as these can not be averaged out. For leveling this is not a big problem as the standard deviation associated to the accelerometer biases is small enough, thanks to the gravity force being much larger than the bias values. However, this effect is highly noticeable for the yaw, where the Earth-rotation angular rate is much smaller than the gyroscope biases, thus resulting in a converged yaw standard deviation greater than 90°. This is practically the same as not knowing the heading of the vehicle, and this summarises the problem of trying to use gyrocompassing with low-cost IMUs.

This way, Fig 6.3 proves the correct operation of the Coarse Alignment stage and its covariance propagation. Once every noise-related attitude standard deviation has reached the configured threshold ($\sigma_{\max} = 1^\circ$ in this case), this stage is complete and the system passes to Fine Alignment. This is shown in Fig. 6.4. In this stage, the KF is initialized and it starts operating, effectively estimating the IMU biases thanks to the ZVU and ZARU updates.

The specific forces correction is shown in the left figures. However, the angular rate corrections are more noticeable, as their values should be around zero during static operation. Thus, in the top right figure, the angular rates clearly start to

(a) Estimated curvilinear position.



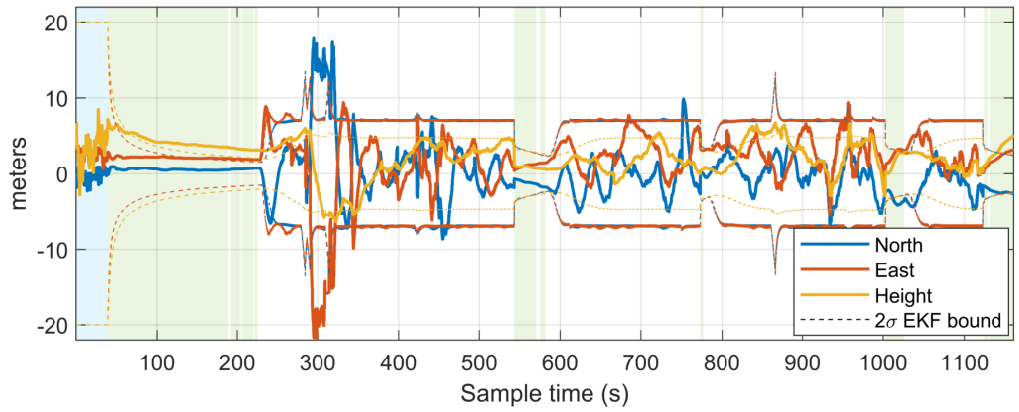
(b) Estimated attitude.

Figure 6.5: Base system navigation solution.

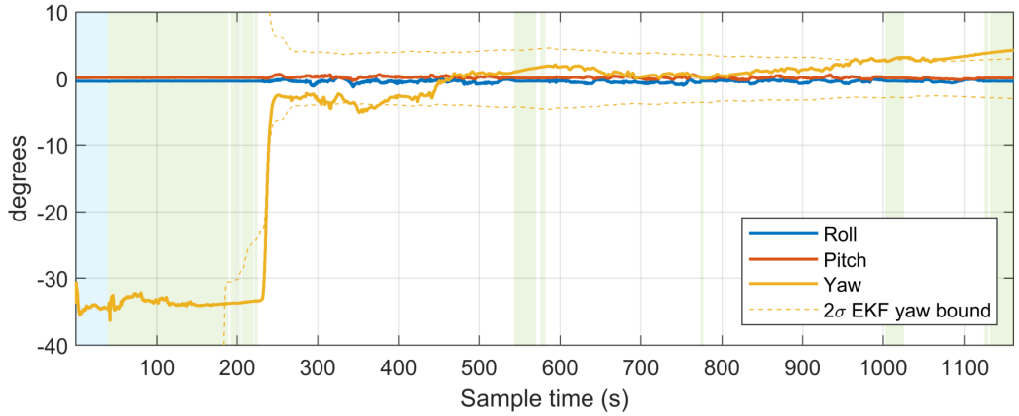
be corrected right after the Fine Alignment stage begins, and their respective bias values are shown in the bottom figure. Moreover, the top right figure offers the IMU measurements perspective to the gyrocompassing problem, where it is seen that it is carrying out the self-alignment process based on the gyroscope biases. This produces the aforementioned wrong initial heading estimation.

Moving on to the state machine and KF performance, Fig. 6.5 shows the navigation solution estimated by the system. The system correctly determines which FSM state needs to be applied at each time, entering Fine Alignment when the vehicle is still (seen in Fig. 6.2), and In-motion mode when it starts moving. As depicted in Fig. 6.5b, the sections with blue, green and white backgrounds indicate that the system is in Coarse Alignment, Fine Alignment or In-motion stages, respectively.

The attitude follows a similar shape as the GT one (Fig. 6.2d) except at the beginning. It is better seen in Fig. 6.6, where the error of the integrated navigation



(a) Error of estimated curvilinear position.

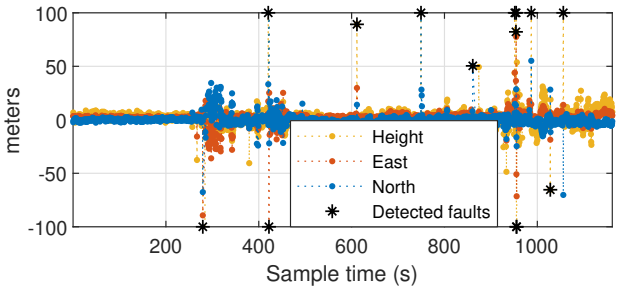


(b) Error of estimated attitude.

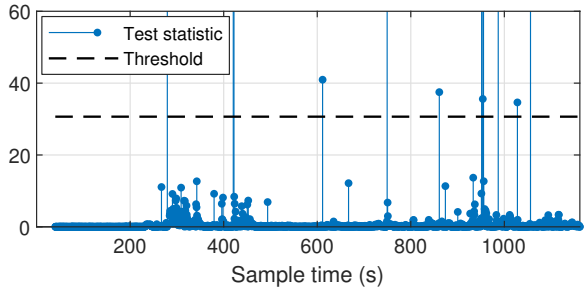
Figure 6.6: Base system navigation solution error (Estimation minus GT).

solution with respect to the GT one is shown. The roll and pitch are well estimated as the IMU has good observability of them. The large initial error in the heading is expected due to gyrocompassing. This causes that the yaw can not be well estimated until the vehicle starts moving and the Kalman Filter has a better observability of its error. This can be noticed in Fig. 6.5a, where the position slightly deviates to the left of the road at the beginning due to this large heading error. Throughout this movement, position updates from GNSS help the KF propagate the errors in the position to the heading error, thus gradually correcting it.

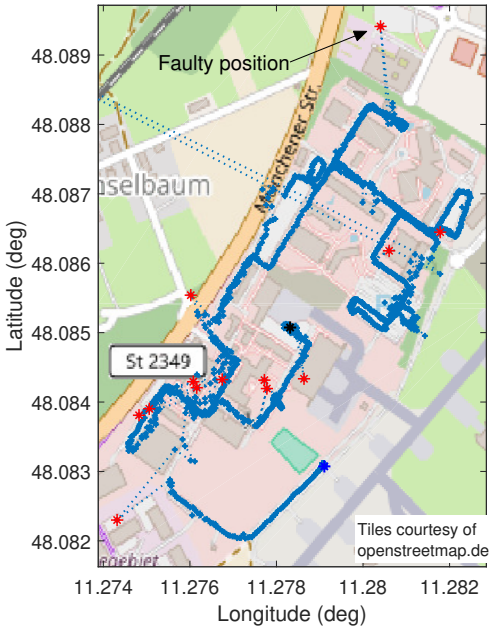
On the other hand, the estimated position is generally correct, as its error tends to be around zero, as Fig. 6.6a shows. However, the loosely-coupled system weaknesses are noticeable, as it deviates from the real position whenever the GNSS receiver measurements are very noisy. These are the sections where the position error surpasses 10 m, due to very limited satellite visibility between buildings. Actually, in these sections sometimes there are even less than four satellites in lock, thus not being able to compute a position solution. In those cases, the IMU alone drives the integrated solution, increasing the position uncertainty until new GNSS measurements arrive.



(a) GNSS receiver position error.



(b) EKF position global test.



(c) GNSS Rx position with fault detection.

Figure 6.7: Base EKF position fault detection.

