DEPARTAMENT DE MICROELECTRÒNICA

I SISTEMES ELECTRÒNICS

Performance Enhancement MEMS Based INS/GPS Integrated System Implemented on a FPGA for Terrestrial Applications

by

Alex Garcia Quinchia

A DISSERTATION

SUBMITTED TO THE POSTGRADUATE SCHOOL OF THE UNIVERSITAT

AUTÒNOMA DE BARCELONA IN PARTIAL FULFILLMENT

OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

IN

MICROELECTRONICS AND ELECTRONICS SYSTEMS

BELLATERRA (BARCELONA) , SPAIN

FEBRUARY 2014

UNIVERSITAT AUTÒNOMA DE BARCELONA

Departament de Microelectrònica i Sistemes Electrònics

PERFORMANCE ENHANCEMENT MEMS BASED INS/GPS INTEGRATED SYSTEM IMPLEMENTED ON A FPGA FOR TERRESTRIAL APPLICATIONS

Dissertation presented to obtain the degree of Doctor of Philosophy in Microelectronics and Electronic Systems.

Author: ALEX GARCIA QUINCHIA

Director : PROF. CARLES FERRER I RAMIS

Dr. Carles Ferrer i Ramis, Full Professor from the Universitat Autònoma de Barcelona.

CERTIFY:

that the dissertation **"Performance Enhancement MEMS Based INS/GPS Integrated System Implemented on a FPGA for Terrestrial Applications"**, presented by Mr. Alex Garcia Quinchia to obtain the degree of Doctor of Philosophy in Microelectronics and Electronic Systems, has been done under his direction at the Universitat Autònoma de Barcelona.

Prof. Carles Ferrer i Ramis

Bellaterra (Barcelona), February 2014

This research work has been financed by support partially funded by the Spanish Ministry of Economy and Competitiveness CICYT under the projects: TEC 2009-09712 and TEC2006-04123, and has been possible thanks to a postgraduate scholarship Research Personnel in Training (PIF) from the Universitat Autònoma de Barcelona.

To my family,

"The way to do is to be."

Laozi.

Resumen

Hoy en día con el desarrollo de sensores inerciales basados en Sistemas Micro-electromecánicos (MEMS), podemos encontrar acelerómetros y giróscopos embebidos en diferentes dispositivos y plataformas, teniéndolos en relojes, teléfonos inteligentes, consolas de video juego hasta sistemas de navegación terrestre y vehículos aéreos no tripulados (UAVs), *etc.* A pesar del amplio rango de aplicaciones donde están siendo utilizados, los sensores inerciales de bajo costo (grado MEMs) son afectados por errores aleatorios y determinísticos que degradan el rendimiento de los sistemas, en especial, los sistemas de navegación, un ejemplo de ello es la integración del Sistema de Posicionamiento Global (GPS) y el Sistema de Navegación Inercial (INS). Aunque diferentes investigaciones se han realizado para minimizar y modelar el error estocástico de los sensores inerciales MEMS, la estimación de los parámetros de este error y su eliminación sigue siendo una tarea poco fácil de realizar. Por este motivo, en esta tesis planteamos soluciones para facilitar la estimación de los parámetros y la compensación del error estocástico, específicamente, para el bias-drift, con el propósito de mejorar el rendimiento del sistema integrado GPS/INS basado en MEMS. Adicionalmente, el sistema de navegación es implementado en un Arreglo de Compuertas Programables en Campo (FPGA), donde se exploran las posibilidades que este tipo de plataformas puede ofrecer y como recursos hardware dedicados podrían ser utilizados para acelerar el filtro de navegación que es donde se presenta el mayor costo computacional. Finalmente, con el fin de evaluar la compensación del error estocástico, el Filtro de Kalman Extendido (EKF) de la estrategia de integración loosely-coupled GPS/INS es aumentado con diferentes modelos del error. Los resultados muestran el rendimiento del sistema de navegación cuando se realiza la compensación del bias-drift bajo bloqueos de la señal del GPS, utilizando datos reales recolectados en un vehículo terrestre en varias campañas.

Abstract

Nowadays with the development of inertial sensors based on Micro-Electromechanical Systems (MEMS), embedded accelerometers and gyroscopes can be found in several devices and platforms ranging from watches, smart phones, video game consoles up to terrestrial navigation systems and unmanned aerial vehicles (UAVs), *etc.* Despite the wide range of applications where such sensors are being used, low cost inertial sensors (MEMS grade) are affected by random and deterministic errors that degrade the systems performance, especially, in navigation systems like the Global Positioning System (GPS)/Inertial Navigation System (INS) integration. Albeit different researches have been conducted to minimize and model the stochastic error of MEMS inertial sensors, the estimation of the random noise parameters and its attenuation is still a non-trivial task. Therefore, in this thesis we propose solutions to facilitate the estimation of the parameters or the stochastic error and its compensation, specifically, for the bias-drift, with the aim of enhancing the performance of MEMS based INS/GPS integrated systems. Additionally, we implement the navigation system on a Field Programmable Gate Array (FPGA), where we explore the possibilities that this sort of platforms could offer and how dedicated hardware resources could be used to accelerate the navigation filter, that presents the highest computational burned. Eventually, in order to assess the compensation of the stochastic error, the Extended Kalman Filter (EKF) of the loosely-coupled GPS/INS integration strategy is augmented with different error models. Results show the performance of the navigation system during the compensation of the bias-drift under GPS signal blockages using real data collected in a land vehicle during several campaigns.

Acknowledgments

Now that my thesis is coming to an end, I would like to express my gratitude to all those who have been part of this work. First, I want to sincerely thank my supervisor, Prof. Carles Ferrer for his continuous encouragement, guidance, for the financial support and for giving me the opportunity to address the challenge that involves a Phd. I also thank members and ex-members of the Department of Microelectronics and Electronics Systems of the Universitat Autònoma de Barcelona, especially, Dr. Elena Martin who gave me advice and the members of the examination committee for the useful comments on this work. I would also like to thank Prof. Joan Oliver for his motivation and for providing me guidance during the hardware development. In addition, I express my gratitude to Jordi Guerrero, Sergio Villar, Victor Soler, Biruk Getachew and Josep Maria Sanz that in one way or another have helped me in various experiments. I thank Carlos Preckler, Alfred Raúl Giménez, José Caballero, Sergi Sanchez, Adria Mendiz and Gustavo Valera for helping me in preparing the field tests and their support during the data collection.

I am also very grateful to Prof. Fabio Dovis for his encouragement, support and for giving me the opportunity to be part of the Navsas research group at Politecnico di Torino, where I learned a lot. I would like to express my sincere thanks to Dr. Gianluca Falco from this group for his guidance, continuous support and for the discussions and comments that helped me to improve my thesis. I also would like to express my gratitude to Dr. Marco Pini, Dr. Emanuela Falleti and Dr. Gianluca Marucco for the support and help they gave me during my stage at the Istituto Superiore Mario Boella.

I am grateful to Dr. Nuria Blanco Delgado for her cooperation and encouragement and to Dr. Isaac Skog for providing me valuable information and the opportunity to visit the Signal Processing Lab at the Royal Institute of Technology.

I gratefully acknowledge the partially funding of my research to the Universitat Autónoma de Barcelona as Personal Investigador en Formación (PIF).

Last but not the least, I would like to thank my friends I spent great time with during these years and my family for their unconditional love and support during this long journey. Thanks to the universe for everything.

Table of contents

List of tables

List of figures

List of Acronyms

ADC	Analog to Digital Converter
AHDL	Altera Hardware Description Language
ARW	Angle Random Walk
ASIC	Application-Specific Integrated Circuit
AV	Allan Variance
BLUE	Best Linear Unbiased Estimator
BPI	Byte-wide Peripheral Interface
CF	Compact Flash
CLB	Configurable Logic Block
CMOS	Complementary Metal-Oxide-Semiconductor
DC	Direct Current
DCM	Direction Cosine Matrix
DDR	Double Data Rate
DPR	Dynamic Partial Reconfiguration
DSP	Digital Signal Processors
ECos	Embedded Configurable Operating System
EDK	Embedded Development Kit
EKF	Extended Kalman Filter

EM	Expectation Maximization
ESC	Electronic Stability Control
FDC	Flight Dynamic Controller
FPGA	Field Programmable Gates Array
FPU	Floating Point Unit
FSL	Fast Simple Link
FSM	Finite State Machine
GMWM	Generalized Method of Wavelet Moments
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
HDL	Hardware Description Language
IC	Integrated Circuit
IEEE	International Electrical and Electronic Engineers
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
ISE	Integrated Software Environment
LC	Loosely-Coupled
LKF	Linearized Kalman Filter
LOD	Levels Of Decomposition
MAC	Multiply Accumulate
MEMS	Microelectromechanical Systems
MODWT	Maximal-Overlap Discrete Wavelet Transform
MOS	MetalâĂŞOxideâĂŞSemiconductor
NED	North-East-Down

NLF	Non-linear Fitting
NMEA	National Marine Electronics Association
NRE	Non-Recurring Engineering
NRMSE	Normalized Root Mean Squared Error
PC	Personal Computer
PCB	Printed Circuit Board
PE	Processing Element
PLB	Processor Local Bus
PLD	Programmable Logic Devices
PROM	Programmable Read-Only Memory
PSD	Power Spectral Density
RRW	Rate Random Walk
RW	Random Walk
SDRAM	Synchronous Dynamic Random-Access Memory
SOC	System-On-Chip
SSM	State-Space Model
TOW	Time Of Week
UART	Universal Asynchronous Receiver Transmitter
UAV	Unmanned Aerial Vehicle
UUV	Unmanned Underwater Vehicle
VHDL	Very High Speed Integrated Circuit Hardware Description Language
VRW	Velocity Random Walk
WAAS	Wide Area Augmentation System
WN	White Noise

WV	Wavelet Variance
XCL	Xilinx CacheLink
XUPV5	Xilinx University Program Virtex 5

Chapter 1

Introduction

1.1 Background and Objectives

Currently many land vehicles are equipped with a Global Positioning System (GPS), which is widely used because of its global availability, portability and low cost. Nonetheless, the GPS is affected by several errors (*i.e.*, multipath, ionosphere and troposphere delays), signal unavailability (*i.e.*, momentary blockage while driving through tunnels, indoor car parks or along urban canyons), voluntary or involuntary signal interference like jamming and spoofing, *etc.* All these errors affect the integrity and reliability of the navigation solution and only some of them can be reduced or mitigated (*e.g.*, multipath and interference). Others are intrinsic in the GPS functioning (*e.g.*, signal blockage and drop in the signal power) and can not be removed. On the other hand, the Inertial Navigation System (INS) provides information about position, velocity and attitude with a higher rate than the GPS. They are inherently immune to the signal jamming and blockage vulnerabilities of GPS, but the accuracy of INS is significantly affected by the error characteristics of the inertial sensors [1]. Since both of these systems have a complementary nature, it is well known that GPS/INS integration provides a higher performance than their stand-alone operation.

In the last few years, advances in the development of Micro-Electromechanical Systems (MEMS) have made possible the fabrication of cheap and small dimension accelerometers and gyroscopes, which have increased the demand of low cost INSs, expanding their usage in several applications where the GPS and the INS are blended, *e.g.*, identifying track defects, terrestrial and pedestrian navigation, unmanned aerial vehicles (UAVs), stabilization of many platforms, *etc.* This inertial sensors based on MEMS technology are fast becoming ubiquitous and the next few years will witness the continued proliferation of these devices with further improvement in cost, performance and integration [2]. Although, low cost and small size of MEMS sensors are attractive characteristics for current navigation systems, they combine electronic and mechanical components, where the small moving parts (mechanical components) that are built inside the MEMS devices are especially susceptible to mechanical noise [3], not to mention the noise caused by the electronic components. As a consequence, those noises degrade the performance of low cost INS/GPS integrated systems in a short period of time. In other words, MEMS based INS presents errors in position, velocity and attitude, which grow rapidly degrading the accuracy of the navigation system in a few seconds [4]. Therefore, a suitable error modelling of MEMS inertial sensors is necessary in order to improve the system performance.

The errors that affect the inertial sensors can be classified as stochastic and deterministic [5]. Modelling the stochastic component of inertial sensors is a challenging task [6], and is not only a current topic but also a relevant one in many areas where the MEMS based Inertial Measurement Units (IMUs) are used. Actually, in recent years different works related to the stochastic error modelling have been achieved using statistical theory as it is described in [7, 8]. Most of the papers are based on Allan variance (AV) technique which is detailed in [9–11] and it has been applied to inertial sensors in several researches such as [12–18] *et al.* A different approach was shown in [19, 20] which make use of Parallel Cascade Identification (PCI) and Autoregressive (AR) models, respectively. Despite this, the estimation of the error model parameters is still non-trivial. Most of the times those unknown parameters are estimated through tuning (that is often challenging and difficult), by using available sensor specifications (high-grade IMUs) or by experience [21]. For this reason, in this thesis we focused our attention on the identification and modeling of the stochastic error, specifically, the bias-drift error.

Since this noise has both high-frequency noise (short-term) and low-frequency noise (long-term), it is necessary to minimize both of them in order to improve the accuracy of the INS. Wavelet de-noising has being used in similar works because of its great effectiveness removing high-frequency noises, as it is shown in [20, 22–24]. However, it has a limited success in removing the long-term inertial sensors errors [25]. Moreover, Allan variance (AV) is a widely used technique in the modeling of inertial sensors which can take into account the long-term noises. Thereby, we present a mixture of the wavelet de-noising technique and Allan variance with the purpose of evaluating the accuracy enhancement of the inertial sensors when these methods are blended together. Even though there are works where AV and wavelet de-noising are performed (*e.g.*, [24]), we developed a suitable combination between AV and wavelet de-nosing showing that MEMS based IMUs require high levels of decomposition in order to have an enhancement of accuracy in the navigation solution, while the vehicle's dynamics is preserved by using a conservative threshold.

Moreover, low cost inertial sensors have noises with complex spectral structures [6]. This is because several random processes are superimposed, which makes difficult the estimation of parameters that enable the modelling of these random errors. A typical example of this situation is the flicker noise, which can be approximated as the combination of several exponentially correlated noise terms as it is stated in [11,26] and it has been detailed in [6,8,27,28] with first order Gauss-Markov processes. Therefore, in this thesis we evaluate a method for estimating the random error (bias-drift) parameters of the inertial sensors based on a non-linear fitting with constraints that we called (NLF). Since most of the reported works in the literature disregard the stochastic error variations at different temperature points [29], we also evaluate the feasibility of developing a stochastic error model temperature dependent using the NLF. The presented method is able to estimate stochastic error model parameters with complex structures of noises that are usually found in sensors based on MEMS technology.

Although recently similar methods have been developed as the Generalized Method of Wavelet Moments (GMWM) proposed by Stebler *et al.* in [8], we implemented the estimator that has been previously described in [30, 31] that is used to minimize the relative distance between the objective function and the estimate variance. It has shown good results fitting log-log Allan variance curves and it could be considered a particular case of the GMWM. It should be highlighted that different from the existing methods, in the NLF we developed constraints for the noises that are identified during the analysis by means of the 95% confidence interval curve. Those constraints are also set taking into account a prior knowledge of the noise characteristics, which makes easy the convergence of the fitting algorithm and guarantee an appropriate non-linear fitting. Additionally, since the classical least square employed for the fitting can lead to local minimum solutions we carried out an optimization of the parameters estimated by means of pattern search technique.

Eventually, according to the literature we noted that most of the efforts have focused on the development of algorithms to enhance the performance of the GPS/INS fusion and less effort has been directed towards the development of practical implementations suitable for compact platforms [32, 33]. Actually, the trend in the navigation systems is to provide small size platforms, this means, systems on a single board and eventually onto a single chip [34, 35]. Although, recent works have been conducted to implement GPS/INS systems on compact platforms, highlighting between them [35–39]. In most of them the FPGA (Field Programmable Gate Array) is used as interface for data acquisition and in approaches where it is used to compute the navigation algorithms the hardware resources available are underutilized, in fact, this is where the FPGA could offer great benefits [33]. It should also be mentioned that in this work we make use of low cost INS (MEMS grade) which imposes constrains on the processor speed since these sensors need an error model that increases the number of states in the Extended Kalman Filter (EKF) and consequently the computational cost.

In the case of in-car navigation systems there are basically four different sources of information available: various global navigation satellite systems (GNSSs), sensors observing vehicle dynamics, road maps and vehicle models [40]. All these sources can be used to enhance the navigation solution provided by the GPS. Since most of the navigation systems require redundant information, we consider that FPGAs (Field Programmable Gate Array) are suitable for their implementation because they are very flexible devices that allow to easily adapt several instruments such as compasses, GPS receivers, odometers, cameras even multiple Inertial Measurement Units (IMUs), *etc.* In this sense, we aim to develop an embedded system that combines GPS/INS with an architecture based on FPGA technology, where we explore the possibilities that the FPGA can offer in the loosely-coupled GPS/INS integration and we study the feasibility of using hardware resources to accelerate the navigation application taking advantage of its parallel processing with DSP blocks for the computational cost involved in the Extended Kalman Filter (EKF), which might decrease the computing time that is a fundamental factor in real-time applications.

- Objectives
  - Develop the loosely-coupled GPS/INS integrated system in platform based on FPGA for terrestrial applications, studying the feasibility of using hardware resources to accelerate the EKF.
  - Develop stochastic error models based on a constrained non-linear fitting and a mixture of Allan variance/wavelet de-noising to compensate the bias-drift that affects the MEMS inertial sensors in order to enhance the performance of a MEMS based INS/GPS loosely-coupled integration.

1.2 Thesis outline

The thesis is organized in seven chapters as follows:

- Chapter 2 - Inertial Navigation System and GPS/INS Integration: it describes the basic blocks of an Inertial Navigation System (INS). Comprising the reference frames, the computation of attitude, velocity and position from measurements of angular velocity and acceleration. Subsequently, the benefits of the GPS/INS integration are presented including different strategies and then the navigation filter used in this dissertation is detailed.
- Chapter 3 - MEMS IMU Inertial Sensors Errors: it provides information about how inertial sensors work, the different types of sensors, how they are classified, their general performance characteristics and the errors that are involved in inertial sensors based on MEMS technology. It also shows the noise sources

that affect an Inertial Measurement Unit (IMU), giving a description of the deterministic and stochastic errors.

- Chapter 4 - Stochastic Modelling of MEMS Inertial Sensors: it presents current approaches to estimate the parameters of the stochastic error modeling in inertial sensors based on MEMS technology. This chapter also carries out an analysis through different techniques such as Allan variance method, autoregressive models, autocorrelation and power spectral density using a MEMS based IMU. Subsequently, it describes the compensation of the short-term noises and long term noises through combination of wavelet de-noising with AV, and then the constrained non-linear fitting (NLF) is explained.

- Chapter 5 - Architecture Based on FPGA for GPS/INS Integration: it presents a review of recent similar platforms where the GPS/INS integration has been implemented. Subsequently, it explains the architecture developed for the loosely-coupled GPS/INS integration based on FPGA, giving details about the software/hardware implementation and utilization resources. Then, a matrix multiplication in hardware is presented as a possibility to speed-up the navigation algorithm.

- Chapter 6 - Results : it presents the performance of different stochastic error models using AV and PSD, wavelet de-noising/AR models and the proposed improvements based on wavelet de-nosing/AV and NLF, they are adapted to the loosely-coupled integration and assessed using real data collected in a land vehicle.

- Chater 7 - Conclusions : the conclusions of the thesis are drawn and topics for further research work are suggested.

Chapter 2

Inertial Navigation System and GPS/INS Integration

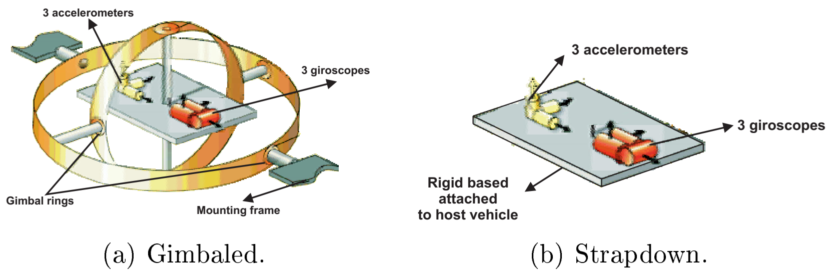
2.1 Introduction

An Inertial Navigation System (INS) calculates the position, velocity and attitude of a vehicle using measurements of acceleration and angular rate obtained from inertial sensors (*i.e.*, accelerometers and gyroscopes); these measurements are processed by navigation equations to get the position, velocity and attitude of the vehicle.

In general there are two ways to build up a INS (see Fig. 2.1). In the first the accelerometers are mounted on an actuated platform (gimbaled system). The gimbal angles are commanded to maintain the platform frame alignment with a specified navigation coordinate system. So the platform does not experience any rotation relative to the navigation frame, in spite of vehicle motion [41].

The second is the strapdown that attaches the inertial sensors directly to the vehicle frame. In this approach, the sensors experience the full dynamic motion of the vehicle, which increases the dynamic range and influences the gyro scale-factor errors and non-linearity. In addition, the relationship among vehicle, navigation, and inertial coordinate frames must be maintained computationally, which increases the on-board computational load [41].

In this dissertation we focused our attention on the strapdown system since it has



(a) Gimbaled. (b) Strapdown.

Figure 2.1: Implementation of Inertial Navigation Systems (adopted from [42]).

been used in the last years because of its cost, small size and low power requirements compared to the gimbaled system that is typically large and more expensive. From here on when we refer to INS it will indicate a strapdown system.

The Inertial Measurement Unit (IMU) is part of the INS and it is the device where the inertial sensors are mounted, so it provides measurements along three mutually orthogonal directions with respect to the body frame. In order to have a reference frame for navigation, these measures must be converted from the body frame by means of a rotation matrix to a reference frame that is usually either a local level frame (n-frame) or an Earth fixed frame (e-frame) [5]. In this chapter we begin with a description of the reference frames that are usually used when the INS is aided with a GPS. Subsequently, we present the navigation equations which are widely used to obtain the inertial navigation solution from the data provided by the IMU. Finally, we describe the benefits of the GPS/INS integration including different fusion strategies and then we explain the navigation filter used in this thesis.

2.2 Reference Frames

For navigation on Earth the inertial measurements provided by the IMU must be transformed to a reference frame where they can be related to the cardinal directions of the Earth. This also allows to have a reference frame to integrate the inertial navigation output with the output of instruments such as the GPS. Therefore this section describes different frames which are important to obtain the navigation solution of a land vehicle. The discussion of each follows from [41] and [43].

The notation used throughout this chapter is summarized as follows:

1. $\mathbf{x}^p$ denotes a vector $\mathbf{x}$ expressed in a coordinate p-frame (*i.e.*, n-frame, e-frame, b-frame, *etc.*);

2. $\mathbf{C}_{from}^{to}$ denotes a coordinate transformation matrix from one coordinate frame (designated by "from") to another coordinate frame (designated by "to") (*e.g.*, $\mathbf{C}_b^n$ denotes the coordinate transformation matrix from b-frame to n-frame);

3. the rotation rate vector between two frames expressed in a specific frame can be represented either by a vector $\boldsymbol{\omega}$ or by the corresponding skew-symmetric matrix $\boldsymbol{\Omega}$ (with $\boldsymbol{\Omega}^T = -\boldsymbol{\Omega}$).

For example

$$\boldsymbol{\omega}_{pq}^r = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^T \tag{2.1}$$

or

$$\boldsymbol{\omega}_{pq}^r \times = \boldsymbol{\Omega}_{pq}^r = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{2.2}$$

describe the rotation rate vector of the q-frame with respect to the p-frame projected to the r-frame.

2.2.1 Inertial Frame (i-frame)

Earth-Centered Inertial (ECI) coordinates has its origin at the center of the Earth and the axes are a non-rotating and non-accelerating with respect to fixed stars. The three axes of the ECI are defined as follows (Fig. 2.2):

X^i pointing to the vernal equinox;

Z^i parallel to the rotation axis (North polar axis) of the Earth;

Y^i completes the right-handed orthogonal coordinate system.

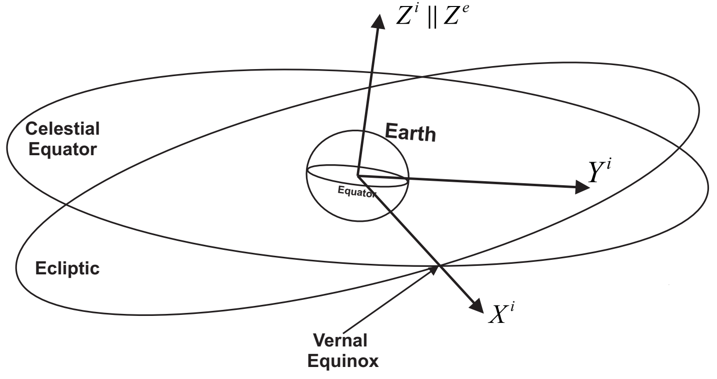


Figure 2.2: The inertial frame.

2.2.2 Earth Frame (e-frame)

The Earth-Centered Earth-Fixed (ECEF) coordinates have the origin fixed to the center of the Earth and the axes are fixed with respect to the Earth. So they rotate relative to the inertial frame with a frequency of

$$\omega_{ie} = 7.292115 * 10^{-5}\ rad/sec \tag{2.3}$$

The three axis are defined as follows (Fig. 2.3):

X^e lies along the interception of the plane of the Greenwich meridian with the Earth's equatorial plane;

Z^e parallel to the Earth's rotation axis;

Y^e is 90° East of Greenwich meridian in the equatorial plane to make a right-handed orthogonal coordinate system.

2.2.3 Navigation Frame (n-frame)

The navigation frame (n-frame) is also known as the local level frame (l-frame). It has its origin at the location of the navigation system. The n-frame can also refer to as the North-East-Down (NED) system. The advantage of using this frame is that the axes coincide with the IMU axis when it is heading North on a levelled road. In addition, the n-frame is based on the perpendicular to the reference ellipsoid, so the geodetic coordinate differences ($i.e.$, geodetic latitude, longitude and height) are the output of the system [5].

The three axis are defined as follows (Fig. 2.3):

X^n points to the true North (N) (*i.e.*, towards the direction where the latitude increased);

Y^n points East (E) (*i.e.*, towards the direction where the longitude increased);

Z^n points downwards (D) perpendicular to the reference ellipsoid to make a right-handed orthogonal coordinate system.

The NED system and the ECEF system are shown in Fig.2.3, where φ denotes the geodetic latitude and λ the geodetic longitude.

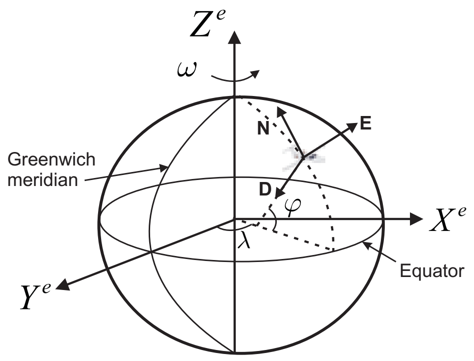


Figure 2.3: The ECEF and NED coordinate systems.

2.2.4 Body Frame (b-frame)

It is an orthogonal frame that is assumed to be aligned with the vehicle. The axis of this frame coincide with the axis of the IMU and the orientation of the axes is defined as follows (Fig.2.4):

X^b points toward the front of the vehicle;

Y^b points toward the right of the vehicle;

Z^b to make a right-handed orthogonal coordinate system.

These axes directions are not unique, but are typical in aircraft and land vehicle applications.

This frame can coincide with the navigation frame (NED) as it was described previously and the variables ϕ, θ and ψ correspond to the Euler angles (*Roll*, *Pitch* and

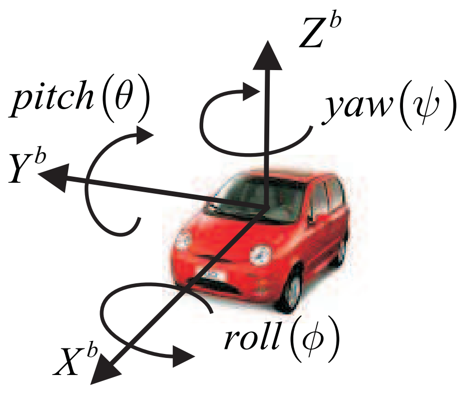


Figure 2.4: The body frame.

Yaw). A common convention for vehicle attitude Euler angles having the body frame axis aligned with NED coordinates (*i.e.*, X^b axis pointing to North and on a levelled road) can be described as [44]:

1. *Yaw/Heading*: the azimuth is considered positive if calculated clockwise from the north direction.
2. *Pitch*: the pitch angle is positive upward or nose up direction from local horizontal plane (elevation).
3. *Roll*: the roll angle takes positive values if the moving body is rolling towards right.

Since the vehicle attitude can be specified in terms of Euler angles in the body frame (Fig. 2.4) and the latitude, longitude can be related to the n-frame and e-frame (Fig. 2.3), the transformations between these coordinate frames can be achieved by considering these terms and using rotation matrices as it is presented below.

2.3 Coordinate Transformations

Before describing the navigation equations it is important to define the relationship between the different frames, which will permit to transform the inertial measurements provided by the IMU in the body frame to navigation frame (*i.e.*, assuming that the IMU is aligned with the body frame), thus the navigation solution can be expressed in a reference frame (navigation frame) and later it can be combined with GPS data.

2.3.1 Rotation Matrices

Although there are several techniques to perform the transformations between the frames; probably the most common is through the direction cosine matrix (DCM). Here we will present the DCM rotation matrices between some frames. For further details about the derivation of these matrices refer to [41,43,45,46].

The DCM from the e-frame to the n-frame is [41]:

$$\mathbf{C}_e^n = \begin{bmatrix} -\sin\varphi\cos\lambda & -\sin\varphi\sin\lambda & \cos\varphi \\ -\sin\lambda & \cos\lambda & 0 \\ -\cos\varphi\cos\lambda & -\cos\varphi\sin\lambda & -\sin\varphi \end{bmatrix} \tag{2.4}$$

The matrix rotation from the b-frame to the n-frame is based on the Euler angles; it requires a procedure following a order of rotations to obtain the DCM matrix $\mathbf{C}_b^n$ [5,44]:

$$\mathbf{C}_b^n = \begin{bmatrix} c\theta c\psi & -c\phi s\psi + s\phi s\theta c\psi & s\phi s\psi + c\phi s\theta c\psi \\ c\theta s\psi & c\phi c\psi + s\phi s\theta s\psi & -s\phi c\psi + c\phi s\theta s\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix} \tag{2.5}$$

where "sin" and "cos" are shortly denoted as "s" and "c", respectively.

The Euler angles (*i.e.*, ϕ, θ and ψ) may be derived from Eq. (A.5) as described below [41,43]:

$$\phi = \arctan 2(\mathbf{C}_b^n[3{,}2],\mathbf{C}_b^n[3{,}3]) \tag{2.6}$$

$$\theta = -\arcsin(\mathbf{C}_b^n[3{,}1]) \tag{2.7}$$

$$\psi = \arctan 2(\mathbf{C}_b^n[2{,}1],\mathbf{C}_b^n[1{,}1]) \tag{2.8}$$

where $arctan2(y,x)$ is a four quadrant inverse tangent function.

2.3.2 Angular Velocity Vectors

For the location of the vehicle on the surface of the Earth in a navigation frame (*e.g.*, the n-frame), we must consider that the movement of the vehicle involves a change in the position of the frame, which in turn causes a rotation of this frame with respect to other frames. Therefore, it is necessary to take into account the rotation between different frames involved in the navigation solution. These rates can be given in terms of Earth's rotation and changes of latitud and longitud. So according to the notation described in Section 2.2, the rotation vector of the e-frame with respect to the i-frame projected on the e-frame can be expressed as [47]:

$$\boldsymbol{\omega}_{ie}^{e} = \begin{bmatrix} 0 & 0 & \omega_e \end{bmatrix}^T \tag{2.9}$$

where ω_e is the magnitude of the Earth rotation rate $\approx 7.292115 * 10^{-5}$ rad/sec. This is due to the fact that the z-axis of the e-frame and i-frame are parallel to each other (Fig. 2.2) but the e-frame is rotating at an angular velocity with respect to the i-frame [5]. If this vector is projected to the n-frame we obtain:

$$\boldsymbol{\omega}_{ie}^{n} = \mathbf{C}_e^n \boldsymbol{\omega}_{ie}^{e} = \begin{bmatrix} \omega_e \cos\varphi & 0 & -\omega_e \sin\varphi \end{bmatrix}^T \tag{2.10}$$

The turn rate of the n-frame with respect to the e-frame as measured in the n-frame is called transport rate, and is expressed in terms of the rate of change of latitude and longitude [43, 47]:

$$\boldsymbol{\omega}_{en}^{n} = \begin{bmatrix} \dot\lambda \cos\varphi & -\dot\varphi & -\dot\lambda \sin\varphi \end{bmatrix}^T \tag{2.11}$$

where the rate of change of the positions components ($i.e.$, $\dot\lambda$ and $\dot\varphi$) can be expressed in terms of the velocity components of the n-frame by:

$$\dot\varphi = \frac{v_N}{M+h} \tag{2.12}$$

$$\dot\lambda = \frac{v_E}{(N+h)\cos\varphi} \tag{2.13}$$

where M and N are the radius of curvature in a meridian at a given latitude and the transverse radius of curvature, respectively. These terms are derived by modelling the Earth by a ellipsoid reference and can be expressed as [43]:

$$M = \frac{a(1-e^2)}{\left(1-e^2\sin^2\varphi\right)^{3/2}} \tag{2.14}$$

$$N = \frac{a}{\left(1-e^2\sin^2\varphi\right)^{1/2}} \tag{2.15}$$

where a = 6378137.0 m and e = 0.0818 are the semi-major axis length and the eccentricity of the WGS-84 ellipsoid [48], respectively.

The transport rate (Eq. (2.11)) takes the following form that is function of position and velocity in the navigation frame [41]:

$$\boldsymbol{\omega}_{en}^{n} = \begin{bmatrix} v_E/(N+h) \\ -v_N/(M+h) \\ -v_E\tan\varphi/(N+h) \end{bmatrix} \tag{2.16}$$

where h is the geodetic height, and v_N,v_E are velocities in the North and East direction, respectively.

Next section we will present the navigation equations involved in the strapdown INS, which can describe the motion of a vehicle taking as input the inertial measurements in the body frame (accelerations and angular rotations) and converting these measurements into the navigation frame.

2.4 Navigation and Mechanization Equations

The strapdown INS involves navigation equations (or kinematic equations) which are the numerical tools to implement the physical phenomenal that relates the inertial sensor measurements to the navigation state (*i.e.*, position, velocity and attitude) [5]. These INS kinematic equations describe mathematically the motion of the vehicle by taking inertial measurements as input, usually they come from an Inertial Measurement Units (IMU) that consists of three accelerometers and three gyroscopes. Gyros provide the angular velocity $\boldsymbol{\omega}_{ib}^{b}$, which represents the rotation of the b-frame with respect to the i-frame, measured in the b-frame. Accelerometers sense the specific force $\mathbf{f}^b$ in the

b-frame. These measures along the three mutually orthogonal directions need to be converted into a reference frame for navigation that is usually either the n-frame or the e-frame. Although in this thesis the INS kinematic equations for both reference frames were implemented, in this section we only focus our attention on the INS equations expressed in the n-frame (NED). At the end of this section we also describe briefly the mechanization equations that are the computer implementation of the motion equations.

2.4.1 Navigation Equations

The navigation equations of the vehicle to be positioned in the n-frame can be expressed in a compact form as follows [43, 47]:

$$\begin{bmatrix} \dot{\mathbf{r}}^n \\ \dot{\mathbf{v}}^n \\ \dot{\mathbf{C}}_b^n \end{bmatrix} = \begin{bmatrix} \mathbf{D}^{-1}\mathbf{v}^n \\ \mathbf{C}_b^n\mathbf{f}^b - (2\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n) \times \mathbf{v}^n + \boldsymbol{\gamma}^n \\ \mathbf{C}_b^n \left(\boldsymbol{\Omega}_{ib}^b - \boldsymbol{\Omega}_{in}^b\right) \end{bmatrix} \tag{2.17}$$

The position in the n-frame is expressed in geodetic (curvilinear) coordinates:

$$\mathbf{r}^n = \begin{bmatrix} \varphi & \lambda & h \end{bmatrix}^T \tag{2.18}$$

The time rate of change of these position components is associated to the velocity components in the n-frame such as:

$$\dot{\mathbf{r}}^n = \begin{bmatrix} \dot{\varphi} \\ \dot{\lambda} \\ \dot{h} \end{bmatrix} = \begin{bmatrix} \frac{1}{M+h} & 0 & 0 \\ 0 & \frac{1}{(N+h)cos(\varphi)} & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} v_N \\ v_E \\ v_D \end{bmatrix} = \mathbf{D}^{-1}\mathbf{v}^n \tag{2.19}$$

where the velocity in the NED frame have the following components:

$$\mathbf{v}^n = \begin{bmatrix} v_N & v_E & v_D \end{bmatrix}^T \tag{2.20}$$

and the diagonal matrix $\mathbf{D}^{-1}$ is defined as follows:

$$\mathbf{D}^{-1} = \begin{bmatrix} \frac{1}{M+h} & 0 & 0 \\ 0 & \frac{1}{(N+h)\cos\varphi} & 0 \\ 0 & 0 & -1 \end{bmatrix} \tag{2.21}$$

Since the kinematic equations are implemented in the NED frame, the rotation matrix $\mathbf{C}_b^n$ that transforms the acceleration measures from the b-frame to the n-frame is considered. So the acceleration components in the three axis of the body frame are:

$$\mathbf{f}^b = \begin{bmatrix} f_x & f_y & f_z \end{bmatrix}^T \tag{2.22}$$

and taking into account Eq. (2.17) the velocity of the vehicle in the n-frame can be to obtained by:

$$\dot{\mathbf{v}}^n = \mathbf{C}_b^n\mathbf{f}^b - (2\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n) \times \mathbf{v}^n + \boldsymbol{\gamma}^n \tag{2.23}$$

the acceleration components $\mathbf{f}^b$ in the b-frame need to be converted to the n-frame by means of $\mathbf{C}_b^n$. In addition, three acceleration compensation are required, the first is the rotation rate of the earth given by Eq. (2.10), the second is the change of rotation of the n-frame with respect to the e-frame expressed by Eq. (2.11), and the third one is the normal gravity vector given by [5]:

$$\boldsymbol{\gamma}^n = \begin{bmatrix} 0 & 0 & \gamma \end{bmatrix}^T \tag{2.24}$$

where γ is function of the latitude and ellipsoidal height and is a dominant factor in the n-frame velocity [49]. So the local gravity formula when the WGS-84 parameters are used is expressed as follows [47]:

$$\gamma = a_1\left(1 + a_2 sin^2\varphi + a_3 sin^4\varphi\right) + \left(a_4 + a_5 sin^2\varphi\right)h + a_6 h^2 \tag{2.25}$$

where

$$\begin{aligned} a_1 &= 9.7803267715\ m/sec^2; & a_4 &= -3.087691089 * 10^{-6}\ 1/sec^2; \\ a_2 &= 5.2790414 * 10^{-3}; & a_5 &= 4.397731 * 10^{-9}\ 1/sec^2; \\ a_3 &= 23.2718 * 10^{-6}; & a_6 &= 0.721 * 10^{-12}\ 1/(m\ sec^2); \end{aligned}$$

The third equation considering Eq. (2.17) is for the attitude angles of the moving platform, which are determined by:

$$\dot{\mathbf{C}}_b^n = \mathbf{C}_b^n \left(\boldsymbol{\Omega}_{ib}^b - \boldsymbol{\Omega}_{in}^b \right) \tag{2.26}$$

where $\boldsymbol{\Omega}_{ib}^b$ and $\boldsymbol{\Omega}_{in}^b$ correspond to the skew-symmetric matrices of $\boldsymbol{\omega}_{ib}^b$ and $\boldsymbol{\omega}_{in}^b$, respectively. The term $\boldsymbol{\omega}_{ib}^b$ is provided by the gyroscopes of the inertial system and is expressed by:

$$\boldsymbol{\omega}_{ib}^b = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^T \tag{2.27}$$

Since the gyroscopes measure the Earth's rotation rate, the change of orientation of the n-frame, and the angular velocities of the moving body. The angular velocities in $\boldsymbol{\Omega}_{in}^b$ are subtracted from $\boldsymbol{\Omega}_{ib}^b$ to remove the first two effects [5]. Next section describes the way the navigation equations are solved to get the parameters of navigation.

2.4.2 Mechanization Equations

The mechanization equations are the computer implementation that uses initial conditions (*i.e.*, position, velocity and attitude), specific force and angular velocity measurements to solve the navigation equations. The procedure to obtain the position, velocity and attitude of the vehicle with the mechanization equations can be summarized in the following steps [50]:

1. Attitude update
2. Velocity and position update.

Before describing the mechanization steps two aspect are worth mentioning : On the one hand, since the IMU provides angular velocity and acceleration measurements each Δt, which is equal to $t_{k+1} - t_k$, the update is performed at the time t_{k+1} using information from the previous epoch t_k (*i.e.*, INS mechanization forward). This is due to the fact that the initial navigation states are known, and they are determined through a preliminary stage called the alignment.

On the other hand, given that the inertial sensors are characterized by high noise and large uncertainties in their outputs: such as bias, scale factor and non-orthogonalities [51], these errors that influence the inertial measurements need to be compensated as much as possible in order to have a minimum error in the mechanization input and as a consequence a more accurate inertial navigation solution. The error sources that affect the inertial sensors will be discussed in Chapter 3.

Attitude Update

Since the attitude is implicit in the rotation matrix $\mathbf{C}_b^n$, the update of this matrix can be performed by different approaches *i.e.*, quaternions, direction-cosine and Euler angles. In this case, we describe the quaternions since they are more computationally efficient than the other two methods [41]. For further details about these algorithms refer to [41, 43, 52].

In order to solve Eq. (2.26), it necessary to determine the body rate with respect to the navigation frame $\boldsymbol{\omega}_{nb}^b$ (Eq. (2.16)). This is derived from the difference between the measure body rates, $\boldsymbol{\omega}_{ib}^b$ (Eq. (2.27)), and the estimates of the components of the navigation frame rate $\boldsymbol{\omega}_{in}^b$ Eq. (2.28) [43].

$$\boldsymbol{\omega}_{nb}^b = \boldsymbol{\omega}_{ib}^b - \boldsymbol{\omega}_{in}^b = \boldsymbol{\omega}_{ib}^b - \mathbf{C}_n^b \boldsymbol{\omega}_{in}^n \tag{2.28}$$

where $\mathbf{C}_n^b = (\mathbf{C}_b^n)^T$ and the term $\boldsymbol{\omega}_{in}^n$ is the rotation vector of the n-frame with respect to the i-frame and can be obtained by adding Eq. (2.16) and Eq. (2.10).

$$\boldsymbol{\omega}_{in}^n = \boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n = \begin{bmatrix} \omega_e \cos\varphi + v_E/(N+h) \\ -v_N/(M+h) \\ -\omega_e \sin\varphi - v_E \tan\varphi/(N+h) \end{bmatrix} \tag{2.29}$$

Thus Eq. (2.28) can be written as the total angular increment of the vehicle by

$$\Delta\boldsymbol{\theta}_{nb}^b = \boldsymbol{\omega}_{ib}^b \Delta t - \mathbf{C}_n^b(\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n)\Delta t = \Delta\boldsymbol{\theta}_{ib}^b - \mathbf{C}_n^b \boldsymbol{\omega}_{in}^n \Delta t \tag{2.30}$$

where Δt is the sampling time of the Inertial Measurement Unit (IMU), and the angular increments $\Delta\boldsymbol{\theta}_{nb}^b$ can be expressed as a vector:

$$\boldsymbol{\mathit{\Delta\theta}}_{nb}^{b} = \begin{bmatrix} \mathit{\Delta}\theta_x & \mathit{\Delta}\theta_y & \mathit{\Delta}\theta_z \end{bmatrix}^T \tag{2.31}$$

After computing the angular increments Eq. (2.30), the components of this vector are used to update the quaternions as [53]:

$$\mathbf{q}_{k+1} = \mathbf{q}_k + 0.5 \begin{bmatrix} c & s\mathit{\Delta}\theta_z & -s\mathit{\Delta}\theta_y & s\mathit{\Delta}\theta_x \\ -s\mathit{\Delta}\theta_z & c & s\mathit{\Delta}\theta_x & s\mathit{\Delta}\theta_y \\ s\mathit{\Delta}\theta_y & -s\mathit{\Delta}\theta_x & c & s\mathit{\Delta}\theta_z \\ -s\mathit{\Delta}\theta_x & -s\mathit{\Delta}\theta_y & -s\mathit{\Delta}\theta_z & c \end{bmatrix} \mathbf{q}_k \tag{2.32}$$

where $s = \frac{2}{\theta} sin\left(\frac{\theta}{2}\right)$, $c = 2cos\left(\frac{\theta}{2} - 1\right)$, $\theta = \sqrt{(\mathit{\Delta}\theta_x)^2 + (\mathit{\Delta}\theta_y)^2 + (\mathit{\Delta}\theta_z)^2}$ and $\mathbf{q}_k$ is the quaternion that is a four-parameter vector defined as:

$$\mathbf{q}_k = \begin{bmatrix} q_{1,k} & q_{2,k} & q_{3,k} & q_{4,k} \end{bmatrix}^T \tag{2.33}$$

Having updated the quaternions the rotation matrix, $\mathbf{C}_b^n$ can be obtained by doing the transformation from quaternions to DCM.

$$\mathbf{C}_b^n = \begin{bmatrix} \left(q_1^2 - q_2^2 - q_3^2 + q_4^2\right) & 2\left(q_1q_2 - q_3q_4\right) & 2\left(q_1q_3 + q_2q_4\right) \\ 2\left(q_1q_2 + q_3q_4\right) & \left(q_2^2 - q_1^2 - q_3^2 + q_4^2\right) & 2\left(q_2q_3 - q_1q_4\right) \\ 2\left(q_1q_3 - q_2q_4\right) & 2\left(q_2q_3 + q_1q_4\right) & \left(q_3^2 - q_1^2 - q_2^2 + q_4^2\right) \end{bmatrix} \tag{2.34}$$

Thus the Euler angles associated with the attitude of the vehicle are calculated as states in Eqs. (2.7), (2.8) and (2.8).

Velocity and Position Update

To obtain the velocity in the navigation frame (NED frame), Eq. (2.23) is expressed as it follows:

$$\mathit{\Delta}\mathbf{v}^n = \mathit{\Delta}\mathbf{v}_f^n - (2\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n) \times \mathbf{v}^n \mathit{\Delta}t + \boldsymbol{\gamma}^n \mathit{\Delta}t \tag{2.35}$$

where $\varDelta\mathbf{v}^n$ is the velocity increment in the n-frame and $\varDelta\mathbf{v}_f^n$ is the specific force increments projected in the navigation frame that is given by:

$$\varDelta\mathbf{v}_f^n = \mathbf{C}_b^n \begin{bmatrix} 1 & 0.5\theta_z & -0.5\theta_z \\ -0.5\theta_z & 1 & 0.5\theta_x \\ 0.5\theta_y & -0.5\theta_x & 1 \end{bmatrix} \varDelta\mathbf{v}_f^b \tag{2.36}$$

where the matrix expressed in terms of θ_x,θ_y and θ_z is the first-order sculling correction [54] and $\varDelta\mathbf{v}_f^b$ is the specific force increments in the body frame. The latter is calculated by considering the three axis acceleration measured by the IMU:

$$\varDelta\mathbf{v}_f^b = \mathbf{f}^b * \varDelta t = \begin{bmatrix} f_x & f_y & f_z \end{bmatrix}^T * \varDelta t \tag{2.37}$$

After computing the velocity increment $\varDelta\mathbf{v}_{k+1}^n$ with Eq. (2.35), the current velocity in the n-frame can be obtained through the previous velocity $\mathbf{v}_k^n$ as:

$$\mathbf{v}_{k+1}^n = \mathbf{v}_k^n + \varDelta\mathbf{v}_{k+1}^n \tag{2.38}$$

Finally, the position of the moving platform in the navigation frame is computed by integrating Eq. (2.38) as shown in Eq. (2.39).

$$\mathbf{r}_{k+1}^n = \mathbf{r}_k^n + \frac{1}{2}\mathbf{D}^{-1}\left(\mathbf{v}_k^n + \mathbf{v}_{k+1}^n\right) \cdot \varDelta t \tag{2.39}$$

where the term $\mathbf{D}^{-1}$ is given by Eq. (2.21) and $\mathbf{r}_k^n$ corresponds to the position obtained in the previous epoch.