The sections where no GNSS position updates are input to the system, either because they have been discarded or there were less than 4 visible satellites, are marked in red in Fig. 6.5a. These are the same sections where the KF bounds increase in Fig. 6.6a

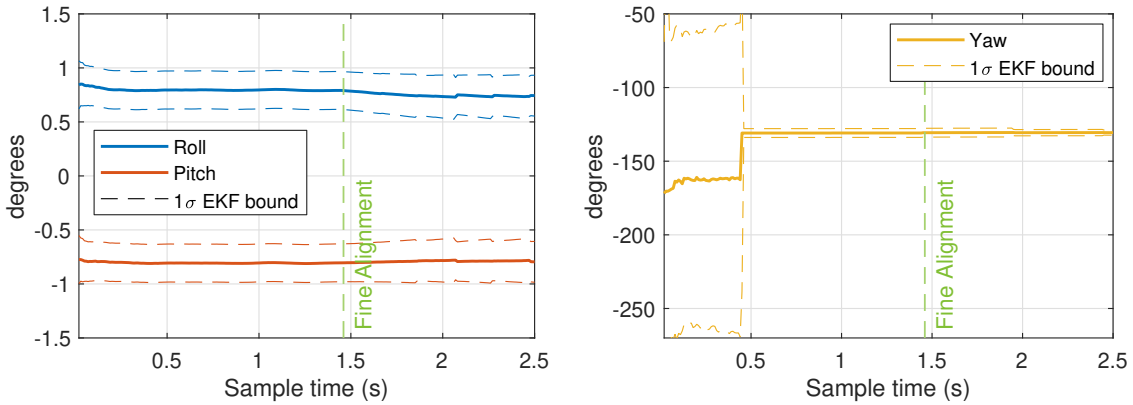
Finally, the KF covariance propagation has the expected behaviour, correctly bounding the solution uncertainty. Particularly, the yaw bound in Fig. 6.6b correctly accounts for its large variance at the beginning and then decreases accordingly during the In-motion stage. For position, the bounds are slightly optimistic when the GNSS measurement is already deviated.

In this loosely-coupled architecture, not much more can be done to improve the position estimation. However, the attitude can be significantly improved with the designed KF attitude updates.

6.2.2 Position Fault Detection

When the KF detects that the position updates measurement innovation is much larger than the expected (based on the previous statistical knowledge and the fault test thresholds), it discards those updates. The threshold is configured for a probability of false alarm of 10^{-6}. Thus, Fig. 6.7 shows the position fault tests performance.

When the test statistic value exceeds that of the threshold, the new GNSS position gets discarded, as denote the black asterisks in Fig. 6.7a. This way, the KF does not use those measurements to update the solution, preventing its corruption. In Fig. 6.7c, the red asterisks are the positions that are discarded from the figures



(a) Estimated roll and pitch.

(b) Estimated yaw.

Figure 6.8: Loose system coarse alignment performance.

on the left, and it is clearly seen that they are effectively erroneous or severely noisy.

In addition, in this figure it is also noticed that, in the sections with low satellite visibility, there are very few and noisy GNSS measurements. In this sense, it is even more appreciable how the integration system correctly filters and bridges the solution in Fig. 6.5a, illustrating its successful performance.

6.3 Loose-Attitude KF Solution

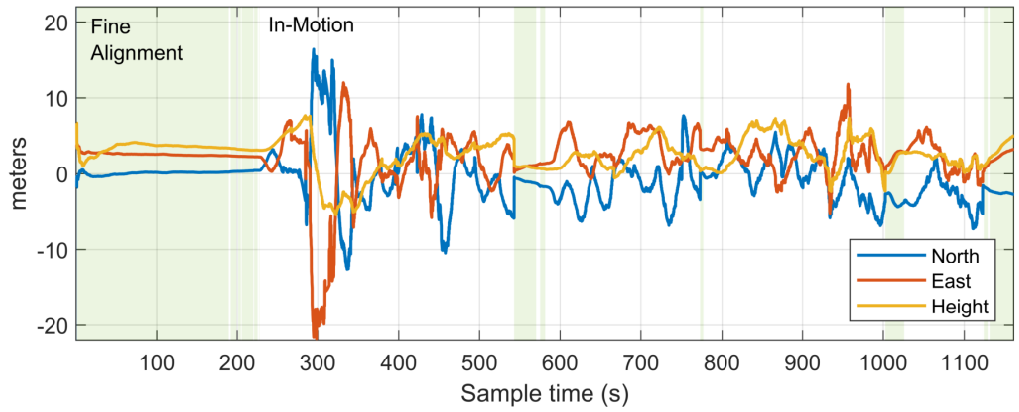
The performance of the second filter configuration with the KF loose attitude updates in assessed in this section.

6.3.1 Performance Improvements

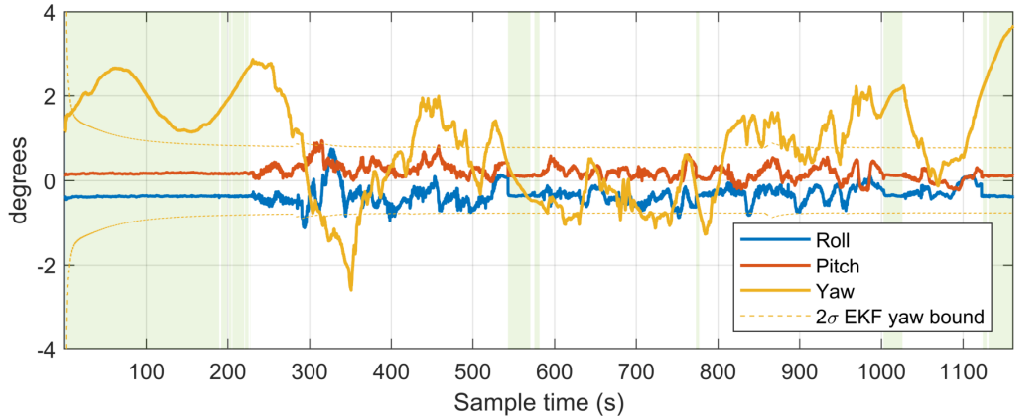
Fig. 6.8 shows that the coarse alignment is much faster than in the base system. Gyrocompassing is aborted when the first GNSS attitude measurement enters the system, thus drastically reducing the yaw uncertainty as a consequence. This way, only leveling is responsible for the change to the Fine Alignment stage. As this process alone obtains a very low noise-related uncertainty for the roll and pitch in significantly less time than gyrocompassing, the next FSM state is already operating in less than two seconds from the start of the system. In comparison, the base system took almost 40 seconds to complete the coarse alignment (Fig. 6.4), and the heading was not even well estimated.

Therefore, with this configuration the Fine Alignment process has more time to estimate the biases, and more effectively as the heading is better known. In this sense, in Fig. 6.9 it is noticed that this FSM stage occupies practically all the time since the begining of the run until the start of the movement.

Compared with the base configuration, now the position solution is better when the In-motion stage is first entered. While in Fig. 6.6a there is a peak in the position errors after finishing the Fine Alignment stage, the transition is smooth in



(a) Error of estimated curvilinear position.



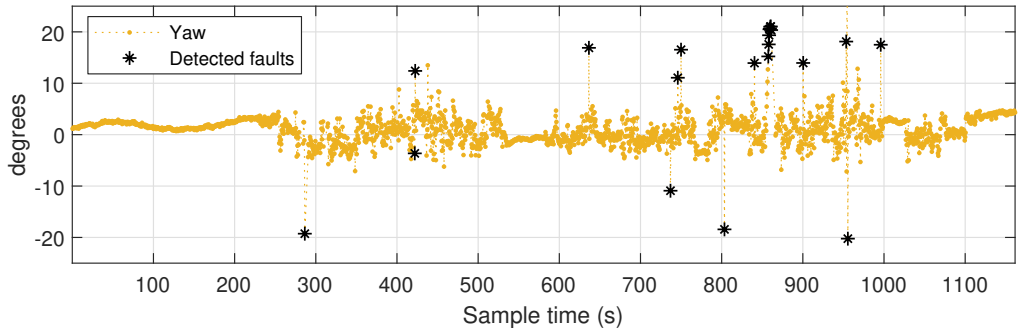
(b) Error of estimated attitude.

Figure 6.9: Loose system navigation solution error (Estimation minus GT).

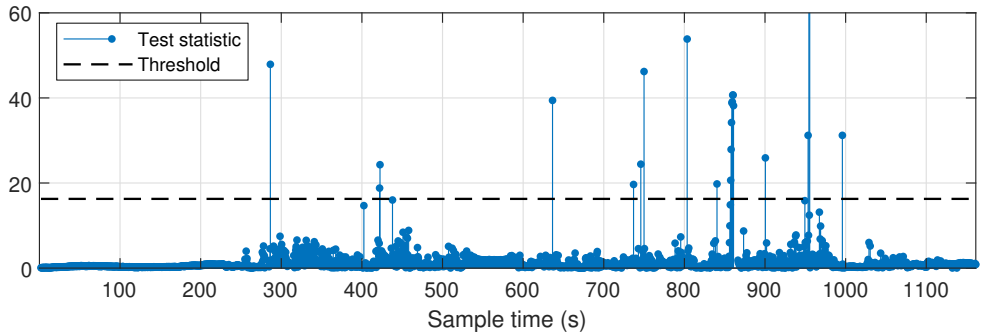
Fig. 6.9a. This is achieved thanks to the use of the GNSS attitude, which offers an accurate heading estimation either in movement or static, and avoids using the (useless) gyrocompassing process. After the first seconds in dynamic operation, the position is mainly driven by the GNSS solution, so this system configuration can not improve it much more.