Figure 2.5 summarizes the mechanization equations in the n-frame. As it was mentioned the input is provided by the IMU's measurements that is adapted in the vehicle and the output supplied information of position, velocity and attitude of the vehicle in the navigation frame (NED frame).

The flow diagram of the INS mechanization algorithm is depicted in Fig. 2.6, where the sequence of terms that are calculated is shown. The algorithm takes as inputs $\mathbf{v}_f^b$ and $\boldsymbol{\omega}_{ib}^b$ and the outputs are the attitude obtained from $\mathbf{C}_b^n$, the position $(\mathbf{r}_{k+1}^n)$ and the velocity $\mathbf{v}_{k+1}^n$ in the navigation frame. It is noteworthy that measures of angular

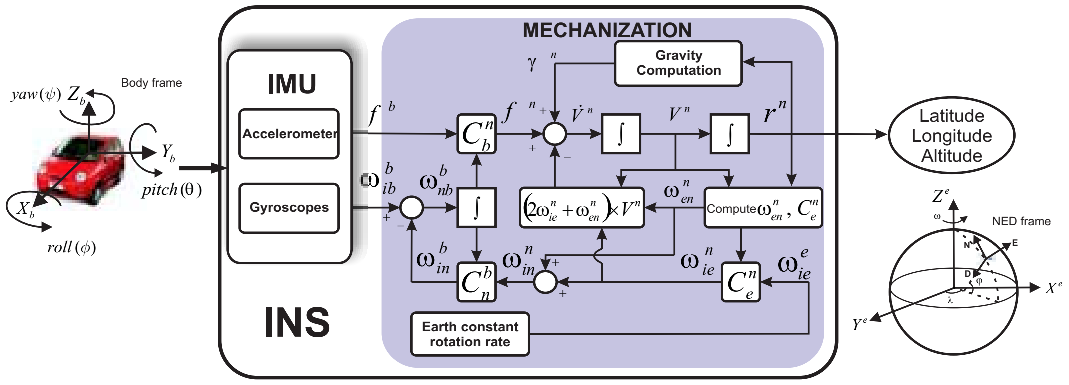


Figure 2.5: INS mechanization in the n-frame (adopted from [55]).

velocity and specific force are corrected in an earlier stage in order to minimize the error in the INS. For a better performance of the INS, it is combined with the GPS, which is described in the below section.

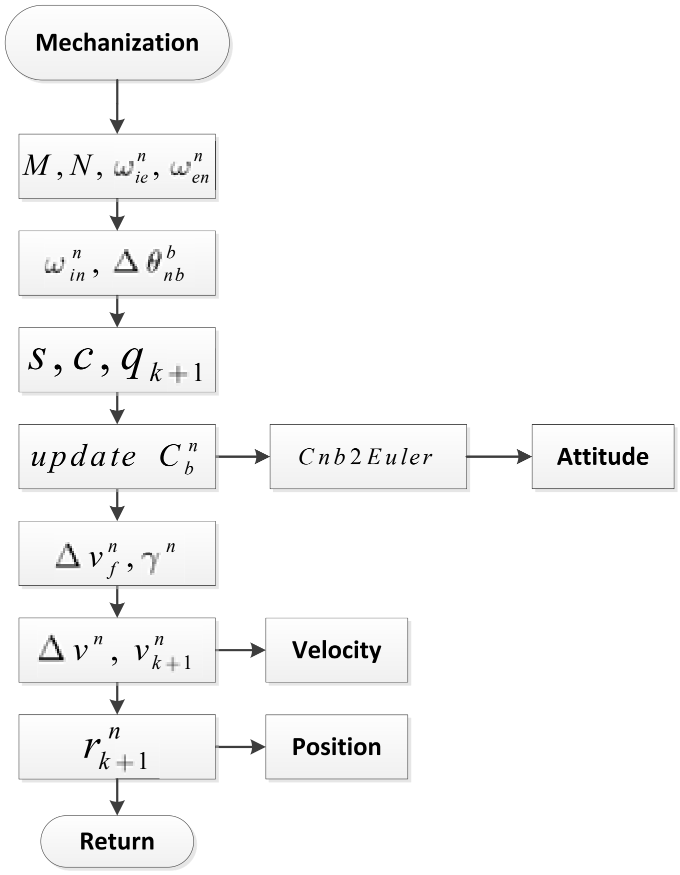


Figure 2.6: Flow diagram for the mechanization.

2.5 GPS/INS Integration

In this section we discuss the GPS/INS integration, the benefits of this integration and two strategies of integration, specifically, we provide details of the tightly coupled and the loosely coupled that is the one used in this work.

2.5.1 Benefits of GPS/INS Integration

The INS measurements exhibits a noise relatively low from second to second, but it tends to drift over time due to the inherent error of the inertial sensors. Since this error is integrated in the mechanization process, it is reflected in an unbounded position and velocity solution. Despite this, the INS output is computed using the data provided by the inertial sensors, which makes it immune to external interferences. Additionally, the INS provides a navigation solution with a higher rate than GPS (typically 100 Hz), which is limited by the choice of the computational approach and equipment [41].

On the other hand, the GPS errors are relatively noisy from second to second but in contrast with the INS the biases are bounded, so it does not exhibit long-term drift [56]. Thus, the GPS provides position and velocity estimation with bounded estimation errors [41]. Nonetheless, this information has a low rate (typically between 1 Hz or 10 Hz) and is susceptible to jamming, blockage, interference, *etc.*

In the last decades the fusion between these two systems has been implemented in many navigation applications because it provides better performance than their stand alone operation, which is a consequence of its complementary nature.

Basically, GPS and inertial measurements are complementary for two reasons: the characteristics of their errors are different and these are measurements of different quantities [56]. Besides the redundancy that the two systems can provide, it seeks to take advantage of the synergy as it describes below [56, 57]:

1. The inertial navigation system provides navigation information when the GPS signal is not available.

2. GPS measurements can be used to correct the INS estimates by an integrated navigation filter that combines inertial system and GPS measurements.

3. The GPS/INS integration exceeds the accuracy of the GPS alone. This is more apparent in scenarios where the GPS is affected by multipath.

4. The adaptation of the INS provides a navigation solution at rates much higher than the GPS receivers.

Table 2.1 summarizes the features and shortcomings of inertial and GPS navigation systems.

Table 2.1: Inertial and GPS advantages and disadvantages (adopted from [56]).

	Advantages	**Disadvantages**
GPS	• Errors are bounded • Low cost	• Low data rate (typically $1-10\ Hz$) • No attitude information • Susceptible to jamming and signal blockage
INS	• Low cost MEMS inertial sensors • High data rate (typically $\geqslant 100\ Hz$) • Attitude information • Self-contained (not susceptible to jamming)	• Unbounded errors • High cost for the quality • Requires initial reference (position, velocity and attitude)

2.5.2 GPS/INS Integration Approaches

It is common to blend GPS and INS using different integration approaches (*i.e.*, loosely-coupled, tightly-coupled or ultra-tightly coupled; see [58–60]). Here we described the most common strategies that are the loosely-coupled (LC) and the tighly coupled approaches.

Loosely Coupled INS/GPS Integration

In this strategy the position and velocity obtained from the mechanization ($\mathbf{r}^n_{INS}$,$\mathbf{v}^n_{INS}$) (see Section 2.4.2) are combined with the GPS output, which delivers velocity and position data ($\mathbf{r}^n_{GPS}$,$\mathbf{v}^n_{GPS}$). There are two ways to implement the LC strategy: feed-forward and feed-back. The first one is used in systems that have a high-performance inertial measurement unit (IMU), as it merges the GPS/INS information, but it has no control over the error that may occur in the IMU; it basically works with an open-loop architecture. On the other hand, the feed-back includes a close-loop that allows us to correct the INS error, and in the case of a GPS outage, the navigation solution will depend only on the INS, which will be corrected by its correspondent inertial sensor error model. The block diagram of the GPS/INS integration with feedback is shown in Fig. 2.7.

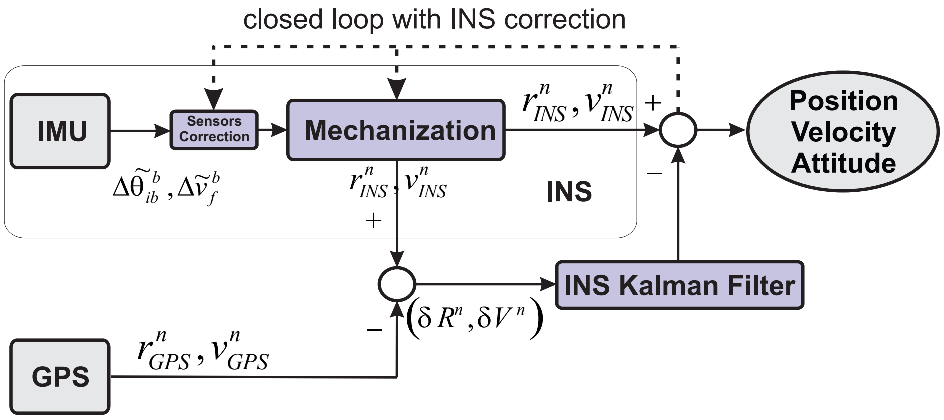

closed loop with INS correction
IMU
Sensors Correction
Mechanization
r^n_{INS}, v^n_{INS}
+
−
Position Velocity Attitude
$\Delta\tilde{\theta}^b_{ib}, \Delta\tilde{v}^b_f$
r^n_{INS}, v^n_{INS}
+
INS
INS Kalman Filter
$(\delta R^n, \delta V^n)$
−
GPS
r^n_{GPS}, v^n_{GPS}

Figure 2.7: Loosely-coupled Kalman Filter (KF) integration with feedback [53].

According to Fig. 2.7 the residual error ($\delta\mathbf{R}^n$,$\delta\mathbf{V}^n$) calculated from the GPS and INS outputs is the input to the Kalman Filter (KF), where a state-space model is built with error states for navigation and IMU errors. The error states related to the IMU errors are fed back though the closed loop in order to correct the INS navigation solution.

The system model for loosely-coupled approach is given by position error, velocity error and attitude error, which represent the navigation error states, *i.e.*, a total of nine states for 3D navigation. Moreover, the scale factors and bias for gyro and accelerometers are included in the IMU error states, and the number of states will depend on the stochastic model employed.

Tightly Coupled INS/GPS Integration

This strategy uses information of pseudoranges (ρ_{INS}) and Doppler ($\dot{\rho}_{INS}$) obtained from the INS output with raw GPS data (*i.e.*, pseudoranges (ρ_{GPS}) and Doppler ($\dot{\rho}_{GPS}$) determined by using satellite ephemeris data). The residuals ($\partial\rho$,$\partial\dot{\rho}$) are the input to the KF, which computes the INS errors estimates. Then, these error states are used to compensate the INS navigation solution and at the same time the IMU errors. The scheme for the tightly coupled GPS/INS integration is shown Fig. 2.8.

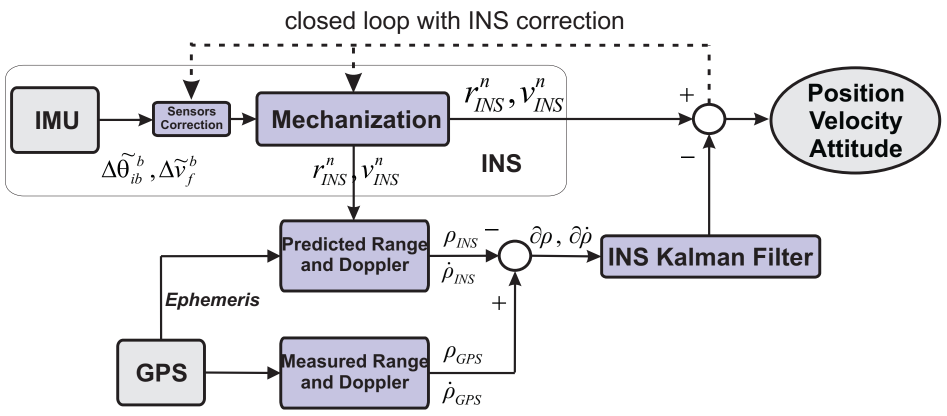

closed loop with INS correction
IMU
Sensors Correction
Mechanization
r^n_{INS}, v^n_{INS}
+
−
Position Velocity Attitude
$\Delta\tilde{\theta}^b_{ib}, \Delta\tilde{v}^b_f$
r^n_{INS} v^n_{INS}
INS
Predicted Range and Doppler
ρ_{INS}
$\dot{\rho}_{INS}$
$\partial\rho, \partial\dot{\rho}$
INS Kalman Filter
Ephemeris
GPS
Measured Range and Doppler
ρ_{GPS}
$\dot{\rho}_{GPS}$

Figure 2.8: Tightly-coupled Kalman Filter (KF) integration [53].

If we compare these two strategies the LC approach requires at least four satellites to provide an acceptable position and velocity, while the tightly coupled with less than four satellites provides a navigation solution. Furthermore, satellites with poor GPS measurements can be discarded from the navigation solution in the tightly coupled. Despite this, LC integration is appropriate for hardware implementation because of its simplicity compared to the tightly coupled, additionally, both of them can be used to study the IMU bias-drift. Therefore, in this thesis we confine our attention in the loosely-coupled (LC) approach, because this strategy can be used to evaluate the behavior of the inertial sensor stochastic model that is one of main goals of this thesis. In addition, this strategy has a lower computational load compared to the tightly coupled, which is more suitable for an FPGA implementation. The loosely coupled integration scheme is performed using the EKF, which is explained in detail in the next section.

2.6 Kalman Filter

The Kalman filter is an optimal estimation tool that provides a sequential recursive algorithm for the estimation of a system states when the system's model is linear [61]. This filtering technique has been used in a wide range of applications in the navigation field from military to civilian since it has an effective and versatile procedure for combining noise sensor outputs to estimate the state of the system with uncertain dynamics [44]. Additionally, it provides an efficient computational method to estimate the state of a linear stochastic process by minimizing the mean and the squared errors and although this is a good quality that makes the KF suitable for real time implementations, the most desirable feature is its robustness [5]. The latter is due to the fact that the KF propagates the uncertainty of the states and measurements through a gain equation that is known as Kalman gain, which will be described latter.

In the derivation of the filter there are two mathematical models involved: the dynamic model that contains the time propagation information for the states and the measurement model that relates the measurements to the states [5].

2.6.1 Dynamic Model

The dynamic model of an INS error model can be represented in a continuous-time form by the following first-order differential equation [51]:

$$\delta\dot{\mathbf{x}}(t) = \mathbf{F}(t)\delta\mathbf{x}(t) + \mathbf{G}(t)\mathbf{w}(t) \tag{2.40}$$

where

- $\delta\mathbf{x}(t) \rightarrow (n \times 1)$ is the vector with the state of the system;
- $\mathbf{F}(t) \rightarrow (n \times n)$ is the system dynamics matrix;
- $\mathbf{G}(t) \rightarrow (n \times p)$ is a system input matrix; and
- $\mathbf{w}(t) \rightarrow (p \times 1)$ is system noise vector, the noise is characterized by zero-mean and normally distributed with spectral density given by $\mathbf{Q}(t)$.

Since the measurements from the inertial sensors and GPS take place in discrete steps, for implementation purposes it is convenient to express Eq. (2.40) in discrete time. Thus, Eq. (2.40) can be expressed such as

$$\delta\mathbf{x}_{k+1} = \boldsymbol{\Phi}_k \delta\mathbf{x}_x + \mathbf{w}_k \tag{2.41}$$

where

- $\delta\mathbf{x}_{k+1}$,$\delta\mathbf{x}_k$ are the system state vectors at time t_{k+1} and t_k, respectively;
- $\boldsymbol{\Phi}_k$ is a transition matrix relating $\delta\mathbf{x}_k$ to $\delta\mathbf{x}_{k+1}$ in the absence of a forcing function; and
- $\mathbf{w}_k$ is the driven response at t_{k+1} due to the white sequence input of the forcing function during the (t_k,t_{k+1}) interval.

For an integrated navigation system the states $\delta\mathbf{x}_k$ that are usually part of the KF are related to the position, velocity, attitude, sensors errors and any other parameter that can be adapted into the equations of the system (Eq. (2.41)) and can contribute to the overall improvement of the navigation accuracy.

The transition matrix $\boldsymbol{\Phi}_k$ that describes the system's natural dynamics can be calculated from $\mathbf{F}(t)$ assuming that this is constant over the (t_k,t_{k+1}) interval of interest [62]. Thus, $\boldsymbol{\Phi}_k$ can be calculated by the matrix exponential of $\mathbf{F}(t_k)\Delta t$, that is,

$$\boldsymbol{\Phi}_k = e^{\mathbf{F}(t_k)\Delta t} = \mathbf{I} + \mathbf{F}(t_k)\Delta t + \frac{(\mathbf{F}(t_k)\Delta t)^2}{2!} + \ldots \tag{2.42}$$

where Δt is the difference between t_{k+1} and t_k, in the case of the IMUs that are used to test the INS, they provide measurements at a frequency of $(1/\Delta t) = 100\ Hz$, so the first two terms are sufficient to estimate the transition matrix [5].

Based on the assumption that the system noise is a white sequence, $\mathbf{w}_k$ will represent the inertial sensors noise with zero mean Gaussian noise and known covariance matrix $\mathbf{Q}_k$,

$$\mathrm{E}\left[\mathbf{w}_k\right] = 0 \tag{2.43}$$

$$\mathrm{E}\left[\mathbf{w}_k \mathbf{w}_i^T\right] = \begin{cases} \mathbf{Q}_k, & i = k \\ 0, & i \neq k \end{cases} \tag{2.44}$$

where E[•] is the mathematical expectation.

The covariance matrix $\mathbf{Q}_k$ can be written in integral form [63, 64]:

$$\mathbf{Q}_k = \int_{t_k}^{t_{k+1}} \boldsymbol{\Phi}\left(t_{k+1},\tau\right) \mathbf{G}\left(\tau\right) \mathbf{Q}\left(\tau\right) \mathbf{G}^T\left(\tau\right) \boldsymbol{\Phi}^T\left(t_{k+1},\tau\right) d\tau \tag{2.45}$$

A solution of the above equation can be obtained by a trapezoidal integration and considering that $\mathbf{G}\left(\tau\right)\mathbf{Q}\left(\tau\right)\mathbf{G}^T\left(\tau\right)$ is constant in the time interval t_k and t_{k+1}, yielding [63]

$$\mathbf{Q}_k \approx \frac{1}{2}\left[\boldsymbol{\Phi}_k \mathbf{G}\left(t_k\right)\mathbf{Q}\left(t_k\right)\mathbf{G}^T\left(t_k\right) + \mathbf{G}\left(t_k\right)\mathbf{Q}\left(t_k\right)\mathbf{G}^T\left(t_k\right)\boldsymbol{\Phi}_k^T\right]\Delta t \tag{2.46}$$

where the spectral density matrix $\mathbf{Q}(t)$ is related to the white noises $\mathbf{w}(t)$ by $\mathbf{Q}\left(t\right)\delta(t-\tau) = E\left[\mathbf{w}(t)\mathbf{w}^T(\tau)\right]$, the operator $\delta[\bullet]$ denotes the Dirac delta function with units $1/time$ [64].

2.6.2 Measurement Model

The measurement model describes the relationship between the states and the measurements by the matrix $\mathbf{H}_k$. The linear relationship between the observations (measurements) and states considering additive noise is given by:

$$\delta\mathbf{z}_k = \mathbf{H}_k \delta\mathbf{x}_k + \mathbf{v}_k \tag{2.47}$$

where

- $\delta\mathbf{z}_k$ is a measurement vector at time t_k ;

- $\mathbf{H}_k$ is a design matrix giving the noiseless connection between the measurement and the state vector at time t_k ; and
- $\mathbf{v}_k$ is a measurement noise vector.

The measurement noise is assumed to be a white sequence with zero mean and uncorrelated with the process noise. For the loosely coupled approach the measurements are provided by the difference between GPS and INS outputs, that correspond to position and velocity. The covariance matrix for the measurement noise vector $\mathbf{v}_k$ is given by:

$$\mathrm{E}\left[\mathbf{v}_k\right] = 0 \tag{2.48}$$

$$\mathrm{E}\left[\mathbf{v}_k\mathbf{v}_i^T\right] = \begin{cases} \mathbf{R}_k, & i = k \\ 0, & i \neq k \end{cases} \tag{2.49}$$

$$\mathrm{E}\left[\mathbf{w}_k\mathbf{v}_i^T\right] = 0 \qquad \textit{for all } k \textit{ and } i. \tag{2.50}$$

where $\mathbf{R}_k = E\left[\mathbf{v}_k\mathbf{v}_k^T\right]$ is the measurement noise covariance matrix.

With the assumption of a prior estimate $\delta\hat{\mathbf{x}}_k^-$, the measurement vector $\delta\mathbf{z}_k$ can be used to improved the prior estimate. For this propose Kalman filter uses a linear blending of the noisy measurement and the prior estimate in accordance with the following equation [62]:

$$\delta\hat{\mathbf{x}}_k^+ = \delta\hat{\mathbf{x}}_k^- + \mathbf{K}_k\left(\delta\mathbf{z}_k - \mathbf{H}_k\delta\hat{\mathbf{x}}_k^-\right) \tag{2.51}$$

where $\delta\hat{\mathbf{x}}_k^+$ is the update estimate and $\mathbf{K}_k$ is the blending factor or better known as the Kalman gain matrix that minimizes the mean-square estimation error and can be expressed by:

$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^T\left(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{R}_k\right)^{-1} \tag{2.52}$$

where $\mathbf{P}_k^-$ is the a priori accuracy estimate for the states given by $\delta\mathbf{x}_{k+1}$, and it can be calculated by the transition matrix ($\boldsymbol{\Phi}_k$), the a priori error covariance matrix ($\mathbf{P}_{k-1}^+$) and the covariance matrix for the involved system noise $\mathbf{Q}_{k-1}$ as states in Eq. (2.53) [5].

$$\mathbf{P}_k^- = \boldsymbol{\Phi}_k \mathbf{P}_{k-1}^+ \boldsymbol{\Phi}_k^T + \mathbf{Q}_{k-1} \tag{2.53}$$

From Eq. (2.52) it can be noted that having a large uncertainty in measurement ($\mathbf{R}_k$) we will have a small Kalman gain, so a small weight will be given to the measure ($\delta\mathbf{z}_k$) when we estimate $\delta\hat{\mathbf{x}}_k^+$ (Eq. (2.51)). Moreover, if the uncertainty of the measurement is small, $\mathbf{K}_k$ will be large so it will give a significant weight when we calculate $\delta\hat{\mathbf{x}}_k^+$. This Kalman gain is also considered for updating the error covariance matrix that is expressed as:

$$\mathbf{P}_k^+ = \mathbf{P}_k^- + \mathbf{K}_k \mathbf{H}_k \mathbf{P}_k^- \tag{2.54}$$

Details of the derivation of $\mathbf{K}_k$, $\mathbf{P}_k^-$ and $\mathbf{P}_k^+$ are available in [65], [64], or [62].

2.6.3 Extended Kalman Filter

Since the Kalman filter is only applied to linear systems and in the case of the inertial system it is described by a non-linear model due to the navigation equations (see Eq. (2.17)), the typical approach to deal with non-linear systems is to linearize about some nominal point or trajectory, achieving a perturbation model or error model [63]. In the inertial navigation the linearization is performed at the current state (nominal point), and the KF that involves the linearization about the current state is referred to as an Extended Kalman Filter (EKF) [66].

Thus the non-linear system represented by equations of inertial navigation system (Eq. (2.17)), can be linearized around the current state. This procedure is performed by perturbing the kinematic equations in order to obtain a dynamic error equations for navigation system errors. In this sense, position, velocity and Euler angles can be written as it follows [67]:

$$\begin{aligned} \hat{\mathbf{r}}^n &= \mathbf{r}^n + \delta\mathbf{r}^n \\ \hat{\mathbf{v}}^n &= \mathbf{v}^n + \delta\mathbf{v}^n \\ \hat{\mathbf{C}}_b^n &= \left(\mathbf{I} - (\boldsymbol{\epsilon}^n\times)\right)\mathbf{C}_b^n \end{aligned} \tag{2.55}$$

where ˆ denotes the computed value position ($\hat{\mathbf{r}}^n$), velocity ($\hat{\mathbf{v}}^n$) and attitude DCM ($\hat{\mathbf{C}}_b^n$), which are represented by the true value plus a perturbation (position errors, velocity errors) denoted by δ, while the term ($\boldsymbol{\epsilon}^n\times$) is associated with perturbation of the attitude and is a skew-symmetric matrix of the attitude errors given by:

$$(\boldsymbol{\epsilon}^n\times) = \begin{bmatrix} 0 & -\epsilon_D & \epsilon_E \\ \epsilon_D & 0 & -\epsilon_N \\ -\epsilon_E & \epsilon_N & 0 \end{bmatrix} \tag{2.56}$$

According to [47], the resulting error equations for this linearization approach in the n-frame are given by the following error state model:

$$\begin{bmatrix} \delta\dot{\mathbf{r}}^n \\ \delta\dot{\mathbf{v}}^n \\ \dot{\boldsymbol{\epsilon}}^n \end{bmatrix} = \begin{bmatrix} \mathbf{F}_{rr} & \mathbf{F}_{rv} & \mathbf{0}_{3\times3} \\ \mathbf{F}_{vr} & \mathbf{F}_{vv} & (\mathbf{f}^n\times) \\ \mathbf{F}_{er} & \mathbf{F}_{ev} & -(\boldsymbol{\omega}_{in}^n\times) \end{bmatrix} \cdot \begin{bmatrix} \delta\mathbf{r}^n \\ \delta\mathbf{v}^n \\ \boldsymbol{\epsilon}^n \end{bmatrix} + \mathbf{w} \tag{2.57}$$

where the error dynamic matrix is comprised with nine states, which are position error ($\delta\mathbf{r}^n$), velocity error ($\delta\mathbf{v}^n$) and attitude error ($\boldsymbol{\epsilon}^n$). All the matrices F_{xx} are detailed in [5, 59], and the derivation of these error equations and similar ones can be found in [47, 67]. ($\mathbf{f}^n\times$) is the skew-symmetric matrix of the specific force corrected and expressed in the n-frame whereas ($\boldsymbol{\omega}_{in}^n\times$) is the skew-symmetric matrix of rotation vector of the n-frame with respect to the i-frame. Eventually the system noise vector $\mathbf{w}$ is given by:

$$\mathbf{w} = \begin{bmatrix} \mathbf{w}_a & \mathbf{w}_g \end{bmatrix}^T \tag{2.58}$$

where $\mathbf{w}_a$ and $\mathbf{w}_g$ represent the white noise on the accelerometers and gyros, respectively. The correspondent spectral density matrix $\mathbf{Q}$ can be expressed by:

$$\mathbf{Q} = \begin{bmatrix} diag(\mathbf{q}_a) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & diag(\mathbf{q}_g) \end{bmatrix} \tag{2.59}$$

$diag(\mathbf{vec})$ denotes a diagonal matrix with the elements of the vector $\mathbf{vec}$. The spectral density for each sensor (*i.e.*, $\mathbf{q}_a$ and $\mathbf{q}_g$) will be computed in Chapter 4.

Since Eq. (2.57) considers the INS error dynamics expressed in state-space form as Eq. (2.40), this system error dynamic model also known as INS filter can be implemented with KF. Note that the system dynamic matrix is in continuous time so it can be expressed in discrete time through Eq. (2.42). In addition, the dynamic model does not include the error states that would represent the IMU errors, those errors need to be adapted to the model as IMU error states. This will be explained in Chapter 3.

As it was shown in Fig. 2.7 in the loosely-coupled integration approach the measurement of the INS Kalman filter is the difference between position and velocity provided by the GPS and INS solution, that is, a residual error, so the measurement matrix is given in discrete time by:

$$\delta\mathbf{z}_k = \begin{bmatrix} \delta\mathbf{R}^n \\ \delta\mathbf{V}^n \end{bmatrix} = \begin{bmatrix} \mathbf{r}^n_{INS} - \mathbf{r}^n_{GPS} \\ \mathbf{v}^n_{INS} - \mathbf{v}^n_{GPS} \end{bmatrix} = \begin{bmatrix} \varphi_{INS} - \varphi_{GPS} \\ \lambda_{INS} - \lambda_{GPS} \\ h_{INS} - h_{GPS} \\ \mathbf{v}^n_{INS} - \mathbf{v}^n_{GPS} \end{bmatrix} \tag{2.60}$$

where φ, λ, h are the latitude, longitude and altitude respectively.

In order to avoid numerical instabilities when computing the EKF because φ and λ are in radians and their values are very small, it is necessary to multiply the first two rows of Eq. (2.60) by $(M + h)$ and $(N + h)\cos\varphi$ to obtain [47]:

$$\delta\mathbf{z}_k = \begin{bmatrix} (M+h)\,(\varphi_{INS} - \varphi_{GPS}) \\ (N+h)\cos\varphi(\lambda_{INS} - \lambda_{GPS}) \\ h_{INS} - h_{GPS} \\ \mathbf{v}^n_{INS} - \mathbf{v}^n_{GPS} \end{bmatrix} \tag{2.61}$$

where M and N are the radii of curvature in the meridian and prime vertical at a given latitude, as reported by Eqs. (2.14) and (2.15), respectively.

The design matrix $\mathbf{H}_k$ becomes [59]:

$$\mathbf{H}_k = \begin{bmatrix} M+h & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & (N+h)\cos\varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{2.62}$$

Finally, the measurement noise matrix obtained from the GPS can be expressed as:

$$\mathbf{R}_k = \begin{bmatrix} (M+h)^2 \cdot \sigma_\varphi^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & (N+h)^2 \cdot \cos^2\varphi \cdot \sigma_\lambda^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_h^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{V_N}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{V_E}^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{V_D}^2 \end{bmatrix} \tag{2.63}$$

The value of each element in the diagonal matrix $\mathbf{R}_k$ depends on the accuracy of the GPS estimates [59]. Since we use a feedback loosely-coupled approach, the error state vector is set to zero after every measurement updates [47,53,59,68].

2.6.4 KF Algorithm

The algorithm for the KF implementation contains two stages, the first one known as prediction and the second one update.

Prediction

This stage predicts the vector states and the uncertainty of the states by the following equations.

$$\delta\hat{\mathbf{x}}_k^- = \boldsymbol{\Phi}_k \delta\hat{\mathbf{x}}_{k-1}^+ \tag{2.64}$$

$$\mathbf{P}_k^- = \boldsymbol{\Phi}_k \mathbf{P}_{k-1}^+ \boldsymbol{\Phi}_k^T + \mathbf{Q}_{k-1} \tag{2.65}$$

Generally, in the integrated navigation this stage is computed every time there is a sample available from the inertial sensors, so it is executed even if there is a GPS update or not. This also propagates the covariance matrix error and the states from the current epoch to the next, and in case of a GPS outage, it will estimate the states in order to compensate the navigation solution and the INS errors.

Update

The update stage is computed only when there is information available to the measurement (*i.e.*, from GPS) and is given by the following equations:

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T \left(\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k \right)^{-1} \tag{2.66}$$

$$\delta\hat{\mathbf{x}}_k^+ = \delta\hat{\mathbf{x}}_k^- + \mathbf{K}_k \left(\delta\mathbf{z}_k - \mathbf{H}_k \delta\hat{\mathbf{x}}_k^- \right) \tag{2.67}$$

$$\mathbf{P}_k^+ = \mathbf{P}_k^- + \mathbf{K}_k \mathbf{H}_k \mathbf{P}_k^- \tag{2.68}$$

A summary of the algorithm for the implementation of the discrete EKF is depicted in Fig 2.9.

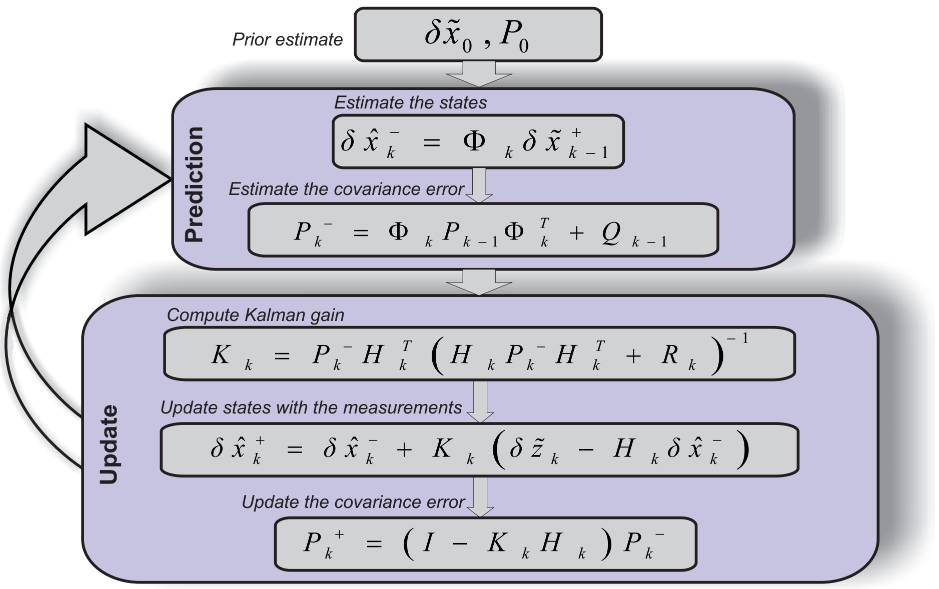

Prior estimate
$\delta\tilde{x}_0, P_0$
Prediction
Estimate the states
$\delta\hat{x}_k^- = \Phi_k \delta\tilde{x}_{k-1}^+$
Estimate the covariance error
$P_k^- = \Phi_k P_{k-1} \Phi_k^T + Q_{k-1}$
Update
Compute Kalman gain
$K_k = P_k^- H_k^T \left(H_k P_k^- H_k^T + R_k\right)^{-1}$
Update states with the measurements
$\delta\hat{x}_k^+ = \delta\hat{x}_k^- + K_k\left(\delta\tilde{z}_k - H_k \delta\hat{x}_k^-\right)$
Update the covariance error
$P_k^+ = \left(I - K_k H_k\right) P_k^-$

Figure 2.9: Discrete Kalman filter algorithm.

Chapter 3

MEMS IMU Inertial Sensors Errors

3.1 Introduction

Although initially the use of strapdown systems was debated because of the full dynamic motion experienced by the sensors and its computational requirements, in the last decades advances in the computer technologies and sensors have led to show more interest in this type of inertial systems [41]. Indeed, an important aspect that has promoted its use in many applications is the progress in the fabrication of inertial sensors based on Microelectromechanical Systems (MEMS). This is because they are low cost, lightweight and small size. Despite the fact that MEMS inertial sensors have such characteristics these sensors require a proper error analysis to improve their performance, which is one of the main objectives in this work. For this reason, in order to have a better understanding of the errors that are involved in this inertial sensors technology this chapter provides information about how they work, the different types of sensors, how they are classified, general performance characteristics and finally a description of the IMU is presented with the MEMS inertial sensors deterministic and stochastic errors.

3.2 MEMS Inertial Sensors

3.2.1 Microelectromechanical Systems (MEMS)

Development of silicon micromachining was introduced by Bell Laboratories in 1950's with the discovery of the piezoresistive effect of silicon [69, 70]. Besides that these micromachined silicon sensors had advantages such as small size, low cost and lightweight, there were two factors that made these devices attractive: the mechanical properties and the readily available fabrication technology [70].

Over the years, advances in silicon micromachined systems and microelectronics systems allowed to integrate mechanical devices together with electronic circuitry on a single chip, one of the first developments was the accelerometer for air bag application that is described in [71].

The fusion between these two system is known as a Microelectromechanical Systems (MEMS) and it involves a mechanical and electronic design for development of these devices. More exactly MEMS can be defined as devices that have characteristic length of less than 1 mm but more than 1 $micron$, that combine electrical and mechanical components, and that are fabricated using integrated circuit batch-processing technologies [72].

Recently, MEMS inertial sensors are being used in a variety of applications, especially in automotive applications, such as lane-keeping assistance, Electronic Stability Control (ESC), preventing vehicle rollover, support to the GPS when there is not a line of sight of the satellites, *etc.* One of the reasons for their high demand in a wide range of application is their low price. To reduce the price, the size of the sensor chip should be minimum and the structure and fabrication process should be simple [73]. Although these aspects can be satisfied it is necessary to consider that they play an important role in the accuracy of the sensors.

Due to the small size, these devices bring a low cost because they can be fabricated using batch-processing production. However, the reduction in size of the sensing element creates challenges for attaining good performance, since as size decreases, then sensitivity (scale factor) decreases and noise increases [74]. According to Honeywell, one of the vendors of inertial sensors, there is a compromise between the nominal size of a inertial sensor and its long-term bias stability, *i.e.*, the smaller the device the lower the performance, while the larger the device the less significant the long-term bias instability and, hence, high performance (see [75] for further details). Next section describes the operational principle of the inertial sensors in order to figure out why the small size affects their performance.

3.2.2 Operational Principle of MEMS Inertial Sensors

The fundamental concept of operation of an inertial sensor can be described from Fig. 3.1. It shows a proof mass, which is suspended to a mechanical frame by means of springs. There is also a pickoff which relates the displacement of the mass to an electrical signal. Thus, when an input force is applied to the structure, there is a displacement of the proof mass, the resultant displacement can be measured by the pickoff that senses the applied force indirectly. Depending on the inertial sensor, its transductor may associate the input force with an acceleration of the mass or a Coriolis acceleration induced by an angular rotation of the mass (*i.e.*, vibratory rate gyroscope), which would be the case of an accelerometer or a gyroscope, respectively.

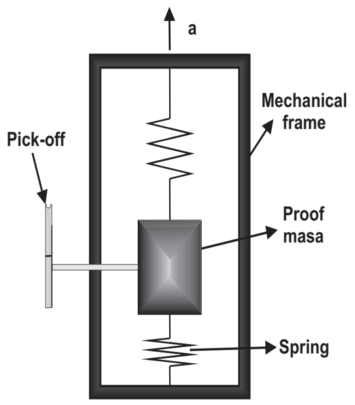


Figure 3.1: Electro-Mechanical mass/spring system for a simple accelerometer.

There are different transductor methods to transform the input force of interest into a response in the proof mass, for further details refer to [43, 76]. A high-quality inertial sensor generally possesses high transduction gain, while rejecting the effects of parasitic forces on the proof mass; these forces affect the performance of the sensors and they are associated to packaging and stresses induced by undesirable forces acting on the proof mass due to motions of a type other than the one to be sensed [2]. The reduction of these parasitic forces involves developing more complex mechanical structures and circuitry in order to obtain the best performance of the sensor.

Since the proof mass that comprises an inertial sensor is small, as well as its movement that is associated with the physical signal to be sensed (*e.g.*, angular velocity or acceleration), there are several factors not only parasitic forces but also temperature effects, noises in the interface circuits *etc*, which make it difficult to obtain an accurate measurement of the movement of the mass.

Accelerometers

MEMS accelerometers measure acceleration that is typically provided in g, where 1 g is equivalent to 9.81 m/s^2. Accelerometers development appears more mature than gyro development, due primarily to the push for the reliable crash detectors in the automobile market, so gyro development is the limiting factor in achievable accuracy in a MEMS IMU [75].

These sensors not only measure Earth gravity but also linear acceleration due to motion. Fig. 3.1 illustrates a simple accelerometer, that, as mentioned, it measures the displacement of the proof mass which is related to the applied force, the input force can be generated by the motion of the sensor or by gravity.

Sensing acceleration due to gravity requires a DC measurement and in consumer-grade accelerometers natural frequencies appear in the low-kilohertz range, so ten of nanometers mechanical displacement occurs at DC for 1 g acceleration [2]. For a given sensitivity, to discriminate between the force due to acceleration and parasitic forces due to others factors such as thermal stress, it is useful to maximize both mass and spring stiffness [2], which implies an increase in the size of the device.

Gyroscopes

These inertial sensors measure angular rate and the units are typically given in deg/h. The vast majority of the reported micromachined rate gyros utilizes a vibratory proof mass suspended by flexible beams above a substrate [76], which is known as a vibratory gyroscope.

MEMS gyros are more challenging to manufacture than accelerometers, in fact, they are larger devices with higher cost and can require as much as 10 times more power than the MEMS accelerometers [75].

The basic architecture of a vibratory gyroscope is comprised by a drive-mode and a sense-mode. The drive-mode generates and maintains a linear drive oscillation or rotatory drive oscillation depending on whether the gyroscope implementation is with a linear or torsional resonator. The sense-mode measures the sinusoidal Coriolis force induced due to the combination of the drive vibration and an angular rate input [76]. Fig. 3.2 depicts a generic z-axis gyro, the proof mass requires to be free in a way that allows movements in two orthogonal directions (*i.e.*, x and y directions), this enables the proof mass to have movements with two degrees of freedom (2-DoF). In this simple gyroscope, the sense-mode consists of a proof mass, the suspension that allows the proof mass to oscillate in the y direction represented by k_y, and the sense-mode detection electrodes associated to C_y. Similar to the sense-mode, the drive-mode is comprised by the proof mass, a suspension that allows to oscillate the mass in the x direction given by k_x, and the drive-mode electrodes related to C_x. When the gyroscope is exposed to an angular rate, in this case, to sense z-axis (yaw) angular rate, a sinusoidal Coriolis force at the frequency of the drive-mode oscillation is induced in the sense direction. The Coriolis force excites the sense-mode accelerometer causing the proof mass to respond in y direction, this response is reflected in the detection electrodes [76].

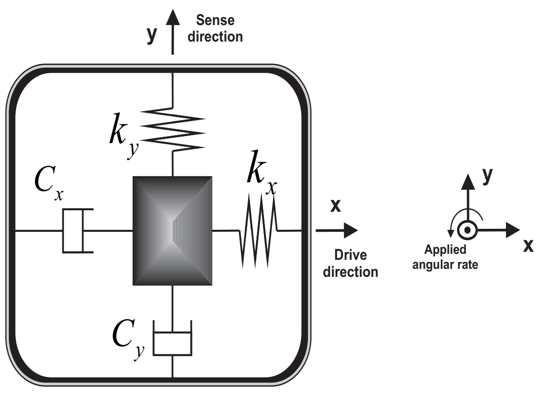


Figure 3.2: Electro-Mechanical mass/spring system for a simple vibratory rate gyroscope.

Figure 3.3 illustrates a simplified version of one of the MEMS gyroscopes used in this thesis, the structure is similar to an accelerometer, which has a proof mass (inner structure) suspended by springs. As mentioned previously the difference in operation is that the angular velocity is derived by measuring the Coriolis force on the vibrating mass. When the structure is exposed to an angular rate, the Coriolis force couples sense into an outer mechanical frame (outer structure), which contains movable fingers that are placed between fixed pickoff fingers. The fingers can also be seen like combs where the capacitance change between them when there is a movement of the frame, for further details about this gyro refer to [77].

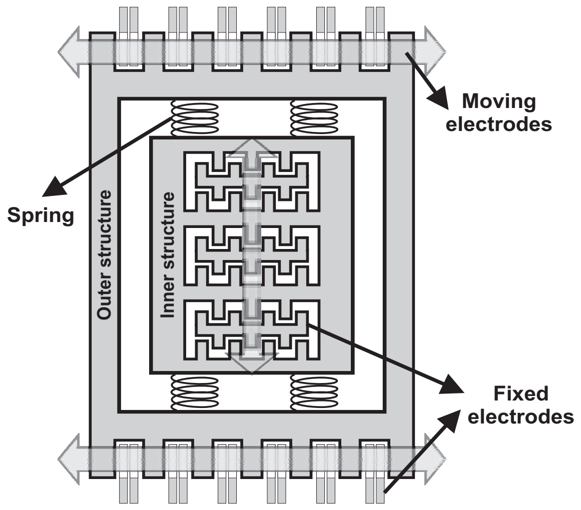


Figure 3.3: Simplified gyro sensing structure of one gyroscope of the 3DM GX3 25 IMU.

The absolute value of the Coriolis force is extremely small in consumer-grade MEMS devices and it is difficult to precisely measure the movement of the structure [73]. Indeed, a small displacement generated by the Coriolis force that is not precisely measured may lead to a large error in an inertial navigation system since it induces a bias in the measurements of the inertial sensor that are used as input of the Mechanization stage described in Section 2.4. Therefore, techniques such as Kalman filtering are being used to correct this error and related errors in navigation systems [73].

In general, there are many contributors to the noise in an inertial sensor, for instance: the readout electronics, mechanical damping, electrical resistances, *etc*; MEMS sensors are so small that just a Brownian motion agitates bacteria and dust motes, which can be a large force on a tiny MEMS component [78]. In [3] the mechanical-thermal noise in micromachined sensors is analyzed by adding a noise force in a simple accelerometer, similar to the one showed in Fig. 3.1. The noise force includes Brownian mechanical noise from air damping and electronic noise from the readout circuit. According to Gabrielson [3], the Brownian force $F = \sqrt{4k_BTD}$ causes Brownian motion of the proof mass m, and in order to get the noise response, the signal excitation displacement is set to zero and the response of the mass/spring system is solved in terms of the noise force, so it gives a Brownian equivalent acceleration noise [79]:

$$g_{b,B} = \frac{\sqrt{4k_BTD}}{mg} \tag{3.1}$$

where k_B is the Boltzmann constant, T is the ambient temperature, D is the damping coefficient of the proof mass m and g is the Earth's gravity. From Eq. (3.1) we see that a large mass help to achieve a low noise floor [79]. Regarding the damping, it results from many sources, but for inertial sensors air damping is typically dominant; for a low damping, it is important to hermetically seal the mechanical elements to allow operation of the sensor at low pressure, hermetic sealing also prevents contaminants, particles, and moisture from interfering with the sensor operation [2]. In order to have a larger mass, bulk micromachining process can be used to produce wafer-thick proof masses. This fabrication process is described in the following subsection.

In addition to mechanics perturbations, the interface circuit also plays an important role in the overall performance of the whole system [80]. This is because inertial sensors require dedicated microelectronics circuitry in order to achieve a signal conditioning to properly relate the sense movement of the proof mass into an electrical signal that will be the sensor output. In this stage, one of the most significant noises that appear in the electronic component of the MEMS sensors is the flicker noise. According to [81] the noise spectrum of the flicker noise in a MOS (MetalâĂŞOxideâĂŞSemiconductor) transistor can be expressed by:

$$v^2_{flicker} = \frac{k_f}{C_{ox}WL}\frac{1}{f} \tag{3.2}$$

where k_f is a process-dependent parameter that indicates how clean the fabrication process is, C_{ox} is the gate oxide per unit area, W and L are the drawn transistor geometry and f is frequency. From Eq. (3.2) it can be seen that the magnitude of the flicker noise depends on the geometry of the transistor, *i.e.*, the smaller is the size greater is the flicker noise affecting the device performance. Further details about low level mechanical and electrical analysis of different noise sources in inertial sensors can be found in [2, 3, 79–81].

3.2.3 Classification of MEMS Inertial Sensors

There are different approaches to classify MEMS inertial sensors, among them: the fabrication process, the method to detect the position of the proof mass and the mode of operation [75]. In this case, the classification by the fabrication process will be described, for further details about other approaches refer to [5, 43].

The essence of all micromachining techniques is a successive patterning of thin structural layers on a substrate and depending on the structural layer forming technique, micromachining processes are usually divided into two categories: bulk micromachining and surface micromachining [76].

Bulk Micromachining

Bulk micromachining process was first developed for pressure sensors in the sixties and it has become a mature technology that has been under intensive development [70]. This entails controlled material removal on the substrate to transfer a desired pattern into the structural layer. Most of bulk micromachining processes bound two or more wafers, and the moving structures are made out of the whole thickness of a silicon wafer [76]. This thick structural layer created in a bulk machined sensor offers advantages because it increases the mass and the area available to detect changes with the capacitive electrodes. Thicker suspension beams also provide higher stiffness, which reduces shock and vibration susceptibility [76]. In general, this process improves the mechanical stability and thus the inertial sensor performance. Nonetheless, the size of bulk micromachined sensor is bigger than a surface micromachining sensor because of its large mechanical structure, which is a factor to take into account for mass production. Additionally, the sensing structure is process-incompatible with the interface circuitry [75], which also makes it difficult to be cost effective.

Surface Micromachining

Surface micromachining process was introduced in the 1980's to achieve a process compatible with CMOS (Complementary Metal-Oxide-Semiconductor) manufacturing technology. In contrast to bulk micromachining that is a subtractive process, surface micromachining is an additive technique, so the devices are built by depositioning multiple stacks of alternating structural layers [76]. It offers better process control of structural thickness and size [70]. Therefore, the size of a surface micromachined sensor is smaller compared to a bulk micromachined, which is a characteristic that makes them attractive for mass production.

Since this is process-compatible with standard integrated circuit manufacturing, it makes easy its integration with other electronic components such as control, filtering and signal conditioning stages. Indeed, an Application-Specific Integrated Circuit (ASIC) is usually adapted to MEMS sensing element in order to implement the electronic circuitry required to interface of the inertial sensors. In spite of this, surface micromachined have relatively thin proof mass and, hence, high mechanical noise which degrades the performance of the inertial sensor [75].

Table 3.1 summarizes the advantages and disadvantages for the MEMS sensors according to the fabrication process. For further information about different fabrication technologies using these two processes refer to [76].

3.2.4 General Characteristics of MEMS Inertial Sensors

The growing demand of micromachining devices has led to combine surface micromachining together with bulk micromachining processes to a point that it is not easy to differentiate between them. Despite this, generally inertial sensor used in low cost INS are built with surface micromachining processes, which implies small size but also low performance.

Table 3.1: Classification of MEMS sensors according to fabrication processes (adopted from [75]).

Category	Sensing element	Advantages	Disadvantages
Surface micromachining	A thin silicon structure located on the surface of a die	Suites well for integration with other electronic components	Relatively low accuracy and small bandwidth
Bulk micromachining	A single crystal inside a block sandwiched	Better accuracy that surface micromachining	Larger size, more expensive and not good for integration

Since inertial sensors are a fundamental component in an INS, errors involved in each sensor need to be minimized in order to enhance the INS solution. This may be carried out through various laboratory tests, where the objective is to analyse and determine the characteristics of the errors that are affecting the inertial sensors. Subsequently, the parameters obtained from the analysis, that provide information of the errors, are adapted into the INS with the purpose of correcting them. In the case of the loosely-couple GPS/INS integration, the error compensation is achieved at the sensors output measurements, *i.e.*, at a stage prior to the navigation equations due to the fact that if they are not attenuated there, they will be amplified during the integral operation, degrading in this way the INS solution. Before analysing the errors, in the following subsections, general specifications that are often provided by the manufacture and some of the most important performance characteristics of MEMS inertial sensors are described. These characteristics are the bias, bias instability, scale factor, temperature-dependent bias/scale factor, vibration sensitivity, nonlinearity and shock survival.

Bias

Although there are several types of bias when we refer to inertial sensors, according to [82], the bias can be defined as the average over a specified time of accelerometer/gyro measured output that at specified operating conditions has no correlation with input acceleration or rotation. This error can be caused by various effects ranging from manufacturing tolerances until temperature gradients. In [83] the bias is defined as any nonzero output when the input is zero. This variation of the sensor output is a deviation of the measurement from the true value, which has consequences in the system where the MEMS sensors are mounted.

The bias is widely used as a performance indicator of an inertial sensor, in general the lower the bias the better the performance. It is necessary to take into account that the size of the bias is independent of any motion to which the sensor may be subjected and is sometimes referred to as the acceleration (or g) independent bias [43]. Furthermore, the bias can be broken down in different categories, when it is constant from turn-on to turn-on and when it varies randomly at each turn-on. It will be studied in Section 3.3.2.

In the case of an INS, a small bias error will have less influence in the integration stages of the INS and thus a minor error in the navigation solution. A simplified example can be given assuming that an accelerometer has a bias of $b_f = 0.005\ g$ and no other noise or disturbance is involved. If there is not compensation of this bias, the velocity error will be proportional to t (*time*), while the position error will be proportional to t^2 (see Eq. (3.3)). That is, if there is not movement of the accelerometer the position error may reach 88.2 m in only 60 sec. Note that this is a simplified analysis since it does not consider all the errors that can potentially affect a low cost INS. A similar analysis for both accelerometers and gyros can be found in [5].

$$v = \int b_f \, dt = b_f t \qquad p = \int v \, dt = \frac{b_f t^2}{2} \tag{3.3}$$

At this point it is worth noting the difference between the terms run-to-run and in-run, for instance, the run-to-run bias, refers to changes in a parameter for each run. This is because every time the sensor is switched on, a slightly different bias value is observed [84]. This error is constant once the sensor is switched on, here on we will call it turn-on bias. Moreover, in-run bias is an error that occurs due to change in bias during a run [5]. This error is related to the bias stability that will be explained below. The same definitions are valid for scale factor.

Bias stability

A term that is normally found in the specifications of inertial sensors is the bias stability, which refers to a measured of the bias changed over time. In other words, bias stability measurement provides information about how stable the bias of the sensor is over a certain specified period of time [85]. In order to obtain this parameter several measures of the sensor should be performed at different time intervals to determine the bias changed over time. Nonetheless, several concerns arise, as for instance: how many intervals should be evaluated? What should be the size of each interval? What is the bias stability? To address these concerns the bias stability is computed by means of Allan variance technique, where the minimum Allan standard deviation corresponds to the bias stability. For a complete procedure to compute the bias stability see [86]. Although Allan variance was initially introduced to characterize noise and stability in clock systems, it has become popular in the INS community hence it is often employed for noise identification and analysis in inertial sensors. A detailed description of this technique will be presented in Chapter 4.

Albeit the bias stability is a reference parameter to compare or select a gyro or accelerometer, and even though it is provided by the manufacture in some low cost sensors, there are others error sources involved in the MEMS sensors that are not less relevant. For instance, they exhibit scale factor errors, sensitive to external factors such as vibrations and temperature.

Scale factor

The scale factor relates the output signal changes of the sensor with the physical signal changes at the sensor input (*i.e.*, acceleration and angular velocity). This term can be determined from the curve where the x-axis is the sensor input and the y-axis is the sensor output, thus the slope of the best fit line provides the scale factor (see Fig. 3.4). An ideal sensor has a scale factor of 1, hence, any scale factor is above or below 1 is contaminated with sensor errors [5]. The difference between the scale factors of the two curves (*i.e.*, ideal and measured) represents the scale factor error, which is commonly expressed as a ratio of output error to input, in parts per million (ppm) or as a percentage (%) for low cost MEMS inertial sensor [43]. By the scale factor curve additional errors can be identified such as scale factor nonlinearity and scale factor asymmetry, which are associated to thermal changes.

Temperature-dependent bias/scale factor

The temperature-dependent variations can be quite pronounced in very low cost MEMS sensors [5]. In fact, changes in the temperature of the device involves changes in the scale factor and the bias of the sensor. Dependence on temperature can be analysed by a thermal test, which determines changes in the sensors characteristics when the sensor is exposed to temperature variations. An additional limiting factor is the temperature hysteresis, that is, the difference in output at a specific temperature when that temperature approached via cooling versus heating [87]. Further details about thermal tests can be found in [43, 84].

Vibration sensitivity

In most of applications where INS are employed the inertial sensors are subjected to movement and vibrations that perturb the measurements. Generally, the inertial sensors are sensitive to accelerations and it changes depending on the frequency of vibration. This sensitivity is due to imperfections in the mechanical structure, which causes bias that are proportional to the magnitude of the applied acceleration (*i.e.*, g-dependent bias, g^2-dependent bias, *etc*). The sensitivity to vibrations can be defined as a steady state error in the output while vibratory disturbances are acting on the sensor [82].

Although the vibratory rejection is not specified for the majority of low cost inertial sensors, this is a characteristic that is becoming more and more important when selecting an inertial sensor, even more than the traditional bias stability. This is because vibration sensitivity is often the more severe performance limitation and the bias stability represents a smaller component of error [87]. Despite this, MEMS inertial sensors are designed using extremely simple and compact mechanical systems that are not optimized for vibration rejection (rather, they are optimized for low cost) and can suffer due to vibration greatly [87]. In order to provide vibratory rejection anti-vibration mounts can be used. Nevertheless, these mechanical assemblies are not easy to design and they also increase the size of the sensors, as well as the vibration reduction characteristics varies depending on the temperature.

Nonlinearity

From the straight line fitted to the plot of input-output characteristics (see Fig. 3.4) a nonlineal trend can be observed if we compare with the ideal linear curve (slope 1). This is sometimes associated to the inherent nonlinearities of the sensors [88] and thermal effects [43], and can be defined as a systematic deviation from the straight line that defines the nominal input-output relationship [82].

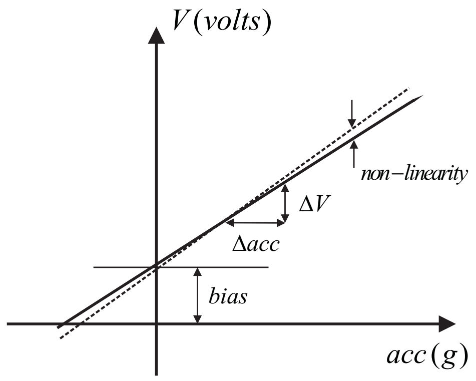


Figure 3.4: Relationship between the output voltage and input acceleration (angular velocity) (Adopted from [88]).

Shock survival

This is defined as the maximum shock that the operating on non-operating device can endure without failure, and conform to all performance requirement after exposure [76]. To evaluate this parameter a shock test is achieved, which consists in measuring the response of a sensor to an applied shock and to establish the resilience of the sensor to such an applied acceleration over a very short duration, typically in the order of few milliseconds [43]. After performing the test characteristics such as bias are compared, that is a comparison before and after applying the shock, so it can be determined whether there has been a transient or a permanent change after exposing the device to the shock.

Similar laboratory test can be assessed in order to evaluate their performance under certain environmental conditions, for example: thermal test, centrifuge test, vibratory test, *etc.* The behavior of specific parameters can be determined through the analysis of the lab tests, and also the ability to endure shocks and vibrations that can be induced by the vehicle where the sensors are assembly. In this sense, the objective of these tests is to have a better understanding of the behavior of the sensors in different situations that may occur in a certain application, as well as to determine performance characteristics of the sensor that can be compensated and thus enhance their accuracy. Hence several errors influence the MEMS inertial sensors, in this case we emphasis in the bias, scale factor and thermal sensitivity due to the fact that these are some of the main error sources when low cost MEMS sensor are used in land inertial navigation systems. Therefore, the following three sections present the MEMS IMU characteristics including different performance categories of IMUs; subsequently, we focus on the deterministic and stochastic errors of inertial sensors where the bias, the scale factor and the temperature effect are analysed.

3.3 Inertial Measurement Unit (IMU)

The inertial measurement unit (IMU), which is a fundamental part of the INS, is the device where the inertial sensors are mounted; it provides accelerations and angular rotations along three orthogonal directions with respect to an inertial frame (Fig. 3.5).

These measurements provide information about the motion of the vehicle, that are then corrected and processed by the mechanization stage with the aim to obtain position, velocity and attitude of the host vehicle.

For some of the experiments conducted in this thesis two inertial measurement units based on MEMS technology were used. On one hand, the Microstrain 3DM-GX3-25 IMU that includes a triaxial accelerometer, triaxial gyro, triaxial magnetometer and temperature sensors (Fig. 3.6(a)). It also has analog anti-aliasing filters that are followed by a digital moving average filter. These stages are implemented on a board processor of the IMU 3DM-GX3-25. This IMU offers a range of output data quantities, including fully calibrated inertial measurements: acceleration, angular rate,

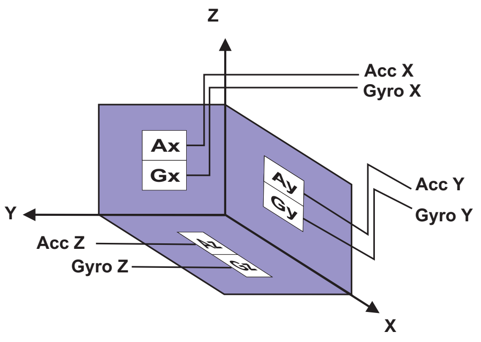


Figure 3.5: Inertial Measurement Unit (IMU).

and magnetic field; it can also output computed orientation estimates: *Pitch*, *Roll*, and *Heading* (*Yaw*) or rotation matrix [89]. The 3DM-GX3-25 MEMS IMU is constituted by surface micromachining gyros, specifically, one is the vibratory rejection ADXRS642 which has a price in the market around $48.82 for quantities between 100 and 499. With respect to the accelerometers, it uses two AD22293 containing surface-micromachined sensors and the cost of a device is less than $10 per unit. The characteristics provided for this IMU (MEMS grade) can be seen in Table 3.2.

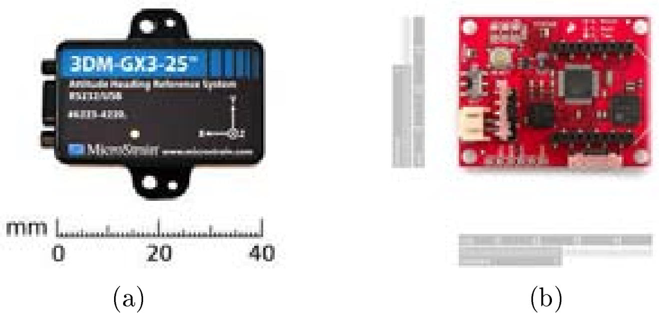


Figure 3.6: (**a**) 3DM-GX3-25 IMU; (**b**) Atomic IMU.

On the other hand, the Atomic IMU from Sparkfun (Fig. 3.6(b)) is equipped with three gyros ST LISY300AL with analog output and a single chip MMA7260Q from Freescale Semicondutors, which allows different sensibility levels. It includes an Atmel ATMega168TM processor running at 10 MHz with 6 dedicated $10-bit$ ADC channels reading the sensors [90]. The price of the inertial sensors is less than $5 per unit. Unfortunately the bias stability is not provided for the Atomic IMU, so it is expected to be very poor (see Table 3.2). Further details about the characteristics of the Atomic IMU provided by the manufacturer can be found in [90].

Table 3.2: MEMS IMUs available in the lab of Microelectronics and Electronics Systems at UAB.

IMU	**Atomic 6DoF**	**3DM-GX3-25**
Price $(k\epsilon)$	0.09494	1.769
Acc technology	MEMS	MEMS
Bias Acc (μg)	–	5000
Gyro technology	MEMS	MEMS
Bias Gyro (deg/h)	–	720
Synchro signals	NO	NO

Given that in this research we adopted low cost MEMS inertial sensors, for the sake of comparison with other IMUs, next section presents the classification of these devices and how they are categorized according to qualities like cost, size and performance.

3.3.1 Classification of IMUs

Typically the IMUs can be classified according to their performance and the type of application where they are implemented. The applications can be divided in consumer-grade, where MEMS inertial sensors are usually low cost and they are suitable for motion sensing, freefall detection, man-machine interface, *etc.* Moreover, tactical IMUs are classified between high performance IMUs and they are employed in unmanned underwater vehicles (UUVs), unmanned air vehicles (UAVs), torpedoes, camera stabilization, land navigation, oil drilling *etc.* Navigation and strategic IMUs are very accurate devices that are used in navigation missiles, autonomous vehicle navigation, stabilization of equipment and weapon platforms. To facilitate the comparison between MEMS IMUs and different IMUs Table 3.3 summarizes the performance characteristics of different inertial sensors, including bias, weight, prices, signal synchronization and errors in position when there is support from an external device like the GPS. Most of the information that is depicted in this table has been collected from [91–95].

Although low cost MEMS IMUs are widely used in several applications due to their small size and low cost, their performance remains as the major limitation. With the purpose of studying specific characteristics that can lead to the improvement of the MEMS IMU performance, the following section is dedicated to explain the errors that

Table 3.3: Inertial Measurement Units categories.

Grade	**Low cost**	**Tactical**	**Navigation**	**Strategic**
Price $(k€)$	< 2	< 15	> 10	> 50
Weight (kg)	< 1	$0.5 - 2$	> 4	> 5
Bias Acc (μg)	> 2000	$100 - 1000$	$10 - 50$	< 1
Bias Gyro (deg/h)	> 10	< 10	< 0.01	< 0.0001
Synchro signals	NO	YES	YES	YES
Positioning error (km/min)	≈ 2	> 0.33	< 0.016	< 0.0005

are typically involved in these devices.

3.3.2 MEMS IMUs Errors

In a low cost INS, the measurement of the accelerometer and gyro sensors is affected by different errors, which can be classified as deterministic and stochastic errors [5]. Fig. 3.7 depicts some of these errors through a simple relationship between IMU physical signal and the sensor output.

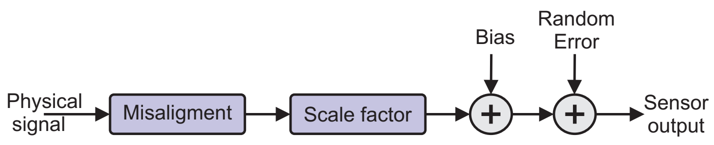


Figure 3.7: Inertial sensor errors including misalignments, scale factors, biases and measurement noise [88].

Deterministic errors are due to manufacturing and mounting defects and can be calibrated out from the data; on the other hand, the stochastic errors are the random errors that occur due to random variations of bias or scale factor over time [5]. There are several errors that affect the MEMS IMU devices and among the most significant are: the misalignment errors that are the result of non-orthogonalities of the sensor axes and are usually treated as deterministic error (see Fig. 3.8). The scale factor that was described in Section 3.2.4, which represents the sensibility of the sensor, and it is the result of manufacturing tolerances or aging; it is usually divided between a linear and a non-linear part, where the linear part is obtained from calibration, while the non-linear is modeled with a stochastic process [13]. In the case of the bias, it is divided between bias turn-on and bias-drift: the bias turn-on is constant, but it varies from turn-on to turn-on and is considered as a deterministic error; the bias-drift presents a random behavior and needs to be modeled with a stochastic process [18]. Regarding the random error (Fig. 3.7), this is an additional signal resulting from noise of the sensor itself or other components that interfere with the signal provided by the sensor; it is also considered part of the stochastic error of the sensor.

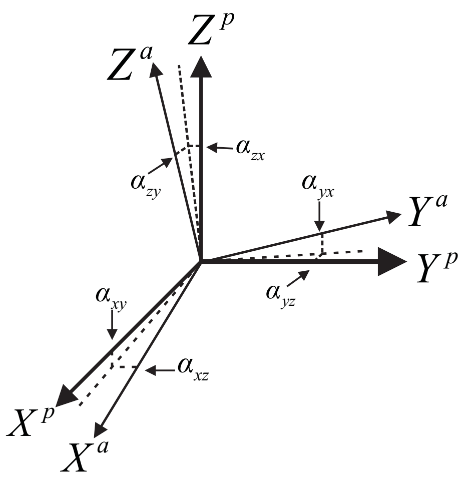


Figure 3.8: IMU misalignments: The nonorthogonal axes of the accelerometers $\{X^a,Y^a,Z^a\}$ can be aligned with the orthogonal body axes $\{X^b,Y^b,Z^b\}$ through the six angles $\{\alpha_{xy},\alpha_{xz},\alpha_{yx},\alpha_{yz},\alpha_{zx},\alpha_{zy}\}$ [88].

As previously mentioned the inertial sensors are also sensitive to environmental factors like temperature, pressure, vibrations, electric and magnetic fields, *etc* [28, 96]. These changes cause the output of MEMS sensors to vary. Since it is very likely to have changes, particularly, in parameters such as scale factor and bias, for the analysis carried out in the next sections the temperature is maintained constant, which will make easier to interpret the results. Since this thesis will not cover all those environmental factors, we will study variations of the scale factor and the bias under different conditions like temperature, especially, temperature dependency of the bias-drift, which will be explained in Chapter 4.

The deterministic errors can be minimized before implementing the mechanization equations by following different procedures in the laboratory. These lab tests are known as the calibration of the IMU and will be presented in the next section.

3.4 Inertial Sensors Deterministic Error

Despite the benefits provided by the GPS/INS integration that were described in Section 2.5.1, the obtained accuracy and convergence time of a GNSS aided INS is highly dependent on the quality of the IMU sensors output [22]. In fact, the ability of the INS to bridge GPS outages depends on the inertial sensor errors, if they are not treated properly it might cause a rapid degradation of the integrated system during GPS outages. Thus, when the GPS is not available, position, velocity and attitude predictions of the vehicle will be strongly affected due to the fact that the errors will be accumulated in the INS. Therefore, the calibration of the IMU is critical for the overall system performance [88].

With the purpose of minimizing deterministic errors such as: turn-on bias, scale factor and non-ortogonalities, the calibration is performed by following a procedure that is detailed below.

3.4.1 Calibration

The calibration can be defined as the process of comparing instrument outputs with known reference information and determining coefficients that force the output to agree with the reference information over range of output values [97]. In order to carry out this process, the measurement of the inertial sensors are described in terms of the parameters associated with the errors as it is stated in Eq. (3.4) and Eq. (3.5) [43,98].

$$\mathbf{f}_{imu} \approx (\mathbf{I} + \mathbf{S}_a + \delta\mathbf{S}_a)\,\mathbf{f} + \mathbf{b}_a + \delta\mathbf{b}_a + \mathbf{w}_a \tag{3.4}$$

$$\boldsymbol{\omega}_{imu} \approx (\mathbf{I} + \mathbf{S}_g + \delta\mathbf{S}_g)\,\boldsymbol{\omega} + \mathbf{b}_g + \delta\mathbf{b}_g + \mathbf{w}_g \tag{3.5}$$

$$\mathbf{b}_{a,g} = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix}_{a,g}, \quad \delta\mathbf{b}_{a,g} = \begin{bmatrix} \delta b_x \\ \delta b_y \\ \delta b_z \end{bmatrix}_{a,g}, \quad \mathbf{w}_{a,g} = \begin{bmatrix} w_x \\ w_y \\ w_z \end{bmatrix}_{a,g},$$

$$\mathbf{S}_{a,g} = \begin{bmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{bmatrix}_{a,g} \quad , \quad \delta\mathbf{S}_{a,g} = \begin{bmatrix} \delta S_{xx} & \delta S_{xy} & \delta S_{xz} \\ \delta S_{yx} & \delta S_{yy} & \delta S_{yz} \\ \delta S_{zx} & \delta S_{zy} & \delta S_{zz} \end{bmatrix}_{a,g} \quad ;$$

where $\mathbf{f}_{imu}$ and $\boldsymbol{\omega}_{imu}$ are the raw measurements provided by the IMU, representing linear acceleration (m/s^2) and angular rate (rad/sec), respectively, $\mathbf{b}_{a,g}$ are vectors comprising biases, while $\delta\mathbf{b}_{a,g}$ are vectors that comprise residual biases, vectors $\mathbf{w}_{a,g}$ correspond to an additional white noise with zero mean. The subscripts a and g are associated with accelerometers and gyros, respectively, while the superscripts $\{x,y,z\}$ are related to the sensor in each direction of the body frame; $\mathbf{I}$ is a 3×3 identity matrix, and $\mathbf{f}$ and $\boldsymbol{\omega}$ are the true linear acceleration and angular rate, respectively. Finally, the terms $\mathbf{S}_{a,g}$ represent the scale factor errors and non-orthogonality errors, which are included in the diagonal and non-diagonal elements, respectively. $\delta\mathbf{S}_{a,g}$ is a matrix with the residual scale factor errors and residual non-orthogonality errors, which are also included in the diagonal and non-diagonal elements, respectively.

Considering Eq. (3.4) and Eq. (3.5), the deterministic components corresponding to the IMU errors are the turn-on bias $(\mathbf{b}_{a,g})$, the misalignment and linear scale factor error $(\mathbf{S}_{a,g})$. These parameters will be obtained off-line (calibration in the lab) and will be introduced to correct the errors of the IMU. Thus, the corrected output of the measurements provided by the sensors would be given by:

$$\mathbf{f}^b \approx (\mathbf{I} + \delta\mathbf{S}_a)\,\mathbf{f} + \delta\mathbf{b}_a + \mathbf{w}_a \tag{3.6}$$

$$\boldsymbol{\omega}_{ib}^b \approx (\mathbf{I} + \delta\mathbf{S}_g)\,\boldsymbol{\omega} + \delta\mathbf{b}_g + \mathbf{w}_g \tag{3.7}$$

Although the deterministic errors are minimized by laboratory calibration, $\mathbf{f}^b$ and $\boldsymbol{\omega}_{ib}^b$ should still be corrected, this is due to the fact that residual errors remain [43]. The residual errors are typically estimated during navigation through the EKF that was described in Section 2.6.3, however the system model needs to be augmented with the IMU error states, that would be related to the residual errors.

The compensation in the loosely-coupled integration is performed by means of the closed loop with INS corrections that was shown in Fig 2.7, so the residual errors are associated with random errors, that according to Eq. (3.6) and Eq. (3.7), they correspond to bias-drift ($\delta \mathbf{b}_{a,g}$), non-linear scale factor and residual non-orthogonality errors ($\delta \mathbf{S}_{a,g}$), whereas ($\mathbf{w}_{a,g}$) is the additive random noise.

These random errors are typically modelled with stochastic models, where the processes that are usually used to compensate them are white noise, random walk, first order Gauss-Markov, autoregressive processes, *etc.* A study of these processes is detailed in Section 3.5.

3.4.2 Multi-position Calibration Method

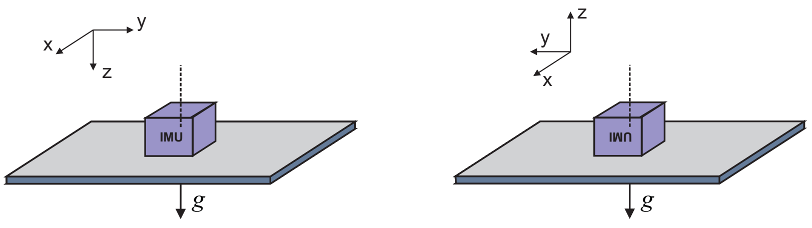
To obtain the parameters that represent these deterministic errors (*i.e.*, $\mathbf{b}_{a,g}$ and $\mathbf{S}_{a,g}$) and minimize their effect during navigation, there are two methodologies. One is the six-position direct method and the second one is the one that we adopted to achieve the laboratory calibration procedure, which El-Diasty in [15] called six-position weighted least squares method. We focused on the last one since it takes into account the non-orthogonality errors. This method consists of placing the IMU in six-positions, *i.e.*, one by each side considering the IMU as a cube.

Since the calibration requires an excitation signal that is used as reference, the gravity is used for the accelerometers and a known angular rate for the gyros. The latter needs a turntable which rotates with a specific angular rate that is used as reference signal.

The following three sections present the laboratory tests that were achieved for the calibration of the IMUs, specifically, two experiments were realized one for accelerometers and one for the gyros.

3.4.3 Accelerometers Calibration

For the accelerometers, the IMU was initially placed on a levelled table using a cube shaped mounting frame, where the excitation signal for the z-axis down is the gravity (g) (Fig. 3.9(a)).



(a) Down position. (b) Up position.

Figure 3.9: Up and down position for calibration of z-axis accelerometer with gravity as reference signal.

In this IMU position the measurements from the accelerometers were taken during 60 *sec* and then the average was computed for each accelerometer $(Av(f_{imu}^{z_{down}})_{x,y,z})$; so the expected observation equations are obtained from Eq. (3.4), where the additive noise and the residual errors are eliminated since their expected values are zeros [15]:

$$\begin{bmatrix} Av(f_{imu}^{z_{down}})_x \\ Av(f_{imu}^{z_{down}})_y \\ Av(f_{imu}^{z_{down}})_z \end{bmatrix} = \left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{bmatrix}_a \right) \cdot \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} + \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix}_a \tag{3.8}$$

The experiment was repeated but this time placing the IMU with z-axis in the up direction (see Fig. 3.9(b)). This procedure was performed for each one of the accelerometers, which led to the following matrix form equation for the six-position static test [15]:

$$\mathbf{X}_a \mathbf{A} = \mathbf{W} \tag{3.9}$$

where

$$\mathbf{X}_a = \begin{bmatrix} S_{xx} & S_{xy} & S_{xz} & b_x \\ S_{yx} & S_{yy} & S_{yz} & b_y \\ S_{zx} & S_{zy} & S_{zz} & b_z \end{bmatrix}_a \tag{3.10}$$

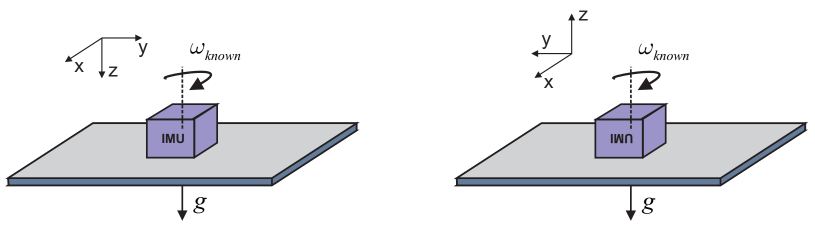
$$\mathbf{A} = \begin{bmatrix} -g & g & 0 & 0 & 0 & 0 \\ 0 & 0 & -g & g & 0 & 0 \\ 0 & 0 & 0 & 0 & -g & g \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{3.11}$$

$$\mathbf{W}^T = \begin{bmatrix} Av(f_{imu}^{x_{up}})_x + g & Av(f_{imu}^{x_{up}})_y & Av(f_{imu}^{x_{up}})_z \\ Av(f_{imu}^{x_{dn}})_x - g & Av(f_{imu}^{x_{dn}})_y & Av(f_{imu}^{x_{dn}})_z \\ Av(f_{imu}^{y_{up}})_x & Av(f_{imu}^{y_{up}})_y + g & Av(f_{imu}^{y_{up}})_z \\ Av(f_{imu}^{y_{dn}})_x & Av(f_{imu}^{y_{dn}})_y - g & Av(f_{imu}^{y_{dn}})_z \\ Av(f_{imu}^{z_{up}})_x & Av(f_{imu}^{z_{up}})_y & Av(f_{imu}^{z_{up}})_z + g \\ Av(f_{imu}^{z_{dn}})_x & Av(f_{imu}^{z_{dn}})_y & Av(f_{imu}^{z_{dn}})_z - g \end{bmatrix}^T \tag{3.12}$$

Now solving Eq. (3.9) by least square we can estimate the calibration parameters for the accelerometers, which are included in matrix $\mathbf{X}_a$.

3.4.4 Gyros Calibration

Although the calibration of navigation and tactical gyroscopes can be performed by using as a reference signal the Earth rotation rate, for low cost inertial sensors this is not valid because the Earth's reference signal can be completely buried in the noise levels [84]. Therefore, it was necessary to mount the IMU on a turntable using a cube shaped frame (Fig. 3.10). In this test the excitation signal was determined by the speed of the motor adapted to the testing table.



(a) Down position. (b) Up position.

Figure 3.10: Up and down position for calibration of z-axis gyro with known angular velocity as reference signal.

Analogous to the accelerometer case, we collected data from the sensors with the z-axis is in the down direction and subsequently the average from the three gyros was estimated:

$$\begin{bmatrix} Av(\omega_{imu}^{z_{down}})_x \\ Av(\omega_{imu}^{z_{down}})_y \\ Av(\omega_{imu}^{z_{down}})_z \end{bmatrix} = \left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{bmatrix}_g \right) \cdot \begin{bmatrix} 0 \\ 0 \\ \omega_{known} \end{bmatrix} + \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix}_g \tag{3.13}$$

After reading the measurements from the six-positions on the testing table, the six observation equations can be written as a single equation in matrix form as it was stated in Eq. (3.9). Then by using the least squares method, the deterministic error coefficient matrix $\mathbf{X}_g$ are estimated.

3.4.5 Thermal Calibration Test

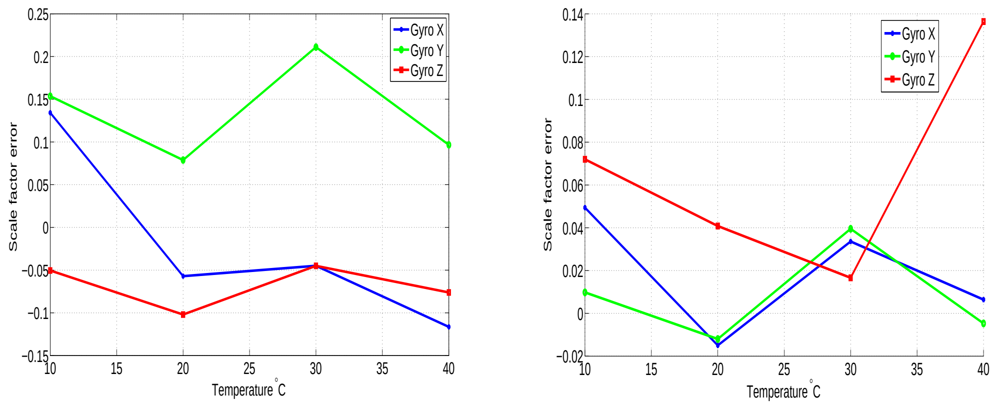
With the purpose of applying the six-position weighted least squares method and determine the variations of the deterministic errors with the temperature, a thermal test was performed in the laboratory of Microelectronics and Electronics Systems at Universitat Autònoma de Barcelona. In this experiment the IMUs available in the lab (See Table 3.2) are adapted to a turntable are enclosed in a thermal chamber, which was configured to remain constant temperatures of 10 $°C$, 20 $°C$, 30 $°C$ and 40 $°C$ degrees, each of them lasting 30 min. The method of recording the data at specific temperature points is called the soak method and is typically used to calibrate the inertial sensors [5]. The IMUs were configured with a sampling frequency of 100 Hz and connected to a battery of 5 v to feed the devices. For the free rotation of the turntable we used a Bluetooth device to send the inertial sensors data to a PC.

The values for accelerometers and gyroscopes that were obtained at a temperature of 20 $°C$ are shown in Table 3.4 and Table 3.5 for the atomic IMU and for the 3DM-GX3 IMU, respectively.

Table 3.4: Accelerometers and gyros deterministic errors for the Atomic IMU at 20 °C.

	Bias (m/s^2)	**Scale Factor Error**	**Non-Orthogonalities**	
Acc X	0.567	0.019	S_{xy}	0.014
			S_{xz}	0.017
Acc Y	0.770	0.025	S_{yx}	0.024
			S_{yz}	0.016
Acc Z	1.357	−0.001	S_{zx}	0.037
			S_{zy}	−0.009
	Bias (rad/s)	**Scale Factor Error**	**Non-Orthogonalities**	
Gyro X	−0.78	−0.06	S_{xy}	−0.07
			S_{xz}	−0.83
Gyro Y	0.22	0.08	S_{yx}	−0.15
			S_{yz}	0.01
Gyro Z	−0.28	−0.10	S_{zx}	0.18
			S_{zy}	0.01

Figure 3.11 shows the variation of the scale factor error for gyroscopes in both IMUs. The Atomic IMU has a more significant change compared to the 3DM-GX3 IMU as it was expected. This temperature variation in the parameters must be taken into account in the error model of the inertial sensors, so it can be compensated to improve performance navigation system.



(a) (b)

Figure 3.11: (**a**) Variation of gyros scale factor error for the Atomic IMU; (**b**) Variation of gyros scale factor error for the 3DM-GX3 IMU.

Table 3.5: Accelerometers and gyros deterministic errors for the 3DM-GX3-25 IMU at 20 $°C$.

	Bias (m/s^2)	**Scale Factor Error**	**Non-Orthogonalities**	
Acc X	-0.018	-0.002	S_{xy}	0.014
			S_{xz}	-0.008
Acc Y	0.008	-0.001	S_{yx}	-0.002
			S_{yz}	-0.026
Acc Z	-0.26	-0.012	S_{zx}	0.027
			S_{zy}	0.020
	Bias (rad/s)	**Scale Factor Error**	**Non-Orthogonalities**	
Gyro X	0.002	-0.015	S_{xy}	0.042
			S_{xz}	0.004
Gyro Y	-0.032	-0.012	S_{yx}	-0.006
			S_{yz}	0.001
Gyro Z	0.013	0.041	S_{zx}	0.003
			S_{zy}	0.002

The following section makes emphasis in the description of the main random errors involved in the inertial sensors, this also presents the state-space form of different stochastic models and how they are adapted into the Kalman filter in order to compensate the random errors. The stochastic processes explained are typically used to model the bias-drift that affects the INS. It is worth pointing out that the identification, analysis and extraction of the parameters of the random errors, specifically for the bias-drift, are performed in Chapter 4.

3.5 Inertial Sensors Stochastic Error

In this section we focus our attention on the stochastic error, specifically, in the bias-drift ($\delta\mathbf{b}_{a,b}$), since the stochastic modeling of this error is a challenging task, not only because of the random nature, but also because it seriously affects the performance of a navigation system. For further details of the impact of this error, refer to [99, 100], where it is showed how the position error grows when different bias-drift are affecting the inertial sensor measurements. Therefore, a suitable estimation of the stochastic model parameters of this error will improve the performance of the INS; as a consequence, the input error to the mechanization stage (Fig. 2.5) can be compensated and, in turn, the position error minimized. Regarding the misalignment errors (*i.e.*, non-diagonal elements of $\delta\mathbf{S}_{a,g}$), they will be not covered in this thesis since the calibration is particularly useful for the removal of them, these should be relatively constant over time assuming a rigid body IMU platform [66].

3.5.1 Noise Terms

This section describes the noise terms that can be identified using Allan variance (AV) and Power Spectral Density (PSD) by fitting straight lines. Most of these errors are shown in Fig. 3.12, where an hypothetical curve of a inertial sensor after computing the PSD is shown. Below the types of noises are summarized as well as Table 3.6 depicts their curve slope and the equivalent coefficient value [9, 11].

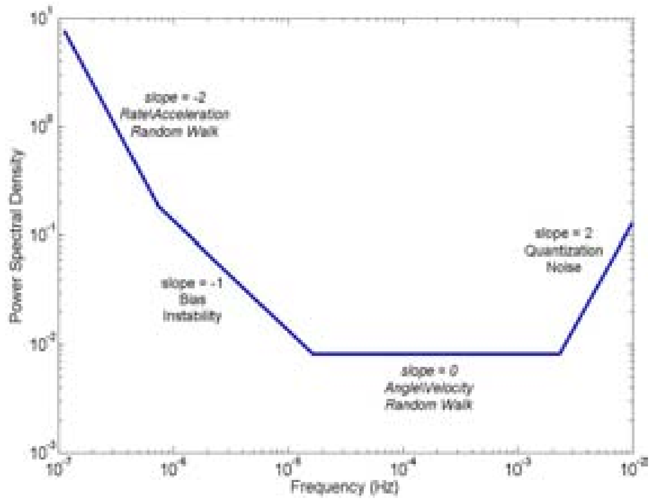


Figure 3.12: Hypothetical PSD in single-sided form of an Inertial Sensor [11].

Rate\Acceleration Random Walk (K)

This is a random process of uncertain origin, possibly a limiting case of an exponentially correlated noise with a very long correlation time. It can contribute to the gyro (rate) or the accelerometer (acceleration). Its coefficient is denoted by K and is represented by a slope of 1/2 in AV and -2 in the PSD.

Table 3.6: Random error for Power Spectral Density analysis [9].

Noise type	**PSD**	**Curve slope**	**Coefficient value**[1]
Quantization (Q)	$(2\pi f)^2 Q^2 T_s$	2	$Q=\frac{\sqrt{S_x(1)/T_s}}{2\pi}$
Angle\Velocity random walk (N)	N^2	0	$N=\sqrt{S_x(f)}$
Correlated (q_c)	$\frac{(q_c T_c)^2}{1+(2\pi f T_c)^2}$	$0, -2$	see [9]
Sinusoidal (Ω_0)	$\frac{\Omega_0^2}{2}[\delta(f-f_0)]$	discrete spectra	see [9]
Bias instability (B)	$\frac{B^2}{2\pi f}, 0$	-1	$B=2.51\sqrt{S_x(1)}$
Rate\Acceleration random walk (K)	$\left(\frac{K}{2\pi f}\right)^2$	-2	$K=2\pi\sqrt{S_x(1)}$
Rate ramp (R)	$\frac{R^2}{(2\pi f)^3}$	-3	$R=15.75\sqrt{S_x(1)}$

[1] $S_x(1)$ power spectral density of x evaluated at $1Hz$

Bias Instability or Flicker Noise (B)

The origin of this noise is the electronics or other components that are susceptible to random flickering [9]. According to Eq. (3.2), in a transistor it is associated with the contamination in the processing of materials. Although it can be reduced by cleanliness practices, it still persists; it occurs at low frequencies and it is temperature and frequency dependent [101]. Flicker noise is widely found in nature, occurring in physics, biology, astrophysics, economics, psychology and even in inertial navigation [102–104], however, no generally recognized physical explanation has been proposed [105]. The flicker noise is represented by the flat region in AV and a slope of -1 in the PSD (see Fig. 3.12).

Angle\Velocity Random Walk (N)

Angle (gyros) and velocity (accelerometer) random walk is characterized by the white noise of the inertial sensors; this noise presents high-frequency terms that have correlation time much shorter than the sample time [9]. It has a slope of $-1/2$ in AV and 0 in the PSD.

Quantization Noise (Q)

The quantization noise is one of the errors introduced when the continuous signal is approximated by a discrete signal. That noise is caused in the analog-digital converter where a continuous value is associated with a level of quantization with finite word length, the amount of levels depends on the resolution of the converter. This noise is represented by a slope of -1 in AV and 2 in the PSD.

Drift Rate Ramp (R)

The error terms considered so far are of random character. However, the drift rate ramp for long, but finite time intervals is more a deterministic error rather than a random noise [9]. The PSD method cannot distinguish between the rate random walk and the rate ramp. Thus, the rate ramp must be removed before applying the PSD method [106]. However, It can be identified in a log-log AV plot with a slope of 1.

The following errors are not as common in the inertial sensors as the five previous noise terms, however, they can be identified using the Allan variance method.

Exponentially Correlated (Markov) Noise (q_c)

This noise is characterized by an exponential decaying function with a finite correlation time (T_c). In a log-log AV plot, for a time cluster much longer than T_c time its behaviour is the same as the angle random walk and for a time cluster much smaller than T_c its behaviour is the same as the rate random walk.

Sinusoidal Noise (Ω_0)

This noise is resulted from periodic environmental changes and its PSD is characterized by one or more distinct frequencies. A low-frequency source could be the slow motion of the test platform due to periodic environmental changes. This error is represented in AV with multiple frequency sinusoids, where the amplitudes of consecutive peaks fall off rapidly and may be masked by higher order peaks of other frequencies making observation difficult [9].

3.5.2 Inertial Sensor Error Models

In the previous section different random noises that can be identified with AV and PSD were presented. This section will described the stochastic processes that are usually used to model some of these noise terms and also the state-space representation that is implemented in the EKF for modelling the random errors.

White Noise (WN)

White noise is defined to be a stationary process having a constant spectral density function [62]. This noise can be considered as a signal containing all the frequency components, similar to white light (hence the name) that includes all the visible frequencies. It has the particularity that its output at any instant of time is independent of previous values, therefore its autocorrelation is characterized by a Dirac delta. Although white noise is an idealized concept it serves as approximation to situations in which a disturbing noise is wideband compared with the bandwidth of a system [64]. It can be used to model the random noise of an inertial sensor showed in Fig. 3.7 that is part of the stochastic error, it is associated with the noise term angle/velocity random walk (N) that is obtained with Allan variance or PSD (see Sections 4.3.3 and 4.3.4). Additionally, a number of random processes can be generated by passing white noise through a suitable filter [64], which is appropriate for KF since Gaussian noise disturbances for the process noises and measurement noises can be represented by white noise. Moreover, the models of the INS residual error can be implemented through a certain shaping filter (*i.e.*, a linear dynamic system that can be adapted into the KF) that uses as input white noise to yield an output of time-correlated (or colored) noise, which will change the correlation characteristics of the white noise [107].

Random Constant (RC)

The random constant is a non-dynamic quantity with a fixed, albeit random, amplitude [64]. The continuous and discrete random constant are described by Eq. (3.14) and Eq. (3.15), respectively.

$$\dot{x} = 0 \tag{3.14}$$

$$x_{k+1} = x_k \tag{3.15}$$

Non-orthogonalities and turn-on bias of sensor triads can be dealt with random constants, bias-drift of the inertial sensors can also be considered as random constant, if the operation time is very short. If the operation time is very long even if the state is constant, it will be preferable to add noise intentionally, which results in a random walk process [51], although this is not suitable for low cost inertial sensors.

Random Walk (RW)

The random walk process results when uncorrelated signals are integrated, for instance when white noise is integrated. Its continuous and discrete representation are

$$\dot{x} = w \tag{3.16}$$

$$x_{k+1} = x_k + w_k \tag{3.17}$$

where w is a white noise with noise covariance $q_k = q(t_{k+1} - t_k) = q\delta t$. The uncertainty of the random walk increases with time, therefore it is a non-stationary process [64], however, it can be considered stationary within small time intervals [108]. Since an INS integrates signals from accelerometers and gyros, the white noise components are integrated, this will increase the uncertainty of velocity and attitude [51]. Parameters that represent random walk are related with the rate/acceleration random walk (N) described in Section 3.5.1, they are obtained through the Power Spectral Density or Allan Variance analysis, which will be explained in Chapter 4.

Random Ramp (RR)

Some random errors exhibit a time-growing behavior, for instance the drift rate ramp (R) described in the previous section; in these cases the random ramp which grows linearly with time can be used to describe them [64]. The growth rate of this function is a random quantity with a given probability density, the states vector differential equation and its corresponding discrete representation are described by two variables [64]:

$$\dot{x_1} = x_2 \tag{3.18}$$

$$\dot{x_2} = 0 \tag{3.19}$$

$$x_{k+1} = x_{1k} + (t_{k+1} - t_k)x_{2k} \tag{3.20}$$

$$x_{2k+1} = x_{2k} \tag{3.21}$$

First-order Gauss-Markov Process

Gauss-Markov (GM) random processes are stationary processes that have exponential autocorrelation functions [15]. This process is important because it is able to represent a large number of physical processes. First order GM process is one of the most common processes for modeling random errors of the inertial sensors when the variation sensor error is slow, in high-end sensors this can be done with a large correlation time T_c. For the first order GM process the continuous model is described by the following equation:

$$\dot{x} = -\frac{1}{T_c}x + w \tag{3.22}$$

where x is a random process with zero mean, correlation time, T_c, and driven noise, w. The corresponding discrete time equation can be written as:

$$x_k = e^{-\Delta t/T_c}x_{k-1} + w_k \tag{3.23}$$

where Δt is the sampling time and w_k is a white noise with noise covariance:

$$\sigma^2_{w_k} = \sigma^2_{x_k}\left(1 - e^{-2\Delta t/T_c}\right) \tag{3.24}$$

where $\sigma^2_{x_k}$ is the covariance of the process.

Once the correlation time (T_c) and the driven noise variance ($\sigma^2_{x_k}$) are obtained, the model of the first order GM process can be implemented with Eq. (3.23). According to

Eq. (3.23), if $T_c = \infty$, then Gauss-Markov model becomes a random walk model. On the other side, if $T_c = 0$, it becomes a white noise. The first order Gauss-Markov model parameters can be estimated using least squares fitting of the estimated autocorrelation values for gyro and accelerometer measurements [29]. Also Allan variance can be used to determine T_c and the variance of the driven noise w_k, a description of how to determine these parameters can be found in [4, 9, 13, 26].

The first order Gauss-Markov process can be used to model the flicker noise, this is because it is sometimes approximated as the combination of several exponentially correlated noise terms as it is stated in [26]. The flicker noise can be approximated over a bandwidth given as a sum of exponentially correlated noises [11], so the sequence of processes formed by many first order GM process can be expressed by:

$$y_k = \sum_{i=1}^{n} x_k^i \tag{3.25}$$

where y_k would be the flicker noise model, x_k would be given by Eq. (3.23) and n would be the number of GM processes to be added. The correspondent power spectral density of the exponential correlated (Markov) noise x_k is denoted by $S_x(f)$ [63, 64]:

$$S_x(f) = \frac{2\sigma_x^2/T_c}{(2\pi f)^2 + (1/T_c)^2} \tag{3.26}$$

In Eq. (3.25) it should be considered that processes x_k^i at different T_c's are independent [109].

The flicker noise is very common in inertial sensors, but why can it be approximated as a sum of first order GM processes? This is because this noise was discovered by Johnson [110] while doing experiments to study the shot noise in vacuum tubes, then Schottky [111] attempted to describe it mathematically with a Lorentzian spectral density, which have similar representation to the power spectral density of a exponential Markov correlated noise (Eq. (3.26)), *i.e.*, a white noise for low frequencies, a random walk for high frequencies and between them there is a slope of -1 that can represent the flicker noise (see Fig. 3.13).

Later Bernamont [112] proposed a superposition of these processes to represent flicker noise and more recently Erland and Greenwood [113] considered a collection of

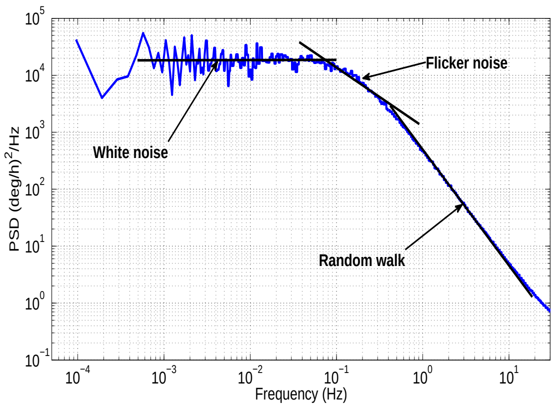


Figure 3.13: Hypothetical power spectral density of a first order Gauss-Markov process for a Gyro.

first order autoregressive processes. In the case of inertial sensors, the flicker noise has already been modelled as a sum of first order GM processes, for further details see [6, 8, 27, 28].