On the other hand, Fig. 6.9b shows the significant improvement in the attitude estimation when making use of the GNSS loose attitude updates. The roll and pitch are mainly driven by the IMU as it has good observability of them. However, the heading follows that of the GNSS measurements, being more noticeable during the static periods. The variation of the yaw in those sections is produced by the errors in the GNSS receiver DOA estimation. Anyway, the attitude does not have a consistent drift with time and it moves around the true value with a total error maintained below 4° for the entire run.

Even in the sections with poor satellite visibility, the system is able to obtain a good estimation of the attitude, demonstrating the benefits of the integration. However, the KF bound for the heading is not very accurate, as it does not have information about the real-time accuracy of the GNSS attitude computed by the



(a) GNSS receiver heading error with fault detection.



(b) EKF attitude global test.

Figure 6.10: Loose system EKF attitude fault detection.

receiver. Consequently, the KF sometimes believes that the attitude estimation is slightly better than it really is.

Furthermore, a small boresight between the GT IMU and the NavChip is perceptible as the roll and pitch values are always over a similar mean value. As the yaw error has a bigger variation, it is harder to obtain a mean value that adequately represents its boresight.

6.3.2 Attitude Fault Detection

The loose attitude updates that this system configuration implements have associated their corresponding fault tests. Its functionality is the same as in the position fault tests. The configured threshold is set for a probability of false alarm of 10^{-3}, which is higher than the position one because of the narrower set of possible attitude values.

Fig. 6.10b shows the results from these tests. The top figure represents the yaw estimated by the GNSS receiver with the discarded faulty measurements in black asterisks. These faults are detected when the test statistic in the bottom figure exceeds the threshold. It is directly observable that the measurements discarded by the KF are the ones with a large error.

These errors can be a consequence of poor satellite visibility conditions, high multipath or even some type of momentary fault in the DOA estimation, resulting

in the internal LS algorithms converging to an erroneous solution. Fortunately, the system catches them and prevents the solution from corrupting.

In addition, it can be seen that the GNSS attitude presents an oscillatory behaviour at the beginning. This is the same effect observed in Fig. 6.9b, as it is basically following the GNSS solution during the static sections where the KF can not estimate the attitude from the dynamics of the vehicle.

6.4 Tight-Attitude KF Solution

The last tested system configuration uses the KF updates that compute the attitude directly from the DOA vectors instead of using the attitude computed by the GNSS receiver.

6.4.1 Performance Improvements

Figure 6.11 shows the position and attitude errors of the processed navigation solution with respect to the GT. There is not noticeable improvement in the position solution respect to the loose attitude configuration. However, the attitude estimation is improved again from the last configuration.

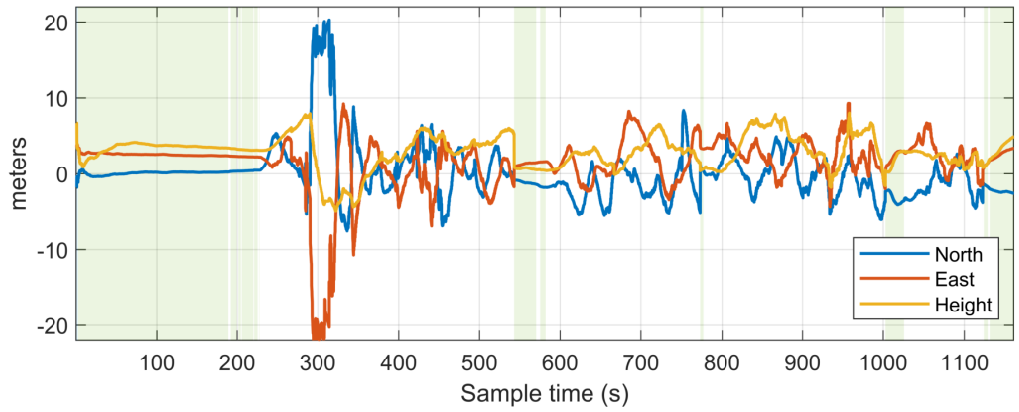
The yaw error is maintained around zero, with a lower variation than in Fig. 6.9b. The maximum attitude error in this case is less than 2°, which is a significant improvement. Furthermore, as the DOAs are directly input to the Kalman filter, it is able to better propagate its uncertainties to the attitude error one. This is seen in Fig. 6.11b, where the yaw bound is much better fitted to the real heading error throughout all the run.

It has to be taken into account that the system still needs an initial attitude value, which is given by the GNSS receiver during the Coarse Alignment stage. After that, when the KF is initialized in the Fine Alignment stage, no more GNSS attitude values are used by the system, and only DOA vectors are input to the KF. Thus, the attitude estimated by the GNSS receiver must be available only at the beginning of the run.

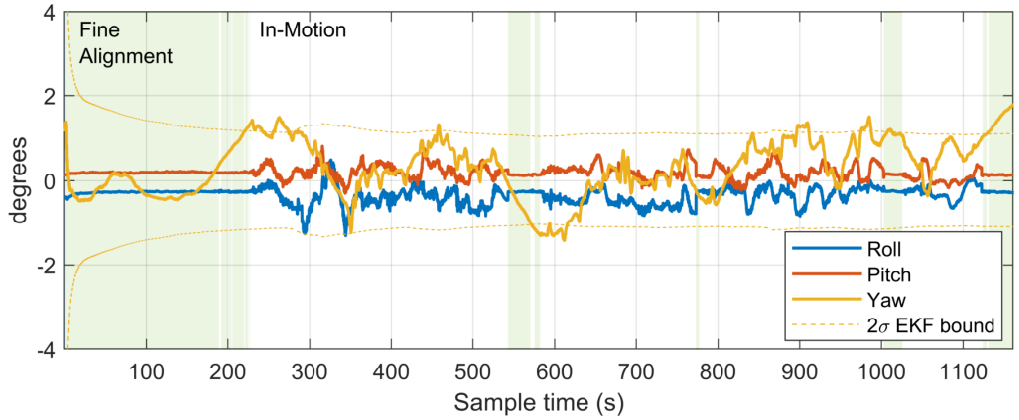
The results from Fig. 6.11b clearly show that the DOA-aided updates entail the best performance of the three studied configurations. Not only the attitude estimation is remarkably accurate for a real-time system, but its uncertainties are also precise.

6.4.2 DOA Filtering and Fault Detection

The tight attitude updates have their corresponding partial fault tests associated to each of the DOAs. This allows for the detection of individual faulty satellite signals, which may be produced by multipath reflections or spoofing devices. In addition, the improvement in the attitude estimation can also be seen from the DOA domain perspective.



(a) Error of estimated curvilinear position.



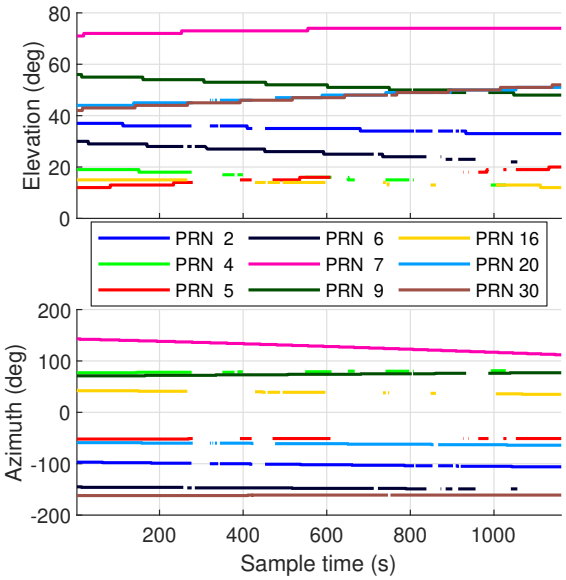
(b) Error of estimated attitude.

Figure 6.11: Tight system navigation solution error (Estimation minus GT).

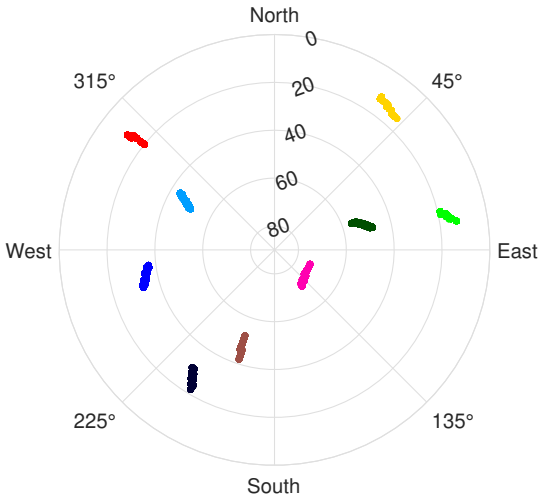
First, Fig. 6.12 shows the expected Ephemeris-based elevation and azimuth of the different visible satellites during the test run. GALANT was configured to only track GPS satellites in this test. This way, there were a maximum number of 9 satellites visible during the test. Moreover, the period of time the satellites remained visible was higher for the ones with a higher elevation, as can be seen in Fig. 6.12a. This is expected as the satellite signals coming from lower elevations are more easily blocked by nearby buildings. In particular, it can be noticed that the section where the sample time is around 300 s there was a very poor satellite visibility due to the buildings around the red line in Fig. 6.5a. This is the same section where the position error is the highest of the whole run, as seen in Fig. 6.11a.