Combinations of Random Processes

A typical bias-drift of a inertial sensor can be represented by a combination of different random processes, such as white noise (WN), random walk (RW) and first order GM processes. These processes can be added into the KF by writing them in a state-space model. According to the previous definitions, a random process that combines WN, RW and first order GM can be generated using the following discrete time-invariant state-space model:

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}_k = \begin{pmatrix} (1-\beta\Delta t) & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}_{k-1} + \begin{pmatrix} \sigma_{GM}\sqrt{(1-e^{-2\Delta t/T_c})} \\ \sigma_{RW}\sqrt{\Delta t} \end{pmatrix} w_k \tag{3.27}$$

$$\delta b_k = \begin{pmatrix} 1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}_k + \left(\sigma_{WN}/\sqrt{\Delta t}\right) v_k \tag{3.28}$$

where σ_{WN} is the standard deviation of the white noise process, σ_{RW} is the standard deviation of the random walk process, β and σ_{GM} are the inverse of the correlation time (T_c) and the covariance of the first order GM process, respectively. δb_k is the result of combining WN, RW and first order GM. Eqs. (3.27) and (3.28) are easily adapted into the KF equations, since they are represented in state-space form. In this example, the bias-drift (δb) would be modeled by the combination of three noises, *i.e.*, $\delta b = WN + 1^{st}GM + RW$.

Autoregressive (AR) Process

An AR process is a time series produced by linear combination of past values, which can be described by the following linear equation [114]:

$$x(n) = -\sum_{k=1}^{p} \alpha_k x(n-k) + \beta_0 w(n) \tag{3.29}$$

where $x(n)$ is the process output, which is a combination of past outputs, plus a white noise, $w(n)$, with standard deviation, β_0; p is the order of the AR process and α_k are the model parameters.

In order to include the AR process in the EKF transition matrix, it is necessary to express Eq. (3.29) in state-space form. If we consider a third order AR process, the corresponding state-space form can be expressed as follow [62]:

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}_n = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -\alpha_3 & -\alpha_2 & -\alpha_1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}_{n-1} + \begin{pmatrix} 0 \\ 0 \\ \beta_0 \end{pmatrix} w(n) \tag{3.30}$$

This represents the AR model in state-space for one of the inertial sensors. It should be noted that if the order of the AR model increases by one, the variables in the state vector of the Kalman filter will increase by six, since this model is applied to each axis of inertial sensors.

The stochastic processes that are used to model the inertial sensors bias-drift are augmented into the Kalman filter, as was explained in this section. In order to obtain the parameters of each stochastic process, an analysis of the sensors data needs to be done. The methods addressed to get these parameters are discussed in Chapter 4, as well as the experimental analysis of each method for low cost IMUs.

For more details of higher orders of GM processes and others stochastic models see [15, 41, 64, 107].

Chapter 4

Stochastic Modelling of MEMS Inertial Sensors

4.1 Introduction

This chapter is divided in two parts. The first part is focused on the identification and modeling of the bias-drift stochastic error applying the most used techniques currently available to analyze these random processes and we present the mixture of the wavelet de-noising technique and Allan variance.

The second part of the chapter begins in Section 4.5, where we continue studying the bias-drift stochastic modelling of inertial sensors and we explain the constrained non-linear fitting (NLF). In order to assess the performance of the NLF, a simulation analysis is achieved by generating noise sources that typically influence the inertial sensors. Then, we compared the NLF with the (EM) [21, 115] that is a recent method that deals with complex noise structures.

4.2 Identifying and Extracting Stochastic Model Parameters

The stochastic modeling of the inertial sensors is a challenging task that in most practical cases, it is performed by tuning the GPS/INS Extended Kalman Filter, which is often sensitive and difficult, by using sensors available specifications, but low cost sensors do not provide enough information to develop this sort of models, or by experience [116]. Therefore, there are different works that have been achieved in order to obtain a suitable estimation of the stochastic model parameters [7, 12, 19, 20, 24, 117]. In this section, we describe the most used methods for noise identification and extraction of the noise parameters for stochastic modeling of inertial sensors. Additionally, an introduction to the wavelet de-noising technique is presented at the end of the section.

4.2.1 Autocorrelation

The autocorrelation is one of the most important functions to describe a random process due to the fact that it indicates the similarity degree of a random signal between two instants of time t_1,t_2. For a random process $x(t)$ it is defined as:

$$R_{xx}(t_1,t_2) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x_1 x_2 f_x(x_1 x_2; t_1 t_2) dx_1 dx_2 \tag{4.1}$$

where $x_1 = x(t_1)$, $x_2 = x(t_2)$ and f_x is the joint density function. The double integral arises as consequence that at the time instant t_1 the amplitude of the signal $x(t_1)$ is associated with the random signal x_1 and at the time instant t_2 the amplitude of the signal $x(t_2)$ is associated with x_2.

The autocorrelation function has been used in previous works to analyze the stochastic error of the inertial sensors [19, 20], and also to obtain the parameters for modeling using the first order Gauss-Markov (GM) process. As it was mentioned in the previous chapter, this process seems to fit a large number of physical processes with reasonable accuracy.

For a random process, x, with zero mean, correlation time, T_c, and driven noise, w, the first order Gauss-Markov (GM) process is described by Eq. (3.22). The parameters needed to implement this process can be extracted from its autocorrelation function (Fig. 4.1), which is given by:

$$R_{xx}(\tau) = \sigma^2 e^{-\beta|\tau|} \tag{4.2}$$

where the correlation time is $T_c = 1/\beta$ and σ^2 is the variance of the process at zero time lag ($\tau = 0$). The most important characteristic of the first order GM process is that it can represent bounded uncertainty, which means that any correlation coefficient at any time lag, τ, is less or equal the correlation coefficient at zero time lag, $R_{xx}(\tau) \leqslant R_{xx}(0)$ [64].

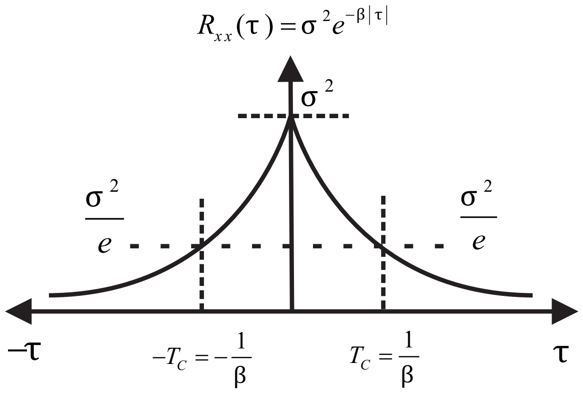
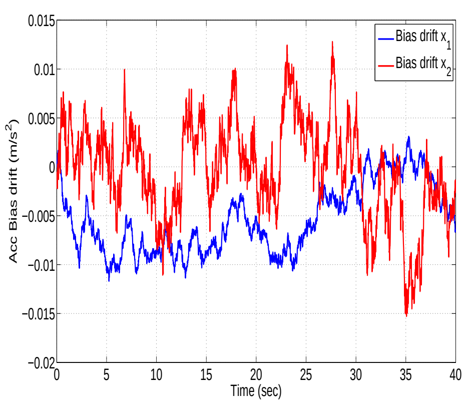


Figure 4.1: The autocorrelation function of the first order Gauss-Markov process.

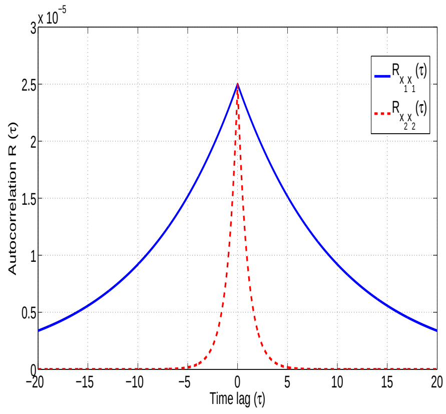
An example of two first order GM processes that may be part of the bias-drift of the inertial sensors are depicted in Fig. 4.2(a), where two bias-drift with different characteristics can be seen (*i.e.*, T_c and σ^2), process x_2 presents faster variations than process x_1. Computing their autocorrelation functions and drawing the result as it is shown in Fig. 4.2(b), it can be observed that a random signal with fast variations (x_2) has a greater value of autocorrelation over short time lag (τ) than over large time lag because of its similarity in short periods of time, while the autocorrelation curve of a process with slow variations (x_1) decreases slowly which indicates that there are similarities even when time lag is large. In MEMS inertial sensors more than two of this noises can be found making complex noises structures.

One of the limitations of this method is that an accurate autocorrelation curve from experimental data is rarely done, due to the fact that the data collected is limited and finite. As it is discussed in [62], the accuracy of the autocorrelation depends on the recorded length data.

In [19, 20, 117], it was shown that the autocorrelation function of experimental inertial sensor data might not be as a first order GM process, which is equivalent to a first order autoregressive process. This means that only a first order autoregressive



(a)



(b)

Figure 4.2: (**a**) Accelerometer bias-drift modeled with two different first order Gauss Markov processes; (**b**)Autocorrelation curve of two first order Gauss Markov processes.

process may not be adequate to model the bias-drift behavior that affects the performance of the inertial navigation system. In fact, in most of the cases when low cost IMU are used, the shape of the autocorrelation follows higher order Gauss-Markov processes. As a consequence, higher orders of autoregressive processes are more appropriate to model inertial sensors stochastic errors [107]. Despite this, the autocorrelation analysis can be useful to determine the correlation grade of the underlying random processes that affect the sensors and, also, if the uncorrelated noise can be highly attenuated after filtering the sensor signal. This issue will be discussed in Section 4.3.1.

4.2.2 Autoregressive Processes

To avoid the problem of inaccurate modeling of inertial sensor random errors, as in the case with the low-precise autocorrelation function, described in previous section, another method, which was introduced in [20], can be applied. There are different works where the autoregressive (AR) models have been evaluated, some of them are well detailed in [19, 20, 114, 117].

Although first order Gauss-Markov (GM) process has been very useful for modeling random errors of inertial sensors, better stochastic modeling can be achieved by modeling these errors as higher order AR models [19]. In addition, the autocorrelation of the random error for MEMS sensors often seems to follow a higher order GM process, which can be modeled using an appropriate AR model.

According to Eq. (3.29), this is assumed that the coefficients (β_0, α_k) are computed so that the linear system is stable, making the model stationary [64]. It should be noted in Eq. (3.29) that if $p = 1$, then the AR process approximates first order GM processes. On the other hand, if $p = 1$ and $\alpha_1 = -1$, it becomes a random walk (RW), and if $\alpha_1 = 0$, it would be a white noise (WN). The coefficients of this process are estimated by Burg's method, since it overcomes some of the drawbacks of other methods by providing more stable models and improved estimates with shorter data records [118].

In this thesis, we focus on AR models up to the third order, since a higher order would increase the computational load and might result in unstable solutions [20]. This method is usually used after applying wavelet de-noising to the static inertial sensor data, which is explained in Section 4.2.5.

4.2.3 Power Spectral Density

Power spectral density (PSD) is an important descriptor of a random process, because it provides information of the signal that is not easy to extract from the time domain. The PSD is related to the autocorrelation function with:

$$S_x(jw) = \mathcal{F}\left[R_{xx}(\tau)\right] = \int_{-\infty}^{\infty} R_{xx}(\tau)e^{-jwt}d\tau \tag{4.3}$$

where, $S_x(jw)$ is the power spectral density of the process, x, $\mathcal{F}\left[\cdot\right]$ indicates Fourier transform, and $R_{xx}(\tau)$ is the autocorrelation of the process, x [11].

Basically, the PSD is used to identify the stochastic errors of the inertial sensors (*i.e.*, bias-drift) from the frequency components, and the parameters obtained from the PSD are eventually used in the stochastic model of the INS.

Fig. 3.12 depicts a hypothetical inertial sensor PSD in single-sided. According to this curve, the noise sources might be identified considering the slopes, *i.e.*, a slope of -2 represents the rate\acceleration random walk noise for gyro and accelerometer, respectively. Obviously, the number of random noises that might be present in the curve depends on the type of sensors. Table 3.6 shows the PSD function associated with different random noises, it includes the corresponding curve slope and coefficient value for various noises terms depicted in Fig. 3.12. The noise terms that can be identified with the PSD are well detailed in [9, 11, 12].

So far, we have presented the autocorrelation, where the stochastic model parameters are extracted from the autocorrelation curve, the autoregressive processes that estimates the coefficients of an AR model applying Burg's method over the de-noised sensor data and the power spectral density that identifies the noise terms based on the slopes in a log-log PSD curve. The following section will describe the Allan variance technique, which is similar to the PSD, but in the time domain.

4.2.4 Allan Variance

The Allan variance (AV) is a time domain analysis technique originally developed to study the frequency stability of oscillators [17]. More recently, this has been successfully applied to the modeling of inertial sensors [14–17, 24], and two key documents to determine the characteristics of the random processes that give rise to the measurement noise of the sensors using this technique are [9, 12]. As such, AV helps in identifying the source of a given noise term in the observed data [9].

The Allan variance is estimated as follows:

$$\sigma^2(T) = \frac{1}{2T^2(N-2n)} \sum_{k=1}^{N-2n} (\theta_{k+2n} - 2\theta_{k+n} + \theta_k)^2 \tag{4.4}$$

where T represents the correlation time, or cluster time, *i.e.*, the time associated with a group of n consecutive observed data samples, N is the length of the data that will be analyzed and θ is the output velocity, in the case of the accelerometers, and output angle, in the case of the gyros; these measurements are made at discrete times from the inertial sensors.

The basic idea to estimate the AV is to take a long sequence of data (N), where the IMU is in a static condition. After having removed the turn-on bias from the gyros' and accelerometer's stored data, the output of the inertial sensor is integrated to get θ. Thus, the AV can be computed through Eq. (4.4).

In AV, the uncertainty in the data is assumed to be generated by noise sources of specific character, as for instance, rate random walk, angle random walk, bias instability, *etc.* In order to obtain the covariance of each noise source affecting the sensor output, it is necessary to analyze the computed AV result by Eq. (4.4). This is usually achieved by plotting a log-log AV curve, as depicted in Fig. 4.3, from which the covariance values for each error can be extracted doing a similar analysis to the one performed with the PSD curve.

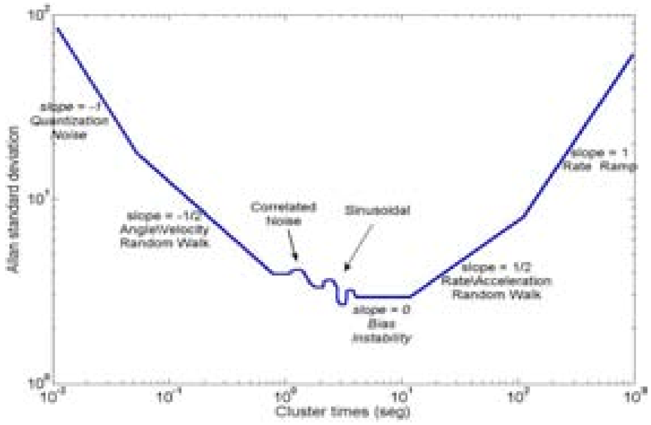


Figure 4.3: Hypothetical Allan variance (AV) of an inertial sensor [9].

Table 4.1 summarizes the AV function for different segment of the curve related to various noise terms, it includes the curve slope and coefficient value representing the random noises shown in Fig. 4.3.

Table 4.1: Random error for Allan variance analysis [9].

Noise type	**AV**	**Curve slope**	**Coefficient value**
Quantization (Q)	$\frac{3Q^2}{T^2}$	-1	$Q = \sigma(\sqrt{3})$
Angle\Velocity random walk (N)	$\frac{N^2}{T}$	$-\frac{1}{2}$	$N = \sigma(1)$
Correlated (q_c)	$\frac{q_c^2 T}{3}, \frac{(q_c T_c)^2}{T}$	$\frac{1}{2}, -\frac{1}{2}$	$q_c T_c = \sigma(1)$, $q_c = \sigma(3)$
Sinusoidal (Ω_0)	$\Omega_0^2 \left(\frac{\sin^2 \pi f_0 T}{\pi f_0 T}\right)^2$	$1, -1$	see [9]
Bias instability (B)	$(0.664B)^2$	0	$B = \frac{\sigma(T_0)}{0.664}$
Rate\Acceleration random walk (K)	$\frac{K^2 T}{3}$	$\frac{1}{2}$	$K = \sigma(3)$
Rate ramp (R)	$\frac{(RT)^2}{2}$	1	$R = \sigma(\sqrt{2})$

The AV obtained from Eq. (4.4) is related to the two-sided PSD by:

$$\sigma^2(T) = 4\int_0^\infty df \cdot S_x(f) \cdot \frac{\sin^4(\pi f T)}{(\pi f T)^2} \tag{4.5}$$

where $S_x(f)$ is the PSD of the random process, x, written in Eq. (4.3).

An interpretation of Eq. (4.5) is that the Allan variance is proportional to the total noise power of the sensor output when passed through a bandpass filter with transfer function $\sin^4(\pi f T)/(\pi f T)^2$. This filter depends on T, which suggests that different types of random processes can be examined by adjusting the correlation time (T). Thus, the AV provides a mean of identifying and quantifying various noise terms that exist in the data [9].

Computation of AV needs a finite number of clusters that can be generated from the raw data measurements of the sensors. Depending on the size of these clusters, AV can identify any noise term that is affecting the data sensor. It is important to mention that the estimation accuracy of the AV for a given T depends on the number of independent clusters within the data set [9]. The bigger the number of independent clusters, the better the estimation accuracy. It has been described in [12] that the percentage error of AV, $\sigma(\delta)$, in certain $\sigma(T)$ and with a data set of N points is given by:

$$\sigma(\delta) = \frac{1}{\sqrt{2\left(\frac{N}{n} - 1\right)}} \tag{4.6}$$

where N is a set of data points collected from the sensors and n is the number of data points of the cluster in estimating $\sigma(T)$. Eq. (4.6) shows that the estimation errors in the region of short cluster length, T, are small, as the number of independent cluster in these regions is large. On the other hand, the estimation error in the region of long cluster length, T, are large, as the number of independent clusters in these regions is small [9, 12].

For example, if 360000 data points are collected from an inertial sensor and if we want to compute the estimation accuracy of the AV for a bias instability (Fig. 4.3) with a characteristic time of 10 min, we will have 60000 *points* with a sampling frequency of the sensor equal to 100 Hz. According to Eq. (4.6), the percentage error of the AV for this random process would be approximately 32%.

The following section presents wavelet de-noising technique, which will be combined with autoregressive processes, as well as Allan variance.

4.2.5 Wavelet De-Noising

The Discrete Wavelet Transform (DWT) is a widely used technique in digital signal processing, and one of its characteristics is that it allows us to do a multiresolution analysis. Basically, when DWT is applied to a signal, $x(n)$, this is filtered with low-pass, $h_0(n)$, and high-pass, $h_1(n)$, filters (the coefficients of each filter depend on the wavelet function). Subsequently, a sub-sampling by two is done. Wavelet multiple levels of decomposition (LOD) are obtained by repeating this stage on the sub-sampled output of the low-pass filter, $h_0(n)$, as shows Fig. 4.4. After applying DWT, the spectrum of the signal, $x(n)$, is divided into different sub-bands with different resolutions, as can be seen in Fig. 4.5. The most significant coefficients of the signal, $x(n)$, are the approximations (A_k). This means, that they have the majority of the information of the signal, while the high-frequency components are know as details (D_k), and as its name says, they are details of the signal, $x(n)$, that in most cases, are high-frequency noise components.

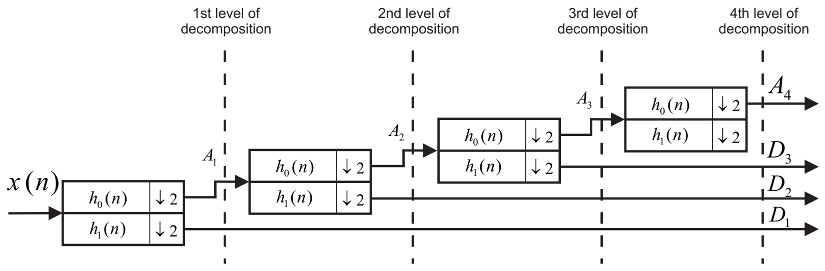

1st level of decomposition
2nd level of decomposition
3rd level of decomposition
4th level of decomposition
$x(n)$
$h_0(n)$ $\downarrow 2$
$h_1(n)$ $\downarrow 2$
A_1
A_2
A_3
A_4
D_3
D_2
D_1

Figure 4.4: Filter banks of the discrete wavelet transform.

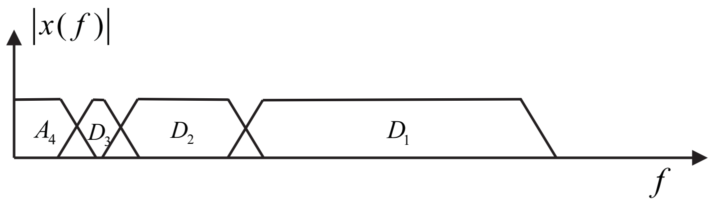


Figure 4.5: Band frequency distribution after applying four levels of decomposition.

Moreover, wavelet de-noising takes advantage of the sub-band decomposition performed by the DWT and removes the noise by eliminating the frequency components that are less relevant; in general, this procedure is called wavelet de-nosing and is well described in [22,107,119,120].

This technique is the current state-of-the-art technique used in the accuracy enhancement of inertial sensors [20, 22, 23, 25]. Since inertial sensors are composed by long-term and short-term noises, wavelet de-noising can be applied in order to remove part of the high-frequency components (short-terms noises). Although wavelet de-noising of INS sensors has had limited success in removing both noise components, it has been combined with AR processes and the autocorrelation function by using the inertial sensor measurements in static conditions. Basically, when it is applied in the autocorrelation method, the uncorrelated noise is removed using wavelet de-noising in order to obtain a smooth autocorrelation function that can be associated to a stochastic process. In the case of the AR process, wavelet de-noising is applied, and then the AR coefficients are estimated from the residual noise.

Wavelet de-noising might be used to remove long-term noises (low-frequency) by increasing the level of decomposition that at the same time, increases the number of frequency bands that can be de-noised. However, in land vehicle applications, these low-frequency components consist not only of long-term noises, but also of vehicle motion dynamics. Since wavelet de-noising can be used to remove the high-frequency components and the AV method can be used to model the long-term noises without removing the vehicles motion, these two methods are combined in order to enhance the INS accuracy. The mixture between these two techniques is addressed in the following section, as well as the experimental analysis for each method explained.

4.3 Experimental Analysis

In order to evaluate and compare the previous methods, the static data for analysis was obtained from the IMU 3DM-GX3-25 MEMS grade of MicroStrain (Fig. 3.6(a)). The IMU was configured with a sampling frequency of 100 Hz, and the second moving average filter stage implemented in the microcontroller was adjusted with a filter width of 15; this means an attenuation of 14.16% at 20 Hz; for further details of this digital filter, which is embedded on the IMU, see [121].

The characteristics provided by the manufacturer can be seen in Table 3.2. The test for static analysis was conducted in a room temperature at the Navsas laboratory, Politecnico di Torino [122]. Seven *hours* of static data were collected in order to analyze the inertial sensors data with the methods that were explained previously. The following sections provide details of the analysis achieved for this IMU data.

4.3.1 Autocorrelation Analysis

After the seven hour-length data collecting, we used the autocorrelation method to achieve the analysis of the random errors that affect the accelerometers and gyroscopes of the IMU. Nevertheless, before processing the raw samples, we removed the turn-on bias for each sensor. Then, the high-frequency terms were attenuated by applying the wavelet de-noising technique. The idea in this step is to minimize the uncorrelated noise that is present in the sensors. Subsequently, the autocorrelation is calculated (Fig. 4.6(b)), and the corresponding parameters should be extracted from the curve. In the case of the first order GM process, they would be stated as T_c and σ, respectively.

Fig. 4.6(a) depicts the normalized autocorrelation function of the accelerometers before applying de-nosing, while Fig. 4.6(b) corresponds to the autocorrelation curve after de-noising with six levels of decomposition using Daubechies 4 as the wavelet function. This autocorrelation shows clearly that the residual noise of the x-axis accelerometer after applying wavelet de-noising is still dominated by terms that are uncorrelated. With respect to the other two-axes accelerometers (*i.e.*, y-axis and z-axis), their correlations seems to have more correlated terms than in the x-axis accelerometer case, so a high order autoregressive model could be used to model their residual noise, since the autocorrelation curve is similar to the curve of high order AR processes (see [20, 64]).

The same wavelet de-noising procedure was repeated to analyze the gyroscope's characteristics. The results are depicted in Fig. 4.7(a). This curve shows that the signal for the three gyroscopes is mainly dominated by short-term noises (high-frequency components), which are related to white noise. After applying wavelet de-noising with six levels of decomposition using Daubechies 4 as the wavelet function (Fig. 4.7(b)); the autocorrelation shows that the three gyros have similar characteristics, and although part of the uncorrelated noise was removed, the remaining signal for the gyroscopes still has a representative white noise component.

In the case of inertial sensors based on MEMS technology, the assumption that the stochastic error follows a first order Gauss-Markov process is not valid in most of the situations. This can be visible by comparing Fig. 4.1 with Figs. 4.6(b) and 4.7(b), where it can be seen that they are different from the autocorrelation function of the first order Gauss-Markov process. This is because these sensors are composed by more complex noise types, and first order Gauss-Markov is only a rough approximation of this complex structure of noises. Nonetheless, for the sake of comparison with the different models and to validate this analysis, a first order AR process is also assessed in Section 4.5.3.

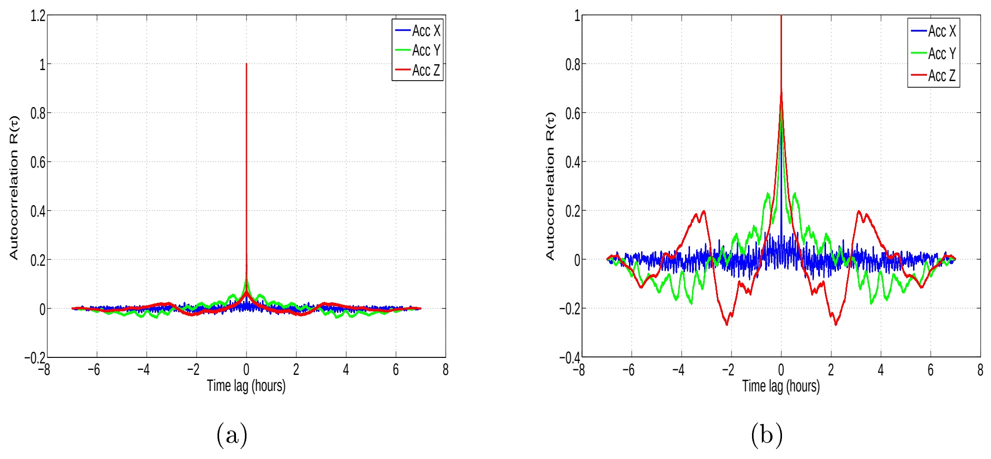


Figure 4.6: (**a**) IMU 3DM-GX3-25 autocorrelation for accelerometers; (**b**) IMU 3DM-GX3-25 autocorrelation for accelerometers after applying wavelet de-noising with six levels of decomposition (LOD).

It is worth mentioning that the uncorrelated noise could be minimized by applying more levels of decomposition during the wavelet de-noising procedure, or a very high order autoregressive model could be used to create the model. However, the use of such a complex AR model in the integration filter would drastically increase the matrices sizes, as well as the computational burden. In addition, due to the fact that the autocorrelation has some other limitations (see Section 4.2.1), the method that will be analyzed in the following section is more appropriate to model higher order autoregressive processes.

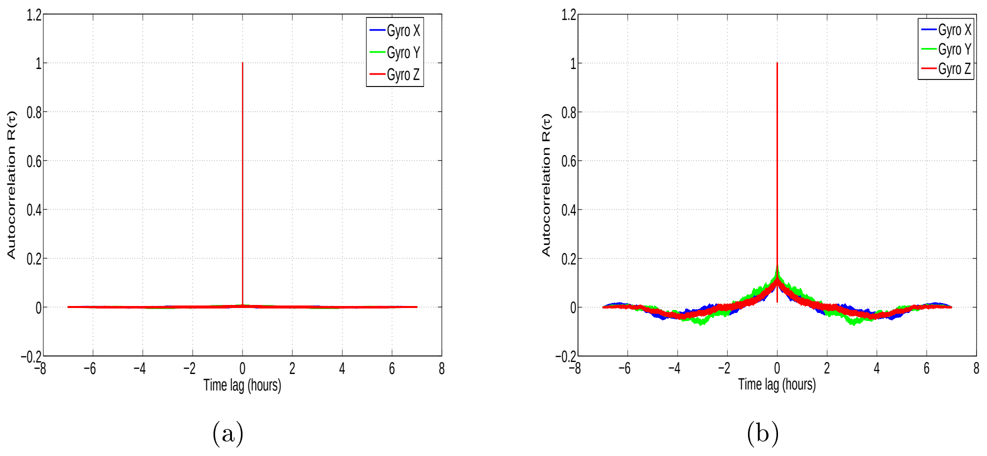


Figure 4.7: (**a**) IMU 3DM-GX3-25 autocorrelation for gyros; (**b**) IMU 3DMGX3-25 autocorrelation for gyros after applying wavelet de-noising with six LOD.

4.3.2 AR Models

Since the autocorrelation is a low-accurate technique to identify the noises affecting a low cost INS, a method based on AR models have been used to overcome this issue (see [20]). It consists in combining AR processes and wavelet de-noising to reduce high-frequency noise and, consequently, to obtain the AR coefficients from the residual noise. In other words, after minimizing the short-term error (high-frequency components) with wavelet de-noising, the residual noise could be modeled by an AR model.

For static drift data of the inertial sensors, the approximation part of the DWT includes the earth gravity, the earth rotation rate frequency components and the long-term error, while the detail part of the DWT contains the high-frequency noise and other disturbances [20, 120].

By working with inertial data collected in a stationary condition, we first applied the wavelet de-noising technique, and then, the AR model coefficients were estimated with Burg's method. This procedure is executed for each sensor and for two AR models: first and third order. In this work, the attention is focused on these two models, because the first order AR models is one of the most used in the navigation field, and also up to the third order, because as it is explained by Nassar *et al.* [20, 117], the higher order would increase the computational load and might result in unstable solutions.

Table 4.2 depicts the parameters obtained with Burg's method for each inertial sensor using the wavelet de-noising characteristics described in the previous section. It shows the coefficients for the first and third order AR process that correspond to the stochastic process explained in Section 4.2.2. These AR model coefficients are estimated after computing wavelet de-noising in stationary conditions, which was described in Section 4.2.5.

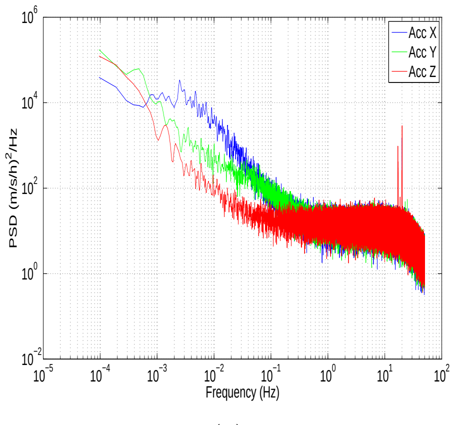
Table 4.2: Autoregressive process coefficients for each inertial sensor obtained with Burg's method after wavelet de-noising with six LOD.

	α_1	α_2	α_3	${\beta_0}^2 (m/s^2)^2$
Acc X	-1			$2.129 * 10^{-10}$
	-2.582	2.166	-0.585	$4.973 * 10^{-13}$
Acc Y	-1			$1.148 * 10^{-9}$
	-2.564	2.136	-0.572	$8.399 * 10^{-12}$
Acc Z	-1			$1.014 * 10^{-9}$
	-2.564	2.136	-0.572	$7.964 * 10^{-12}$
	α_1	α_2	α_3	${\beta_0}^2 (rad/s)^2$
Gyro X	-0.999			$1.014 * 10^{-10}$
	-2.564	2.131	-0.567	$2.739 * 10^{-13}$
Gyro Y	-0.999			$7.04 * 10^{-11}$
	-2.565	2.133	-0.568	$1.905 * 10^{-13}$
Gyro Z	-0.999			$8.066 * 10^{-11}$
	-2.562	2.128	-0.566	$2.181 * 10^{-13}$

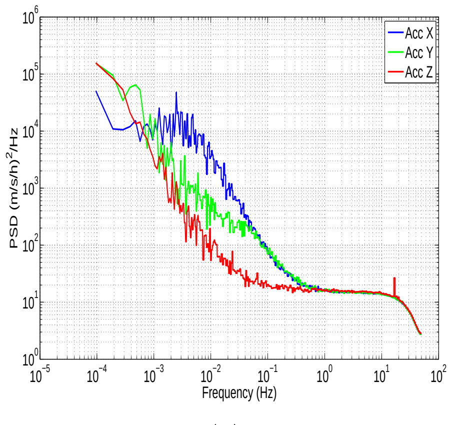
4.3.3 PSD Analysis

The power spectral density was implemented using Welch's method, since this has been found to have the widest application in engineering and experimental physics [123]. In this case, we have applied a Fast Fourier Transform with 2^{20} data points for the seven *hours* of the data collection. The results for the PSD are shown in Fig. 4.8(a) for accelerometer data.

Figure 4.8(a) depicts the one-sided PSD for accelerometers data. This log-log plot shows a bunch of high-frequency components, which makes it difficult to identify noise terms and obtain parameters of the stochastic model. The variance in these short-term noises may be decreased by averaging adjacent frequencies of the estimated PSD [18]; this task can be accomplished by using a technique that is called frequency averaging; further details of this technique can be found in [11]. Fig. 4.8(b) shows a PSD curve



(a)



(b)

Figure 4.8: (**a**) Power spectral density accelerometer IMU 3DM-GX3-25; (**b**) power spectral density accelerometer IMU 3DM-GX3-25 after frequency averaging.

after applying frequency averaging; it can be noticed that the noise term identification is easier than in Fig. 4.8(a), and although the low-frequency part of the PSD plot has a high uncertainty, it still conveys some information [11]. According to Fig. 3.12, which was presented in Section 4.2.3, there are three types of noise: the acceleration random walk (K), the bias instability (B) and the velocity random walk (N). Fig. 4.8(b) shows that the z-axis accelerometer has a bias instability (slope -1) smaller than the other two accelerometers, and the velocity random walk is almost the same for all the accelerometers (slope 0).

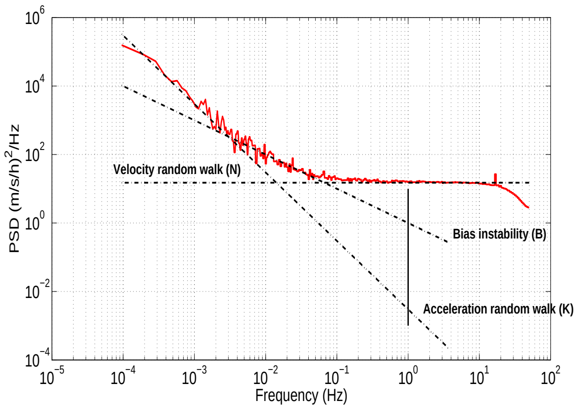
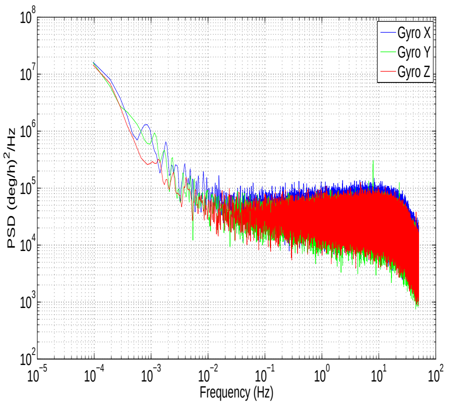


Figure 4.9: Power spectral density accelerometer Z IMU 3DM-GX-25.

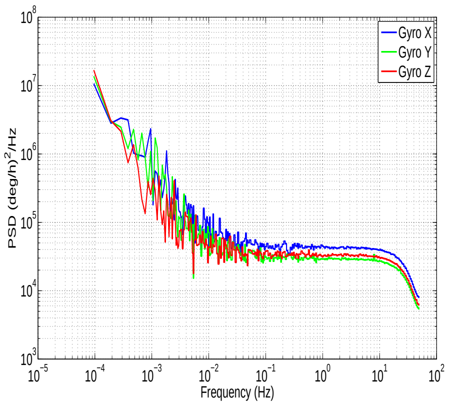
The values for each noise parameter (B,N,K) were extracted drawing straight lines for each frequency band influenced by the noise. The interception of each line with a specific point was taken into account. For instance, the PSD curve for the z-axis accelerometer is plotted in Fig. 4.9; it also includes straight dotted lines for each noise, N, B and K, with their respective slopes, 0, -1,-2. The acceleration random walk (K) is present in the low-frequency components between 1×10^{-4} Hz and 2.29×10^{-3} Hz. This parameter is obtained by fitting a straight line with a slope of -2, starting from 1×10^{-4} Hz, until it meets the vertical line of $f = 1$ Hz. Thus, the acceleration random walk for the z-axis accelerometer is determined as:

$$K = 14.60\ (m/s/h^{3/2}) \tag{4.7}$$

The bias instability (B) is the dominant noise between 2.29×10^{-3} Hz and 7.1×10^{-2} Hz, with a slope of -1, while the velocity random walk (N) is present between 0.1248 Hz and 20 Hz. After 20 Hz, there is an attenuation, because of the digital moving average filter, which is used to minimize high-frequency spectral noise produced by the MEMS sensors.



(a)



(b)

Figure 4.10: (**a**) Power spectral density gyro IMU 3DM-GX3-25; (**b**) power spectral density gyro IMU 3DM-GX3-25 after applying frequency averaging.

Regarding the gyroscopes, Fig. 4.10(a) represents the power spectral density, while Fig. 4.10(b) corresponds to the gyros PSD after applying frequency averaging; in the latter, it was identified as angle random walk (N) and bias instability (B), following the same procedure as with the accelerometers.

Table 4.3 summarizes the values of different errors that affect the inertial sensors using PSD method. In order to check the validity of these noise coefficients obtained with the power spectral density, AV analysis is presented in the following section.

Table 4.3: Identified error coefficients for accelerometers and gyro of the 3DM-GX3-25 IMU with PSD.

	Velocity Random Walk (N) $(m/s/(\sqrt{h})$	**Bias Instability** (B) $(m/s/h)$	**Acceleration Random Walk** (K) $(m/s/h^{3/2})$
Acc X	0.045	4.6447	168.60
Acc Y	0.044	4.6700	26.66
Acc Z	0.047	1.7733	14.60
	Angle Random Walk (N) $(deg/\sqrt{h})$	**Bias Instability** (B) (deg/h)	**Rate Random Walk** (K) $(deg/h^{3/2})$
Gyro X	2.297	43.438	
Gyro Y	1.937	39.614	
Gyro Z	2.058	30.705	

4.3.4 AV Analysis

For Allan variance analysis, the acceleration and the angular rate were integrated to obtain the instantaneous velocity and angle. Subsequently, the log-log plot of Allan variance standard deviation *versus* cluster times (T) was obtained after evaluating Eq. (4.4). The results are plotted in Fig. 4.11(a) for the accelerometer and Fig. 4.12 for gyro data.

Figure 4.11(a) shows the AV estimated on the 3DM-GX3-25 accelerometers. According to Fig. 4.3, which was presented in Section 4.5.1, the accelerometers are affected by three types of error: velocity random walk (N), bias instability (B) and acceleration random walk (K). It confirms that z-axis accelerometer has a bias instability (slope 0) smaller than the other two accelerometers, and the velocity random walk is almost the same for all the accelerometers (slope $-1/2$), which is coherent with the results obtained with the PSD.

The values for each noise parameter were extracted as in the PSD, drawing straight lines for each error with its corresponding slope, but in this case, the interceptions are different. To clarify, Fig. 4.11(b) depicts straight lines for each noise of the z-axis accelerometer. In this case, the accelerometer has N, B and K with slopes $-1/2$, 0 and $1/2$, respectively. It can be seen that the dominant noise in short cluster times is the velocity random walk, while the dominant error in long cluster times is the acceleration random walk. From the straight line with slope $-1/2$ fitted to the beginning of the

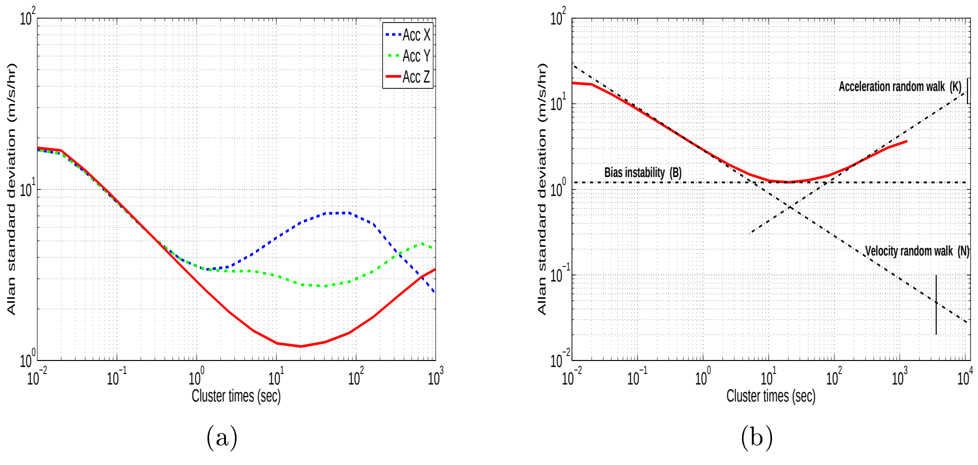


Figure 4.11: (**a**) IMU 3DM-GX3-25 Allan variance for accelerometers; (**b**) IMU 3DM-GX3-25 Allan variance for accelerometer Z.

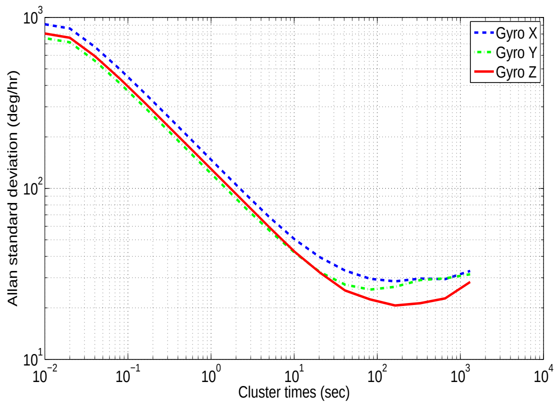


Figure 4.12: IMU 3DM-GX3-25 Allan variance for three gyro axes.

N noise, a value, $\sigma = 0.047\ (m/s/h)$, at a cluster times of 1 h can be read. Since the velocity random walk (N) is present in a cluster time interval where the number of independent clusters is very large, the estimation accuracy of the AV is approximately 1.1%. Thus, the velocity random walk or, in other words, the noise term (N) for the z-axis accelerometer is determined as:

$$N = 0.047 \pm 0.00050\ (m/s/\sqrt{h}) \tag{4.8}$$

The Allan variance standard deviation *versus* cluster times (T) for gyro data is depicted in Fig. 4.12. Unlike accelerometers, the gyroscopes have all similar characteristics, where two types of noises can be recognized: angle random walk (N) for short cluster times and bias instability (B) for long cluster times.

For the x-axis gyro (blue curve), the bias instability is present in the time range between 321.92 (s) and 654.01 (s). The value of this error can be measured with a flat line at 29.57 (deg/h). Dividing this standard deviation by the factor 0.664, as suggested in [9], the B coefficient can be achieved:

$$B = 44.533 \pm 5.14 \; (deg/h) \tag{4.9}$$

For further details of the intercepts of each noise term in the log-log AV curve, see [9, 12, 13].

Table 4.4 summarizes the error coefficients with their respective uncertainty for accelerometers and gyro data. The correlation time, (T_c), of the bias instability, (B), and the standard deviation for each sensor (STD) of the IMU 3DM-GX3-25 are shown in Table 4.5. The correlation time, (T_c), might be used in Eq. (3.22) for modeling the bias instability (B) as a first order Gauss-Markov process; this value is obtained from the segment of the curve where the bias instability is the dominant noise, *i.e.*, the flat segment of the log-log Allan variance curve. It should be mentioned that not only these parameters, but also the whole parameters obtained from AV need to be manually tuned in the KF, since the values obtained from AV are considered an initial approximation of the bias-drift [124].

Table 4.4: Identified error coefficients for accelerometers and gyro of the 3DM-GX3-25 IMU with AV.

	Velocity Random Walk (N) ($m/s/\sqrt{h}$)	**Bias Instability** (B) ($m/s/h$)	**Acceleration Random Walk** (K) ($m/s/h^{3/2}$)
Acc X	0.045 ± 0.00023	5.1581 ± 0.0370	166.30 ± 4.6398
Acc Y	0.045 ± 0.00022	4.5507 ± 0.0506	24.95 ± 2.8368
Acc Z	0.047 ± 0.00050	1.8336 ± 0.0524	13.53 ± 1.8685
	Angle Random Walk (N) ($deg/\sqrt{h}$)	**Bias Instability** (B) (deg/h)	**Rate Random Walk** (K) ($deg/h^{3/2}$)
Gyro X	2.420 ± 0.0974	44.533 ± 5.14	
Gyro Y	1.988 ± 0.0565	38.810 ± 2.51	
Gyro Z	2.164 ± 0.0599	31.717 ± 2.29	

This verifies the results that were obtained with PSD analysis, where velocity random walk (N), bias instability (B) and acceleration random walk (K) for accelerometers data and angle random walk (N) and bias instability (B) for gyro data

Table 4.5: Identified correlation time, (Tc), for the bias instability (B) and standard deviation for each inertial sensor of the 3DM-GX3-25 IMU.

	Acc X	**Acc Y**	**Acc Z**	**Gyro X**	**Gyro Y**	**Gyro Z**
T_c (s)	4.56	7.26	20.74	490.89	623.25	735.17
σ $(m/s^2 - rad/s)$	0.0068	0.0065	0.0063	0.0055	0.0045	0.0048

were also identified. It can be seen that most of the estimated values in PSD (see Table 4.3) are within the confidence interval computed by AV (Table 4.4).

The next section presents the inertial sensor error model that mixtures of AV and wavelet de-noising techniques.

4.3.5 Wavelet De-Noising with Allan Variance

In order to combine wavelet de-noising (WD) and Allan variance under dynamic conditions, it is necessary to process the inertial sensors measures with wavelet de-noising before computing the mechanization, which leads to the following question: how many levels of decomposition should be applied? In this case, the number of levels of decomposition (LOD) for the DWT are chosen based on the the spectrum of the signal after the DWT is applied. We have to consider that each level of decomposition divides the spectrum of the signal, $x(n)$, into different sub-bands, as was explained in Section 4.2.5. This means that if the sampling frequency of the inertial sensor is $f_s = 100\ Hz$, after applying one LOD, we will have a spectrum between $0 - 25\ Hz$ for the approximations coefficients (A_1) and a spectrum between $25 - 50\ Hz$ for the details coefficients (D_1), considering perfect filters. Therefore, the frequency band of the wavelet de-nosing output will be limited to $f_s/(2 \times 2^k)$ for the more relevant coefficients (A_k), where k is the level of decomposition (LOD). Since the idea is to preserve the frequency components that are associated with the motion dynamics of the land vehicle, we consider that these motion dynamics are low-frequencies components for land-vehicle applications (*e.g.*, between 0 and 5 Hz), as is mentioned in [114]. Therefore, we evaluated the number of LOD from the one that nearly reaches 5 Hz and higher levels, *i.e.*, considering the approximation coefficients (A_k).

Thus, the test was achieved using the Matlab Wavelet Toolbox from three LOD, where the band of approximation coefficients is limited to 6.25 Hz $(100/(2\times2^3) = 6.25)$, up to eight levels of decomposition, where the output band is limited to 0.1953 Hz $(100/(2 \times 2^8) = 0.1953)$. This is taking into account that we use a sampling frequency of $f_s = 100$ Hz for the inertial sensors. These experiments were assessed using the Daubechies family, specifically, "db4", as the wavelet function, with soft thresholding based on Stein's Unbiased Risk Estimate (SURE), since these parameters are typically used in pre-filtering inertial sensors [23,25,114]. After selecting these wavelet de-noising parameters, the data collected in the laboratory was de-noised and, subsequently, processed with the AV algorithm. Fig. 4.13 depicts the Allan variance standard deviation *versus* cluster times (T) for the z-axis accelerometer (red curve) after applying wavelet de-noising with three and eight levels of decomposition (blue curves). According to this plot, wavelet de-noising removed the short-term noises, while the long-term noises remain without attenuation, as was expected. It is also noticed that the higher the level of decomposition, the more high-frequency components are removed.

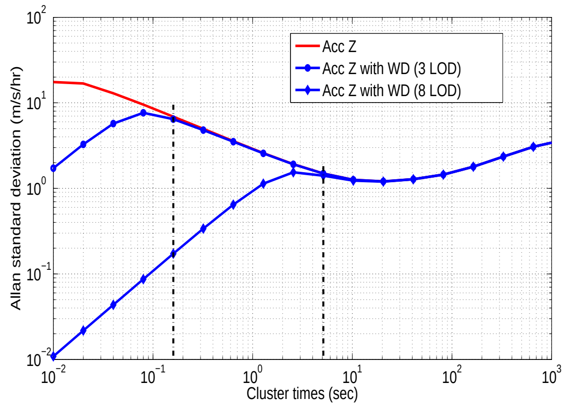
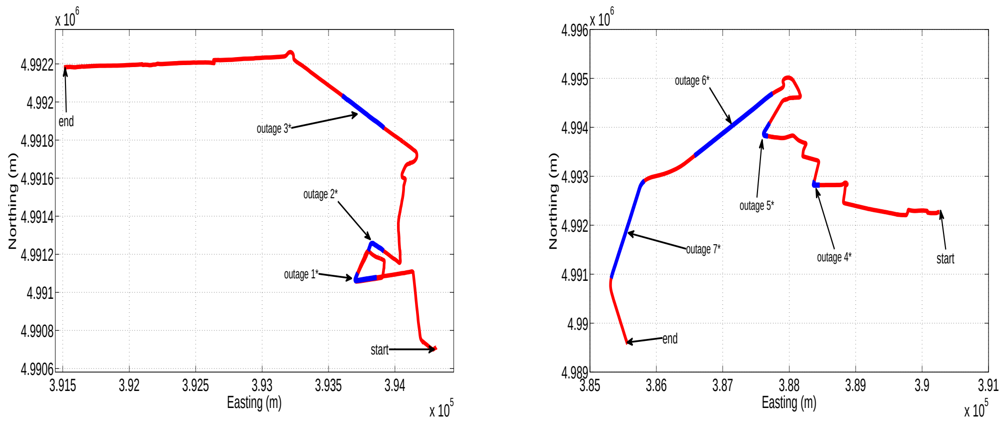


Figure 4.13: Allan variance accelerometer Z IMU 3DM-GX-25 after applying wavelet de-noising with three and eight levels of decomposition.

If we consider these two cases–the first one applying wavelet de-noising with three LOD and the second one applying eight LOD (Fig. 4.13)–the most relevant components that correspond to the motion dynamics of the vehicle would have to be above 0.16 *sec* and 5.12 *sec* (vertical black dotted lines) for each case, respectively. If these components that relate to the motion dynamics are not above these cluster time values, they would be attenuated by the de-noising filters, which could degrade the INS accuracy.

Given that these motions of the vehicle are mixed with the long-term noises, a suitable LOD should be selected with the purpose of not removing relevant components that would compromise the performance of the navigation system. Therefore, to analyze the effect of wavelet de-nosing, we evaluated the enhancement accuracy of the GPS/INS solution with two vehicle tests, where a total of seven GPS outages were introduced under different dynamic conditions with a duration of 30 *sec* and 60 *sec* (see Fig. 4.14). A similar procedure was achieved in [107] with a tactical-grade (medium-accuracy) and navigation-grade (high-accuracy) IMUs. The performance of the GPS/INS solution (*i.e.*, without error models) during GPS outages with wavelet de-nosing under different LOD is summarized in Table 4.6. It depicts the outage number, the average speed and the maximum horizontal error for each GPS outage that was assessed. The LOD 0 corresponds to the navigation solution without applying wavelet de-noising. In the case of three LOD, we apply one level of decomposition less for y-axis and z-axis inertial sensors, since the uncorrelated noise is not so dominant for the other inertial sensors, as can be seen in the autocorrelation analysis described in Section 4.3.1.



(a) (b)

Figure 4.14: (**a**) First and (**b**) second trajectory test in Matlab with the GPS outages that were introduced intentionally to analyze the effect of wavelet de-nosing with different LOD.

Table 4.6 shows that the navigation solution performs slightly better for most of the GPS blockage when seven LOD are applied, compared to the navigation solution without applying wavelet de-nosing (*i.e.*, zero LOD), with an improvement of almost 4.3% in terms of horizontal positioning error.

The wavelet de-nosing parameters that provided the most significant enhance accuracy of the GPS/INS solution are summarized in Table 4.7. It represents the levels of decomposition where the most relevant energy associated with the motion dynamics of the vehicle remain. In this case, the most significant frequency components of the vehicle motion dynamics for the 3DM-GX3 IMU are below 0.78 Hz for y-axis and z-axis accelerometers, while for the rest of inertial sensors, it is below 0.39 Hz.

The use of Stein's Unbiased Risk Estimate (SURE) as a threshold rule helps us not to lose significant coefficients associated with the vehicle, since it is a conservative threshold that is usually used when small details of the signal lie in the noise range [125].

Table 4.6: Maximum horizontal position error during GPS outages before and after applying wavelet de-noising.

Outage (#)	**Duration** (sec)	**Average Speed** (km/h)	**Levels of Decomposition**		
			0	**3**	**7**
			max (m)	**max** (m)	**max** (m)
1*	30	25.16	85.12	85.13	76.08
2*	30	18.86	162.98	162.67	157.73
3*	30	42.53	189.67	189.69	188.82
4*	30	23.56	52.20	52.20	51.62
5*	30	39.32	54.03	54.01	43.99
6*	60	103.36	232.54	232.74	217.28
7*	60	122.73	279.22	279.03	274.74

Table 4.7: Wavelet de-noising parameters for each sensor under kinematic conditions.

	LOD	**Frequency Limit for A_k Coefficients** (Hz)	**Thresholding**
Acc X	7	0.39	soft, SURE
Acc Y	6	0.78	soft, SURE
Acc Z	6	0.78	soft, SURE
Gyro X	7	0.39	soft, SURE
Gyro Y	7	0.39	soft, SURE
Gyro Z	7	0.39	soft, SURE

Having selected the LOD for wavelet de-noising, the long-term noises are modeled and compensated by the AV parameters obtained in Section 4.3.4. Overall, under dynamic conditions, wavelet de-noising will be computed for inertial sensor measurements prior to the INS mechanization, and the AV model will be in charge of compensating the long-term noises. The next section explains the way the AV model and each model obtained so far is adapted into the loosely-coupled strategy.

4.4 INS Bias Model Adapted to the Loosely-Coupled KF

Having identified the random errors using AV and PSD, the parameters obtained with AV were used in the loosely-coupled GPS/INS integration scheme (Fig. 2.7) to model the errors of accelerometers and gyros of the IMU under test. The stochastic model parameters for each sensor are taken from Tables 4.4 and 4.5. Thus, the 3DM-GX3-25 accelerometers stochastic error a_{se} was modeled as:

$$a_{se} = WN(N) + 1^{st}\,GM(B) + RW(K) \tag{4.10}$$

where the noise term associated to N is modeled as white noise (WN), the noise term associated to K as a random walk (RW), while the bias instability (B) is modeled as a first order Gauss-Markov process ($1^{st}\,GM$).

Regarding the 3DM-GX3-25 gyro stochastic error, g_{se}, the model was defined as:

$$g_{se} = WN(N) + 1^{st}\,GM(B) \tag{4.11}$$

where the noise term associated to N is modeled as white noise (WN) and the bias instability (B) is modeled as a first order Gauss-Markov process ($1^{st}\,GM$). The latter noise can be modeled by a combination of Markov noise states [11], and there are also different approaches to model the bias instability noise terms; some of them are presented in [13, 28]. In this case, a first order Gauss-Markov process was fitted to the flat part of the AV curve taking into account B and its correspondent correlation time (T_c) (see Table 4.5). Regarding the noise term angle random walk (N), it presents dominant high-frequency components that have a correlation time much shorter than the sample time. Therefore, this noise is modeled as additive noise with noise variance obtained from the parameter, N (see Table 4.4).

On the other hand, the AR coefficients obtained from Burg's method are adapted into the KF taking the parameters that were shown in Table 4.2. These stochastic error models were implemented in the KF according to the state-space forms that were presented in Section 3.5.2. Further details about IMU error state-space implementation in the Kalman filter can be found in Appendix A. The performance of the different stochastic error models for the bias-drift when they are adapted into the LC integration are shown in Chapter 6.

4.5 Constrained Non-linear Fitting

4.5.1 Allan Variance Limitations

Although AV is the most widely used method for modeling of inertial sensors, this error analysis technique presents some drawbacks, (*e.g.*, it requires very long data set to obtained a consistent AV curve) high uncertainty for long clusters times, it may also lead to different interpretations when a AV curve is observed since many models can be used to fit the log-log AV curve [28]. Despite these limitations Allan variance is still the most accepted method for modeling inertial sensors based on MEMS technology because it provides a complete analysis of the error and uses a simple algorithm for the identification of the different error sources.

In previous works where Allan variance has been implemented [14–16, 24] *et al.* it is often assumed that each noise source is considerably separated in frequency, thus each noise term can be associated with a slope. In fact, according to the hypothetical AV curve Fig. (4.3) the flicker noise (bias instability) should have a slope of 0, the angle random walk a slope of $-1/2$, *etc.* This can be suitable for high-quality inertial sensors, where the parameters of the stochastic model are extracted by fitting straight lines with different slopes that corresponds to each noise term. However, this procedure is not entirely valid for the majority of low cost inertial sensors since the noise terms might be strongly overlapped in frequency, so they might form complex structures that are not easy to identify by the conventional method of the straight line-fitting [6]. This can be clearly seen in the flicker noise because it is usually combined with short-term and long-term noises, so the identification of the 0 slope might be ambiguous due to the fact that different noise terms might be mixed in the frequency band where it is supposed to be the dominant noise.

Despite the fact that AV presents these limitations, we consider this technique because we can overcome some of the drawbacks by using non-linear functions to fit the log-log curves instead of using traditional line-fitting, which is more suitable to estimate the unknown parameters of the random error.

Additionally, recent methods using different techniques have been developed to estimate the stochastic model parameters more accurately, for instance the Best Linear Unbiased Estimator (BLUE) estimator and Expectation Maximization (EM) algorithm described in [7, 21, 115]. Only the latter is briefly explained in the following section, specifically the constrained EM, since it is more adequate for low cost MEMS sensors than the BLUE estimator presented in [7] that is suitable for high-end IMUs.

4.5.2 Constrained Expectation Maximization (EM)

An algorithm that has been used in many areas to estimate unknown parameters is described in [126–129] and it relies on the Expectation Maximization technique that was first proposed in [130]. The EM algorithm can be applied to a generalized State-Space Model (SSM) in discrete time as stated in Eq. (4.12) and Eq. (4.13):

$$\mathbf{x}_{k+1} = \boldsymbol{\Phi}_k \mathbf{x}_k + \mathbf{w}_k \tag{4.12}$$

$$\mathbf{z}_{k+1} = \mathbf{H}_{k+1} \mathbf{x}_{k+1} + \mathbf{v}_{k+1} \tag{4.13}$$

where $\boldsymbol{\Phi}_k$ is the dynamic coefficient matrix, $\mathbf{H}_{k+1}$ is the design matrix which converts the unobserved stochastic vector $\mathbf{x}_{k+1}$ into the observed space $\mathbf{z}_{k+1}$. The sequences $\mathbf{w}_k$ and $\mathbf{v}_{k+1}$ represent the process errors characterized by a zero-mean, uncorrelated and normally distributed noise such that $\mathbf{w} \sim (\mathbf{0},\mathbf{Q})$ and $\mathbf{v} \sim (\mathbf{0},\mathbf{R})$, where $\mathbf{R}$ and $\mathbf{Q}$ are related to the observation and the state equations, respectively [127]. This linear system is adapted with the stochastic models that will predict the random noises that affect the IMU.

Since the random errors of a low cost inertial sensor might be composed by elaborated structures, which makes difficult the estimation of the parameters specified in the SSM, in [21, 115] the EM algorithm was modified and properly used in order to estimate the parameters of the stochastic model and so improve the inertial sensors performance. In this case, we focused our attention on the constrained EM method proposed in [21] due to the fact that some elements of the stochastic model remain fixed in the SSM and some others are freely estimated. Thereby, the SSM is less complicated and allows to estimate more complex stochastic errors that are involved in any INS MEMS grade.

The EM is an iterative procedure that consists of switching between two steps: one is the expectation E-step and the other one is the maximization M-step [131]. The E-step involves the calculation of the expected states $\mathbf{x}_k^N$ and of the covariance matrices $\mathbf{P}_k^N$ and $\mathbf{P}_{k,k-1}^N$. These quantities can be calculated using a Kalman smoother (see [127]). Then, in order to obtain the unknown parameters of the State-Space Model (*e.g.*, the parameters of the stochastic model that will describe the behaviour of the bias-drift) defined by Eq. (4.12) and Eq. (4.13), a (log) likelihood function $\log \mathbf{L}\left(\boldsymbol{\theta}|\mathbf{y}_k,\mathbf{x}_k\right)$ is used, where $\boldsymbol{\theta}$ is the vector with the unknown parameters included in the SSM. The idea is to update $\boldsymbol{\theta}$ until the (log) likelihood is maximized and after a certain number of iterations the parameters monitored in the SSM will converge to a Maximum Likelihood value that, in case of INS, will represent a reliable estimation of the bias-drift noise parameters. A detailed description of the algorithm and a complete derivation of the equations for the constrained case can be found in [21, 127, 132].

Next section will explain the NLF developed in this thesis, which is based on a constrained non-linear fitting, it will be evaluated and further compare in Section 4.5.3 with the previous mentioned.

4.5.3 Constrained Non-linear Fitting (NLF)

The traditional method using line-fitting in log-log AV curves as it is described in [12–14, 16, 133] is not easy to apply to low cost inertial sensors, since AV curves are usually more complex than in the case of high-end IMUs. For instance, the flicker noise that is usually modelled with a first order Gauss-Markov process is not easy to estimate in a low cost IMU because the band where it is dominant presents an overlapping between the long-term and short-term noises [134], so it could be influenced by a combination of a ARW plus a RRW processes (*i.e.*, for a gyro). Now if we consider that the flicker noise is composed by not only one first order Gauss-Markov process, the estimation of these error parameters is even more challenging. That is, the flicker noise might be a dominant noise in a wideband frequency, where only one first order GM process is not enough to model it.

In order to improve the estimation of the unknown parameters that represent the bias-drift of inertial sensors, it is supposed that the existing noise terms are all statistically independent, so it can be shown that the Allan variance at any given cluster time T is the sum of Allan variances due to the individual random processes at the same T [9].

$$\sigma^2(T) = \sigma_N^2(T) + \sigma_K^2(T) + \sigma_{flicker}^2(T) + \cdots \tag{4.14}$$

where $\sigma_N^2(T)$, $\sigma_K^2(T)$ are the Allan variance of the angle/velocity random walk and rate/acceleration random walk, respectively, whereas $\sigma_{flicker}^2(T)$ is the AV of the flicker noise. In the case of flicker noise, it can be modelled as a combination of several first order Gauss-Markov (GM) processes as it was explained in Section 3.5.2. According to Voss [109], flicker noise modeling requires a specific distribution of independent processes with different correlation time, which is valid in the case of AV because processes are independent. So the variance of the flicker noise can be expressed as:

$$\sigma_{flicker}^2(T) = \sigma_{M1}^2(T) + \sigma_{M2}^2(T) + \cdots \tag{4.15}$$

where $\sigma_{M1}^2(T)$ is the Allan variance of one first order GM process. The continuous time representation for the first order GM process is stated in Eq. (3.22), which has a noise covariance σ_w^2 that can be expressed as:

$$\sigma_w^2 = 2\beta\sigma_{GM}^2 \tag{4.16}$$

where σ_{GM}^2 is the covariance of the process and β is the inverse of the correlation time (T_c). The power spectral density $S_x(f)$ of Eq. (3.22) is given by [64]:

$$S_x(f) = \frac{2\beta\sigma_{GM}^2}{(2\pi f)^2 + \beta^2} \tag{4.17}$$

Expressing the PSD of the first order GM in terms of the noise covariance (σ_w^2) yields:

$$S_x(f) = \frac{(\sigma_w T_c)^2}{1 + (2\pi f T_c)^2} \tag{4.18}$$

According to the relationship between Allan variance and the two-sided PSD, performing the integration that is stated in Eq. (4.5) yields the Allan variance of the first order Gauss-Markov process:

$$\sigma_M^2(T) = \frac{(\sigma_w T_c)^2}{T}\left[1 - \frac{T_c}{2T}\left(3 - 4e^{-\frac{T}{T_c}} + e^{-\frac{2T}{T_c}}\right)\right] \tag{4.19}$$

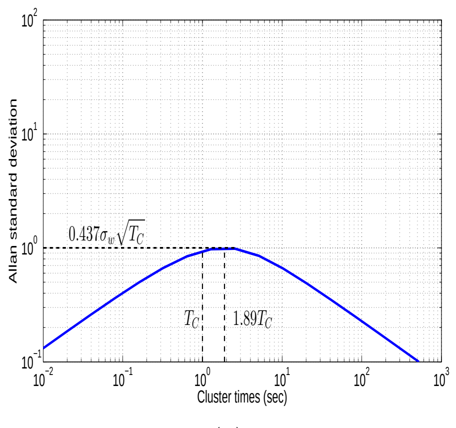
Plotting this function on a log-log scale with $T_c = 1$ and $\sigma_w = 1/(0.437\sqrt{T_c})$, we obtain the AV curve for a first order GM process (see Fig. 4.15(a)), where the cluster times for the midpoint of the flat region is equal to

$$T_{mp} = 1.89T_c \tag{4.20}$$

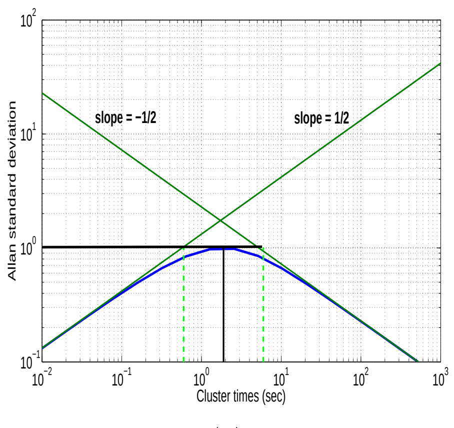
with a standard deviation at T_{mp} given by [9]:

$$\sigma_M(T_{mp}) = \sigma_{mp} = 0.437\sigma_w\sqrt{T_c} \quad \rightarrow \quad \sigma_{mp} = 0.437\sigma_w\sqrt{T_{mp}/1.89} \tag{4.21}$$

Thus the Allan variance of the flicker noise as reported in Eq. (4.15) can be replaced by Eq. (4.19) with various σ_{wi} and T_{ci} in order to approximate the flicker noise as a



(a)



(b)

Figure 4.15: (**a**) Allan variance of a first order GM process; (**b**) Random walk and white noise for the Allan variance of a first order GM process.

sum of multiple first order GM processes. Given the fact that low cost MEMS sensors are not only composed by flicker noise but also by white noise, random walk, ramp rate, *etc*, which makes more difficult the parameters estimation for the stochastic error modelling due to the complex noise structures, we propose constraints for each noise term identified with AV. In this way, we can obtain a more appropriate estimation of the bias-drift and at the same time we can provide additional information to the algorithm that will estimate the parameters. The determination of the constrains will be explained in the following subsections where we also describe the stochastic model identification, the estimator and the optimization algorithm that are used in the NLF.

Identifying the Stochastic Model

In the NLF first a stochastic model is set from the noise terms identified in a log-log AV curve. For instance, if we consider the AV curve depicted in Fig. 4.16, the objective function $\sigma^2(\theta,T)$, *i.e.*, the function to be optimized given certain constraints and that will be fitted to the estimated variance, would be given by:

$$\sigma^2(T,\theta) = \sigma_N^2(T,\theta) + \sigma_K^2(T,\theta) = \frac{N^2}{T} + \frac{K^2T}{3} \tag{4.22}$$

where N and K are associated with the ARW (slope $-1/2$) and RRW (slope $1/2$), which can be represented by white noise (WN) and random walk (RW) processes described in Section 3.5.2. Thus, we would have a stochastic process that combines WN plus RW. In this case, the unknown parameters (θ) would be the standard deviation of the white noise and the standard deviation of the random walk processes, that is, σ_{WN} and σ_{RW}, respectively. Considering the state-space form that was described in Section 3.5.2 the sum of these random processes can be expressed by:

$$x_k = x_{k-1} + (\sigma_{RW}\sqrt{\Delta t})w_k \tag{4.23}$$

$$y_k = x_{k-1} + (\sigma_{WN}/\sqrt{\Delta t})v_k \tag{4.24}$$

where Δt would be the sampling time of the IMU raw measurements. Although Fig. 4.16 can be found in some inertial sensors, most of MEMS inertial sensor are affected by flicker noise, which increases the complexity of the function to be fitted. This issue will be considered below.

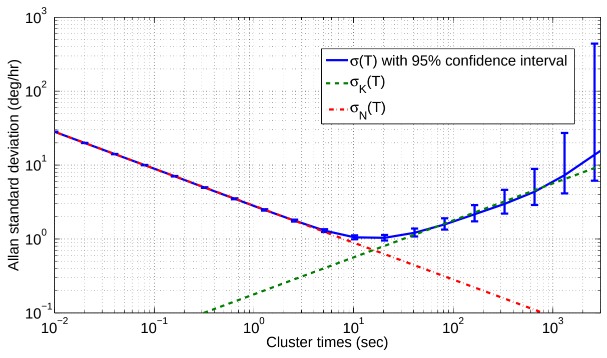


Figure 4.16: Allan variance of angle random walk (N) plus rate random walk (K).

Constraints of the Objective Function

Once the stochastic model is identified, we defined constrains for the objective function to be fitted. In order to set the constraints in the non-linear fitting for different combination of noises, it is necessary to compute the 95% confidence interval of the analysed log-log AV curve. This can be performed by estimating the coefficients related to the noise terms, *i.e.*, ARW, VRW, flicker noise, *etc.* as it is described in [10,135,136]. Another way to get the confidence interval is by means of wavelet variance (WV). This method is based on the Maximal Overlap Discrete Wavelet-Transform (MODWT) and it uses a modified version of the DWT explained in Section 4.2.5. Actually, it carries out the same steps as filtering the discrete wavelet transform ordinary but does not subsample [137]. The wavelet variance based on the MODWT can be expressed as [138]:

$$\sigma^2_{WV}(T_j) = \frac{1}{M_j(N)} \sum_{t=L_j-1}^{N-1} \tilde{W}^2_{j,t} \tag{4.25}$$

where $\tilde{W}_j$ are the MODWT wavelet coefficients for levels of decomposition $j = 1,...,J$, N is the length of the data to be analysed, T_j is defined for dyadic scales 2^{j-1}, L_J is the filter length for level J and $M_j(N) = N - L_j + 1$. Although, different wavelet functions can be used to filter the signal, when Haar wavelet is used, Allan variance can be interpreted in terms of the coefficients of the Haar wavelet transform [139]. This is due to the fact that the wavelet variance is equal to half the Allan variance [138]. Since wavelet variance is related to Allan variance and it does not require a *priori* knowledge of the noise parameters associated with the noise terms, the calculation of the 95% confidence interval is achieved by computing wavelet variance. Details about the properties of wavelet variance and the calculation of the confidence interval with WV can be found in [137, 138].

Considering Fig. 4.16 the constraints for the noise term N, *i.e.*, upper (N_U) and lower (N_L) limits, can be determined by:

$$N_L = \sigma_{cil}(T_{NL})\sqrt{T_{NL}} \quad , \quad N_U = \sigma_{ciu}(T_{NU})\sqrt{T_{NU}} \tag{4.26}$$

where (σ_{cil},σ_{ciu}) are the low and up 95% confidence interval of $\sigma(T)$, respectively, and (T_{NL},T_{NU}) are the cluster times where (σ_{cil},σ_{ciu}) will be evaluated to set the lower and upper bounds of the noise term N. (T_{NL},T_{NU}) should be chosen to cover the largest segment where the noise term is dominant and the uncertainly is relatively low. In other words, we expect to have the parameter N between those constraints due to the fact that the influence of the noise term (K) is minimal in the segment of the curve where N is dominant.

If we do the same for K we can obtain the upper (K_U) and lower (K_L) bounds with:

$$K_L = \sigma_{cil}(T_{KL})\sqrt{3/T_{KL}} \quad , \quad K_U = \sigma_{ciu}(T_{KU})\sqrt{3/T_{KU}} \tag{4.27}$$

where (T_{KL},T_{KU}) are the cluster times that are used to evaluate (σ_{cil},σ_{ciu}) and set the lower and upper bounds of the noise term K. Parameters (T_{KL},T_{KU}) should be selected near the cluster times where the noise term related to K (*i.e.*, 1/2 slope) starts being dominant because this is where its uncertainty is lower. Eq. 4.26 and Eq. 4.27 are considered from the AV coefficient values for each noise term (see Table 4.1). The constraints for parameters K and N for the log-log AV curve (Fig. 4.16) are shown in Fig. 4.17 with green dashed lines. T_{NL} and T_{NU} were set at $T = 0.64$ *sec*, whereas T_{KL} and T_{KU} were set at $T = 40.96$ *sec* and $T = 81.92$ *sec*, respectively. Note that bounds for N are chosen in the segment of the curve where this noise term is dominant. The difference between upper and lower bounds can not be distinguished because of the low uncertainty at $T = 0.64$ *sec*. In contrast, the lower and upper bounds for K are clearly seen in the figure and they are selected to cover most of the segment where K is dominant.

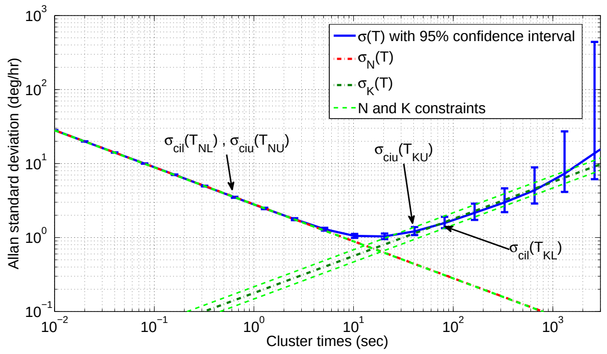


Figure 4.17: Allan variance of angle random walk (N) plus rate random walk (K) with constraints.

Moreover, in a log-log AV curve with dominant noises N, K and B, the constraints for N and K could be defined as it is stated in Eqs. (4.26) and (4.27). However, we would need to specify the constraints for the correlation time T_{ci} and the noise covariance σ^2_{wi} for each i first order GM process that will compound the flicker noise.

Before analysing the sum of several GM process, we first plot the noise components for one of them in Fig. 4.15(b), it shows that 0 slope is dominant for 1/3 of *decade* approximately, which will be the segment of the curve that would represent the flicker noise. This is because for $T \gg T_c$, it becomes a white noise ($-1/2$ slope) after almost one *decade* above the midpoint (T_{mp}), whereas for $T \ll T_c$ it becomes a random walk (1/2 slope) after almost one *decade* below the midpoint (T_{mp}). The equations for these noises that are part of the first order GM process can be deduced from Eq. (4.19).

$$\sigma_M^2(T) \Rightarrow \frac{(\sigma_w T_c)^2}{T} \qquad T \gg T_c \tag{4.28}$$

$$\sigma_M^2(T) \Rightarrow \frac{\sigma_w^2}{3} T \qquad T \ll T_c \tag{4.29}$$

When multiple first order GM processes are superimposed a better approximation of the flat segment in an log-log AV curve (flicker noise) can be obtained. According to Keshner [104], a pole of first order can approximate the flicker noise in a bandwidth of one *decade*. So we can define one *decade* as constraint for each term T_{mpi} of the GM processes. *i.e.*, if the flicker noise is dominant in two *decades*, between 10 *sec* and 1000 *sec*, we will set two GM processes. The midpoint correlation time T_{mp1} would be constrained by $10 \leqslant T_{mp1} \leqslant 100$, whereas the parameter T_{mp2} for the second GM process would be constrained by $100 \leqslant T_{mp2} \leqslant 1000$.

To determine the constrains of the parameters σ_{wi}^2, we made the following analysis:

If we consider three first order GM processes that are uniformly distributed in a log-log AV curve, it means that each noise has equal standard deviation (*e.g.*, $\sigma_{mp1} = 1$, $\sigma_{mp1} = \sigma_{mp2} = \sigma_{mp3}$) and they are separated a *decade* (*e.g.*, $T_{c1} = 1$, $T_{c3} = 10T_{c2} = 100T_{c1}$) as it is shown in Fig. 4.18, we can compute Allan variance $\sigma^2(T)$ using Eq. (4.15), so we will have

$$\sigma^2(T) = \sigma_{M1}^2(T) + \sigma_{M2}^2(T) + \sigma_{M3}^2(T) \tag{4.30}$$

To determine the value of $\sigma^2(T)$ where $\sigma_{M1}^2(T)$ is more dominant, we can evaluate Eq. (4.30) for the cluster times $T = T_{mp1}$ yielding:

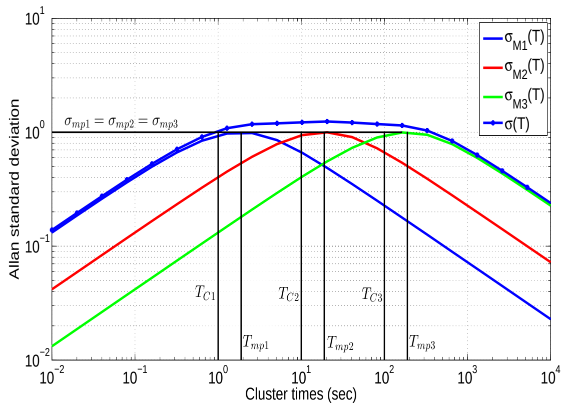


Figure 4.18: Allan variance of a flicker noise with three first order Gauss Markov process.

$$\sigma^2(T_{mp1}) \simeq \sigma^2_{mp1} + \frac{\sigma^2_{w2}T_{mp1}}{3} + \frac{\sigma^2_{w3}T_{mp1}}{3} \tag{4.31}$$

where we have considered Eq. (4.29) for $\sigma^2_{M2}(T_{mp1})$ and $\sigma^2_{M3}(T_{mp1})$ since at T_{mp1} they are dominated by a random walk process. Then, isolating σ_w in Eq. (4.21) and replacing it in Eq. (4.31) yields

$$\sigma^2(T_{mp1}) \simeq \sigma^2_{mp1} + \frac{3.3\sigma^2_{mp2}T_{mp1}}{T_{mp2}} + \frac{3.3\sigma^2_{mp3}T_{mp1}}{T_{mp3}}, \quad T_{mp1} \ll T_{c2}, T_{mp1} \ll T_{c3} \tag{4.32}$$

In this equation we can see that if T_{mp2} and T_{mp3} are large compared with T_{mp1}, $\sigma^2(T_{mp1})$ approximates the value of σ^2_{mp1} since the influence of $\sigma^2_{M3}(T_{mp1})$ and $\sigma^2_{M3}(T_{mp1})$ would be minimal.

For the midpoint of $\sigma_{M2}(T)$, we have that the Allan variance $\sigma^2(T_{mp2})$ is given by

$$\sigma^2(T_{mp2}) \simeq \frac{(\sigma_{w1}T_{c1})^2}{T_{mp2}} + \sigma^2_{mp2} + \frac{\sigma^2_{w3}T_{mp2}}{3} \tag{4.33}$$

Here the term $\sigma^2_{M1}(T_{mp2})$ was replaced by Eq. (4.28) whereas the term $\sigma^2_{M3}(T_{mp2})$ was approximated using Eq. (4.29), this is due to the fact that at T_{mp2} they become white noise and random walk, respectively. Using Eqs. (4.20) and (4.21) we obtain Eq. (4.34). Note that if T_{mp2} is large compared with T_{mp1} and small compared with

T_{mp3} the value of Allan variance $\sigma^2(T_{mp2})$ approaches to σ^2_{mp2}, that is, as long as $\sigma_{mp1} = \sigma_{mp2} = \sigma_{mp3}$.

$$\sigma^2(T_{mp2}) \simeq \frac{2.7\sigma^2_{mp1}T_{mp1}}{T_{mp2}} + \sigma^2_{mp2} + \frac{3.3\sigma^2_{mp3}T_{mp2}}{T_{mp3}}, \; T_{mp2} \gg T_{c1}, T_{mp2} \ll T_{c3} \tag{4.34}$$

Following the same procedure for $\sigma^2(T_{mp3})$ we obtain

$$\sigma^2(T_{mp3}) \simeq \frac{2.7\sigma^2_{mp1}T_{mp1}}{T_{mp3}} + \frac{2.7\sigma^2_{mp2}T_{mp2}}{T_{mp3}} + \sigma^2_{mp3}, \; T_{mp3} \gg T_{c1}, T_{mp3} \gg T_{c2} \tag{4.35}$$

This is worth noting that Eq. (4.32), Eq. (4.34) and Eq. (4.35) might be applied to any flicker noise in an inertial sensor as long as the inequalities provided are satisfied. For the specific case of Fig. 4.18, where the separation between the GM processes is one *decade*, the contribution of each to the flicker noise at T_{mp1}, T_{mp2} and T_{mp3} could be determined through those equations, as it follows:

$$\sigma^2(T_{mp1}) \simeq \sigma^2_{mp1} + 0.33\sigma^2_{mp2} + 0.033\sigma^2_{mp3}, \quad 1 \ll 5.29, 1 \ll 52.91 \tag{4.36}$$

$$\sigma^2(T_{mp2}) \simeq 0.28\sigma^2_{mp1} + \sigma^2_{mp2} + 0.33\sigma^2_{mp3}, \quad 10 \gg 0.52, 10 \ll 52.91 \tag{4.37}$$

$$\sigma^2(T_{mp3}) \simeq 0.028\sigma^2_{mp1} + 0.28\sigma^2_{mp2} + \sigma^2_{mp3}, \quad 100 \gg 0.52, 100 \gg 5.29 \tag{4.38}$$

This way, we have three equations with three unknown variables σ_{mp1}, σ_{mp2} and σ_{mp3}. Solving Eqs. (4.36), (4.37) and (4.38) we can obtain a preliminary estimation of σ_{mp1}, σ_{mp2} and σ_{mp3}. Thus, knowing an approximate curve of the flicker noise and identifying the cluster times where this is more dominant (*i.e.*, T_{mp1}, T_{mp2} and T_{mp3}), one could make a preliminary determination of σ_{mp} for each first order GM process, which could help us to define the constraints. It must be taken into account that these equations are simply an aid to determine constraints for parameters σ_{wi}. In general, we consider a constraint of half *decade* below the Allan standard deviation $\sigma(T)$ for σ_{wi}, this is because after doing experiments with different T_c and σ_w through Eqs. (4.32), (4.34) and (4.35), we noticed that half *decade* is a sufficient range for the σ_{mp} of a first order GM to contribute significantly in the log-log AV curve of an inertial sensor.

In order to have more information that might be useful to set the constraints of the GM processes, we noted that the flicker noise can be partially observed in a log-log AV curve of an inertial sensor. For this purpose, we consider two log-log AV curves that are typically found in inertial sensors. The first one is a mixture of WN, RW and flicker noise as it is shown in Fig 4.19, where we have set constraints for N and K as we described previously. The mathematical representation can be expressed by:

$$\sigma^2(T) = \sigma_N^2(T) + \sigma_K^2(T) + \sigma_{flicker}^2(T) \tag{4.39}$$

Assuming that Allan variances for velocity random walk (N) and acceleration random walk (K) are given by σ_{NL}^2 and σ_{KL}^2, that are the variances in terms of the lower bounds N_L and K_L,

$$\sigma_{NL}^2(T) = \frac{N_L^2}{T} \quad , \quad \sigma_{KL}^2(T) = \frac{K_L^2 T}{3} \tag{4.40}$$

then replacing Eqs. (4.40) in Eq. (4.39) and isolating $\sigma_{flicker}^2(T)$, we can obtain an approximation of the flicker noise that we will be denoted as $\sigma_f^2(T)$:

$$\sigma_f^2(T) \simeq \sigma_{flicker}^2(T) \simeq \sigma^2(T) - \sigma_{NL}^2(T) - \sigma_{KL}^2(T) \tag{4.41}$$

Figure 4.19 shows the segment of the curve where $\sigma_f(T)$ (in red) better approximates $\sigma_{flicker}(T)$. When there is a low uncertainty, $\sigma_f(T)$ approximates very well to $\sigma_{flicker}(T)$. However, for cluster times above 80 *sec* the uncertainty increases and its estimate is unreliable.

The second log-log AV curve that is typically found in gyros is the mixture of flicker noise and angle random walk as it is shown in Fig. 4.20. The mathematical representation can be written as:

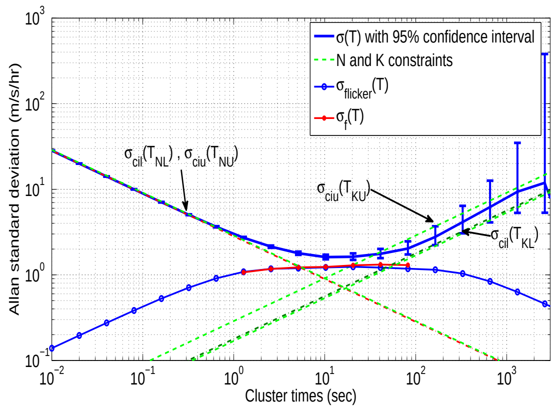


Figure 4.19: Allan variance of a mixture of flicker noise, WN and RW for an accelerometer.

$$\sigma^2(T) = \sigma_N^2(T) + \sigma_{flicker}^2(T) \tag{4.42}$$

Assuming that $\sigma_N(T)$ is given by the lower bound $\sigma_{NL}(T)$, we could have an estimation of the flicker noise with $\sigma_f^2(T)$:

$$\sigma_f^2(T) \simeq \sigma^2(T) - \sigma_{NL}^2(T) \tag{4.43}$$

The resultant $\sigma_f(T)$ can be seen in Fig. 4.20, which shows a very good approximation of $\sigma_{flicker}(T)$. In fact, this could be better in most of the cases compared to the previous curve since there are only two noises and $\sigma_N(T)$ can be estimated accurately by means of $\sigma_{NL}(T)$, this is due to the low uncertainty for the short term noises. It is also noteworthy that when computing Eq. (4.41) or Eq. (4.43) the larger the data set to be analysed the better the accuracy.

To summarize, the constraints of the non-linear fitting are determined taking into account the following observations:

- *N (angle/velocity random walk) and K (rate/acceleration random walk) constraints*: these bounds are set using the confidence interval of the analysed log-log AV curve (*i.e.*, 95% confidence interval expressed by upper and lower limits). The selected bounds should cover the largest segment possible where the noise term is dominant and the uncertainly is relatively low.

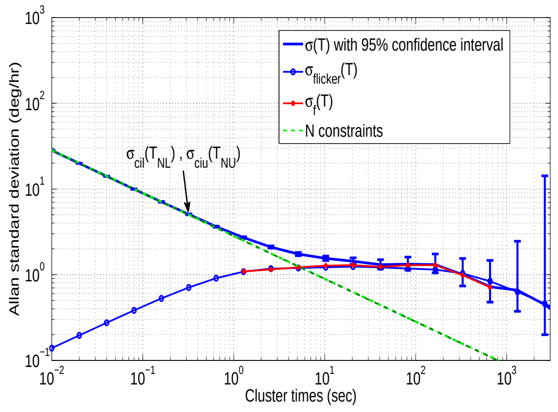


Figure 4.20: Allan variance of a mixture of flicker noise and WN for a gyro.

- $T_{mp} \rightarrow T_c$ *constraints*: they are set in the segment where the flicker noise is dominant. Having a *decade* of difference between the upper and lower limits. In case of several first order GM process these limits may be slightly overlapped.
- σ_{GM} *constraints*: these limits are set from the Allan standard deviation curve where the flicker noise is the dominant noise (upper bound) until half *decade* below it (lower bound).

Estimator

After having identified the stochastic model and defined the constraints, we estimate the unknown parameters for each noise term computing AV and performing a non-linear fitting between the log-log AV curve and the objective function (*i.e.*, the function that is defined according to the different noise sources that have been identified). The condition for the minimization of the relative differences between the variance measurements and the theoretical variance is stated in Eq. (4.44) and it can be solved in the least-squares sense:

$$\underset{\theta}{min} \sum_i \left[\frac{\hat{\sigma}^2(T_i) - \sigma^2(\theta,T_i)}{\hat{\sigma}^2(T_i)} \right]^2 \tag{4.44}$$

In this case the objective function is $\sigma^2(\theta,T_i)$, θ are the unknown parameters (*i.e.*, σ_{WN}, σ_{RW}, σ_{GM1}, T_{c1}, σ_{GM2}, T_{c2}, *etc*), T_i is the cluster times and $\hat{\sigma}^2(T_i)$ is the estimate variance. Further details about similar estimators and the classical least square fit on the log-log Allan variance plots by minimizing the relative distance between the curve and the estimate $(estimate - curve)/estimate$ can be found in [6, 8, 30, 31, 140]. By using wavelet variance instead of Allan variance the estimator used for the NLF might be seen as a particular case of the GMWM described in [8].

Optimization

Since the solution of Eq. (4.44) with the classical least square might lead to a local minimum, we have optimized the parameters estimation by means of pattern search. This technique consists in a set of vectors (patterns) that are used to determine which point to search at each iteration, so if the pattern search algorithm finds a point that improves the objective function at the current point, the new point becomes the current point at the next step of the algorithm [141]. This algorithm for optimization is based on augmented Lagrangian patter search and the complete description can be found in [142, 143].

It is also worth mentioning that if one of the bounds (constraints) is reached by the unknown parameter after the optimization, the bound must be modified, since it might be an indication that the parameter is not limited by the constraints.

In order to evaluate the benefits of the NLF, in the next section we will analyse one simulation assuming that the inertial sensors are influenced by different noise sources that are typically found in MEMS IMUs.

Simulation

In this section we analyse the performance of the non-linear fitting with constrains (NLF) compared to the constrained Expectation Maximization (EM) described in [21], these algorithms were developed in MATLAB, where the optimization stage for the NLF was achieved using the Global Optimization Toolbox. In order to evaluate different stochastic processes that are often found in inertial sensors based on MEMS technology, we simulated and analysed a noise composed by white noise (WN), random walk (RW) and a flicker noise with two first order Gauss-Markov processes. The simulation generates 100 times the noise source and each time with 2^{20} samples, having a sampling frequency of 100 Hz. The true values for each parameter are given by

$\sigma^2_{WN} = 6.13 * 10^{-5}$, $\sigma^2_{GM1} = 1.14 * 10^{-7}$, $T_{c1} = 20.74$, $\sigma^2_{GM2} = 2.36 * 10^{-8}$, $T_{c2} = 2.07$ and $\sigma^2_{RW} = 7.36 * 10^{-11}$, the units are in *seconds* and $(m/s^2)^2$ for the correlation time and the variances, respectively. To estimate these parameters with the NLF we use the following objective function:

$$\sigma^2(\theta,T) = \frac{N^2}{T} + \sigma^2_{M1}(T) + \sigma^2_{M2}(T) + \frac{K^2T}{3} \tag{4.45}$$

This function is used to fit the log-log AV curve of the simulated process. The velocity random walk (N) and the acceleration random walk (K) are directly related to the WN and the RW processes, respectively, while $\sigma^2_{M1}(T)$ and $\sigma^2_{M2}(T)$ are the Allan variance functions of the first order GM process stated in Eq. (4.19). So the goal is to estimate the unknown parameters of Eq. (4.45), which are given by:

$$\theta = \{\sigma_{WN},\sigma_{GM1},T_{c1},\sigma_{GM2},T_{c2},\sigma_{RW}\} \tag{4.46}$$

The log-log AV curve of one simulated noise is depicted in Fig. 4.21. According to this plot, we set the constrains for parameter K at 327.7 *sec* where it meets the upper and lower bounds of the 95% confidence intervals curve, which corresponds to 3.94 $m/s/h$ and 2.31 $m/s/h$, respectively. The upper and lower bounds are set drawing straight lines with a slope of 1/2, these lines meet the two points of the confidence interval curve 3.94 $m/s/h$ and 2.31 $m/s/h$, respectively. The two points need to be carefully selected in order to cover the segment of the curve where the noise (K) is dominant. In case of parameter N, we defined the upper bound at 3.59 $m/s/h$ and the lower bound at 3.50 $m/s/h$ Allan standard deviation, taking into account that the 95% confidence interval curve was evaluated at 0.64 *sec*. These constraints are shown with green dashed lines of slope $-1/2$ (see Fig. 4.21). We defined these constraints since they cover most of the segment of the curve where the long-term noise (K) and the short-term noise (N) are dominant. Regarding the constraints of the first order GM processes, the correlation time is determined considering the segment of the curve where the flicker noise is dominant, this is approximately between 2 *sec* and almost 200 *sec*. According to the observations given in Section 4.5.3, two first order GM process could be suitable for modelling this flicker noise since it seems to be dominant in two *decades*. Therefore, the lower and upper bounds for the correlation time were placed between (20,200) *sec* and (2,20) *sec* for T_{c1} and T_{c2}, respectively. In order to cover the region where the flicker noise is more likely to exist, the limits for the standard deviation of the two first order GM processes (σ_{GM1},σ_{GM2}) were set between 1.57 $m/s/h$ and 0.49 $m/s/h$. That is, half *decade* below the Allan standard deviation corresponding to the segment of the curve where the flicker noise is dominant). The region where the two first order GM processes are suspected to be dominant is also shown in Fig. 4.21. It should be mentioned that when we applied the NLF we only fit the objective function to Allan standard deviation below 655.33 *sec* cluster times since the uncertainty above this time is high.

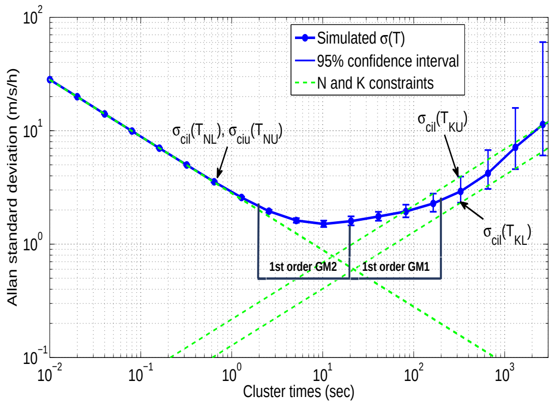


Figure 4.21: Allan variance of a mixture of WN, two first order GM processes and a RW.

The results of the 100 simulations are presented in Fig. 4.22 and Fig. 4.23, where the true values of the parameters (θ) are drawn with horizontal red lines. Note that AV is not included because it is limited when dealing with signals composed by a combination of various processes.

It is also noticed that the parameters estimated are highly affected in case of the EM algorithm, opposite to the NLF which provides consistent results of the parameters that represent the noise modelled, and despite the EM has smaller variance in parameters such as σ_{RW}, the bias of the NLF is smaller than the EM in most of the parameters estimated. As it was shown in [21] the constrained EM can be applied in a short data set compared with the NLF that requires the AV estimation. Despite this, the EM

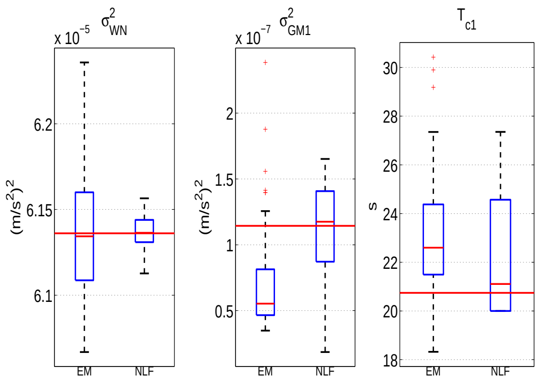


Figure 4.22: Performance comparison between EM and NLF for 100 realizations of a combination of WN, two first order Gauss-Markov processes and a RW.