The position of these satellites is also represented in the skyplot from Fig. 6.12b, which is a common visualization of the satellites geometry in the sky related to the user position. The satellites position change over time as they follow their orbits during the almost 20 minutes test.

On the other hand, the skyplot of the DOAs estimated by the GNSS receiver is shown in Fig. 6.13a. This skyplot is obtained in the antenna frame, with the 0°



(a) Elevation and Azimuth with respect to time.



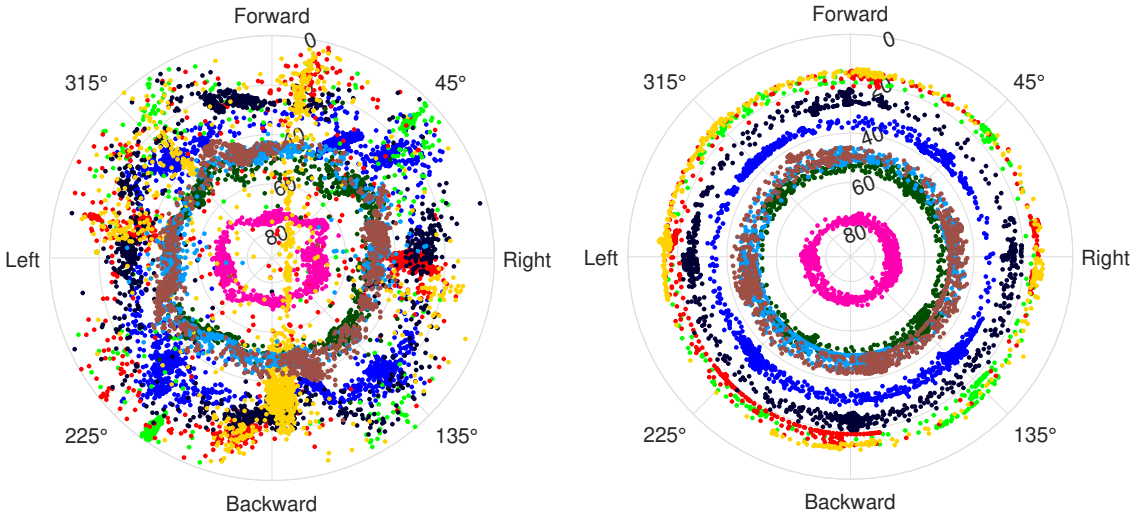
(b) Skyplot.

Figure 6.12: DOAs computed from the Ephemeris by the GNSS receiver ($\mathbf{d}_j^n$).

azimuth representing the GNSS antenna forward direction. In the skyplot on the right, the DOAs estimated by the KF in the antenna frame are shown, which already seem much cleaner than the estimated by the GNSS receiver directly. However, as the vehicle keeps changing its orientation during the test, the DOA of each satellite respect to the antenna frame moves in accordance, making the skyplot look messy. In order to facilitate the visualization of these plots, both are rotated to the local frame through the GT attitude, giving the skyplots in Fig. 6.14.

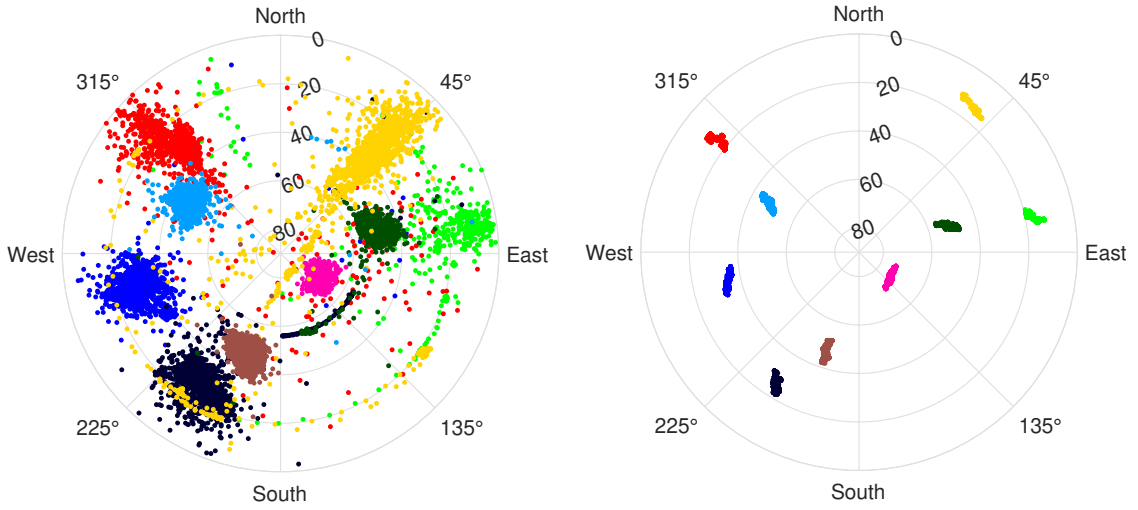
With the skyplots already in the local frame, it is easier to observe the satellite signal DOAs that the GNSS receiver estimates. They are effectively centred around the expected directions based on the Ephemeris. These DOAs are noisy, as expected, and some of them suffer from dispersion problems, more accentuated for low elevation satellites, which have a worse visibility. Consequently, their signals are weaker and they have more multipath. Moreover, in general, lower elevation satellite DOAs are more difficult to be accurately estimated by the antenna array due to the signal coming from a direction increasingly parallel to the antenna plane. Furthermore, isolated faulty measurements can occur sometimes, acquiring a wrong estimated DOA due to momentary signal outages or the algorithm not converging correctly. These possible effects increase the probabilities for the GNSS receiver attitude-estimation algorithm of calculating a wrong value, as seen in the attitude fault detection in Fig. 6.10.

The DOAs from Fig. 6.14a are used by the integrated system to estimate the DOAs shown in Fig. 6.14b. This last skyplot is very important as it represents the integrated-navigation-solution estimated attitude error in the DOA domain. It is evident that the skyplot is highly similar to the one obtained from the Ephemeris (Fig. 6.12b), which agrees with the low error observed from the attitude domain



(a) Antenna-obtained DOAs ($\mathbf{d}_j^a$). **(b)** DOAs estimated by KF ($\hat{\mathbf{C}}_n^a \mathbf{d}_j^n$).

Figure 6.13: Comparison of DOA skyplots in antenna frame.



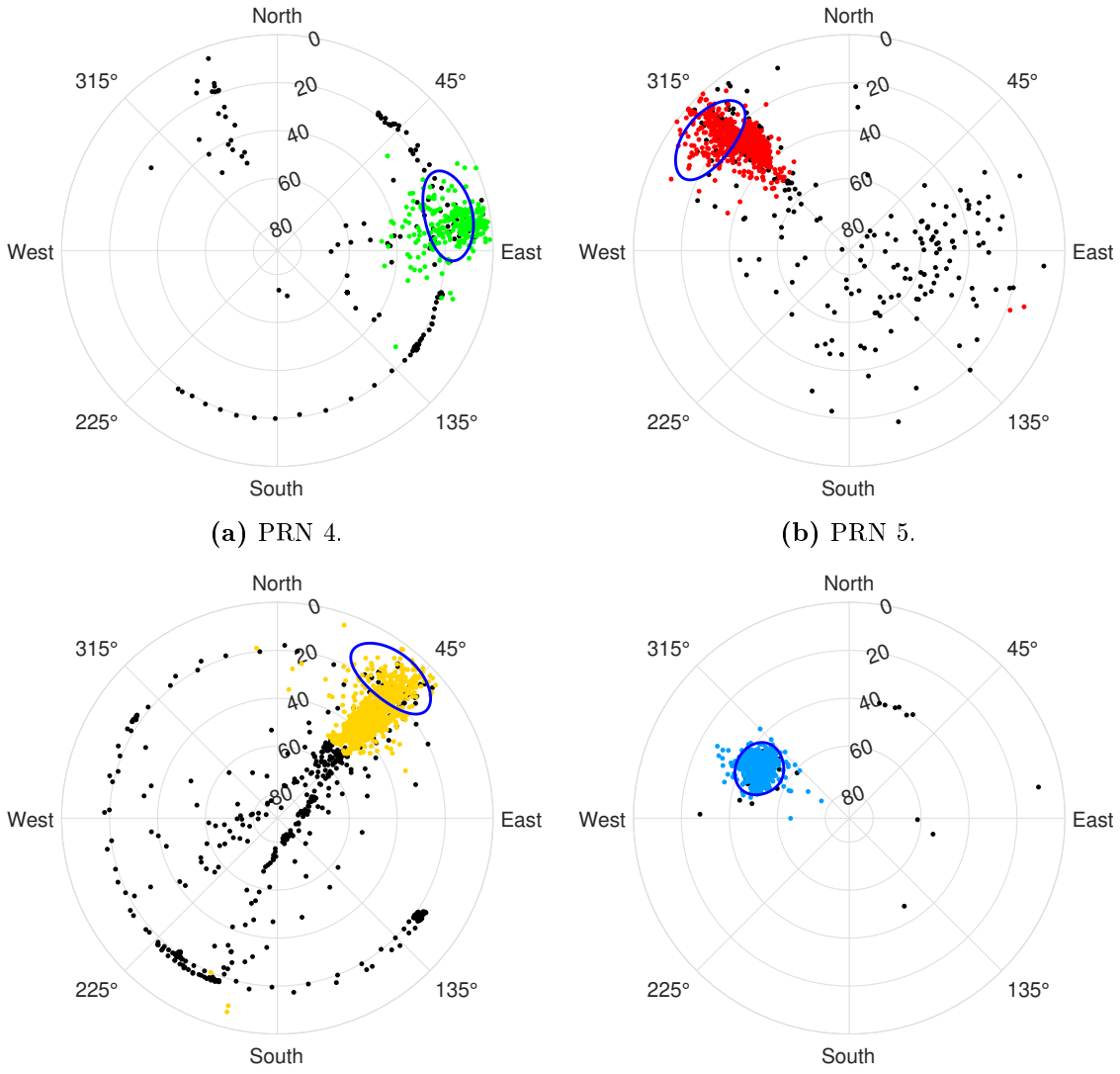
(a) Antenna-obtained DOAs ($\mathbf{C}_a^n \mathbf{d}_j^a$). **(b)** DOAs estimated by KF ($\mathbf{C}_a^n \hat{\mathbf{C}}_n^a \mathbf{d}_j^n$).