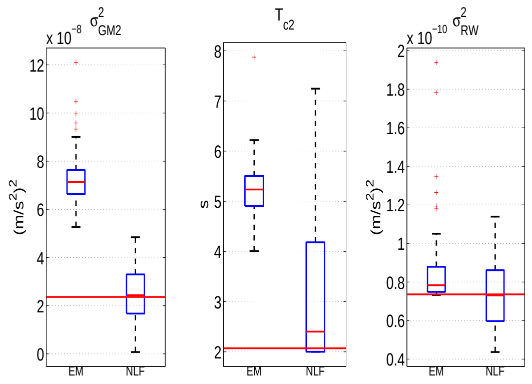


Figure 4.23: Performance comparison between EM and NLF for 100 realizations of a mixture of WN, two first order Gauss-Markov processes and a RW.

is very likely not to converge to a local maximum of the likelihood function when the SSM of the signal modelled becomes complex (*e.g.*, a noise source composed by WN and various first order GM processes). Additionally, it is very dependent on the initial conditions.

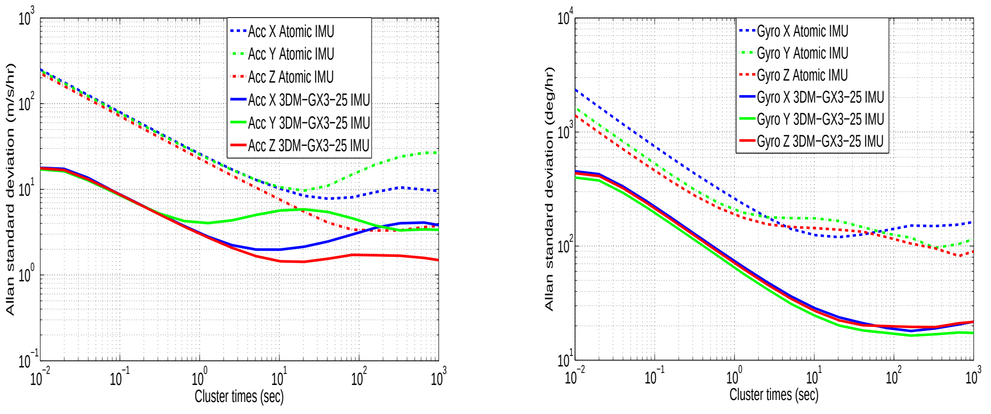
It should be mentioned that the number of parameters (θ) defined for each Allan variance curve depends on the number of noise sources that are suspected to exist, therefore to analysed the curves it is necessary to take into account the slopes that are shown in Fig. 4.3. In case of flicker noise, the number of Gauss-Markov process that we consider is related to the number of *decades* where it is the dominant noise, as it was mention previously, if the segment curve where the flicker noise is dominant occupies two *decades* the number of GM process used to create the stochastic model will be two. According to the typical Allan variances curves for MEMS inertial sensors, one can see that the flicker noise is presented in the middle of N and K, just in the curve segment where they are less dominant.

Next section we analyse the performance of the NLF with a real data set collected from two low cost Inertial Measurement Units.

NLF Applied on a Real Data Set

Two Inertial Measurement Units were involved to estimate the stochastic error model parameters by using the constrained non-linear fitting. Both of them were configured with a sampling frequency of 100 Hz. The test experiment was conducted in a room temperature and seven *hours* of static data was collected to analyse the raw measurements with the NLF. For further details of these IMUs refer to [89, 90].

Figure 4.24(a) shows the estimated AV for the accelerometers of both MEMS IMUs. It can be noted that the velocity random walk (N) of the Atomic IMU is larger than the noise term (N) for the 3DM-GX3 IMU, (*i.e*, approximately 10 times). On the other hand, Fig. 4.24(b) shows the log-log AV curve for the gyros of both MEMs IMUs, where it can be noted that both of them are affected by an angle random walk and a flicker noise.



(a) (b)

Figure 4.24: (**a**) 3DM-GX3-25 IMU and Atomic IMU Allan variance for accelerometers; (**b**) 3DM-GX3-25 IMU and Atomic Allan variance for gyros.

For illustrative purposes we only present the stochastic error parameters of two inertial sensors *i.e.*, one accelerometer and one gyro corresponding to the Atomic and the 3DM-GX3 IMUs, respectively. The curves of the other sensors are similar to those analysed as well as the procedure that is achieved to obtain the parameters. In order to applied the NLF method the z-axis accelerometer was chosen for the IMU 3DM-GX3-25 while for the Atomic IMU we selected the z-axis gyro.

Figure 4.25 shows the correspondent AV curve for the z-axis accelerometer in blue (circle market), where we assumed that the objective function is composed by a WN, a first order GM process and a RW, since it seems to be the model that would better represent this log-log AV curve. Thus, the unknown parameters are $\theta = \{\sigma_{WN}, \sigma_{GM}, T_c, \sigma_{RW}\}$ and the constraints for the N and K parameters were set as it was explained in the previous section, considering the 95% confidence interval values at 0.32 *sec* and 327.70 *sec*, respectively. Concerning the constraints for the parameters of the first order GM process, they are set between 10 *sec* and 100 *sec* for T_c and between 0.38 $m/s/h$ and 1.21 $m/s/h$ for σ_{GM}, which covers the region where the flicker noise is suspected to be. Fig. 4.25 also depicts the result after applying the NLF with the previous constraints, the cyan solid line that superimposes the AV of the z-axis accelerometer represents the fitted curve performed by the NLF method, while the blue line (triangle marker) represents the first order GM process estimated. The parameters obtained for this inertial sensor are $\theta = \{0.0079, 2.4830^{-4}, 18.1401, 6.2559 * 10^{-6}\}$ in *seconds* and m/s^2.

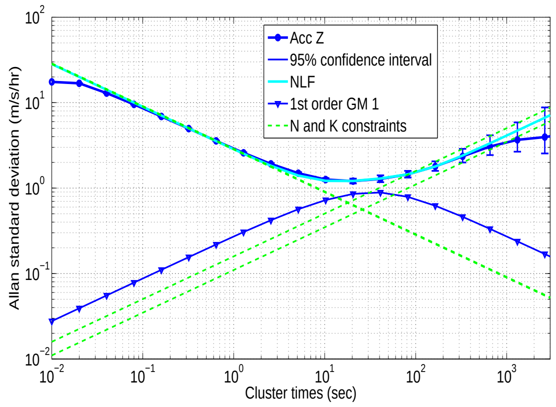


Figure 4.25: Performance of the NLF for the IMU 3DM-GX3-25 accelerometer Z.

Moreover, the log-log AV curve for the Atomic IMU z-axis gyro is depicted in Fig. 4.26. It shows that the inertial sensor is dominated by a WN in the short-term clusters while for the long-term cluster it is dominated by a flicker noise. So in this test we considered a objective function formed by a WN and three first order GM processes since the flicker noise seems to be dominant at least for three *decades*. In this way, the goal is to estimate the parameters $\theta = \{\sigma_{WN},\sigma_{GM1},T_{c1},\sigma_{GM2},T_{c2},\sigma_{GM3},T_{c3}\}$. The constraints for the parameter N were set at 0.16 *sec* which is associated to 372.17 *deg/h* and 368.73 *deg/h* according to the 95% confidence interval curve. In order to cover the region where the flicker noise is more likely to exist, the bounds for the standard deviation of the three first GM processes were set between 142.1 *deg/h* and 44.93 *deg/h*, while the lower and upper bounds for the correlation time we selected at (10,100) *sec*,(1,10) *sec*, and (100,1000) *sec* for T_{c1},T_{c2} and T_{c3}, respectively.

Fig. 4.26 plots the fitted curve (cyan line) after performing the NLF with the mentioned constraints. It can be seen that it matches very well to the AV curve of the z-axis gyro, showing that the sum of multiple first order GM processes is suitable in this inertial sensor. The three first order GM processes estimated by the constrained non-linear fitting are also depicted, which provides information of the underlying noise sources that are combined to build up the flicker noise. The set of parameter in units of *seconds* and *deg/s* are given by $\theta = \{0.41,0.03,17.12,0.03,1.59,0.02,211.63\}$.

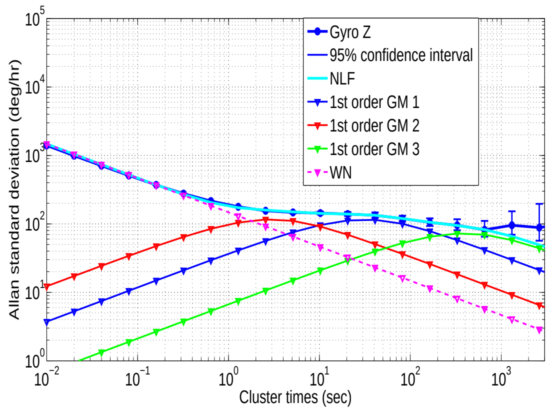


Figure 4.26: Performance of the NLF for the Atomic IMU gyro Z.

Despite the fact that there is not a reference to compare with the estimated parameters as in the previous section, we evaluate the goodness of the fit by means of the Normalized-Root-Mean-Squared-Error(NRMSE) (see [144]). Thus, we consider that the NLF makes a suitable fitting of the AV curve when the fitness value is greater than 95%.

NLF Bias Model Adapted to the Loosely-Coupled KF

Having applied the NLF, we adapted the obtained parameters in the loosely-coupled GPS/INS integration in order to compensate the bias-drift of accelerometers and gyros of the IMUs under test. For the accelerometers of the Atomic IMU, the stochastic error a_{se} was modeled as:

$$a_{se} = WN(N) + 1^{st}\, GM_1(B) + 1^{st}\, GM_2(B) \tag{4.47}$$

where the noise term associated to N is modeled as white noise (WN) and the flicker noise is modeled as a superposition of two first order Gauss-Markov process. Regarding the gyros, the stochastic error g_{se} was defined as:

$$g_{se} = WN(N) + 1^{st}\, GM_1(B) + 1^{st}\, GM_2(B) + 1^{st}\, GM_3(B) \tag{4.48}$$

where we have a white noise (WN) plus a flicker noise modeled as a superposition of three first order Gauss-Markov process.

On the other hand, it is noteworthy that the log-log AV curves obtained for the 3DM-GX3-25 IMU in Section 4.3.4 vary with respect to the ones showed in Fig. 4.24(a). This is because we work with three MEMS IMUs, (*i.e.*, two 3DM-GX3-25 IMUs and one Atomic IMU), one 3DM-GX3-25 IMU from the Navsas research group and the others from the Department on Microelectronics and Electronics Systems of Universitat Autónoma de Barcelona (UAB). So the analysis described in previous sections was achieved using the 3DM-GX-25 IMU available in the Navsas research laboratory [122]. For the 3DM-GX3-25 IMU available in the UAB, we applied the NLF and we noted that the error model that fitted better for this IMU was the following:

For the accelerometers of the 3DM-GX3-25 the stochastic error a_{se} was:

$$a_{se} = WN(N) + 1^{st}\, GM_1(B) + 1^{st}\, GM_2(B) \tag{4.49}$$

it is composed by a white noise (WN) and a flicker noise modeled as a superposition of two first order Gauss-Markov process. For the gyros, we obtained a white noise (WN) plus a flicker noise modeled as a sum of two first order Gauss-Markov process ($1^{st}\, GM$).

$$g_{se} = WN(N) + 1^{st}\,GM_1(B) + 1^{st}\,GM_2(B) \tag{4.50}$$

In order to compare the performance of the stochastic models obtained with the NLF, we not only estimate the noise parameters with this method but also with AV following the procedure described in Section 4.5.1. The comparison between both of them will be presented in Chapter 6.

Temperature Test for the Stochastic Error

Most of the reported works in the literature disregard the stochastic error variations at different temperature points [29]. Indeed, the analysis of the Allan variance curves is typically achieved at an specific temperature point, except for some researches where AV curves have been taken into account with various temperature points [145, 146]. Therefore, El-Diasty in [29] introduces a temperature-dependent stochastic model for inertial sensors, where autoregressive models are employed to estimate the parameters of a first order GM process at different temperature points. Despite the fact that this is a novel error model, we consider that a more appropriate error modelling can be determined. For this reason, we decided to develop a temperature-dependent stochastic error based on the NLF. Thus, in order to analyse the behaviour of the bias-drift at different temperatures points with the NLF we focused on the 3DM-GX3-25 MEMS based IMU. For this purposed, we placed the IMU in the thermal chamber available in the Department of Microelectronics and Electronics Systems at the UAB. During the temperature test, we collected static data sets from the IMU at different temperature points, specifically, 10 $^{\circ}C$, 20 $^{\circ}C$, 30 $^{\circ}C$ and 40 $^{\circ}C$. The 3DM-GX3-25 IMU was configured in the same way as in Section 4.3 with a sampling rate of 100 Hz. Then, we recorded seven $hours$ of static data at each temperature point ($i.e.$, a total of 28 $hours$). Subsequently, the AV algorithm was computed for each seven $hours$ of the different temperature points. The y-axis accelerometer and gyro of the 3DM-GX3-25 IMU were chosen to illustrate this analysis. The other sensors gave similar results. Fig 4.27(a) and Fig. 4.27(b) show the log-log Allan variance curves for y-axis accelerometer and y-axis gyro of the IMU at different temperature points, respectively. A temperature dependency of the noise terms that are affecting the inertial sensors can be appreciate in the plots. For instance, the y-axis accelerometer has a significant variation in the flicker noise, this is in a segment of the curve where the AV estimation has a relatively low uncertainty (*i.e.*, for cluster times between $2 * 10^{-1}$ *sec* and 20 *sec*). On the other hand, it can be noted that the y-axis gyro presents a temperature dependency for the short term noise angle random walk (slope $-1/2$), (*i.e.*, for cluster times below 10 *sec*, where the noise term (N) is more dominant). For cluster times above 20 *sec* is not easy to determine an accurate variation of the noise term involved, this is due to the fact that the AV uncertainty is high. Despite this, a temperature dependency of the noise terms is observed in cluster times where the log-log AV curve has a low uncertainty, which means, that we could estimate the noise parameters through the NLF at each temperature point in order to examine how these parameters that can be well estimated at different temperature points affect the performance of the system.

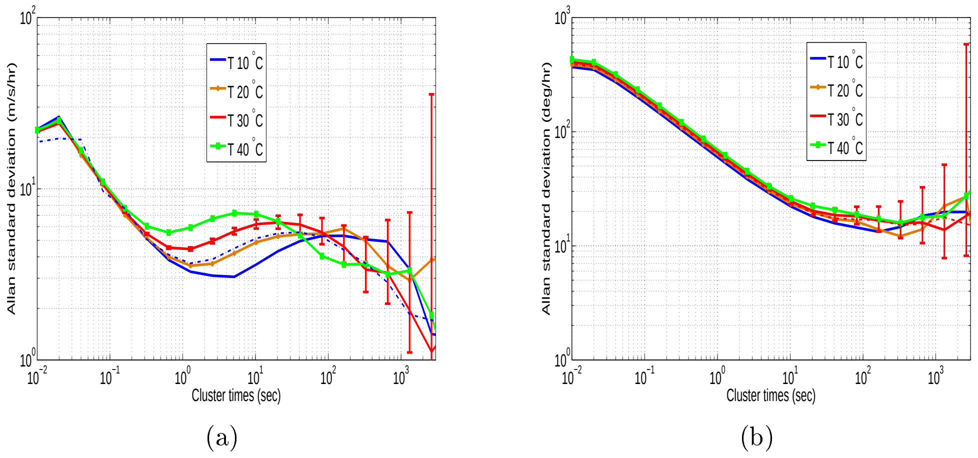
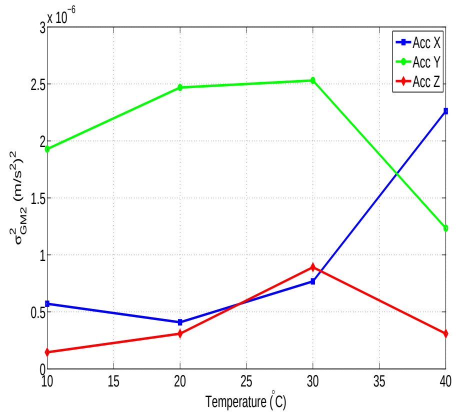


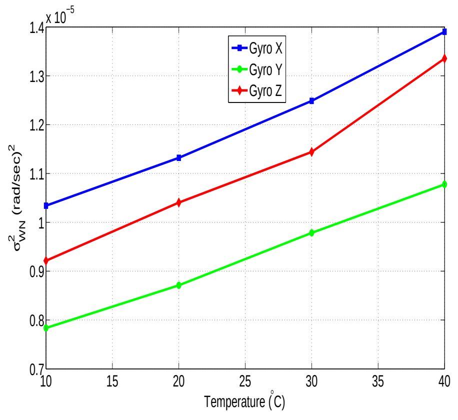
Figure 4.27: (**a**) IMU 3DM-GX3-25 Allan variance for accelerometer Y at different temperature points; (**b**) IMU 3DM-GX3-25 Allan variance for gyro Y at different temperature points.

For building the stochastic error model temperature dependent, we applied the NLF to each AV curve obtained at different temperature points. The objective function that we used to fit each curve is the one related to the stochastic models described in Eqs. (4.49) and (4.50).

Figures 4.28(a) and 4.28(b) show some of the estimated parameters for accelerometers ans gyros (*i.e.*, the covariance of a first order GM process (σ^2_{GM2}) for the three accelerometers and the covariance of the white noise (σ^2_{WN}) for the three gyros). These parameters were obtained after computing the NLF and it is clear that the angle random walk (N) has a linear tendency. In fact, the experiment was repeated on different occasions and we observed that the angle random walk noise increased with temperature.



(a)



(b)

Figure 4.28: (**a**) Covariance of a first order GM process (σ^2_{GM2}) for the three different accelerometers at different temperature points; (**b**) Covariance of the white noise (σ^2_{WN}) for the three gyros at different temperature points.

After this temperature test, we concluded that it was necessary to include the stochastic error model temperature dependent in the loosely-coupled GPS/INS integration in order to examine its performance. The results for this error model are shown in Chapter 6.

Chapter 5

Architecture Based on FPGA for GPS/INS Integration

5.1 Introduction

Although the GPS is highly portable, with low power consumption and dominates the market in positioning and navigation [75], the reability, continuity and availability in the navigation applications where it is used is compromised in different situations, *e.g.*, in urban canyons and tunnels where there is not line-of-sight between the GPS receiver and the satellites. In fact, these are not the only errors that affect the GPS, there are also factors such as jamming, multipath, interference, *etc.* Given the vulnerability of the GPS receivers, in the last years it has been supported by additional information sources in order to improve the navigation solution. Since most of the navigation systems require redundant information, we consider that FPGAs (Field Programmable Gate Array) are suitable for their implementation because they are very flexible devices that allow to easily adapt several instruments such as compasses, GPS receivers, odometers, cameras even multiple Inertial Measurement Units (IMUs), which can lead to redundant information sources and in turn a better performance of the navigation system. Additionally, they are being used in a wide range of applications that require computation-intensive, due to the fact that they have different features such as embedded processors, DSP blocks, high speed serial communication and high density in gates. FPGAs can be programmed to achieve several applications with a high performance, taking advantage of their high grade of parallel processing and their capacity of run-time re-configurability.

Although, recently some works have being conducted to implement GPS/INS integration on compact and flexible platforms, highlighting between them [35–39], we noted that in most of the cases the FPGA is used as an interface for data acquisition and in approaches where it is used to compute the navigation algorithms the hardware resources available are underutilized. In fact, this is where the FPGA could offer more benefits if several information sources are included. It is also worth noting that the use of low cost INS implies high computational cost due to the complex error models that are needed to enhance its performance. Therefore, in this chapter we aim to develop an embedded system that combines GPS/INS with an architecture based on FPGA technology, where we will analyse the feasibility to use parallel processing with DSP blocks for the computation involved in the Extended Kalman Filter (EKF).

This chapter is organized as it follows. The first section presents a review of recent similar platforms where the GPS/INS integration has been implemented. Second section gives a general description of FPGAs and the requirements and specification of the embedded system. The third section, explains the architecture developed for the loosely-coupled GPS/INS integration based on FPGA. This also gives details about the software/hardware implementation, utilization resources and matrix multiplication in hardware as a possibility to speed-up the system. Finally, the applications where the navigation platform was assessed are presented.

5.2 Previous Work

Recently, studies have been conducted to carry out the GPS/INS integration using platforms such as DSPs (Digital Signal Processors) and FPGAs (Field Programmable Gates Arrays) [35–38, 147]. References [37, 147] present the integration GPS/INS using DSP and FPGA. In this case a floating-point DSP is used to implement the Kalman filter and navigation equations while the FPGA is used as interface with instruments such as the GPS. Basically, the FPGA is responsible for the acquisition, communication and control. Although these papers propose novel systems combining

FPGA and DSP, platform size could be reduce using only one of them. At the time, one of these works was under development and the other one had been tested using a spacecraft simulator called Flight Dynamic Controller (FDC). A similar work using DSP/FPGA was described in [35]. It summarizes various GPS/INS configurations developed in different boards that were reported in the literature at the time and also details the computational cost required by the KF and the INS algorithms involved in the loosely-coupled approach with 9 states (*i.e.*, position, velocity and attitude). Although this work implements some hardware modules on the FPGA, such as a GPS receiver serial link interface, these modules are available in most of the current FPGAs development boards. Even though the PCB designed for the inertial sensors interface provides little versatility, this paper shows a novel approach in acquiring signals from IMU and GPS for a given Kalman filter to fuse the data by means of DSP/FPGA boards. In [36] the GPS/INS integration is carried out using an FPGA. This uses the soft-core processor Nios II in conjunction with eCos (Embedded Configurable Operating System), which is responsible for managing the user interface, the synchronization time, the INS and Kalman filter algorithm. The are some components that are designed by using the AHDL (Altera Hardware Description Language). This project shows good results and also proposes a compact platform. However, it uses AHDL language which limits the design since this is specifically for Altera, consequently it is not an IEEE standard as VHDL (Very High Speed Integrated Circuit Hardware Description Language). In [38], a compact platform based on FPGA is described. It makes the GPS/INS fusion by using the hard core processor PowerPC 440, the navigation algorithm is developed in C language and the complete system is evaluated with simulated trajectories. Despite this, there is no any details about the computational cost required by each stage and neither the utilization resources. In [148], a new strapdown algorithm in a single FPGA chip is developed. Although it does not implement the GPS/INS integration it describes the parallelization of the mechanization and its computational complexity. It shows that hardware can greatly decrease the execution time, nonetheless, we consider that the effort to reduce the computational cost in the GPS/INS integration should be focused on the EKF, which is a critical block of the navigation system as it will be pointed out in the following sections.

5.3 Platform Based on FPGA

Despite the fact that there are different hardware platforms that can be used to develop an embedded system such as microcontrollers, digital signal processor DSP, application specific integrated circuits (ASIC) [147, 149–151]. The selection of a platform over others depends not only on the requirements of the system to be developed such as the performance, power consumption, cost per chip, but also on the ease of the tools accompanied by a specific platform to assist the developers [152].

On one hand, the DSP can be suitable for the GPS/INS integration since some of these processors are able to work with floating point, they also have a specialized data-path for digital signal processing and can be programmed in C language. Nonetheless, adding new sensors to the system might result difficult in most of the situations. Although there is a wide discussion in regard to the use of FPGA and DSP technologies, we consider that FPGA can offer great advantages in this sort of navigations systems since it would be enough to implement the loosely-coupled. However, the FPGA implementation presents greater challenges than the DSP. On the other hand, from the works that combine DSP/FPGA, we noted that the FPGA makes easier the reading from the navigation instruments and the DSP is appropriate to achieve the digital signal processing involved in the KF to fuse the data. However, it might imply higher power consumption, a less compact platform, additional synchronization stages between the two devices and a design of different PCBs to fed all the circuit boards.

For the loosely-coupled approach focused on this thesis, the FPGA was chosen because of the wide variety of advantages that it offers over similar platforms. The FPGA is very flexible to adapt new information sources in the navigation system (*i.e.*, magnetometers, cameras, RF receivers, bluetooth devices *etc.*), it has great capacity for parallel processing that could reduce the computation time, which is an important factor in real-time implementations. Additionally, FPGAs have a low risk design methodology. Nowadays they come together with resources like processors, modules to handle high-speed communications, Floating Point Unit (FPU), DSP blocks, *etc.* These devices have the benefits of the hardware such as speed and flexibility, apart from their simple design cycle. This means that if there is an error in the design, it just needs to modify the code and reprogram the FPGA, and does not have a high non-recurring engineering (NRE) as in the case of an Application Specific Integrated Circuit (ASIC).

The navigation platform based on FPGA that was developed will be explained in the following sections, but first we will make a brief introduction about FPGA devices and subsequently the details of the prototype will be presented.

5.3.1 Field Programmable Gate Array (FPGA)

The field programmable gate array is an integrated circuit (ICs) that belongs to the family of programmable logic devices (PLDs). Since its creation in the decade of the 70's it has been characterized by offering flexibility in the designs, due to the fact that these devices have a large amount of gates that are programmable in the field to perform different functions. An FPGA chip basically includes I/O blocks and a core programmable fabric [153]. The I/O blocks are located around the periphery of the chip and provide programmable inputs and outputs that allow interaction with external devices. On the other hand, the core programmable fabric consists of an array of logic blocks also called configurable logic blocks (CLBs) that are interconnected by programmable connections. Depending on the technology of FPGAs, it can be programmed only once while others can be programmed many times.

Modern FPGAs incorporate resources like RAM blocks, DSP blocks for the realization of MAC operations, high logic density, communication modules in the order of Giga bits, clock management modules, embedded processors, *etc.* These components have made FPGAs an attractive alternative for the development of System-On-Chip designs [153].

Figure 5.1 illustrates the general layout of the FPGA architecture of a Xilinx device. In this scheme shows the programmable logic blocks, the programmable interconnections that are part of the core programmable and the I/O blocks. All the blocks are highly configurable and can be used to implement large and complex functions that had previously been the domain only of ASICs [154]. The functions developed by the user are mapped into the logic blocks of the device performing

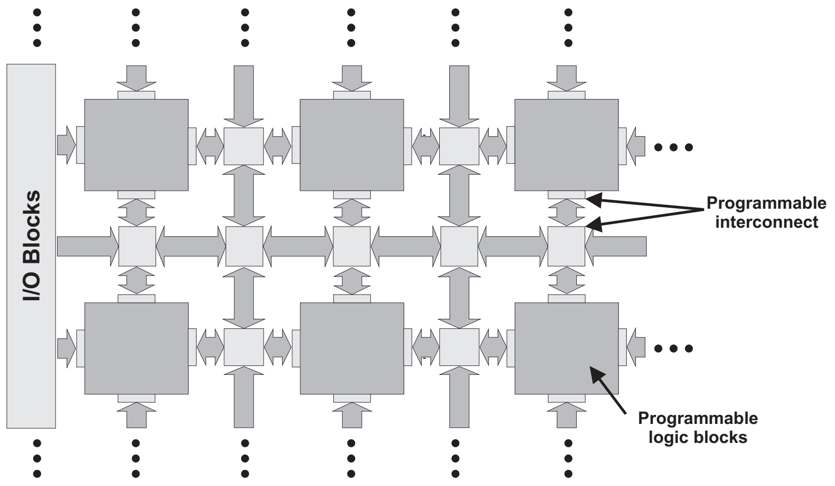


Figure 5.1: Generic FPGA architecture.

the necessary connections between them. Thus, configurable interconnection lines are programmed making connections between different logic blocks in order to achieve a specific task.

The design methodology for FPGAs consists in two basic methods, bottom-up and top-down [155]. The bottom-up description of a system begins with the implementation of basic components that are gathering to form more complex modules until the complete system is developed. In contrast, the top-down begins with a high level description where the main blocks of the system are identified and then it is divided into blocks that have less complexity and are easier to implement. Here we adopted the top-down methodology as it allows error corrections at early stages of the development cycle.

5.3.2 Requirements and Specifications

In order to make the implementation of the GPS/INS integrated system on the FPGA, we defined the following specifications and functional requirements for this navigation embedded system:

- Interface with the Atomic IMU, 3DM-GX3-25 IMU, u-blox LEA-6T GPS receiver and a bluetooth component for communication with external devices, it could be a PC or an Android device that will interact with the platform *e.g.*, to collect data during campaigns or to display the trajectory of the vehicle. Each of these devices are provided with a UART serial communication interface.

- Synchronization between different IMUs and GPS (*i.e.*, one IMU at the time) using the Time Of Week (TOW) provided by the GPS. The inertial sensors will provide data every 10 *msec* while the GPS receiver every second.

- Drivers for both IMUs and GPS that allow to obtain information that will be used later in the navigation algorithm. It requires programming the Atomic IMU microcontroller to send inertial sensor samples every 10 *msec* and also adapt a sensor to monitor the temperature of the device. The 3DM-GX3-25 IMU needs to be configured to send a specific packet every time the FPGA is power up, so it will be able to provide raw measurements of angular velocity, acceleration and Euler angles every 10 *msec*. The drivers for the GPS should extract position and velocity information every second.

- Implement the loosely-coupled GPS/INS integration algorithm using an architecture based on FPGA, which will be assessed with simulated and real data collected during campaigns.

- Store the navigation solution in a external memory or send it via bluetooth to a remote device such as a mobile device or PC. The developed architecture for the navigation embedded system should be stored in an external non-volatile flash memory, thus it does not need to be programmed every time a vehicle test is performed.

To test the performance of the navigation platform based on FPGA, the following two approaches are considered:

- Acquisition: to evaluate the data acquisition of the different devices, the FPGA platform is mounted in a robot and a land vehicle under several scenarios, where it is configured to read data coming from IMU and GPS. The data collected is synchronized and stored in a compact flash memory or sent to a PC via bluetooth.

- Navigation algorithm: in order to assess the performance of the GPS aided INS, the FPGA is reconfigured with the navigation algorithm. Then, the raw measurements that were collected during the campaigns are stored in an external DDR SDRAM memory available on the XUPV5 board from Xilinx, where once the algorithm stars running the measurements are loaded from the memory and fed into the loosely-coupled GPS/INS integration. To analyse the computational cost, a timer is added to measure the execution time needed for each stage involved in the navigation algorithm. The off-line solution is also compared with the algorithm developed in Matlab. It is worth mentioning that the extracted information from both IMUs will be evaluated independently in the integration with the GPS and not as a data fusion of multiple IMUs.

5.4 GPS/INS Integration Based on FPGA

The platform for the GPS/INS data fusion is shown in Fig. 5.2. It presents the Virtex-5 ML509 board, the IMUs, the GPS and an Android device. There are three UART interfaces which carry out the data acquisition from the navigation instruments. The measurements are stored in a 1 GB compact flash memory that is in the bottom of the board, although there is also the option to send them via Bluetooth through a UART peripheral. That is, using a serial communication interface, where we connect a UART-Bluetooth converter (Bluemore200), which is suitable for integration in microprocessor systems without operating system since it does not need a driver to work [156]. Thus, the platform is able to send position data to an Android device, where we can monitor the trajectory of the vehicle. The communication with external devices and visualization of the trajectory were used during the robot test since the user interface developed in the Android device not only receives information from the FPGA but it is also designed to send commands to the FPGA in order to have interaction with the Virtex-5 ML509 platform.

The following sections describe the customization of hardware and software development that is necessary to build the embedded navigation system.

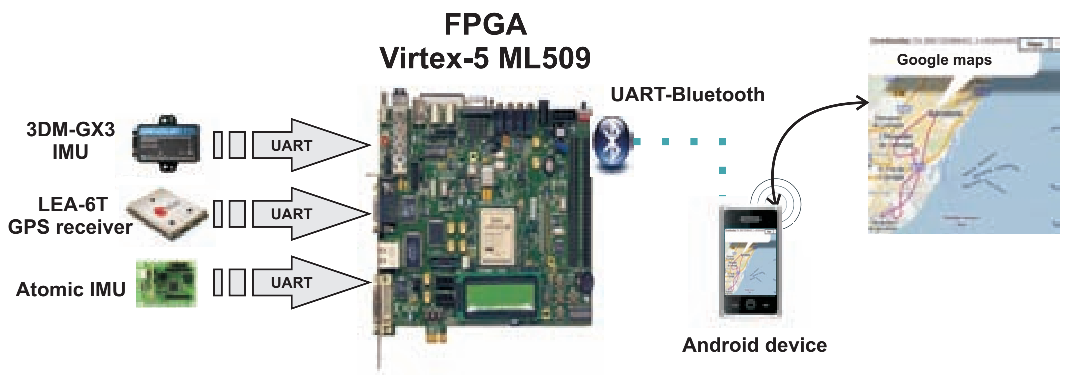


Figure 5.2: General system components.

5.4.1 Hardware Implementation

To implement the navigation system on the Virtex-5 ML509 development kit, we used the ISE Design suite 12.1 and also the EDK (Embedded Development Kit) from Xilinx. The Microblaze processor version 7.10.*d* was customized to run at 125 MHz with an IEEE-745 compliant single-precision Floating Point Unit (FPU). This soft-core processor is implemented using the logic primitives of the FPGA and this is part of the Xilinx embedded solutions. This soft-core processor is an Intellectual property (IP) core that has a rich instruction set optimized for embedded applications, which gives complete flexibility to select the combination of peripherals, memories and interfaces, so it provides the exact system that is necessary at the lowest cost possible [157, 158]. Thereby, the architecture was customized with three UART peripherals, two of them to receive data from the IMU and GPS, whereas the third one to communicate with the Android device through a UART-Bluetooth converter (see Fig. 5.3).

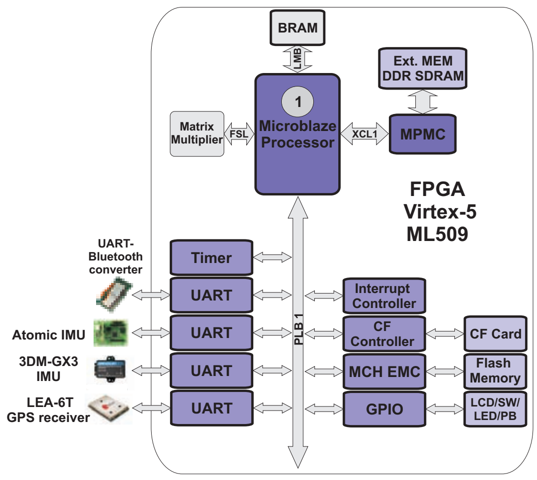


Figure 5.3: Architecture for the GPS/INS integration.

The Microblaze processor is connected to the Multi-Port Memory Controller (MPMC) which manages the external memory (DDR SDRAM) where the software application will be stored. The access to the external DDR SDRAM is performed by the Xilinx CacheLink (XCL) that is a high performance solution to connect directly to the memory controller MPMC [158]. Regarding the Timer IP core, this is added in order to log the IMU samples and also to measure the execution time that takes different stages of the algorithm. The architecture includes the system ACE CF controller that uses the memory based on the Compact Flash (CF), which not only permits to store the raw measurements but also supports multiple-bitstream files if additional architectures are required. Moreover, the GPIO is the general purpose I/O core, which enables interface to leds, dip switches, LCD display and push bottoms. These I/O peripherals are employed to start/end the system and also to check if the application is running properly. It can be noted that in the same way all these hardware resources were included any information source with a standard interface can be adapted in the architecture by just attaching a custom core or an IP core, which will increase the redundancy of the navigation system and in turn its performance. This is one of the main advantages that FPGAs can offer in the development of navigation systems.

Since XUPV5 board includes a platform flash PROM as most FPGAs development kits, we used the non-volatile memory available on the board to store the complete system (*i.e.*, a linear flash device that provides 32 MB of flash memory). Thus, the System on Chip design can be loaded every time the FPGA is turned-on. Finally, an interrupt controller is attached into the PLB bus for the acquisition of the inertial sensors measurements and the information provided by the GPS. This issue will be explained in the next section.

5.4.2 Software Development

The software development for the navigation application is made up of different blocks as it is depicted in Fig. 5.4. The EKF explained in Section 2.6.3 was implemented considering 15 states, nine for the navigation states and six for the IMU error states as it is stated in Eq. (5.1). The flowchart for this algorithm can be seen in Section 2.6.4.

$$\delta \mathbf{x} = \begin{bmatrix} \delta \mathbf{r}^n & \delta \mathbf{v}^n & \delta \boldsymbol{\psi}^n & \delta \mathbf{b}_{a,b} & \delta \mathbf{b}_{g,b} \end{bmatrix}^T \tag{5.1}$$

The algorithms were developed in m-code with Matlab and also in C language. They were debugged and validated and latter on the C language code was ported to the soft core Microblaze processor. It should be mentioned that although currently there are tools that can be used to accelerate the development of prototypes in hardware, for instance, Matlab HDL coder that converts m-code to HDL code or Matlab coder that converts m-code to C code, in this case we implemented the GPS/INS integration in m-code and we manually converted m-code to C code. Despite the fact that it is a slower process than using Matlab toolboxes, we chose this option because we have more control over the code *e.g.*, it is readable, it is not necessary to alter the Matlab code and besides the code can be optimized more easily to adapt the navigation system to a specific processor, such as Microblaze.

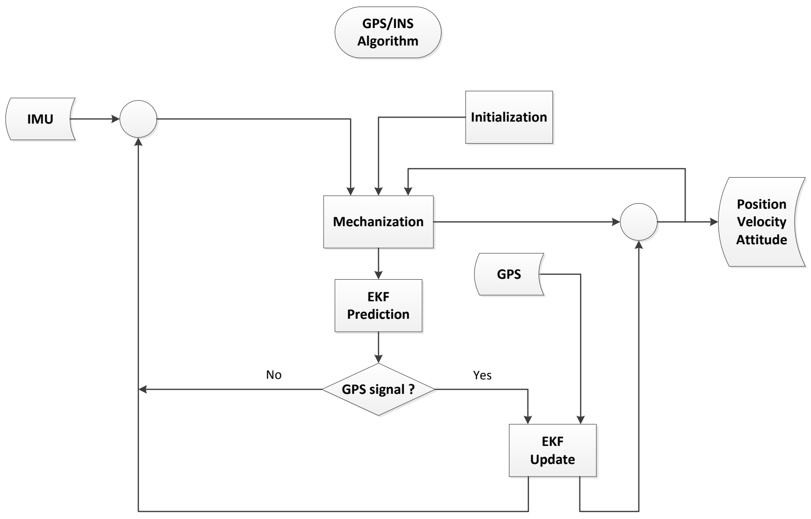


Figure 5.4: Flow diagram for the loosely-coupled GPS/INS integration.

According to the flow diagram shown in Fig 5.4, the initialization provides the initial conditions of the navigation system, *i.e.*, information of position, velocity and attitude. This block can be performed as a separate algorithm, which can be seen as a software application that runs the alignment and it is stored as an image in the flash PROM. The initial position and velocity is given by the GPS while the Euler angles are read from the 3DM-GX3-25 IMU. The process to acquire the measurements from the GPS and IMUs is explained in the following section.

Acquisition

The data acquisition from the GPS and the Inertial Measurement Units was carried out through universal asynchronous receiver transmitter (UART) peripherals. These peripherals are included during the set up of the hardware architecture and are connected to the PLB (Processor Local Bus); each of them were configured with a baud rate of 115200 *baud*.

To obtain the measurements from the Atomic IMU, the inertial sensors signals are passed through a signal conditioning and a low-pass filters stage providing a high level analog output proportional to the acceleration and angular rate measurements. These analog outputs are then sampled by the on-board microcontroller ATMega168TM, that runs at 10 MHz with 8 dedicated 10-bit ADC channels. In order to obtain temperature data during the campaigns, we modified the Atomic IMU from Sparkfun electronics adapting the LM35 temperature sensor. This is a precision integrated-circuit temperature sensor that provides an output voltage linearly proportional to the centigrade temperature and does not require any external calibration to provide accuracies of $\pm 0.75°C$ over a full temperature range $-55°C$ to $+150°C$ [159]. In this case the ATMega168TM is programmed to perform the conversion of each channel of the ADC as it is shown in Fig. 5.5, the seven ADC channels are sampled by the microcontroller to obtain a message packet with acceleration, angular velocity in the three axis and temperature *i.e.*, A_x, A_y, A_z, ω_x, ω_y, ω_z and T, respectively. Given the fact that the ADC is connected to an 8-channel analog multiplexer, the seven entries can not be simultaneously sampled, so there is a delay between each digital sample, this indicates that the samples obtained from the accelerometers, gyroscopes and temperature sensor not correspond to the same instant of time due to the delays caused by the multiplexer. In spite of this, if the ADC conversion time is significantly high, then the effect of errors due to the phase delays will be minimal. This nonsimultaneous sampling will have an impact if the computational cycle time (*i.e.*, approximately 10 ms) is comparable with the conversion time [35]. In the present case it is not critical since the conversion time is about 83.6 μs (13 *cicles*) with a frequency of 156.25 KHz for the ADC. After getting the digital samples for each sensor, the Atomic IMU message packet is sent to the FPGA through the USART interface available in the microcontroller.

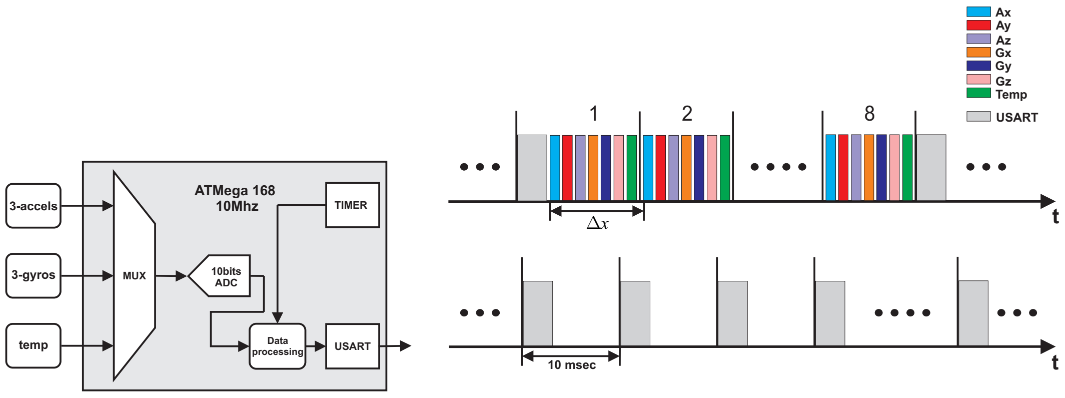


Figure 5.5: Timing diagram for Atomic IMU acquisition.

Since GPS and IMUs provide different sampling frequency, *i.e.*, 1 Hz and 100 Hz, respectively, the synchronization between these instruments is done by the GPS, that is typically used as time reference in multi-mobile sensors systems [160]. Therefore, we used the time of week supplied by the GPS as time stamp on each data received from the IMUs. Although a better synchronization procedure can be implemented, in this first prototype of the navigation platform we consider a conventional method. For further details of recent methods of synchronization refer to [160–162].

Thus, whenever a packet is received from one IMU, it is tagged with the current time of week given by the GPS. Thus, an IMU with a sampling frequency of 100 Hz will have 100 *samples* with the same time of week. To achieve this task we enabled the interrupts in the UART interface, giving priority to the interrupt of the UART associated with the GPS. The interrupts are generated when a valid data exists in the receive register of the UART, then, it is stored into a buffer and when it is complete, this is copied to another buffer, which is processed to extract the data (*e.g.*, position and velocity for the GPS and angular velocity, acceleration and Euler angles for the IMU). Each packet has a specific structure that supports a header for synchronization, payload with a group of messages which are related to position, velocity, angular velocity, acceleration, attitude, *etc*, and a check sum. The algorithms implemented on the FPGA to read the Atomic IMU and the GPS are slightly similar, they are summarized in one flowchart depicted in Fig. 5.6. For the 3DM-GX3-25 IMU, each time the FPGA is turned-on a command needs to be sent to configure the data-stream format. This message format is saved on its on-board processor and then when the IMU is enabled it sends a continuous stream with acceleration, angular velocity and Euler angles with a sampling frequency of 100 Hz. See [121,163] for further details about the structure of each packet sent by the GPS receiver and the 3DM-GX3-25 IMU.

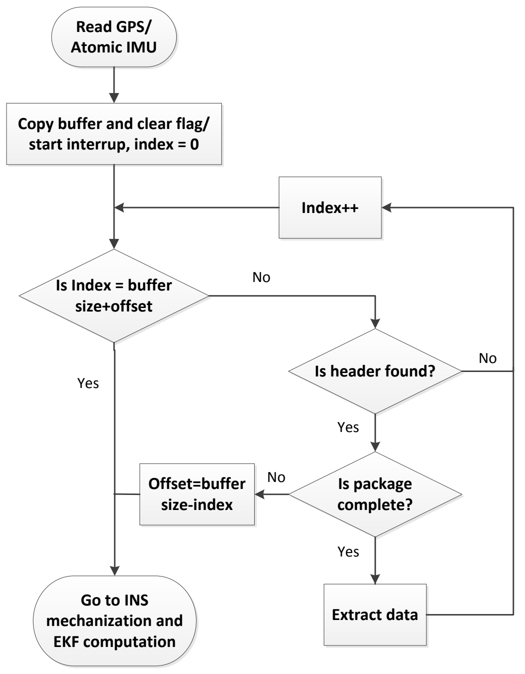

Read GPS/ Atomic IMU
Copy buffer and clear flag/ start interrup, index = 0
Index++
Is Index = buffer size+offset
No
Yes
Is header found?
No
Yes
Offset=buffer size-index
No
Is package complete?
Yes
Go to INS mechanization and EKF computation
Extract data

Figure 5.6: Flow diagram to read GPS receiver and Atomic IMU.

Memory Mapping

Since the Block RAM available to run a Microblaze application is by default 64 KB, we used the external memory DDR2 SDRAM of 256 MB to avoid limitations with regards to the space required to execute the GPS/INS application. Furthermore, in order to evaluate the platform in a terrestrial vehicle, we used a non-volatile device PROM as a platform to store the software application and the configuration data of the FPGA during power-down. The SDRAM and the flash PROM memories are typically included in the development kits based on FPGA; the latter memory is sometimes dedicated to load the FPGA configuration data upon power-up. This non-volatile device can be also employed to hold small amounts of user data or several images to configure the FPGA, it might be useful to store images with different proposes. For instance, in the GPS/INS application one image could have the navigation alignment and another one the computation of the navigation solution; it would be convenient and flexible.

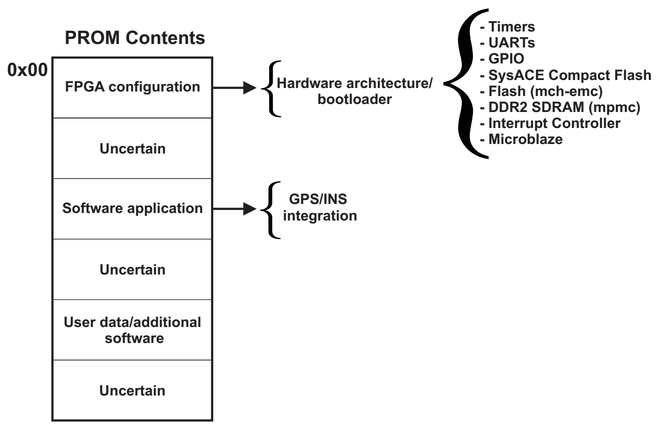


Figure 5.7: Memory map for the embedded system.

Figure 5.7 shows the contents of the flash memory, in this case, the linear flash chip available on the XUPV5 board is used. This non-volatile memory provides 32 MB with 16-bit wide to storage data, software, or bitstream files. The section that includes the software applications storages the executable loosely-coupled GPS/INS integration, while the FPGA configuration section contains the information to configure the logic cells and routing within the FPGA (*i.e.*, the hardware architecture described previously). Additionally, it also storages a bootloader application to initialize the BRAM of the Microblaze.

FPGA Configuration

Having customized the architecture with the features shown in Fig. 5.3, the hardware design is synthesized and the bitstream file is generated with the EDK. Then, in order to configure the FPGA with the navigation system two steps are required. Firstly, the hardware architecture and software application need to be stored in the flash memory and mapped as it was explained in the previous section. Secondly, the GPS/INS software application need to be run from the external DDR SDRAM memory. This is achieved by following these steps:

- Programming the flash memory with the GPS/INS application image. This is loaded into the flash with a offset address, that corresponds to the address memory where the bootloader will look for the software application once the FPGA is switched-on.

- Programming the flash memory with a image containing the hardware platform and the BRAM initialized with the bootloader.

Once these steps are accomplished every time the platform is switched-on the hardware architecture described in Fig. 5.3 is configured on the FPGA from the linear flash. Subsequently, the bootloader, stored in the BRAM, accesses to the pre-determined memory location where the loosely-coupled application was stored in the flash, then copies it from the flash to the DDR SDRAM. Finally, the bootloader application jumps to the extern memory where the the navigation application has already been stored and starts running. The complete process is implemented using the Byte-wide Peripheral Interface (BPI) mode. See further details about this procedure in [164, 165].