Figure 6.14: Comparison of DOA skyplots in GT local frame.

perspective already shown in Fig. 6.11b. Furthermore, the KF estimated attitude does not present any outlier DOA as happens with some antenna-obtained ones. This is the corroboration of the INS/GNSS integration advantages related to the attitude estimation from the DOA domain perspective.

In order to obtain an accurate estimation of the attitude from the noisy DOAs in Fig. 6.14a, first the KF needs to detect which are faulty and discard them. An individual test is applied to each DOA vector as explained in Section 4.5. The probability of false alarm configured for these tests was 10^{-2}. Four examples of these tests performance are shown in Fig. 6.15.

In the fault detection skyplots, the black points represent the DOA measurements



(a) PRN 4. **(b)** PRN 5.

(c) PRN 16. **(d)** PRN 20.

Figure 6.15: Fault detection with antenna-obtained DOAs in GT local frame.

that the KF has categorised as faulty (or discarded) and, thus, have not been used for updating the attitude. The blue ellipses have an elevation radius of 10° and 15° in azimuth, and represent the 2σ bounds that the KF uses to check if the DOAs are inside the expected value range. It is seen that the coloured DOAs are close to their expected position, validating the choices made by the KF tests.

In addition, the KF discards the DOA measurements that arrive when no Ephemeris data is available. Consequently, if the satellite information can not be decoded by the GNSS receiver because the signal is too weak, and yet the DOA of that satellite is estimated, the KF simply discards those DOAs even though they could possibly be correct.

On the other hand, it can be seen that some outliers do not get discarded for satellites 5 and 16. It happens with DOAs that are on the other side of the skyplot

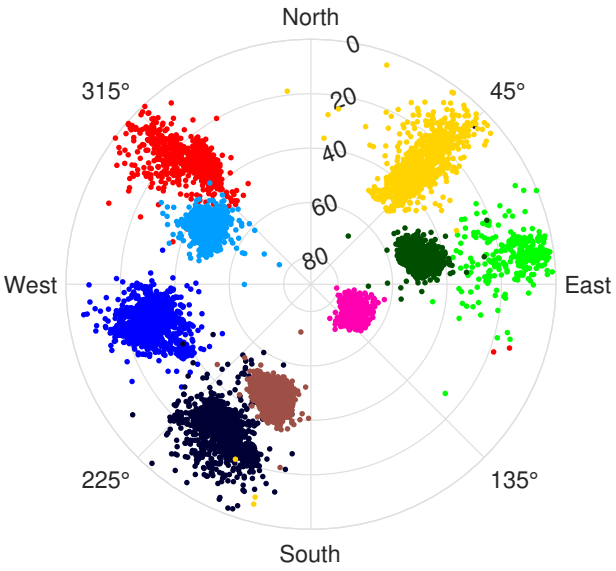


Figure 6.16: Antenna-obtained DOAs in GT local frame without discarded measurements.

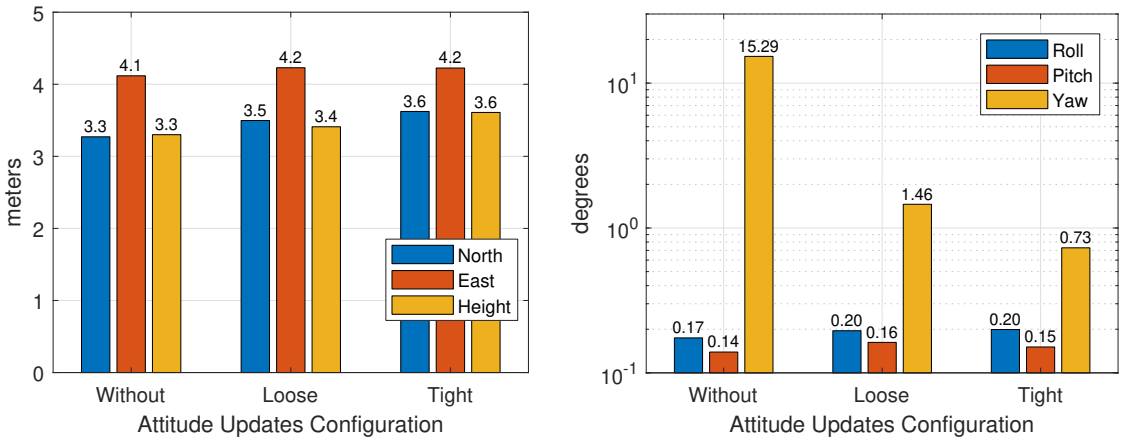
and have a very low elevation. This behaviour is caused by the type of covariance propagation used for the DCVs, described in Subsection 4.4.2. The geometrical particularities of the elevation and azimuth angles cause some troubles when their variances are propagated. Hence, a better set of descriptive values for the satellite position could be used to assess this problem. In any case, these outliers do not cause an appreciable degradation of the attitude estimation.

Finally, removing all the discarded DOAs by the KF, the *cleaned* antenna-estimated skyplot is shown in Fig. 6.16. This figure only contains the DOA measurements that the KF actually inputs to the system to update the attitude. The improvement in the skyplot is significant, better containing the DOAs where they are expected to be. This way, the integrated system can be used to obtain a filtered version of the DOAs estimated by the multi-antenna GNSS receiver. Therefore, the tight-attitude system configuration not only improves the estimated attitude solution, but can be potentially helpful for the DOA measurements evaluation and filtering in real-time.

In this test, all the faulty DOAs were caused by multipath reflections and estimation errors. However, in the case that spoofed signals were also present, the system would detect and discard them in the same way, assuming that they come from a direction different to the expected one. Hence, this system can assist in the detection of multipath and nearby spoofing devices.

6.5 Comparison between KF Configurations

As a summary, an explicit comparison is made between the solution estimated by the three tested system configurations. The Root-Mean-Square Errors (RMSE) of the



(a) RMSE of estimated position.

(b) RMSE of estimated attitude.

Figure 6.17: Comparison of the root-mean-square error (RMSE) of the different EKF configurations.

different solutions are shown in Fig. 6.17. However, before obtaining these values, the boresight between the NavChip and the ground truth IMU is removed from the roll and pitch by subtracting their average values. This gives a better overview of their true errors.

As discussed, the position estimation is mainly driven by the GNSS measurements, so there is not a notable change between the different configurations, as Fig. 6.17a shows. However, the improvement in the attitude estimation as the system introduces tighter attitude updates is remarkable.

As already mentioned, the roll and pitch are well estimated in the three systems as they are mainly driven by the IMU, which has a good observability of them.

With respect to the heading, the base system has an error above 10° due to its lack of observability during the initial static period. This error is significantly reduced when the loose attitude updates are applied, already obtaining an error below 2°. Finally, the tight attitude configuration achieves a RMSE below 1°, which is noteworthy for a real-time system equipped only with a low-cost IMU and a GNSS receiver. Therefore, it is clear that the DOA-aided KF configuration gives the best performance of the three systems.

Chapter 7

Conclusions and Outlook

This last chapter attempts to collect the most relevant achievements reached throughout this master thesis. Some conclusions are extracted related to the obtained results, focusing in the attitude estimation, which is the main point of this work. In addition, different possibilities for future work on this topic are pointed out.

7.1 Achievements

A fully functional real-time INS/GNSS integration architecture for precise attitude estimation has been created. The implemented filter is able to integrate data from inertial sensors and a multi-antenna-capable GNSS receiver to provide a complete navigation solution. The designed algorithms have been tested under real conditions through a measurement campaign to prove their performance.

First, it is relevant to explicit the work that was developed in parallel or already available before the start of this work, either theoretical or tasks carried out at DLR:

- The theoretical background on Kalman filter-based basic INS/GNSS integration is widely known, along with its basic system model and the inertial strapdown algorithms. An implementation of these algorithms in MATLAB previously carried out by the research group at DLR was used as code reference.
- The used hardware platform, along with the multi-element array antenna, were already designed and built at DLR facilities.
- The GALANT multi-antenna GNSS receiver software was already developed by DLR researchers. This includes its internal DOA and attitude estimation algorithms.
- The *Picozed Launcher* program, which serves as the communication interface with both the NavChip IMU and GALANT from the Linux system, was mostly developed. Some modifications and improvements were made to this program during the development of this work.

On the basis of this previous work, the following list summarises the particular tasks that have been carried out during this thesis work, separated by categories:

- **Research and development:**
  - Design of the navigation state machine.
  - KF algorithms derivation for the GNSS attitude and DOA integration.
  - Derivation of the covariance propagation equations for the initial attitude and DOA uncertainties.
  - GNSS receiver modelling.
- **Implementation:**
  - Interface for the available input data.
  - Data structures for IMU measurements, GNSS solution and DOA vectors.
  - Alignment of the measurement timestamps from different sources.
  - Context-aware state machine controller.
  - Strapdown and KF algorithms with the developed attitude-related updates.
  - Fault detection tests.
  - Efficient data logger and offline replayer.
- **Evaluation:**
  - Measurement campaign with the DLR MessBus to collect the needed real data.
  - Navigation state machine identification of the vehicle dynamics.
  - Performance of the different FSM stages.
  - Fault tests operation and multipath detection capabilities.
  - Comparison between different KF configurations: without any attitude external help, with loose attitude updates and with DOA-aided tight updates.
  - DOA domain perspective analysis.

As a summary, new algorithms for improving the accuracy of the attitude estimation have been developed. The system is able to operate in real-time and it has been implemented in a C++ framework, having proved its adequate functionality.

7.2 Conclusions

Several conclusions can be drawn based on the analysis of the results obtained in Chapter 6:

First, the integration between IMU and multi-antenna GNSS has proven to be a successful approach to obtain a navigation solution with a high attitude accuracy. The heading observability problem of the IMU is compensated by the use of the attitude estimated by the GNSS receiver. The IMU, on the other hand, helps to

filter and bridge the attitude solution for a high availability even in critical satellite visibility scenarios. This way, precisions below one degree have been obtained in real conditions feeding the system directly with the directions of arrival of the satellites. It is therefore suggested that this integration system could be suitable for applications where high attitude precision and availability is needed.

Second, the implemented navigation state machine has shown to correctly apply the different stages depending on the solution uncertainty and the dynamics context. This way, it can execute the corresponding algorithms for each stage in a successful manner, thus ensuring a proper initialization of the integration filter and an adequate operation of the system.

Third, from the comparison between system configurations, it is clear that the DOA-aided KF configuration has the best performance at attitude estimation. It can also potentially offer a better uncertainty bound for the estimated attitude given that an adequate measurement modelling is provided. In addition, it is able to detect faulty satellite signals, which can be helpful for multipath and spoofing detection. These advantages indicate that the proposed system may be potentially helpful for safety-critical applications.

Even though the direct use of DOAs in the system is more advantageous, using the attitude computed by the GNSS receiver already offers a great improvement over the basic INS/GNSS integration scheme. Therefore, this strategy is also interesting when there is limited access to the GNSS receiver data and no DOAs can be obtained.

On the other hand, the implemented loosely-coupled integration makes the system highly dependent on the GNSS receiver computed position. Although the fault tests successfully remove the most severe errors and prevent the solution from corrupting, it is clear that a tighter integration is needed to improve the position solution.

Finally, related to the DOA estimation, it has been shown that the tight attitude KF together with the partial fault tests can be beneficial in order to filter these DOAs. By removing the faulty ones, this system can serve as an intermediate filtering step for other systems that may need to use these DOAs.

7.3 Outlook

During the development of this master thesis, several new ideas for improvement emerged. While many of them were implemented throughout the thesis, some of the ideas involve more research and time to be carried out. These are proposed here as outlook for future work, along with some objectives related to the logical progression of this work:

- **Tightly-coupled KF.** The most obvious step to follow after this work is to implement a complete *tightly-coupled* Kalman filter architecture. This way, the system would directly use the GNSS pseudoranges to compute the PVT solution inside the integration filter. This will allow a better position estimation. Furthermore, it would be a great improvement for position accuracy

in limited satellite visibility conditions, as the system could benefit from the pseudoranges information even when less than four satellites are visible.

- **Navigation FSM behaviour.** The Degraded state operation has been out of scope in this work. In this sense, the design of the different FSM stages actions could be further developed. In addition, more required conditions to trigger the changes between states could be implemented, taking the solution uncertainty more into account after the Coarse Alignment is complete.

- **IMU error modelling.** The IMU statistics have been assumed to have values that resulted in an adequate system operation. Moreover, only the IMU biases have been estimated by the KF. Thus, a more extensive analysis on the IMU errors should be carried out. In addition to a proper error modelling, the low-cost IMU would greatly benefit from estimating more of its errors with the KF, as the scale factors. This would improve the INS solution accuracy.

- **DOA variance model.** The DOA skyplots showed that the satellites with a lower elevation had a higher dispersion in their DOA estimation. Consequently, an elevation-dependent model of the DOA errors could be obtained to feed the KF with a better statistical knowledge of these vectors. This way, the KF would trust the lower-error DOAs more, improving the attitude estimation.

- **Complete integrity monitoring.** Some fault tests have been implemented in this system, either in the input interfaces and the KF algorithms. However, a better integrity monitoring could be implemented to verify the correct operation of the system in real-time and the computed solution validity.

- **GNSS receiver aiding.** The attitude estimation and DOA filtering outputs could be fed back to the GNSS receiver in order to aid the DOA estimation. This way, the receiver would have a better starting point for the convergence of its internal algorithms.

- **Blind calibration approach.** The measurement campaign was carried out while providing the GNSS receiver with a reference signal for the antenna-array coefficients calibration. Recently, a new *blind calibration* method has been developed at DLR which avoids using any reference signal for the antenna coefficients computation. Thus, it would be interesting to obtain measurements using this technique to check its performance in dynamic operation.

Pablo Mateos Ruiz
September 24, 2021

Appendix A

KF Transmission Matrices

The INS state propagation matrices resolved in a local frame are shown below, extracted from [1]. Each of these matrices, named by $\mathbf{F}_{ij}^{n}$, represents how the error j propagates into the i one. These subscripts can be "a" (attitude error), "v" (velocity error) or "p" (position error).

The following matrices relate to Eq. (3.22) and, to calculate them, Eqs. (2.15), (2.23) and (2.20) are needed:

$$\mathbf{F}_{aa}^{n} = -\left[\boldsymbol{\omega}_{in}^{n}\times\right] \tag{A.1}$$

$$\mathbf{F}_{av}^{n} = \begin{bmatrix} 0 & \dfrac{-1}{R_E + h_b} & 0 \\ \dfrac{1}{R_N + h_b} & 0 & 0 \\ 0 & \dfrac{\tan L_b}{R_E + h_b} & 0 \end{bmatrix} \tag{A.2}$$

$$\mathbf{F}_{ap}^{n} = \begin{bmatrix} \omega_{ie}\sin L_b & 0 & \dfrac{v_E}{(R_E + h_b)^2} \\ 0 & 0 & \dfrac{-v_N}{(R_N + h_b)^2} \\ \omega_{ie}\cos L_b + \dfrac{v_E}{(R_E + h_b)\cos^2 L_b} & 0 & \dfrac{-v_E \tan L_b}{(R_N + h_b)^2} \end{bmatrix} \tag{A.3}$$

The following matrices relate to Eq. (3.23). In order to compute them, the required expressions are Eqs. (2.15) and (2.29):

$$\mathbf{F}_{va}^{n} = -\left[\left(\mathbf{C}_{b}^{n}\mathbf{f}_{ib}^{b}\right)\times\right] \tag{A.4}$$

$$\mathbf{F}_{vv}^{n} = \begin{bmatrix} \dfrac{v_D}{R_N + h_b} & -\dfrac{2v_E \tan L_b}{R_E + h_b} - 2\omega_{ie} \sin L_b & \dfrac{v_N}{R_N + h_b} \\ \dfrac{v_E \tan L_b}{R_E + h_b} + 2\omega_{ie} \sin L_b & \dfrac{v_N \tan L_b + v_D}{R_E + h_b} & \dfrac{v_E}{R_E + h_b} + 2\omega_{ie} \cos L_b \\ -\dfrac{2v_N}{R_N + h_b} & -\dfrac{2v_E}{R_E + h_b} - 2\omega_{ie} \cos L_b & 0 \end{bmatrix} \tag{A.5}$$

$$\mathbf{F}_{vp}^{n} = \begin{bmatrix} -\dfrac{(v_E \sec L_b)^2}{R_E + h_b} - 2v_E \omega_{ie} \cos L_b & 0 & \dfrac{(v_E)^2 \tan L_b}{(R_E + h_b)^2} - \dfrac{v_N v_D}{(R_N + h_b)^2} \\ \begin{pmatrix} \dfrac{v_N v_E \sec^2 L_b}{R_E + h_b} + 2v_N \omega_{ie} \cos L_b \\ -2v_D \omega_{ie} \sin L_b \end{pmatrix} & 0 & -\dfrac{v_N v_E \tan L_b + v_E v_D}{(R_E + h_b)^2} \\ 2v_D \omega_{ie} \sin L_b & 0 & \dfrac{(v_E)^2}{(R_E + h_b)^2} + \dfrac{(v_N)^2}{(R_N + h_b)^2} - \dfrac{2g_0}{r_{eS}^e} \end{bmatrix} \tag{A.6}$$

where r_{eS}^e is the geocentric radius at the surface, given by:

$$r_{eS}^e(L) = R_E \sqrt{\cos^2 L + \left(1 - e^2\right)^2 \sin^2 L} \tag{A.7}$$