Software Profiling

Here we present the execution time of the code implemented on the FPGA that requires a higher computational cost. To measure how long the program takes to execute a piece of code we used a timer/counter peripheral, which count the number of clock cycles between two instructions, where each clock cycle is related to the Microblaze frequency, that is 125 *Mhz*. As shown in Table 5.1 the time required for one mechanization computation is 2.7 *msec*, one Kalman filter computation requires 111.30 *msec*. During the acquisition of the IMU (*i.e.*, considering the 3DM-GX3-25 IMU that provides more samples than the Atomic IMU), the reading takes approximately 0.118 *msec* on average, while for the GPS read and parse take 0.006 *msec* on average. Taking into account that the GPS provides samples every *second* while for the 3DM-GX3-25 IMU every 10 *msec*. Since the algorithm runs in an infinite loop the navigation solution is calculated continuously, providing position, velocity and attitude outputs every 57.72 *msec* which is equivalent to approximately 17 results of position every *second*. This is in the case when the GPS is not available, otherwise if there is a GPS update, the number of results provided by the navigation system is reduced to 16 due to the calculation of the EKF update stage.

A time frame within a *second* for the calculation of the navigation algorithm in μ-Blaze can be seen in Fig. 5.8. The time segment that takes 114.12 *msec* to compute

Table 5.1: GPS/INS integration real-time feasibility using XUP-V5 board.

	Acquisition GPS (*msec*)	Acquisition IMU (*msec*)	Mechanization (*msec*)	EKF Prediction (*msec*)	EKF Update (*msec*)	Exec. Time (*msec*)
μ-blaze (125 *Mhz*)	0.006	0.118	2.7	54.9	56.4	114.12

the navigation solution is related to the EKF update stage (green time slot), which is computed every *second* when there is a GPS update. If there is not GPS update the solution is provided every 52.72 *msec* having the highest computational burned during the EKF prediction stage (purple time slot).

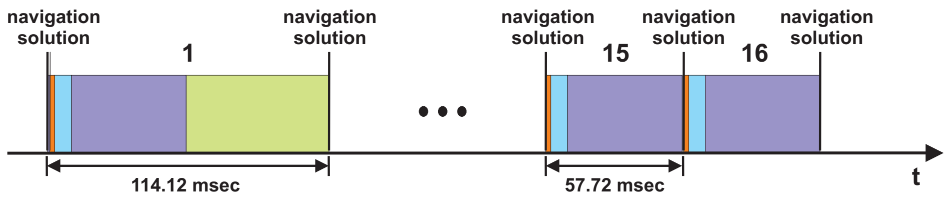


Figure 5.8: Time frame within 1 *sec* of μ-blaze computation at 125 MHz.

Thereby, the platform provides a navigation solution at approximately 16 Hz in real-time, which might be appropriate for terrestrial applications. Despite this, expanding the capabilities of the platform, *e.g.*, adding new sensors, using the platform in applications with high dynamic vehicle movement environment that require a navigation solution with a higher rate or considering more complex stochastic models to compensate low cost IMUs errors, would increase the computational cost of the navigation algorithm. From the software profiling, it was noted that the highest computational burned occurs in the matrix multiplication operation of the EKF, specifically, for the calculation of $\mathbf{P}_k^+$ and $\mathbf{P}_k^-$, since it involves matrices with a size of 15×15. Indeed, every time these matrices are computed the soft core processor takes on average around 11 M clock cycles, that is, an equivalent of 88 *msec*, 77.11% of the total execution time. Therefore, we decided to evaluate the feasibility of accelerating the application with a hardware multiplier using available resources on the FPGA. Further details about this custom core will be given in Section 5.4.4.

5.4.3 Hardware Resources

Table 5.2 is listing the amount of resources required to implement the architecture of the GPS/INS integrated system. Most of the slice LUTs and slice registers resources are attributed to the μ-blaze processor (*i.e.*, around 10000 between slice LUTs and slice registers), it also uses 7% of the DSP blocks available on the FPGA. The architecture uses 121 Blocks RAM dedicated hardware resources from 148 available on the Virtex-5 XC5VLX110T. Moreover, the code that is executed by the microblaze consumes 0.1 percent of the external memory DDR SDRAM. The results show that there are many hardware resources available in the FPGA that could be used to add new sensors into the navigation system. Furthermore, customization of the architecture provides information about the necessary hardware resources to achieve the development of a GPS/INS integration in a system on chip.

Table 5.2: Device and memory utilization summary.

	Used (#)	**Available** (#)	**Utilization** (%)
Slice Register	8.324	69.120	12
Slice LUTs	7.438	69.120	10
DSP48Es	5	64	7
Block RAM	121	148	81
DDR SDRAM	284 KB	256 MB	0.1

5.4.4 Matrix Multiplication in Hardware

Matrix multiplication is an essential operation in many applications and in this particular case in the Kalman filter implementation for the loosely-coupled GPS/INS integration, where depending on the IMU, the number of states might be increased significantly. Although there are several architectures that perform a parallel implementation of the matrix multiplication [166–168], they are used with small matrix dimensions. Nowadays, there are embedded devices that can be used to accelerate the matrix multiplication, for instance, in [153, 169], an efficient matrix multiplication is described using DSP blocks available on Xilinx FPGAs. Therefore, we focused in the development of an architecture using DSP blocks in order to compute the matrix multiplication operations that are typically found in the Kalman filter algorithm. Different from those designs where two dimensional systolic arrays are used, the architecture implemented organizes the matrix elements with only one dimension processing elements to reduce the utilization of DSP blocks available on the FPGA. The hardware design presented in this section is also flexible, scalable and parametrisable and is not restricted to the matrix size or the number of matrices to be multiplied.

In this way, if we focus on the XC5VLX110T FPGA, which is integrated in Xilinx University Program (XUP) kit, it has 64 DSP48E slice that can be used to evaluate the calculation of the matrix multiplication. The DSP48E slices support many independent functions. They include multiply, multiply accumulate (MAC), three-input add, among others. The architecture of these DSP48E blocks also supports cascading multiple DSP slices to form wide math functions, DSP filters, and complex arithmetic operations without the use of FPGA fabric [170].

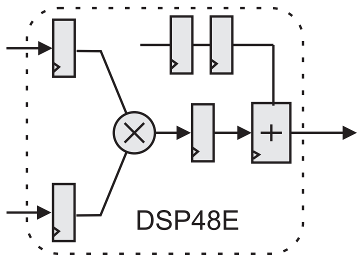


Figure 5.9: Proccesing element with a DSP48E block.

The very simplified form of the DSP48E slice for the Virtex-5 FPGA is shown in Fig. 5.9. It includes 25×18 multiplier and a three-input adder/subtracter/accumulator. So the partial output from the multiplier is 43-bit that is extended, forming 48-bit input datapath. This number of bits can be increased concatenating two DPS48E slices.

Matrix Multiplication Architecture

The architecture developed for matrix multiplication is shown in Fig. 5.10, it consists of processing elements (PEs) that are built with DSP48Es blocks. In this designed they are configured to perform the MAC operation. Moreover, dual port memories are used to store the matrix elements. Each memory has been tagged taking into account the matrix elements stored, *e.g.*, memory B stores all the elements of $\mathbf{B}$ matrix, memory C stores all the elements of $\mathbf{C}$ matrix, whereas memory A_1 stores the first row of matrix $\mathbf{A}$, memory $A \times B\_1$ stores the first row of the result $\mathbf{A} \times \mathbf{B}$ and so on.

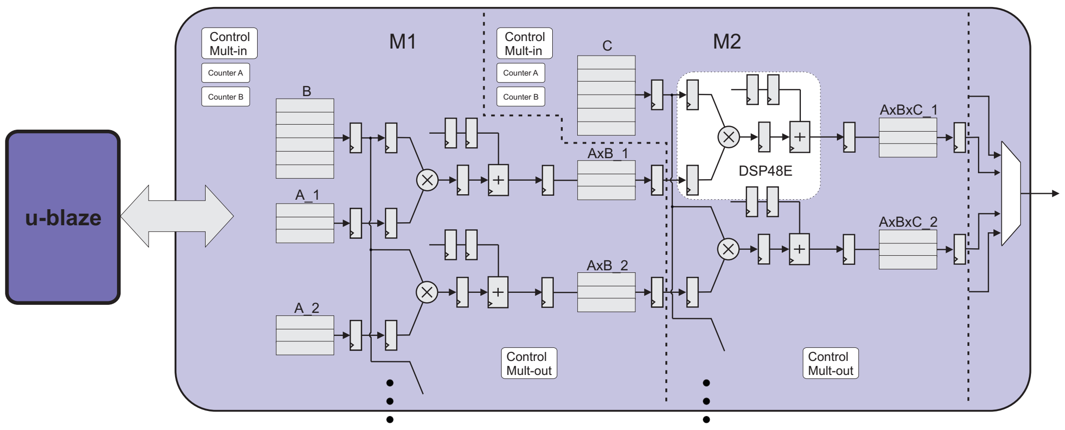


Figure 5.10: Matrix multiplier array with DSP48E blocks adapted to Microblaze.

This designed is scalable because depending on the number of rows of matrix **A**, the architecture can be extended. For instance, adding dual port memories A_4, A_5 and A_6, we could multiply matrices with six rows. This is easily achieved with the generate statement in VHDL (VHSIC Hardware Description Language), which allows to replicate hardware. Similarly, considering that a multiplier stage is given by $M1$ (*i.e.* enclosed by the dotted line in Fig. 5.10 and it computes $\mathbf{A} \times \mathbf{B}$ only), this can be replicated as many times as necessary to multiply several matrices, which is the case when we are computing the Kalman filter. Here we only implement two stages that correspond to $\mathbf{A} \times \mathbf{B} \rightarrow M1$ and $\mathbf{A} \times \mathbf{B} \times \mathbf{C} \rightarrow M2$. The matrix multiplication developed in VHDL includes modules such as control mult-in, counter A/B and control mult-out, the functionality of these modules is described below:

- Counter A: this module is a counter that acts as bus address for memories A_1, A_2, A_3, *etc.* Each counter has two inputs to maintain a value or reset the counter besides the *clk* signal. The input signals come from the control mult-in module. Each stage has its own counter A, *e.g.*, it works as the bus address of memories $A \times B\_1$ and $A \times B\_2$ in stage $M2$.

- Counter B: this module has the same hardware description as the previous one, but it acts as the bus address for memory B. Each stage has its own counter B, *e.g.* it works as the bus address of memory C in stage $M2$.

- Control mult-in: this module is included in each stage M and it is implemented by a Finite State Machine (FSM). This is the one that provides the control signals to perform the matrix multiplication. One of its functions is to reset the processing elements and counters A/B until the multiplication is enabled by the start signal ($matrix\_mult = 1$). This module also stores in a register the size of the matrices to be multiplied *i.e.*, the number of elements of **B** and the number of columns of **A**, so its value can be modified to multiply matrices of different sizes. When it receives the signal to start the multiplication ($matrix\_mult = 1$), it sends a signals to enable the counters and at the same time to performed the accumulation operation in the DSP blocks. While counters A and B are increasing, the DSP blocks are accumulating the multiplication results. So if counter A reaches the number of columns of matrix **A**, a signal is sent to control mult-out module to store the result in $A \times B\_1$ and $A \times B\_2$. Subsequently, counter A is reset together with the PEs with the purpose of starting the multiplication of the next column of matrix **B**. Then, counter B is incremented and the previous steps mentioned are repeated. Once counter B reaches the equivalent value of the number of elements of matrix **B**, the multiplication ends and control mult-in returns to its initial state.

- Control mult-out: this module was developed with a Finite State Machine (FSM) and its main function is to store the result of each stage. It also sends the control signal in order to start the multiplication in the next stage, this is in case the matrices to be multiplied are more than two.

With the purpose of clarifying the functionality of the modules and evaluating the architecture with operations that are used in the navigation algorithm, *i.e.*, multiplications between two and three matrices and all of them not necessarily square matrices, we adapted the generate statement in VHDL language to create the architecture showed on the right side of Fig. 5.11, which was tested computing a matrix multiplication of three matrices: $\mathbf{A}_{2\times2} \times \mathbf{B}_{2\times2} \times \mathbf{C}_{2\times3}$. The matrix multiplication was assessed with a test bench, loading into the memories matrices like **A**, **B** and **C** with the elements that are presented on the left side of Fig. 5.11.

The resultant simulation for this architecture is depicted in Fig. 5.12. It begins storing the elements of the matrices in their respective memories, in this case, for

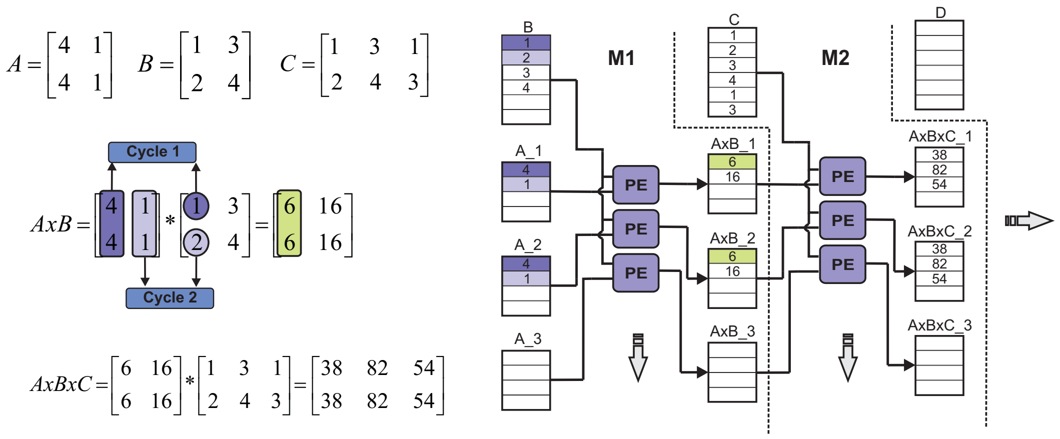


Figure 5.11: Matrix multiplier with two stages and six DSP48E blocks.

illustrative purposes we used the same input address bus for each memory (*addressa*). Moreover, memories A_1, A_2 and A_3 are connected to the same input data bus (*input_data*1*a*) so both memories are loaded with 4 and 1 (*i.e.*, the rows of matrix **A**). In the same way, memories B and C are loaded with their matrix elements through the input data bus (*input_data*2*a*). Matrix multiplication starts when the *matrix_mult* signal is set to 1 in the control mult-in module for one clock cycle (*i.e.*, at 605 *sec*), which enables counter A, that is connected to the bus address of memories A_1, $A\_2 \rightarrow address\_ca[0]$ and $A \times B\_1$, $A \times B\_2 \rightarrow address\_ca[1]$, while counter B is also enabled and is connected to the bus address of memories $B \rightarrow address\_cb[0]$ and $C \rightarrow address\_cb[1]$. At the same time, it allows the PEs to start accumulating. This first step lasts 5 clock cycles, which is due to synchronization states and the intermediate latches that are between the memory and the DSP blocks (see Fig. 5.10). Once the data arrives in the PEs, it takes one clock cycle to multiply the first element of matrix **B**[1,1] = 1 with the first column of matrix **A** (see Fig. 5.11), which is achieved in parallel. Then, while both counters A/B are increasing the DSP blocks are accumulating each result until counter A reaches the number of columns of matrix **A**, so at this step we have 3 additional clock cycles due to the two columns of matrix **A** to get the first column result of $\mathbf{A} \times \mathbf{B}$ in the accumulator. Subsequently, counter B keeps its value and the result 6 is stored in memories $A \times B\_1$ and $A \times B\_2$, while counter A and the DSP blocks of the first stage are reset in order to begin the multiplication with the next column of matrix **B**, these steps last around 2 clock cycles. Thereby, the first result is produced approximately after 10 clocks cycles that is equivalent to

100 ns as it can be seen in Fig. 5.12 from 605 ns until 705 ns, the result appears in the output data bus $ram\_output3B[0 \cdot \cdot 2]$ that is connected to the output port of memories $A \times B\_1$, $A \times B\_2$ and $A \times B\_3$.

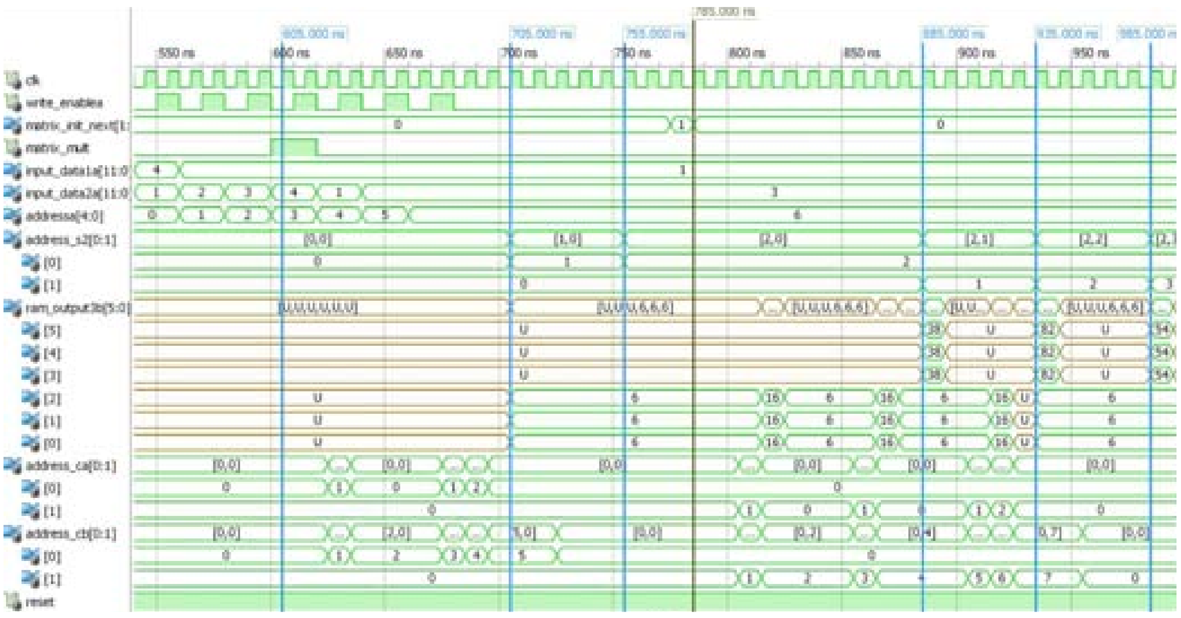


Figure 5.12: Simulation of Matrix multiplier $\mathbf{A} \times \mathbf{B} \times \mathbf{C}$ implemented in VHDL.

After computing the second column of $\mathbf{A} \times \mathbf{B}$, stage $M2$ begins when control mult-out sets $matrix\_init\_next$ signal to 1 for one clock cycle (*i.e.*, at 785 sec). The same as in the previous stages it takes around 100 ns to get the first column of $\mathbf{A} \times \mathbf{B} \times \mathbf{C}$, that corresponds to 38 and it is shown at 885 sec. This result come outs in the output data bus $ram\_output3B[3 \cdot \cdot 5]$ that is connected to the dual port memories $A \times B \times C\_1$, $A \times B \times C\_2$ and $A \times B \times C\_3$.

Total computation time

According to the previous description, we can summarize the computation of the first column for each stage in three steps: the first step is the initialization that enables counters A/B, enables the DSP blocks and move the data from memory to the PEs, it lasts 5 clock cycles and we will denote as k. The second step consists in the MAC operation, the number of cycles at this step (n_{mac}) depends on the number of columns of the first factor to be multiplied, *e.g.*, for the first stage $M1$, it is equivalent to $(\#\ columns\ of\ \mathbf{A}) + 1 = 3$ clock cycles, while for the second stage $M2$ it is the same but this time is given by $(\#\ columns\ of\ (\mathbf{A}\times\mathbf{B}))+1=3$ clock cycles. The third step, that we will denote as n, takes 2 clock cycles, this is to store the result provided by the DSP blocks in the output memory, reset counter A and reinitialize the MAC operation in the PEs. Thus, we can express the computation time to get the first column of stage $M1$ ($i.e.$, the first column of matrix $\mathbf{A}\times\mathbf{B}$) by Eq. (5.2).

$$computation\ time_{1st\ column\ M1} = (k + n_{mac} + n) * (1/f_{clk}) \tag{5.2}$$

where f_{clk} is the clock frequency. Eq. (5.2) can be applied to any stage, for stages $M1$ and $M2$ in Fig. 5.12 both of them take 100 ns . The computation time of the next columns for each stage is smaller, this is because only steps 2 and 3 are repeated. This can be seen more clearly in Fig. 5.12, where the result of the second column of $\mathbf{A}\times\mathbf{B}\times\mathbf{C}$ comes out at 935 ns, that is, after $n_{mac}+n = 50\ ns$ of getting the first column of $\mathbf{A}\times\mathbf{B}\times\mathbf{C}$. The computation time for each column at stage $M1$ can be obtained with

$$\begin{aligned} computation\ time_{2nd\ column\ M1} &= (k + (n_{mac} + n) * 2) * (1/f_{clk}) \\ computation\ time_{3rd\ column\ M1} &= (k + (n_{mac} + n) * 3) * (1/f_{clk}) \\ &\vdots \\ computation\ time_{N\ column\ M1} &= (k + (n_{mac} + n) * N) * (1/f_{clk}) \end{aligned} \tag{5.3}$$

where N is the number of columns of the second factor to be multiplied, $e.g.$, for the first stage $M1$ shown in Fig. 5.11, the second factor would be matrix $\mathbf{B}$) while for the second stage $M2$ would be matrix $\mathbf{C}$. Therefore, we can write the total computation time for computing the result at one stage M by Eq. (5.4).

$$computation\ time_{M} = (k + (n_{mac} + n) * N) * (1/f_{clk}) \tag{5.4}$$

According to this equation the number of cycles to perform a matrix multiplication of two square matrices 2×2 after setting $matrix\_mult$ to 1 would be equal to 15 clock cycles. For the example given in Fig. 5.12, we can also applied Eq. 5.4 in stage $M2$, since it begins at 785 ns we obtained the complete result of $\mathbf{A} \times \mathbf{B} \times \mathbf{C}$ at 985 ns, that is after 200 ns:

$$computation\ time_{M2} = (5 + (3 + 2) * 3) * (10\ ns) = 200\ ns \tag{5.5}$$

Now considering a matrix multiplication that can be found in the KF algorithm, for instance, two square matrices 21×21, the number of clock cycles to perform this operation after setting $matrix\_mult$ to 1 would be equal to 509. Using a clock frequency of 100 MHz it would take more than a few 5.1 μsec to produce the resultant matrix taking into account the write/read of the memories, which could improve significantly the computation time of the matrix multiplication.

Even though this module was developed in order to study the possibility of accelerating the navigation algorithm using the FPGA hardware resources, it is recommended to concatenate DSP blocks to obtain a double precision before connecting it to the μ-blaze processor. Note that this module could be adapted to the μ-blaze processor using the Fast Simple Link (FSL) which would be an custom IP tailored as it can be seen in the architecture presented in Fig. 5.10. The architecture described in this section is scalable since several matrices can be multiplied by increasing the number of stages $M1$, $M2$, $M3$, *etc.* Two matrices can be multiplied in an architecture with three stages, this is due to the flexibility of the hardware implementation that can be enabled to read intermediate values. Moreover, the size of each matrix is loaded in a register, which makes the architecture parametrisable. Thus, an architecture with one stage an 20 PEs can be used to perform matrix multiplications with a size up to 20×20 as well as matrix multiplications with a size of 2×2.

Hardware Resources for the Matrix Multiplication

The utilization of hardware resources for two matrix multiplication architectures is described in Table 5.3. The used of the FPGA resources is very small, actually, is less than 3% between slice register and slice LUTs considering both designs. The number of dedicated DSP blocks is 3 for $\mathbf{A} \times \mathbf{B}$ and 6 for $\mathbf{A} \times \mathbf{B} \times \mathbf{C}$, which would be including a second stage $M2$ as in the architecture presented in Fig. 5.11. It also shows the maximum frequency for both hardware implementations.

Table 5.3: Matrix multiply resources utilization summary.

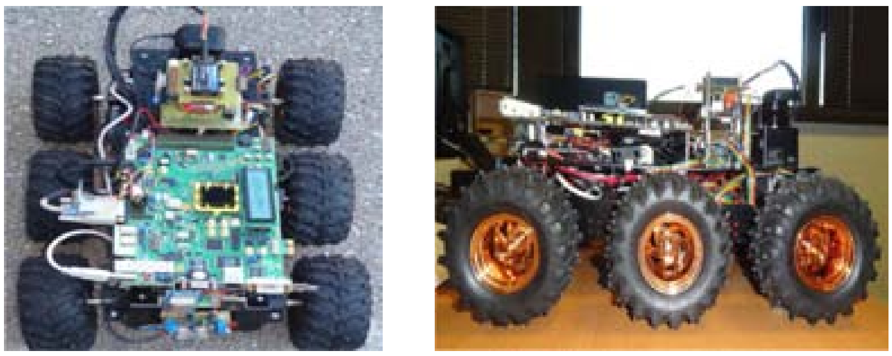
	Slice Register		**Slice LUTs**		**DSP48Es**		**Max. Freq.**
	(#)	(%)	(#)	(%)	(#)	(%)	(Mhz)
$\mathbf{A} \times \mathbf{B}$	214	0.3	174	0.25	3	4.69	271.08
$\mathbf{A} \times \mathbf{B} \times \mathbf{C}$	420	0.61	216	0.31	6	9.38	221.24

5.5 Testing the Navigation Platform Based on FPGA

5.5.1 Mobile Robot

The navigation system implemented on the FPGA was mounted on a mobile robot. The robot was in-house constructed using a chassis from Dagu Electronics that allows to adapt different sensors and additional hardware with a maximum payload of 5 kg. It has 6-wheels with independent suspension for each of the wheels and it is designed to move around rough terrain and steep inclines making this chassis suitable to perform tasks in different environments. To get the chassis moving we used a dual serial motor controller qik 2s12v10 from Pololu, when it is powered at 7.2 v each motor could have a stall torque of roughly 11 $kg - cm$ [171]. We adapted the chassis with seven infrared sensors, the ATmega128 microcontroller, the u-blox LEA-6T receiver, the 3DM-GX3 IMU, the Atomic IMU and two packs of batteries of 7.2V with 4200 mAh and 5000 mAh to power the Xilinx FPGA evaluation board and the motor controller, respectively. The experimental setup of the sensors and the FPGA platform on the robot can be seen in Fig. 5.13. The robot was designed to operate in two modes: autonomous and remote control.

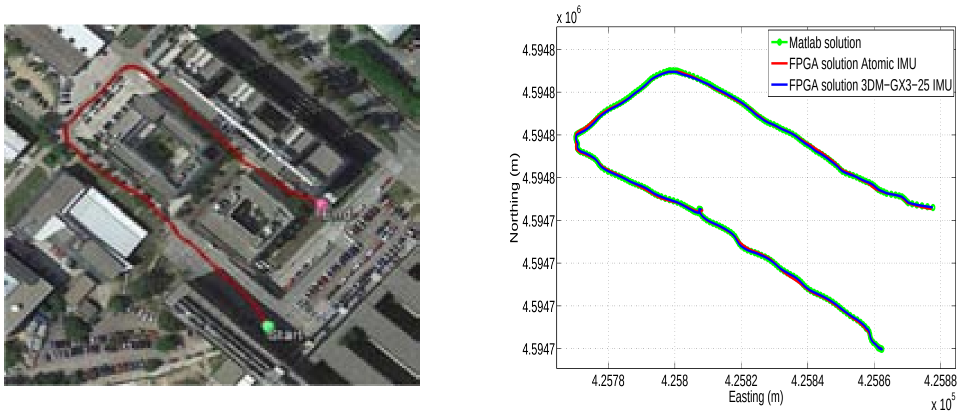
In this application, the FPGA is dedicated to collect data from the GPS and Inertial Measurement Units, the raw measurements are stored in the flash compact memory. The Android device sends commands to control the robot movements ($i.e.$, in remote control mode). The messages are sent to a microcontroller which is responsible for reading, parsing and sending the necessary signals to control the robot motors.



(a) (b)

Figure 5.13: Terrestrial robot.

The trajectories acquired with the mobile robot were performed around the campus of the Universitat Autònoma de Barcelona. Fig. 5.14(a) shows one of the robot tests, which has a duration of 275 *sec*. The navigation system was not exposed to any GPS signal blockage in the area where the robot test was carried out. The information obtained during this campaign was later used to verified the loosely-coupled algorithm developed in Matlab and the one implemented on the FPGA.



(a) (b)

Figure 5.14: (**a**) Terrestrial robot trajectory; (**b**) Matlab and FPGA solution for the two MEMS based IMUs.

Figure 5.14(b) shows the position solution calculated with the platform based on FPGA and the position solution obtained with Matlab. In this case, the reference is the GPS/INS integration in Matlab that was fed with the measurements provided by the 3DM-GX3-IMU, using a loosely-coupled with 15 states as it is described in Eq. (5.1).

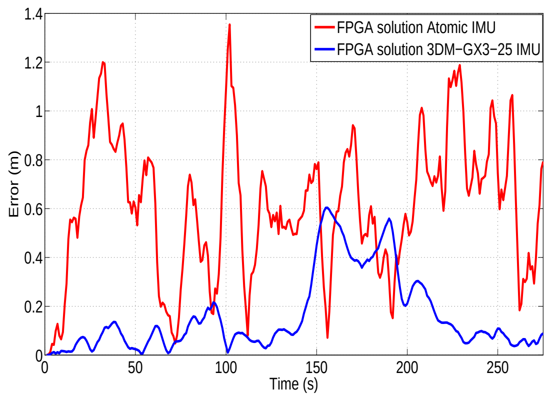


Figure 5.15: Horizontal position error for FPGA solutions.

The position error in the horizontal plane between the position solution obtained in Matlab and the ones calculated in microblaze with the the two MEMS based IMUs is shown in Fig. 5.15. As it was expected, the Atomic IMU has a larger error with respect to the 3DM-GX3-25 IMU, this error is on average 0.60 m whereas for the 3DM-GX3-25 IMU is around 0.15 m.

Table 5.4: Accuracy comparison between μ-blaze and Matlab computation.

IMU	P_N (m)	P_E (m)	P_D (m)	V_N (m/s)	V_E (m/s)	V_D (m/s)	$Roll$ (deg)	$Pitch$ (deg)	Yaw (deg)
Atomic (RMSE)	0.53	0.42	1.69	0.34	0.42	0.35	2.61	1.46	0.83
3DM-GX3 (RMSE)	0.20	0.08	0.40	0.12	0.07	0.06	1.43	0.66	0.48

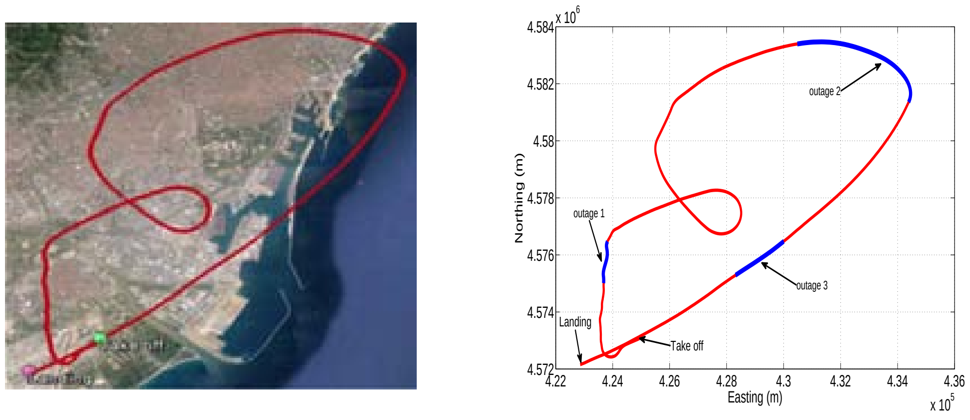
Table 5.4 shows the computation accuracy in terms of the root-mean-square error (RMSE) between the navigation solution computed with the algorithm implemented on μ-blaze and the one developed in Matlab. The difference between the FPGA solutions for the 3DM-GX3 and the one computed in Matlab might be due to the fact that microblaze uses a single-precision floating point FPU, which can be improved by using the software libraries that emulate double-precision floating point.

5.5.2 Flight Simulator

The performance of the navigation system implemented on the XUPV5 development kit was also evaluated collecting navigation data from several flights using a flight simulator framework called Flightgear (FG). It uses JSBsim, that is the default Flight Dynamic Model (FDM), which can be used to model and simulate a small autonomous Unmanned Aerial vehicle (UAV). This software is mainly developed in C++, both of them (FG, FDM) are open source for used in research or academic environments which can model the flight dynamic of several aircrafts [172]. The data coming from the aircraft, such as acceleration, angular velocity, position and velocity was obtained via the generic protocol of FlightGear (FG), this protocol allowed us to store the data from the flight simulator in a plain text file. In this case, the Cessna C172P aircraft was used since it is easy to pilot and provides enough data to test the platform. The sampling frequency for the inertial sensors was set at 100 Hz. After getting different trajectories, the plain text file was read and the inertial sensor measurements were perturbed with a bias-drift error, *i.e.*, $a_{se} = WN(N) + 1^{st}\ GM(B)$ and $g_{se} = WN(N) + 1^{st}\ GM(B)$ for accelerometers and gyros, respectively. It includes a white noise and a flicker noise modelled with a first order Gauss-Markov process. The parameters that represent this stochastic model were set taking into account the Allan variance analysis that was made on the real devices (see Section 4.4). Subsequently, the noisy measurements were downloaded into the external memory (DDR SDRAM) of XUPV5 board to assess the performance of the platform. It is noteworthy that Flightgear is not an ideal data set, since the simulated GPS position is not 100% accurate, in fact, an error function is being under develop to improve its accuracy.

Once the data collected is stored into the DDR SDRAM, the navigation algorithm implemented on the FPGA is executed until each data is processed. Then, the result of the loosely-coupled integration is sent by means of the UART interface to a PC, where it is analysed.

One of the trajectories that was performed with Flightgear is depicted in Google Earth (see Fig. 5.16(a)). This flight was acquired around Barcelona with a total time flight of 767 *sec*. With the purpose of testing the navigation platform when there are absences of the GPS signal, three GPS outages were introduced intentionally as it is



(a) (b)

Figure 5.16: (**a**) Aircraft trajectory in Google Earth; (**b**) Aircraft trajectory with three artificial GPS outages.

depicted in Fig. 5.16(b).

The code implemented in the microblaze processor was modified to simulate GPS signal blockages at different time instants. They were introduced in the trajectory at 160, 500 and 650 *seconds* with a duration of 60, 60 and 30 *seconds*, respectively.

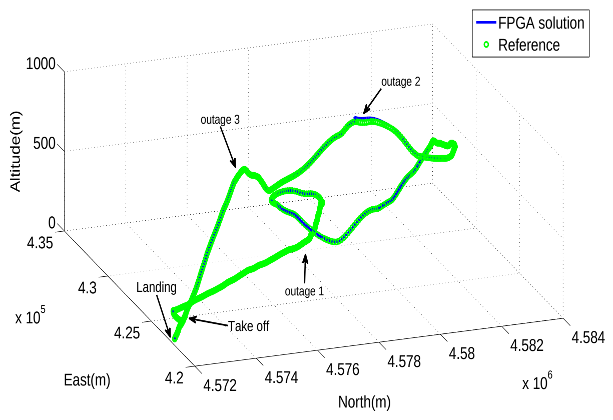


Figure 5.17: Reference and FPGA solution trajectory in 3D plot with three outages.

The 3D position of the aircraft in the local navigation frame (NED) is shown in Fig. 5.17, the blue line represents the estimated position provided by the embedded system and the green line is the reference trajectory provided by GPS/INS integration of the Cessna C172P airplane in Matlab.

Figure 5.18 shows the position error in the horizontal plane (*i.e.*, the square root of the sum of the square error between north and east position) and the altitude error for the different outages that were introduced intentionally. The outages were inserted

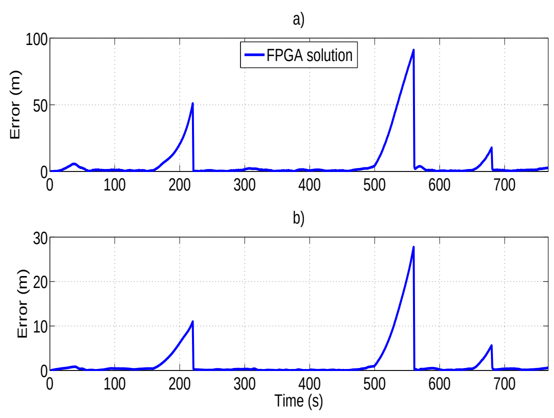


Figure 5.18: (**a**) Maximum horizontal position error during GPS outages; (**b**) Maximum altitude position error during GPS outages.

considering different dynamics of the aircraft and the average speed for each of them is listed in Table 5.5. It also presents the maximum error and the mean error for the μ-blaze solution. From this information, it can be noticed that the largest error occurs in the GPS outage 2, which is performed while the aircraft makes a turn towards south-east with an average speed of 307.64 km/h during 60 sec. This GPS outage has a maximum horizontal error of 91.12 m with the FPGA solution. After 2.5 min there is the third GPS outage with a duration of 30 sec, which has an average speed of 251.71 km/h and a maximum horizontal error of 17.85 m.

Table 5.5: Altitude and horizontal plane position error during GPS outages.

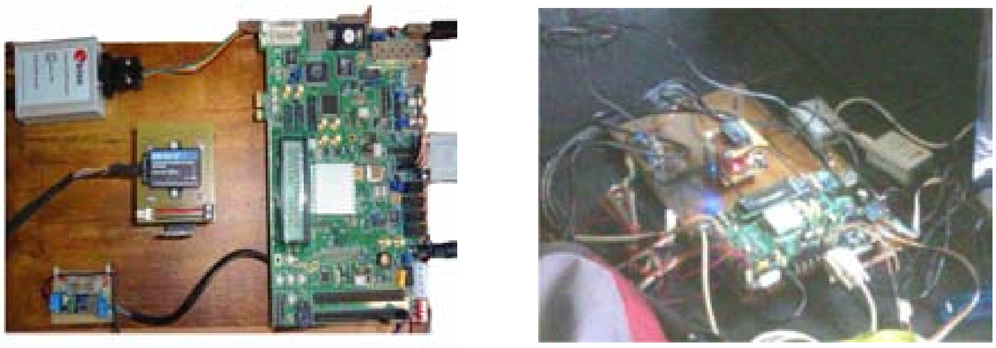
Outage (#)	**Dur.** (sec)	**Av. Speed** (km/h)	**Alt. Error**		**Hor. Error**	
			mean (m)	**max** (m)	**mean** (m)	**max** (m)
1	60	90.44	4.69	11	17.24	50.99
2	60	307.64	11.72	27.77	43.90	91.12
3	30	251.71	2.50	5.62	7.32	17.85

Despite the fact that we were focused on an embedded system for terrestrial applications, the flight simulator was used with the propose of easily acquire trajectories to debug and test the navigation platform during the GPS blockages. In addition, it also shows that the navigation platform might not be limited to terrestrial applications only.

5.5.3 Land vehicle

The development board was mounted in a land vehicle and the experimental setup is depicted in Fig. 5.19. It shows the XUPV5 board, a voltage regulator, the GPS receiver u-blox LEA-6T and the MEMS based IMUs.

During the campaigns the navigation platform was powered through the car battery, using the regulator, that was configured to provide an output voltage of 5 v to feed the board. The navigation instruments are not externally powered since all of them are connected to the XUPV5 board. It supplies 5 v output pins for the IMUs and 5 v through the USB port for the GPS receiver.



(a) (b)

Figure 5.19: Land vehicle equipment.

The results obtained for the different trajectories collected with the FPGA platform mounted in the land vehicle are analysed in the following chapter.

Chapter 6

Results and Discussion

This chapter it is divided in two parts, land vehicle test and land vehicle test using FPGA platform. The first part shows the performance of the stochastic models that were analysed in Section 4.3. The data sets were obtained in the city of Turin, Italy, during a stage that was held at the Navsas research group of the Istituto Superiore Mario Boella. For those results the 3DM-GX3-25 IMU available in the same group was under test. The second part of this chapter presents the performance of the stochastic models obtained with the NLF described in Section 4.5.3. The datasets for the vehicle test were collected with the navigation platform based on FPGA explained in Chapter 5, they were carried out around the campus of the Universitat Autònoma de Barcelona in Spain. During these campaigns the two MEMS IMUs available in the laboratory of Microelectronics and Electronics Systems (see Table 3.2) were under test, except for the temperature test.

6.1 Land Vehicle Test

As explained in Section 2.5, we use loosely-coupled integration with feed-back, which corrects the INS error through a close-loop. The INS error dynamics equations are built in the KF, having initially nine states for position, velocity and attitude error plus additional states to estimate the bias of each sensor of the IMU.

The EKF was adapted for each designed bias model in order to evaluate the

Table 6.1: Number of states in the loosely-coupled integration architecture for different error models.

	15 States	18 States	27 States
AV\PSD a_{se}	$WN(N) + GM(B)$	$WN(N) + GM(B) + RW(K)$	
AV\PSD g_{se}	$WN(N) + GM(B)$	$WN(N) + GM(B)$	
AR $a_{se}\backslash g_{se}$	$1^{st} order AR$		$3^{rd} order AR$

accuracy of the stochastic processes that were obtained from the previous analysis. Firstly, the two models extracted from AV\PSD were implemented, so the vector error states of the Extended Kalman Filter was augmented with 6 and 9 states, respectively. The latter error model was combined with wavelet de-noising in order to evaluate the enhancement accuracy when Allan variance parameters and wavelet de-noising techniques are blended together. Finally, two autoregressive models were assessed augmenting EKF with 6 and 18 states. Table 6.1 summarizes the stochastic models for the 3DM-GX3 sensors and the number of states that are required in the loosely-coupled GPS/INS integration.

The EKF for the loosely-coupled integration has 15 states for two models: one is the model obtained with AV\PSD where the bias instability (B) of both accelerometers and gyro are modelled with a first-order Gauss-Markov process (GM) plus velocity\angle random walk (N) that is modelled as white noise (WN) for accelerometers and gyros, respectively . The second model with 15 states is a first-order AR model. Although it is not depicted in Table 6.1, the AV model that was mixtured with wavelet de-noising corresponds to the case of EKF with 18 states. From here on, the abbreviations 15AR, 27AR, 15AV, 18AV and 18AVWD may be used when referring to the 15 states AR, 27 states AR, 15 states AV, 18 states AV and 18 states AV with wavelet de-noising models, respectively.

In order to assess the performance of the inertial sensor error models, a car was equipped with the 3DM-GX3-25 MEMS grade IMU, which was integrated with the

Sat-Surf platform with u-blox LEA-5X receiver [122]. The experimental setup that was installed inside the car is provided in Fig. 6.1. This platform with the navigation instruments was mounted in the car rear, including the power supply that was delivered by one battery of 12 *volts* dc.

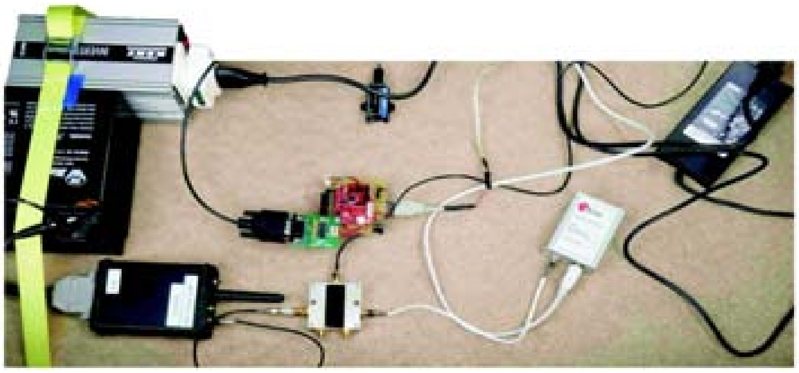


Figure 6.1: Experimental setup mounted inside the test vehicle.

Two data sets were collected in urban roadways inside the city of Turin, Italy. After the data collection campaign, the loosely-coupled integration architecture with the stochastic error models were evaluated. Although there were no GPS outages during the campaigns, we introduced intentionally several GPS outages off line, lasting 30 *sec* and 60 *sec*. During an outage the system works in prediction mode only and the accuracy of the loosely-coupled's performance relies entirely on the INS error model and in particular on the INS bias model. Therefore, it is straightforward to consider different outage lengths and different vehicle's dynamic conditions in order to have a clearer answer on the accuracy of the bias models under investigation. It is really worth mentioning that since this results are based on the loosely-coupled strategy, the simulated outages have complete GPS signal blockages. The GPS/INS solution without any outages was used as a reference to compare the performance of the different error models during the simulated GPS signal blockages.

6.1.1 First Trajectory Turin

The first trajectory that was used to asses the different sensor error models is shown in Google Earth map (Fig. 6.10(a)). This road-test is part of the whole trajectory and lasts near 17.3 *min*, we have acquired the data from the IMU with a sampling frequency of $100 Hz$. The GPS signal blockages that were intentionally introduced during postprocessing are depicted in Fig. 6.10(b) (shown as blue lines overlaid on the red trajectory), in which there are three outages with a duration of 30, 60 and 30 *seconds* for outage 1, 2 and 3, respectively. These artificial GPS outages includes straight and turns portions of the trajectory in a urban roadway, that comprise typical conditions of a real GPS signal degradation inside a city.

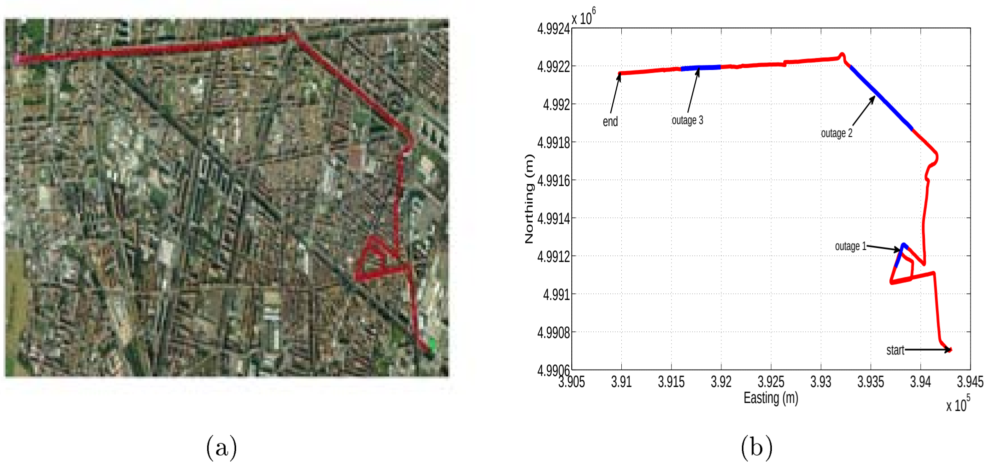
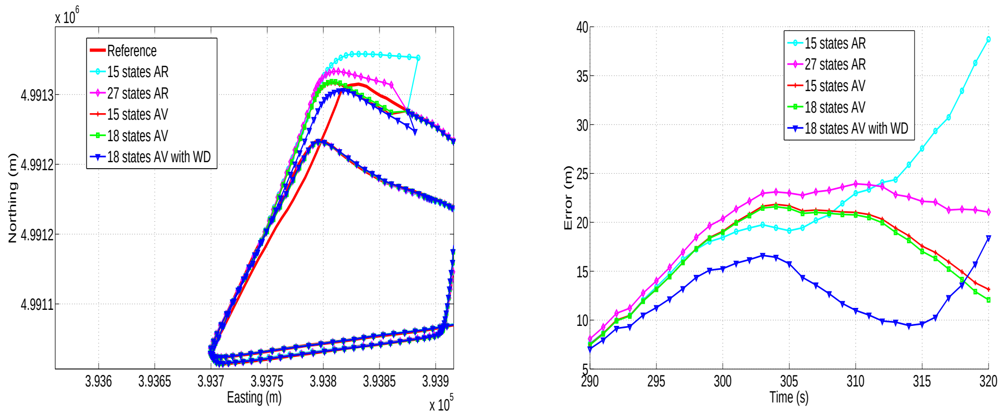


Figure 6.2: **(a)** First trajectory test in Google Earth; **b)** First trajectory test in Matlab with the GPS outages that were introduced intentionally.

The easting and northing position for two of the three outages (outage 1 and outage 2) are presented in Figs. 6.3(a) and 6.4(a), while the corresponding horizontal position error of these two outages are shown in Figs. 6.3(b) and 6.4(b). Table 6.2 summarizes the computation of the maximum and the mean horizontal position error for the error models solutions during the outages of the first trajectory. This table also shows the duration of each outage and the average speed during the 3 outages that were introduced in this road-test.