Finally, the following matrices relate to Eq. (3.24). Once again, Eq. (2.15) is used, along with Eq. (2.14):

$$\mathbf{F}_{pv}^{n} = \mathbf{T}_{r(n)}^{p} \tag{A.8}$$

$$\mathbf{F}_{pp}^{n} = \begin{bmatrix} 0 & 0 & -\dfrac{v_N}{(R_N + h_b)^2} \\ \dfrac{v_E \sin L_b}{(R_E + h_b) \cos^2 L_b} & 0 & -\dfrac{v_E}{(R_E + h_b)^2 \cos L_b} \\ 0 & 0 & 0 \end{bmatrix} \tag{A.9}$$

(A.10)

These matrices can be found in several textbooks with slightly different components depending on the conventions used. The convention followed here is the same as in [1].

Appendix B

Resumen

B.1 Motivación y Objetivos

Los Sistemas Globales de Navegación por Satélite (GNSS) son actualmente una parte esencial de la mayoría de aplicaciones donde se necesita conocer la posición del usuario. Sin embargo, estos sistemas son sensibles a interferencias y a las condiciones de visibilidad, lo cual limita su uso en aplicaciones donde un posicionamiento robusto es necesario. Debido a esto, en este tipo de servicios suele ser de gran ayuda combinar el uso de GNSS con unidades de medición inercial (IMU), ya que éstos proporcionan mediciones a elevada frecuencia que siempre están disponibles y no se ven afectadas por interferencias.

Desde hace algunas décadas, avances en la tecnología han permitido miniaturizar estos sensores, consiguiendo también que sean de bajo coste. Sin embargo, cuando se usan para sistemas de navegación inercial (INS), éstos presentan grandes errores en la estimación de la posición y la orientación a medio-largo plazo debido a las grandes incertidumbres en las mediciones del sensor. Además, es muy complicado para estos sensores de bajo coste estimar el rumbo (dirección en la cual apunta el vehículo) debido a que no existe una gran observabilidad de ello excepto si los giroscopios son de muy alta calidad.

En los últimos años, ha habido un auge en el interés por algunas aplicaciones emergentes en las que la seguridad es crítica, como pueden ser los coches autónomos o pequeños vehículos aéreos no tripulados (UAV). Estas aplicaciones necesitan saber su posición y orientación de manera precisa en todo momento. Así, la integración de GNSS e INS suele ser un enfoque interesante para combinar las ventajas de ambos sistemas y conseguir así un posicionamiento más robusto y continuo. Sin embargo, la estimación del rumbo sigue siendo problemática, sobre todo en condiciones de baja dinámica, ya que GNSS de por sí no proporciona una estimación de la orientación del vehículo.

En este contexto, las ventajas de usar receptores GNSS multiantena están siendo estudiadas en el Instituto de Navegación y Comunicación de la Agencia Espacial Alemana (DLR). Estos receptores son capaces de estimar la orientación del vehículo a partir del desfase entre las señales recibidas por los distintos elementos del array

de antenas. Además, también permiten limitar las posibles interferencias mediante técnicas de conformación del haz [4, 5, 6]. Aún así, estas técnicas siguen en desarrollo y aún no se han integrado con otros sistemas.

De esta manera, la combinación entre receptores GNSS multiantena y sensores inerciales puede ser un planteamiento sólido para obtener una estimación no sólo de la posición sino también de la orientación y rumbo que sea robusta y precisa. Así, este enfoque puede resultar beneficioso para este tipo de aplicaciones que necesitan seguridad y que además tengan restricciones de coste y espacio.

Para conseguir esta integración, se necesitan nuevos algoritmos que permitan incorporar las medidas adicionales de los receptores GNSS multiantena al filtro de integración. Este trabajo pretende contribuir en este camino, teniendo los siguientes objetivos principales:

- Diseño teórico de los modelos matemáticos y algoritmos necesarios para integrar las medidas de IMUs y receptores GNSS multiantena.
- Desarrollo del software que permita ejecutar los algoritmos diseñados.
- Evaluación de los algoritmos con medidas reales usando un IMU de bajo coste y GALANT, el receptor GNSS con soporte multiantena desarrollado por el DLR. Comparar su rendimiento con respecto a la integración INS/GNSS básica (sin ayuda de la orientación proporcionada por el receptor GNSS).

B.2 Logros

En este trabajo se ha creado una arquitectura integrada INS/GNSS totalmente funcional que permite una estimación precisa de la orientación en tiempo real. El filtro implementado es capaz de integrar información de sensores inerciales y un receptor GNSS multiantena para proporcionar una solución de navegación completa. Los algoritmos diseñados han sido probados en condiciones reales mediante una campaña de medidas para demostrar su eficacia.

En primer lugar, es relevante explicitar el trabajo que se estaba desarrollando en paralelo o que ya estaba disponible al comienzo de este trabajo:

- A nivel teórico, los esquemas básicos de integración INS/GNSS con filtros de Kalman son ampliamente conocidos. La implementación de estos algoritmos se basó en código MATLAB previamente realizado por el grupo de investigación del DLR.
- La plataforma hardware y el array de antenas multielemento usados ya habían sido diseñados y fabricados en las instalaciones del DLR.
- El software de GALANT junto con sus algoritmos internos de estimación de la dirección de llegada (DOA) de los satélites y de la orientación ya habían sido previamente desarrollados.

- Por último, el programa encargado de servir como interfaz de comunicación entre el hardware y los algoritmos de navegación (*Picozed Launcher*) también estaba mayormente desarrollado.

A partir de este trabajo previo, la siguiente lista resume las tareas concretas que se han realizado durante este TFM, separadas por categorías:

- **Investigación y Desarrollo:**
  - Diseño de la máquina de estados de navegación.
  - Derivación de los algoritmos del filtro de Kalman para la integración de la orientación proveniente del receptor GNSS así como de las DOAs.
  - Derivación de las ecuaciones de propagación de covarianzas para obtener las incertidumbres de la orientación inicial y las DOAs.
  - Modelado del receptor GNSS.
- **Implementación:**
  - Interfaz de entrada para los datos disponibles de IMU y GNSS, junto a estructuras para almacenar esos datos.
  - Alineamiento de las marcas de tiempo de medidas con diferentes orígenes.
  - Controlador de la máquina de estados dependiente del contexto.
  - Algoritmos strapdown y del filtro de Kalman con los modelos desarrollados relacionados con la orientación.
  - Pruebas de detección de fallos.
  - Grabador de datos eficiente y reproductor en diferido.
- **Evaluación:**
  - Realización de una campaña de medidas con el MessBus del DLR para recabar las medidas reales necesarias.
  - Identificación del estado dinámico del vehículo por parte de la máquina de estados de navegación.
  - Rendimiento de las distintas etapas de la máquina de estados.
  - Funcionamiento de las pruebas de detección de fallos y de la capacidad de detección del multicamino.
  - Comparación entre las distintas configuraciones del filtro.
  - Análisis desde el dominio de las DOAs.

B.3 Conclusiones

Se obtienen algunas conclusiones relacionadas con los resultados obtenidos de las medidas experimentales.

En primer lugar, la integración de sensores inerciales y receptores GNSS multiantena ha demostrado ser un enfoque exitoso para conseguir una estimación de la orientación muy precisa. El problema de observabilidad del rumbo de la IMU se compensa con el uso de la orientación estimada por el receptor GNSS. La IMU, por su parte, ayuda a filtrar y aumentar la continuidad de la solución para obtener una alta disponibilidad incluso en escenarios donde las condiciones de visibilidad de los satélites es crítica. De este modo, se ha conseguido mantener un error por debajo de un grado en condiciones reales alimentando el sistema directamente con las direcciones de llegada de los satélites. Por lo tanto, se sugiere que este esquema de integración podría ser adecuado para aplicaciones en las que se necesita una alta precisión y disponibilidad de la orientación.

En segundo lugar, la máquina de estados de navegación implementada ha demostrado aplicar correctamente las diferentes etapas en función de la incertidumbre de la solución y de las condiciones dinámicas del vehículo.

Finalmente, a partir de la comparación entre las distintas configuraciones del sistema, se extrae que el uso de las DOAs directamente dentro del filtro da los mejores resultados en la estimación de la orientación. Además, este esquema permite detectar señales problemáticas, lo que puede permitir la detección de multicamino y *spoofing*. En este sentido, gracias a las pruebas de detección de fallos, las distintas DOAs son filtradas descartando las que se detectan como erróneas, lo cual también puede ayudar a otros sistemas que quieran hacer uso de ellas. Estas ventajas hacen a este sistema el mejor preparado para aplicaciones donde la seguridad es esencial de entre las configuraciones estudiadas.

B.4 Líneas futuras

Se proponen algunas ideas como posible continuación de este TFM:

- **Integración fuertemente acoplada.** Como mejora más inmediata, las *pseudodistancias* a cada satélite se podrían introducir directamente en el filtro de Kalman para así obtener una mejor estimación de la posición, consiguiendo así lo que se suele denominar como un esquema *fuertemente acoplado* completo. De esta manera, la precisión de la posición podría potencialmente mejorarse incluso en condiciones de poca visibilidad, ya que las pseudodistancias seguirían proporcionando cierta ayuda aunque haya menos de cuatro satétlites visibles.

- **Modelado de las medidas.** Las características de la IMU utilizada se han asumido de tal manera que el filtro tuviera un funcionamiento correcto. Sin embargo, un modelado más detallado de los errores de este sensor debería llevarse a cabo para un funcionamiento óptimo del filtro en este sentido. Además, respecto al receptor GNSS, también podría realizarse un modelado más extenso del error de las DOAs dependiendo de la elevación del satélite. Como se ha comprobado que satélites con menor elevación producen mayor error en la estimación de sus DOAs, esta información puede usarse para que el filtro de

Kalman confíe más en los satélites con mayor elevación, lo que podría conllevar una estimación de la orientación aún más precisa.

- **Calibración "a ciegas".** La campaña de medidas se ha llevado a cabo proporcionando una señal de referencia continuamente al receptor GNSS. Éste usa esa señal para estimar correctamente los coeficientes de calibración que necesita para poder obtener las señales de cada elemento del array de antenas adecuadamente. Recientemente, se ha desarrollado en el DLR una nueva técnica de *calibración a ciegas* que no necesita de señal de referencia para estimar estos coeficientes. Por tanto, sería interesante obtener medidas usando este método para comprobar su funcionamiento en condiciones dinámicas.

Bibliography

[1] P. Groves, *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*, ser. GNSS technology and applications series. Artech House, 2008.

[2] A. E. R. Shabayek, C. Demonceaux, O. Morel, and D. Fofi, "Vision Based UAV Attitude Estimation: Progress and Insights," *Journal of Intelligent and Robotic Systems*, vol. 65, pp. 295–308, Jan. 2012. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00759018

[3] Z. Liu, N. El-Sheimy, C. Yu, and Y. Qin, "Motion constraints and vanishing point aided land vehicle navigation," *Micromachines*, vol. 9, no. 5, p. 249, May 2018. [Online]. Available: https://doi.org/10.3390/mi9050249

[4] M. Cuntz, A. Konovaltsev, M. Heckler, A. Hornbostel, L. Kurz, G. Kappen, and T. Noll, "Lessons Learnt: The Development of a Robust Multi-Antenna GNSS Receiver," in *ION GNSS 2010*, Sep 2010. [Online]. Available: https://elib.dlr.de/63990/

[5] M. Meurer, A. Konovaltsev, M. Cuntz, and C. Hättich, "Robust Joint Multi-Antenna Spoofing Detection and Attitude Estimation using Direction Assisted Multiple Hypotheses RAIM," in *Proc. 25nd International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS'2012)*, Sep 2012. [Online]. Available: https://elib.dlr.de/81071/

[6] M. Appel, A. Konovaltsev, and M. Meurer, "Robust Spoofing Detection and Mitigation based on Direction of Arrival Estimation," in *ION GNSS+ 2015*, Nov 2015. [Online]. Available: https://elib.dlr.de/99721/

[7] K. M. Borkowski, "Accurate algorithms to transform geocentric to geodetic coordinates," *Bulletin géodésique*, vol. 63, no. 1, pp. 50–56, Mar 1989. [Online]. Available: https://doi.org/10.1007/BF02520228

[8] U. S. D. M. Agency, *Department of Defense World Geodetic System 1984: its definition and relationships with local geodetic systems.* Defense Mapping Agency, 1987, vol. 8350.

[9] P. G. Savage, "Strapdown Inertial Navigation Integration Algorithm Design Part 1: Attitude Algorithms," *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 1, pp. 19–28, 1998. [Online]. Available: https://doi.org/10.2514/2.4228

[10] D. Titterton, J. Weston, J. Weston, I. of Electrical Engineers, A. I. of Aeronautics, and Astronautics, *Strapdown Inertial Navigation Technology*, ser. IEE Radar Series. Institution of Engineering and Technology, 2004.

[11] A. E. R. Shabayek, C. Demonceaux, O. Morel, and D. Fofi, "Vision Based UAV Attitude Estimation: Progress and Insights," *Journal of Intelligent and Robotic Systems*, vol. 65, pp. 295–308, Jan 2012. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00759018

[12] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, Mar 1960. [Online]. Available: https://doi.org/10.1115/1.3662552

[13] M. El-Diasty and S. Pagiatakis, "Calibration and Stochastic Modelling of Inertial Navigation Sensor Erros," *Journal of Global Positioning Systems*, vol. 7, pp. 170–182, Dec 2008.

[14] O. G. Crespillo, M. Joerger, and S. Langel, "Overbounding gnss/ins integration with uncertain gnss gauss-markov error parameters," in *2020 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, 2020, pp. 481–489.

[15] P. Montgomery, T. Humphreys, and B. Ledvina, "Receiver-Autonomous Spoofing Detection: Experimental Results of a Multi-Antenna Receiver Defense Against a Portable Civil GPS Spoofer," *Proceedings of the Institute of Navigation, National Technical Meeting*, vol. 1, pp. 124–130, Jan 2009.

[16] O. G. Crespillo, "Map based multisensor railway localization enhanced by raw gnss data," Master's thesis, Universidad de Málaga, 2013. [Online]. Available: https://elib.dlr.de/89240/

[17] O. G. Crespillo, A. Grosch, and M. Meurer, "Detection of DME ranging faults with INS coupling," in *2017 Integrated Communications, Navigation and Surveillance Conference (ICNS)*, 2017, pp. 4B2–1–4B2–9.

[18] J.-T. Lin, "Approximating the cumulative chi-square distribution and its inverse," *Journal of the Royal Statistical Society. Series D (The Statistician)*, vol. 37, no. 1, pp. 3–5, 1988. [Online]. Available: http://www.jstor.org/stable/2348373

[19] Avnet, "Picozed | Avnet Boards," Accessed 2021-09-15. [Online]. Available: https://www.avnet.com/wps/portal/us/products/avnet-boards/avnet-board-families/picozed/picozed-board-family

[20] M. Cuntz, A. Konovaltsev, A. Hornbostel, E. Schittler-Neves, and A. Dreher, "GALANT - Galileo Antenna and Receiver Demonstrator for Safety Critical Applications," Feb 2007.

[21] *NavChip. Precision 6-Axis MEMS Intertial Measurement Unit*, Thales Visionix, 2016, Rev. B. Accessed 2021-09-15. [Online]. Available: https://media.digikey.com/pdf/Data%20Sheets/Thales%20Visionix%20PDFs/NavChip_Precision_MEMS_IMU_DS_RevB.pdf

[22] E. Muñoz Diaz, O. Heirich, M. Khider, and P. Robertson, "Optimal sampling frequency and bias error modeling for foot-mounted IMUs," in *International Conference on Indoor Positioning and Indoor Navigation*, 2013, pp. 1–9.

[23] O. J. Woodman, "An introduction to inertial navigation," University of Cambridge, Research report 696, Aug 2007. [Online]. Available: https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-696.pdf

[24] DLR, Institute of Communications and Navigation, "GALANT. Galileo Antenna Demonstrator," Accessed 2021-09-16. [Online]. Available: https://www.dlr.de/kn/en/desktopdefault.aspx/tabid-2081/6941_read-9224/

[25] K. Betke, *The NMEA 0183 Protocol*, Aug 2001. [Online]. Available: https://www.tronico.fi/OH6NT/docs/NMEA0183.pdf

[26] M. Cuntz, A. Konovaltsev, and M. Meurer, "Concepts, development, and validation of multiantenna gnss receivers for resilient navigation," *Proceedings of the IEEE*, vol. 104, no. 6, pp. 1288–1301, 2016.

[27] C. Sanderson and R. Curtin, "Armadillo: a template-based C++ library for linear algebra," *Journal of Open Source Software*, vol. 1, no. 2, p. 26, 2016. [Online]. Available: https://doi.org/10.21105/joss.00026

[28] ——, "A User-Friendly Hybrid Sparse Matrix Class in C++," *Lecture Notes in Computer Science*, pp. 422–430, 2018. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-96418-8_50