During the first GPS outage (Fig. 6.3(a)) there is a turn out of approximately 90 *deg*, this is a challenging segment of the trajectory to evaluate the bias models since there is an abrupt change in heading angle. From the correspondent horizontal position error (Fig. 6.3(b)), it can be noticed that the 18AVWD model produces the minimum horizontal error, less than 15 m for almost the whole GPS outage. The mean horizontal error for the 18AVWD model is 12.56 m, while the same error parameter for the 27AR model is 19.62 m. Regarding the 15AV and 18AV states models based on



(a) (b)

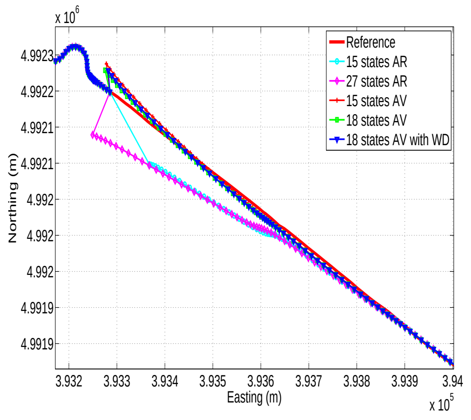
Figure 6.3: **(a)** Horizontal position during GPS outage 1; **(b)** Horizontal position error during GPS outage 1.

Allan variance parameters, it can be seen that they are slightly similar since the only difference is the acceleration random walk ($RW(K)$, see Table 6.1) that is added in the bias model of the accelerometers. On the other hand, 15AR model presents the worse result having a maximum horizontal error of 38.72 m and a mean horizontal error of 20.74 m.

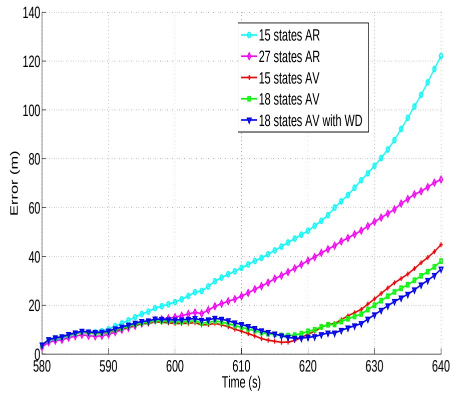
Figure 6.4(a) shows the northing easting plane for the five compared solutions during outage 2. According to Table 6.2 this outage lasts 60 sec and the average speed is about 42.31 km/h. This outage introduced in a straight portion of the trajectory shows that the 18AVWD model is better than the AR models and the other stochastic error models based on AV parameters.

Table 6.2: Maximum and mean horizontal position error during GPS outages for trajectory 1. 15AR, 15 state AR; 27AR, 27 state AR; 15AV, 15 state AV; 18AV, 18 state AV; 18AVWD, 18 state AV with wavelet de-noising.

Out.	**Dur.**	**Av.**	**Stochastic error model**									
		spd.	**15AV**		**18AV**		**18AVWD**		**15AR**		**27AR**	
(#)	(sec)	(km/h)	**mean** (m)	**max** (m)	**mean** (m)	**max** (m)	**mean** (m)	**max** (m)	**mean** (m)	**max** (m)	**mean** (m)	**max** (m)
1	30	23.58	17.11	21.84	16.82	21.60	12.56	18.42	20.74	38.72	19.62	23.94
2	60	42.31	14.41	44.84	14.10	38.04	13.16	34.81	42.82	122.07	29.70	71.49
3	30	47.60	13.34	49.97	11.86	40.41	11.80	39.69	17.48	64.26	19.02	70.21



(a)

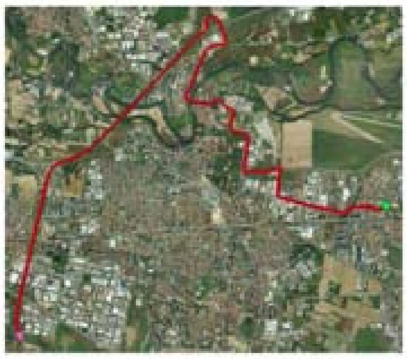


(b)

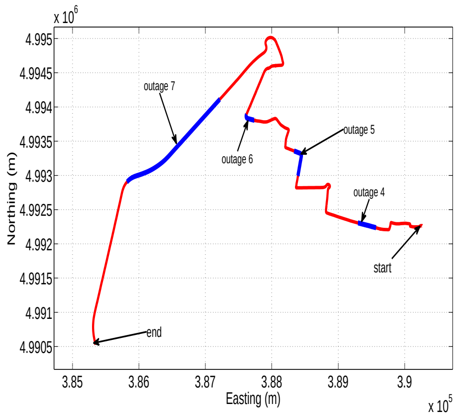
Figure 6.4: **(a)** Horizontal position during GPS outage 2; **(b)** Horizontal position error during GPS outage 2.

6.1.2 Second Trajectory Turin

To further validate the performance of the different stochastic error models, a second road-test trajectory was collected in some urban roadways in the city of Turin, there is also a part of the path on a highway in the outskirts of the city. The road-test trajectory is 15.05 min long and it is depicted in Fig. 6.5(b).



(a)

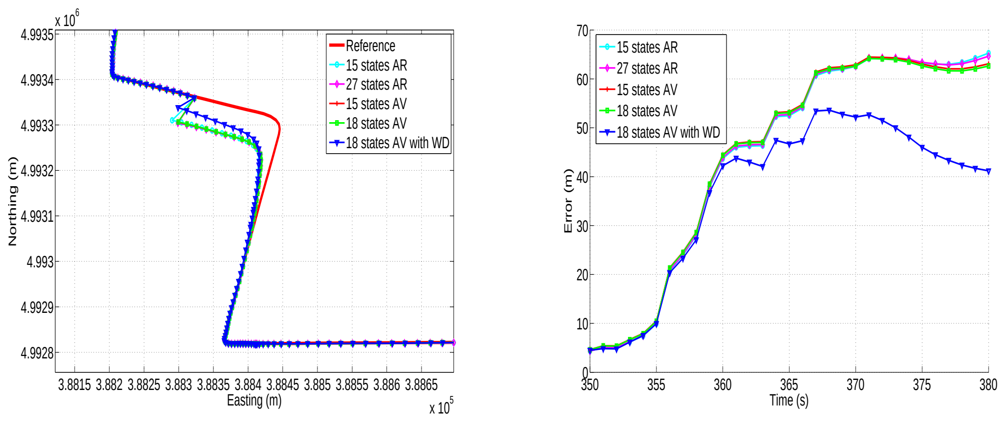


(b)

Figure 6.5: **(a)** Second trajectory test in Google Earth; **b)** Second trajectory test in Matlab with the GPS outages that were introduced intentionally.

Figs. 6.6(a) and 6.7(a) show two of the four GPS outages performed during the second road-test and their respective horizontal errors can be seen in Figs. 6.6(b) and 6.7(b). The same as in the previous trajectory, Table 6.3 summarizes the mean and the maximum error for each error model analysed, as well as the average speed and the duration of each outage that was introduced off-line.

Regarding the GPS outage 5 (Fig. 6.6(a)), it includes a turn out with an average speed of 52.25 km/h. According to the correspondent horizontal error (Fig. 6.6(b)), it can be observed that the correction that is achieved by the 18AVWD error model is bigger with respect to the one applied through the other methods and it has a maximum horizontal and mean position error of 53.61 m and 36.50 m, respectively, during 30 *sec* of absence of the GPS signal. As far as the GPS outage 7 is concerned, it has been simulated along a straight portion of the path including a slight curve at the end of the outage (Fig. 6.7(a)). This GPS blockage lasts 60 *sec* having an average speed of the vehicle of 112.42 km/h. In this GPS outage it was intended to evaluate the stochastic error models under high speed conditions and the same as in the previous GPS blockages, the 18AVWD performed better than the other models.

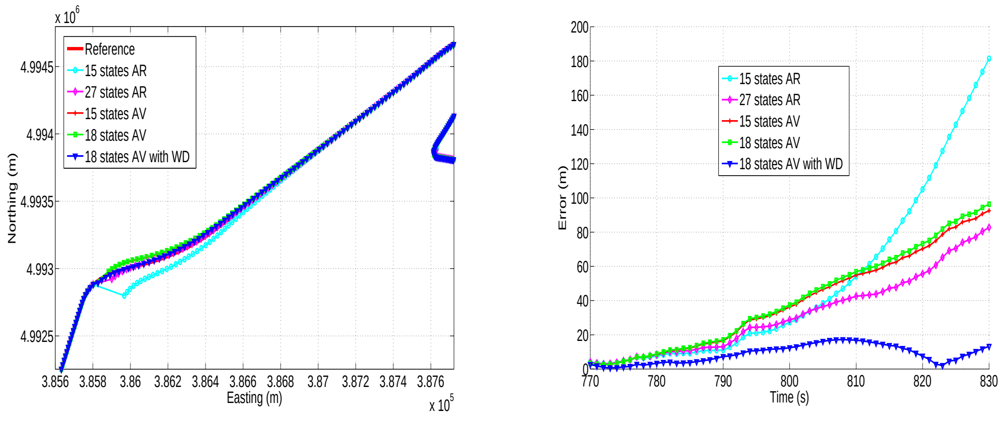


(a) (b)

Figure 6.6: **(a)** Horizontal position during GPS outage 5; **(b)** Horizontal position error during GPS outage 5.

Table 6.3: Maximum and mean horizontal position error during GPS outages for trajectory 2.

Out. (#)	**Dur.** (*sec*)	**Av. spd.** (km/h)	**Stochastic error model**									
			15AV		**18AV**		**18AVWD**		**15AR**		**27AR**	
			mean (m)	**max** (m)	**mean** (m)	**max** (m)	**mean** (m)	**max** (m)	**mean** (m)	**max** (m)	**mean** (m)	**max** (m)
4	30	33.60	10.85	22.70	10.44	21.13	7.29	10.81	6.72	14.72	9.99	29.26
5	30	52.25	44.57	64.46	44.36	64.17	36.50	53.61	44.34	65.28	44.43	64.65
6	30	20.44	2.74	6.19	2.71	5.91	2.25	5.32	4.12	13.17	4.18	13.41
7	60	112.42	39.40	92.51	40.85	96.32	8.74	17.20	49.62	181.38	32.27	82.74



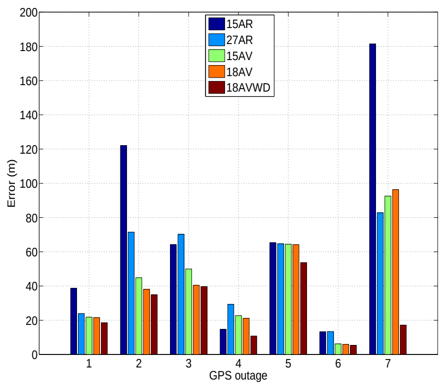
(a) (b)

Figure 6.7: **(a)** Horizontal position during GPS outage 7; **(b)** Horizontal position error during GPS outage 7.

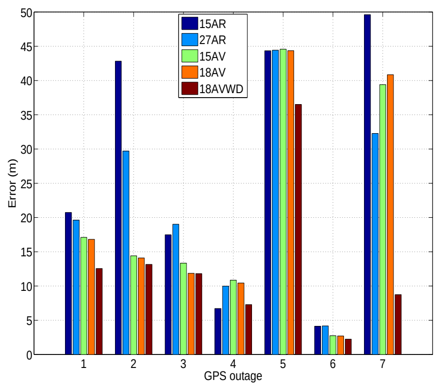
In order to summarize the maximum and mean position error for both trajectories and each GPS outage performed, Figs. 6.8(a) and 6.8(b) show a comparison between the error model solutions for the seven GPS outages introduced during the test campaigns made in the city of Turin.

Overall, the model based on AV and wavelet de-noising is the one that has provided the best accuracy in most of the cases under investigation. For instance, taking into account the results that are depicted in Fig. 6.8(a), the combination of the Allan variance parameters and wavelet de-nosing model (18AVWD) has got an improvement in terms of horizontal positioning error of 50.95% over the the first-order AR model (15AR) maximum horizontal position error. Furthermore, the 18AVWD provides an improvement of 48.20% over the 3rd order AR model (27AR). Regarding the models obtained from AV, the model 18AVWD has shown an improvement of 31.89% and 26.06% over the 15AV and 18AV, respectively. Considering the mean error in horizontal positioning (Fig. 6.8(b)), the blending of the Allan variance parameters and wavelet de-nosing (18AVWD) has allowed to get an improvement of 39.75% over the the first-order AR model (15AR) and it has also got a better accuracy of almost 41.94% with respect to the 3rd order AR model (27AR). In the same way the 18AVWD has provided an improvement of 27.67% and 25.13% over the 15AV and 18AV, respectively.

We can also clearly appreciate how the 18AV shows better results compared with the 15AV in most situations where the GPS signal is not available (see Figs. 6.8(a)



(a)



(b)

Figure 6.8: **(a)** Maximum horizontal position error for whole the GPS outages introduced in both trajectories; **(b)** Mean horizontal position error for whole the GPS outages introduced in both trajectories.

and 6.8(b)) since it offers a more adequate representation of bias-drift according to the noise terms identified with AV and PSD (*i.e.*, the addition of the noise source associated to the acceleration random walk (K) for each of the accelerometers - 18AV). Moreover, the performance of the AR models are lower than the ones obtained with AV, the explanation of this fact is commented next.

As far as the AR technique is concerned, the main objective of using AR models and wavelet de-noising is to remove the uncorrelated noise of the inertial sensors as much as possible. In fact, if we are able to remove the main quantity of the uncorrelated noise we can then obtain a smooth autocorrelations curve and the noise can be modeled with an higher order Gauss-Markov process (*e.g.*, third order AR model) with a consequently benefit on the accuracy and performance of the GPS/INS system. Unfortunately, this is not the case of the low cost inertial sensors (MEMS IMUs) we have used since as it is shown in Section 4.3.1, the autocorrelation function of some of the inertial sensors after processing the data with the de-noising technique does not have a smooth autocorrelation curve, which makes the estimation of the parameters less accurate compared to the parameters obtained with AV (*i.e.*, 15AV and 18AV). Another option to get a more accurate estimation of the bias-drift can be achieved by using higher order AR models (for instance in reference [19] the authors use an AR model with 120 states). In this case, we adopted a tradeoff between complexity and accuracy and we selected 27 states in the AR modeling.

At last, the mixture between AV and wavelet de-noising has shown much better enhancement accuracy of the INS than the others methods presented in this dissertation compensating the short-term and long-term noises that affect the inertial sensors.

6.2 Land Vehicle Test Using FPGA Platform for Acquisition

To evaluate the stochastic error models obtained with the NLF and compare with a conventional method, we got the AV models for each IMU following the procedure described in Section 4.3.4. Thus, the EKF filter was augmented with the correspondent model for each Inertial Measurement Unit. This is worth mentioning that the raw measurements read from the inertial sensors were not filtered with wavelet de-noising since the idea was to evaluate the bias-drift compensation with NLF models only.

Table 6.4: Stochastic error models for each IMU adapted to the loosely-coupled GPS/INS integration.

	Atomic IMU	**3DM-GX3 IMU**
AV a_{se}	$WN(N) + GM(B)$	$WN(N) + GM(B) + RW(K)$
AV g_{se}	$WN(N) + GM(B)$	$WN(N) + GM(B)$
NLF a_{se}	$WN(B) + GM(B) + GM(B)$	$WN(B) + GM(B) + GM(B)$
NLF g_{se}	$WN(B) + GM(B) + GM(B) + GM(B)$	$WN(B) + GM(B) + GM(B) + GM(B)$

The stochastic error for the accelerometers a_{se} and gyros g_{se} of the Atomic IMU were modeled by a white noise and a first order GM process and the parameters are taken from AV. Moreover, the error model identified with the NLF for the accelerometers of the Atomic IMU is composed by a white noise plus two first order GM processes; whereas the stochastic error for the gyros is modelled as a sum of a white noise and three first order GM processes. Table 6.4 summarizes the stochastic error models for the 3DM-GX3-25 and for the Atomic using AV and NLF. In order to assess the error models, they are adapted into the loosely-coupled GPS/INS integration.

To validate the performance of the different stochastic error models, the experimental setup using the navigation platform based on FPGA (see Fig. 5.19) was mounted in the car rear, as shows Fig. 6.9. During the vehicle tests the data acquisition was performed on the FPGA for the 3DM-GX3-25 and the GPS receiver u-blox LEA-6T through the UART peripherals, whereas the data acquisition of the Atomic IMU and the same GPS receiver (*i.e.*, using the available USB port) was performed from a PC using a software developed in Visual Basic. Several trajectories were collected between March and July 2013 near the campus of the Universitat Autonòma de Barcelona (UAB) using the navigation platform. During the vehicle tests different roads and driving conditions were considered in order to evaluate the error models under real world situations, so two sample trajectories were selected and the behaviour of the models was put to the test as it is presented below.

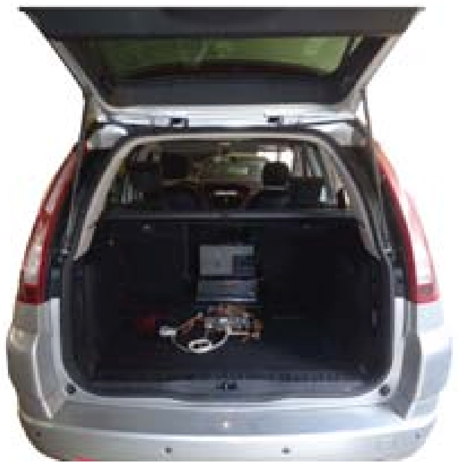
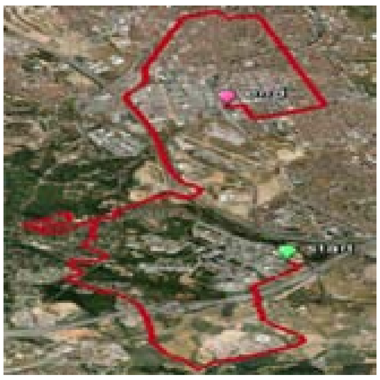


Figure 6.9: Land vehicle experimental setup.

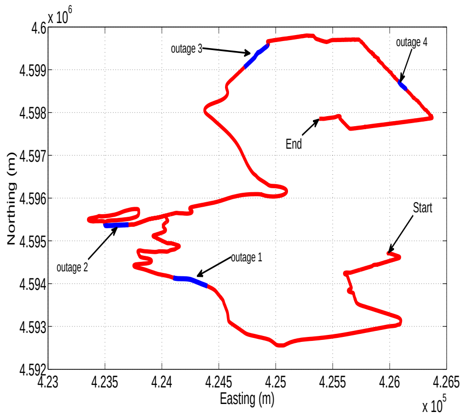
6.2.1 First Trajectory Barcelona

At the time this road-test was performed the temperature measurements were not available in the platform. Therefore, for this trajectory the parameters that were used in each error model were the ones estimated at 20 $^{\circ}C$, which was considered as a temperature close to the real temperature during the kinematic test.

Figure 6.10(a) shows one of the vehicle test, which includes not only the highway but also the urban area near the UAB campus. The trajectory starts at the UAB and ends in Sabadell, it has a duration of nearly 40.85 min of continuous navigation. The data collected from the MEMS based IMUs and the GPS receiver is stored in a PC and in the CF memory available on the FPGA, the platform runs the algorithm that was explained in Section 5.4.2. After the campaign, the raw measurements are processed off line with the loosely-coupled algorithm developed in Matlab, then we used the GPS/INS solution as a reference to compare the performance of the different error models during GPS outages.



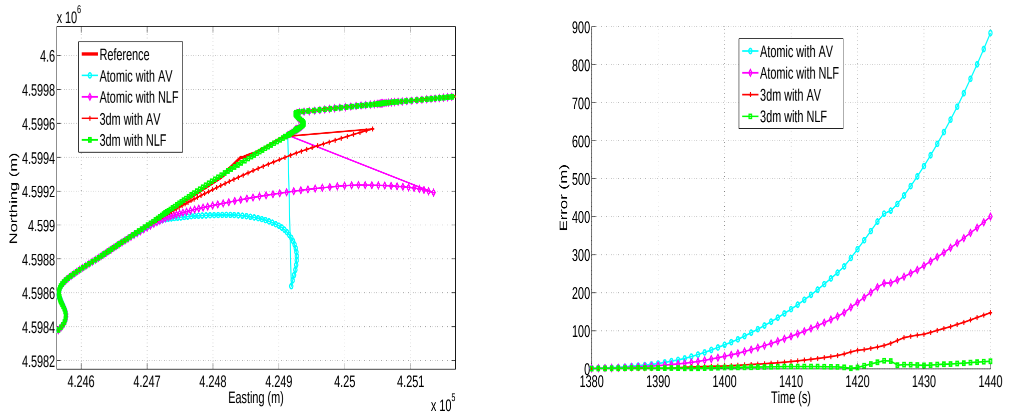
(a)



(b)

Figure 6.10: **(a)** First trajectory test in Google Earth BCN; **b)** First trajectory test in Matlab with the GPS outages that were introduced intentionally.

Since during this test only one natural GPS blockage occurred, we intentionally inserted off line three GPS outages, each of them lasting 30 sec. Fig. 6.10(b) shows the trajectory with the simulated outages, where the natural GPS outage corresponds to outage 3. Although this is a short natural GPS blockage ($<$ 15 sec), we have extended its duration to 60 sec in order to examine the system performance when there is a reliable reference, *i.e.*, before and after the outage when the GPS is available.



(a) (b)

Figure 6.11: **(a)** Horizontal position during GPS outage 3; **(b)** Horizontal position error during GPS outage 3.

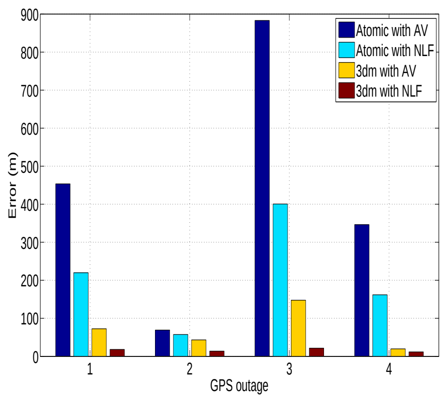
The easting and northing position for GPS outage 3 is presented in Fig. 6.11(a) and the horizontal position error is depicted in Fig. 6.11(b). This GPS outage is present in a straight portion of the path, where the vehicle has an average speed of 40.24 km/h. The minimum horizontal position error is obtained with the NLF error model of the 3DM-GX3-25 IMU (3dm with NLF), which is less than 22 m, compared with the error model based on AV (3dm with AV) that is 147.63 m. Similarly, the NLF of the Atomic IMU (Atomic with NLF) has a superior performance to the one obtained with AV (Atomic with AV), having a difference between them of 482.65 m in the maximum horizontal position error. Table 6.5 summarizes the computation of the maximum and mean horizontally position error for each model during different GPS outages. Here the abbreviations AT-AV, AT-NLF, 3DM-AV and 3DM-NLF refer to Atomic IMU with AV, Atomic IMU with NLF, 3DM-GX3 IMU with AV and 3DM-GX3 IMU with NLF, respectively.

The natural GPS outage can be seen in Fig. 6.11(a), the fact is seen at $(4.2484 * 10^5, 4.5994 * 10^6)$ m easting and northing position coordinates. In this segment of the trajectory the reference (in red) is not reliable because the integrated solution trust in the GPS position. In other words, in this portion of the trajectory the GPS/INS integration (in red) relies on the information provided by the GPS, so the navigation solution is misled by the GPS. During the GPS outage the 3dm with NLF model does not follow the reference trajectory and continues following the true path, which is an indication that it properly corrects the bias-drift error. Since we are evaluating the error models we can see their behaviour after the GPS outage has finished. It happens nearly 1427 *sec* where the error of the 3DM-NLF is around 11.08 m (see Fig. 6.11(b)). In a real situation, the degrade GPS signal would be detected and the solution would rely in the EKF prediction stage that is built with the stochastic models, in case of the GPS outage 3 with the 3DM-GX3-25 IMU, the solution provided by the NLF would follow the green line.

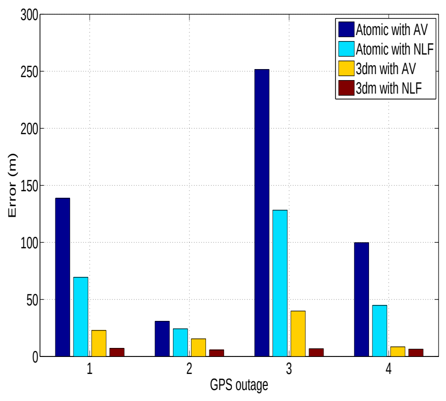
Table 6.5: Maximum and mean horizontal position error during GPS outages for trajectory 1 BCN. AT-AV, Atomic IMU with AV; AT-NLF, Atomic IMU with NLF; 3DM-AV, 3DM-GX3 IMU with AV; 3DM-NLF, 3DM-GX3 IMU with NLF.

Out. (#)	**Dur.** (*sec*)	**Av. spd.** (*km/h*)	**Stochastic error model**							
			AT-AV		**AT-NLF**		**3DM-AV**		**3DM-NLF**	
			mean (*m*)	**max** (*m*)	**mean** (*m*)	**max** (*m*)	**mean** (m)	**max** (*m*)	**mean** (*m*)	**max** (*m*)
1	30	45.26	138.61	453.52	69.42	219.75	22.84	72.33	7.132	18.05
2	30	32.91	30.88	69.08	24.27	57.30	15.48	43.19	5.69	13.70
3	60	40.24	251.58	883.19	128.28	400.54	39.92	147.63	6.89	21.43
4	30	20.59	99.76	346.37	44.77	161.70	8.53	19.66	6.46	11.60

Fig. 6.12(a) and Fig. 6.12(b), compare the maximum and mean position error in the horizontal plane, respectively, for each error model as well as for each MEMS based IMU during the four GPS outages introduced. Clearly the overall system performance benefited from the NLF stochastic models, that is, when the INS is in stand-alone mode the NLF stochastic model of the 3DM-GX3 provides a higher performance than the error models obtained from AV technique, and it also occurs for the Atomic IMU. The average percentage improvement taking into account the four GPS signal blockages for the vehicle maximum horizontal position error is 67.44% when the NLF is used in the 3DM-GX3-25 IMU (3DM-NLF) compared with the model given by AV (3DM-AV). For the Atomic IMU the average percentage of improvement by comparing the AT-NLF and AT-AV for the maximum horizontal position error is 44.14%. The accuracy enhancement provided by the NLF models is more significant when the absence of the GPS signal lasts 60 *sec* (*i.e.*, for GPS outage 3, 85.49% and 54.65% for the 3DM-GX3-25 and the Atomic MEMS IMUs, respectively), it might be due to the fact that the NLF error models can compensate more noise components that the AV models.



(a)



(b)

Figure 6.12: **(a)** Maximum horizontal position error for the GPS outages introduced in trajectory 1 BCN; **(b)** Mean horizontal position error for the GPS outages introduced in trajectory 1 BCN.

6.2.2 Second Trajectory Barcelona

To examine the performance of the stochastic models that were obtained with the NLF at different temperature points (see Section 4.5.3), the navigation platform based on FPGA was installed in a land vehicle and adapted with the 3DM-GX3-25 MEMS IMU and the GPS receiver u-blox LEA-6T. Additionally, the LM35 temperature sensor was used to record temperature data during the road-test.

The data set was collected on July 17, 2013, starting and ending the vehicle test at the UAB campus. The trajectory has urban roadways sections in Sabadell and there is also a segment of the path through a highway near the campus. Fig. 6.13(a) shows the trajectory in Google Earth, which was performed following a counterclockwise direction. The duration of the road-test was around 46 *min* of continuous navigation. It has some natural GPS outages (< 10 *sec*) that were extended to evaluate the position estimation with the EKF in prediction mode and augmented with the stochastic error models. Fig. 6.13(b) depicts the GPS signal outages (blue lines) lasting 30 *sec* and 60 *sec*, those outages were intentionally introduced during postprocessing considering highway and urban area segments, where different driving conditions were under test.

Figure 6.14 shows the recorded temperature during the kinematic test; it varies between 30 $°C$ and 34 $°C$. The raw measurements collected in the campaign are saved in the CF of the Xilinx development board, and then, they are downloaded in a PC

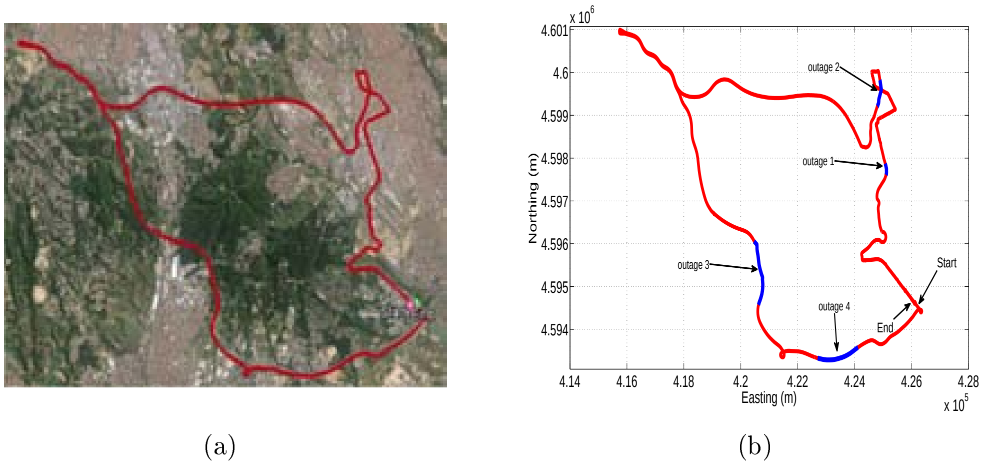


Figure 6.13: **(a)** Second trajectory test in Google Earth; **b)** Second trajectory test in Matlab with the GPS outages that were introduced intentionally.

in order to introduce them in the loosely-coupled GPS/INS integration implemented in Matlab. The navigation algorithm has been adapted with the four stochastic error models that were built up as explained in Section 4.5.3, and the parameters of the deterministic errors that were measured during the calibration test (see Section 3.4).

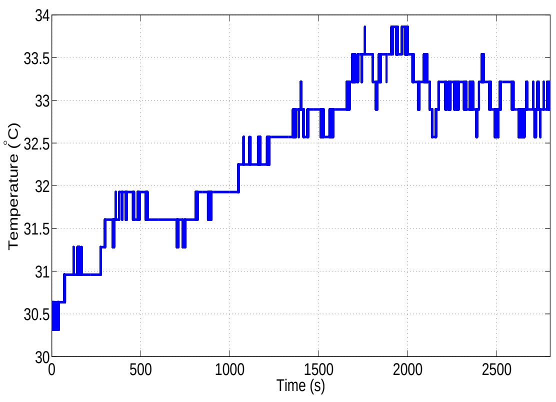


Figure 6.14: Temperature during the second trajectory.

The EKF is augmented with the four stochastic error models using the same combination of random processes presented in Table 6.4, but in this case the error model parameters correspond to the ones estimated by means of the NLF at different temperature points, *i.e.*, at 10 $^{\circ}C$, 20 $^{\circ}C$, 30 $^{\circ}C$ and 40 $^{\circ}C$. From here on, the abbreviations NLF TD 10 $^{\circ}C$, NLF TD 20 $^{\circ}C$, NLF TD 30 $^{\circ}C$ and NLF TD 40 $^{\circ}C$ may be used when referring to the constrained non-linear fitting error model temperature dependent at 10 $^{\circ}C$, 20 $^{\circ}C$, 30 $^{\circ}C$ and 40 $^{\circ}C$, respectively.

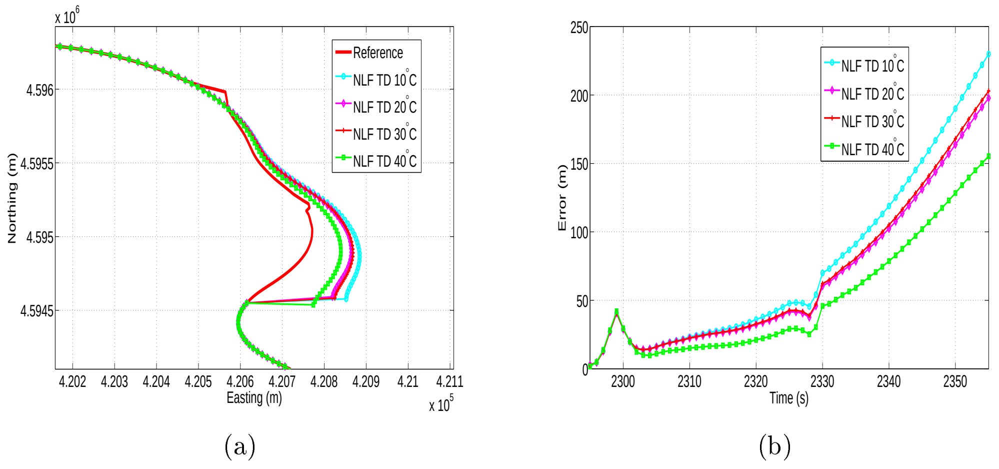


Figure 6.15: **(a)** Horizontal position during GPS outage 3; **(b)** Horizontal position error during GPS outage 3.

Figures 6.15(a) and 6.15(b) show the position and the position error in the horizontal plane during GPS outage 3. Since two natural GPS blockages are presented in this segment of the trajectory (*i.e.*, the outages begin approximately at 1295 *sec* and 1320 *sec*) we introduced intentionally an outage of 60 *sec* between 2995 *sec* and 2355 *sec*, which combines the two outages into one longer outage. This GPS blockage is one of the four outages inserted in this road and it has the highest maximum horizontal position error. Fig. 6.15(a) depicts a comparison between the estimated 2-D horizontal position for the different error models and the GPS/INS integrated solution (in red). This GPS outage is really challenging as it consists in a turn while driving at relatively high speed, *i.e.*, an average speed nearly 96.11 km/h. From Fig. 6.15(b) it can be seen that the performance of the NLF TD 40 $°C$ is superior over the error models computed at 10 $°C$, 20 $°C$ and 30 $°C$, having a maximum horizontal position error of 155.31 m, while the worst performance is given by the NLF TD 10 $°C$ with a maximum horizontal position error of 229.90 m. The error models corresponding to NLF TD 20 $°C$ and NLF TD 30 $°C$ show similar behaviour having a difference between them of 5.17 m in the maximum horizontal position error.

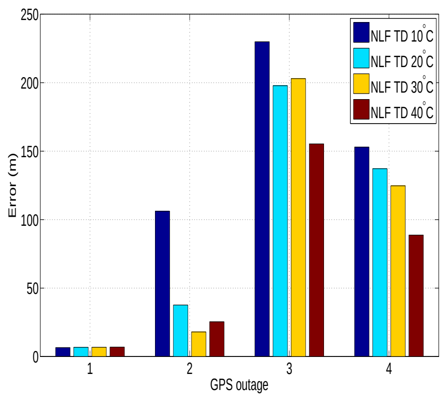
Table 6.6 summarizes the maximum and the mean position error in the horizontal plane for each one of the error models during the GPS outages, it also shows the characteristics of the four GPS blockages that were simulated. The bar graph representation of the maximum and mean position error can be seen in Figs. 6.16(a)

Table 6.6: Maximum and mean horizontal position error during GPS outages for trajectory 2 BCN. NLF TD 10 °C; NLF TD 20 °C; NLF TD 30 °C; NLF TD 40 °C.

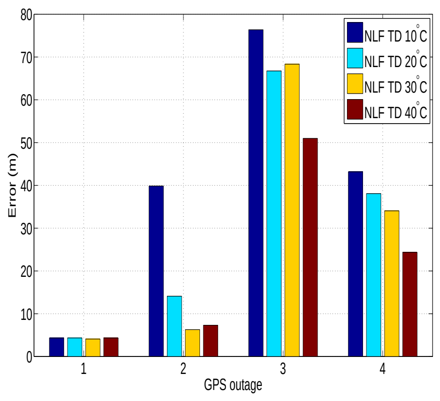
Out. (#)	**Dur.** (*sec*)	**Av. spd.** (*km/h*)	**Stochastic error model**							
			NLF TD 10 °C		**NLF TD** 20 °C		**NLF TD** 30 °C		**NLF TD** 40 °C	
			mean (*m*)	**max** (*m*)	**mean** (*m*)	**max** (*m*)	**mean** (*m*)	**max** (*m*)	**mean** (*m*)	**max** (*m*)
1	490/520	31.88	4.32	6.53	4.34	6.74	4.08	6.78	4.33	6.87
2	1220/1280	39.09	39.86	106.10	14.11	37.64	6.28	17.96	7.32	25.37
3	2295/2355	96.11	76.37	229.90	66.75	197.79	68.32	202.96	50.99	155.31
4	2500/2560	95.26	43.17	152.96	38.08	137.16	34.05	124.68	24.34	88.70

and 6.16(b), respectively, while the temperature *versus* time and the time interval when the GPS outages were inserted can be seen in Fig. 6.14 and Table 6.6, respectively. It can be noted that the best performance is given by the NLF TD 30 °C and NLF TD 40 °C error models. For the first two outages the performance of the NLF TD 30 °C is slightly better than the others, and the difference becomes more significant for relatively long GPS outages (*e.g.*, GPS outage 2 that lasts 60 *sec*), it seems to work as it was expected since the temperature was close to 30 °C at the time the GPS outages were introduced. Although, it was supposed to have a similar tendency for the last two outages, as it can be observed from Figs. 6.16(a) and 6.16(b), the NLF TD 40 °C shows a superior performance for GPS outages 3 and 4. Taking into account the temperature measurements during the road-test, the NLF TD 30 °C should perform better than the other error models due to the fact that the temperature at the time was close to this model, but it does not. We consider that it might be caused by the uncertainty involved during the estimation of the error parameters that are used in the models. Thus, the best performance of the NLF TD 40 °C for the last two GPS outages might be attributed to the uncertainty of the variance computation. In other words, for GPS outages above 2295 *sec* the estimated error parameters related to the NLF TD 40 °C model seem to be more adequate than the ones estimated for the NLF TD 30 °C model. This might be due to the uncertainty of the variance computation that affects the parameters estimation.

To this end, we conclude that the NLF models enhance the accuracy of the GPS/INS



(a)



(b)

Figure 6.16: **(a)** Maximum horizontal position error for whole the GPS outages introduced in trajectory 2 BCN; **(b)** Mean horizontal position error for whole the GPS outages introduced in trajectory 2 BCN.

integration compare with a traditional method such as AV. In spite of this, it has been noted that high position errors occurred, especially, for GPS outages of 60 *sec* when the vehicle has a high speed and at the same time there is a change in the vehicle dynamics. We consider that this position errors might be attenuated by adapting the procedure described with AV/wavelet de-noising to the NLF since the complex composite models do not seem to mitigate this errors. This is also worth mentioning that further improvement can be obtained by aiding information from the vehicle dynamics such as non-holonomic constraints and odometer signal, which gave significant advantages regarding the position accuracy when the GPS signal is not available. On the other hand, the NLF TD has shown satisfactory results, but a stochastic error model with a better resolution in temperature might be set up, actually, the temperature test developed in [29] is performed at 20 $^{\circ}C$ interval. Although collecting a larger data set during the temperature static test might not be a practical solution, the uncertainty of the variance computation can be reduced and as a consequence a better resolution of the NLF TD in temperature could be obtained.

Chapter 7

Conclusions and Future Activities

In this work, different stochastic error models for the measurement noise of MEMS-based IMUs have been implemented using experimental data, specifically, autoregressive/wavelet de-noising models, Allan variance, Allan variance/wavelet de-noising and constrained non-linear fitting (NLF). These stochastic models were adapted to the loosely-coupled strategy integration. Additionally, their performance was assessed in a low cost navigation application by means of intentionally introducing several GPS outages in different trajectories collected in real highways and urban roadways. The artificial GPS blockages were introduced in straight and curved portions of the trajectories comprising conditions of real GPS signal degradation.

For the first stochastic error models implemented (*i.e.*, autoregressive/wavelet de-noising models, Allan variance and Allan variance/wavelet de-noising - Section 4.3), we concluded that although AR processes combined with wavelet de-noising are commonly used for modeling INS stochastic errors, due to the fact that they have more modeling flexibility than first order Gauss-Markov, random walk and white noise processes, it is necessary to consider that the autocorrelation function of the stationary raw inertial sensors measurements is desirable to be a smooth curve (after de-nosing), in order to use a low order AR model, which most of the time is not the case for low cost inertial sensor (MEMS grade).

On the other hand, the inertial sensors are affected not only by short-term noises, but also by long-term noises. Minimizing the latter is not an easy task, since these are combined with vehicle motion dynamics. Therefore, we evaluated an error model that is a mixture of AV parameters and wavelet de-noising techniques (18AVWD); this model showed better performance than the other traditional methods based on AV and AR models during different GPS outages. The 18AVWD stochastic error model uses the parameters obtained from AV to compensate the long-term noises, while wavelet de-noising is employed to minimize the short-term noises that affect the inertial sensor of the IMU. Albeit, wavelet de-noising technique has once again demonstrated its utility for removing the short-term noises of the inertial sensors, we noted that MEMS based IMU requires many levels of decomposition to attenuate part of the uncorrelated noise and observe small enhancement in the position accuracy when using only wavelet de-nosing. Evaluating the combination of AV/Wavelet de-nosing showed that although some vehicle motion components might be attenuated the selected LOD provide more benefits concerning position accuracy.

The methodology adopted to study Allan variance together with wavelet de-nosing in the same log-log curve after applying different levels of decomposition, allowed us to analyze the attenuation of the error terms and the vehicle motion dynamics. By exploiting a combined use of the AV and wavelet de-noising, we have shown how to enhance the position accuracy in a GPS/INS integrated system without excessively increasing the complexity of the INS error model.

From the AV and wavelet-denoising analysis followed in this thesis one could consider not only static data but also kinematic data in order to better study the de-noising under dynamic conditions. Moreover, noise suppression methods that take into account the vehicles dynamics might be developed to enhance the INS performance.

Regarding the constrained non-linear fitting (NLF) (Section 4.5.3), we proposed constraints using the 95% confidence interval curve and taking into account characteristics of the noises that are typically found in low cost inertial sensors. The constraints not only provide information of noises that are possibly affecting the sensors but also can be used to facilitate the convergence of the fitting algorithm. It is worth mentioning that one of the limitations of the NLF is that it uses the variance curve estimated to fit the objective function, which has a considerable uncertainty for long-term cluster times. Therefore, a large static data set need to be recorded in order to have a good accuracy.

We investigated the effect of temperature changes on the stochastic modelling of MEMS based IMU and developed an error model temperature-dependent based on the NLF. This model is more appropriate that the ones reported in the literature since most of the models only take into account the deterministic error temperature variations and the few that consider the temperature-dependent stochastic errors are not adequate for low cost sensors since most of them are based in AR models and one first order GM process. We assessed the NLF TD model collecting temperature data during a vehicle test, and then various GPS outages were inserted under different driving conditions. We noted that it requires a large data set in order to reduce the uncertainty of the variance estimation, so it might yield a better resolution in temperature.

The error models could be adapted in more complex GPS/INS integration strategies, such as tightly-coupled, in order to enhance the position accuracy by using GPS estimates of pseudoranges and Doppler. It is worth noting that they can be applied to all inertial sensors grades and they are not limited to terrestrial applications.

As for the platform based on FPGA (Chapter 5), we developed a relatively compact and flexible navigation platform that can be easily customized. The platform allows to adapt different information sources, which is a feature suitable for current navigation systems. We analysed the software profiling of the loosely-couple GPS/INS integration, where we found that the highest computational cost was in the matrix multiplication computed in the EKF. Therefore, we studied the possibility of adapting dedicated hardware by means of DSP blocks in order to accelerate the navigation application. The custom IP developed is scalable, flexible and parametrisable and is not restricted to the matrix size or the number of matrices to be multiplied.

From the architecture developed on the FPGA, we concluded that although hardware implementation is a challenging task it can significantly speed-up the algorithm due to its capability to parallelize the design. It should be mentioned that high accuracy in the navigation solution requires a lot of hardware resources but current tools are being enabled to assist the developers in this sort of implementations. A future activity for the embedded navigation system would be to improve the synchronization algorithm and also study the possibility to use the partial dynamic reconfiguration in the GPS/INS integration, *e.g.*, modifying the hardware architecture when the EKF is in prediction mode or while navigating under different scenarios indoor/outdoor.

Appendix A

IMU Error State-Space Implementation in the KF

The equations that represent the error dynamics in the n-frame for the loosely-coupled approach are given by position error, ($\delta\mathbf{r}^n$), velocity error, ($\delta\mathbf{v}^n$), and attitude error, ($\delta\boldsymbol{\psi}^n$). The description of the transition matrix for these nine states is detailed in [5,59], and the derivation of these error equations can be found in [47,173]. In order to evaluate the models, it is necessary to increase these nine states, ($\delta\mathbf{r}^n$,$\delta\mathbf{v}^n$,$\delta\boldsymbol{\psi}^n$), with the IMU error states. For illustrative purposes, in this section, we only present the state-space form of the model associated with 18 states calculated with Allan variance and the 27 state model, where the third order AR model is adopted.

A.1 IMU Error State-Space for the 18 State AV Model

To include the bias of the inertial sensors (*i.e.*, Eqs. (4.10) and (4.11)), the transition matrix in the discrete time is augmented from the initial nine states, as in Eq. (A.1):

$$A_{k,9x9} = \begin{bmatrix} I_{3\times3} - diag(\boldsymbol{\beta}_a \Delta t) & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & I_{3\times3} - diag(\boldsymbol{\beta}_g \Delta t) & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & I_{3\times3} \end{bmatrix} \tag{A.1}$$

where Δt is the sampling time of the INS *i.e.*, 0.01 s, while $\boldsymbol{\beta}_a$ and $\boldsymbol{\beta}_g$ correspond to the reciprocal of correlation time, (T_c), presented in Table 4.5 for accelerometers and gyros, respectively. This transition matrix was described for one inertial sensor in Eq. (3.27); in the case of the three accelerometers, the expression, $diag(\boldsymbol{\beta}_a \Delta t)$, is given by:

$$diag(\boldsymbol{\beta}_a \Delta t) = \begin{bmatrix} \beta_{ax} & 0 & 0 \\ 0 & \beta_{ay} & 0 \\ 0 & 0 & \beta_{az} \end{bmatrix} \cdot \Delta t \tag{A.2}$$

The complete error states after adapting Eq. (A.1) into the first nine state of the KF is presented in Eq. (A.3):

$$\delta \mathbf{x} = \begin{bmatrix} \delta \mathbf{r}^n & \delta \mathbf{v}^n & \delta \boldsymbol{\psi}^n & \delta \mathbf{b}_{a,bi} & \delta \mathbf{b}_{g,bi} & \delta \mathbf{b}_{a,k} \end{bmatrix}^T \tag{A.3}$$

where $\delta \mathbf{b}_{a,bi}$ and $\delta \mathbf{b}_{g,bi}$ are the bias-drift associated to the first order GM process for accelerometers and gyros, respectively, while $\delta \mathbf{b}_{a,k}$ is the bias-drift of the accelerometers that represents the random walk process.

The design matrix, G, for the 18 error states AV can be written as:

$$G = \begin{bmatrix} 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ C_b^n & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & -C_b^n & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & I_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & I_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & I_{3\times3} \end{bmatrix} \tag{A.4}$$

where C_b^n is the frame rotation matrix from the body to the n-frame [44, 47].

$$C_b^n = \begin{bmatrix} c\theta c\psi & -c\phi s\psi + s\phi s\theta c\psi & s\phi s\psi + c\phi s\theta c\psi \\ c\theta s\psi & c\phi c\psi + s\phi s\theta s\psi & -s\phi c\psi + c\phi s\theta s\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix} \tag{A.5}$$

where "sin" and "cos" are shortly denoted as "s" and "c", respectively. The variables, ϕ, θ and ψ, correspond to the Euler angles (*Roll*, *Pitch* and *Yaw*).

The noise covariance matrix Q of the model is:

$$Q = \begin{bmatrix} diag(\mathbf{q}_{a,n}) & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & diag(\mathbf{q}_{g,n}) & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & diag(\mathbf{q}_{a,bi}) & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & diag(\mathbf{q}_{b,bi}) & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & diag(\mathbf{q}_{a,k}) \end{bmatrix} \tag{A.6}$$

where $\mathbf{q}_{a,n}$, $\mathbf{q}_{a,bi}$ and $\mathbf{q}_{a,k}$ are the spectral densities of the process driving noises for each noise term that are used to model the bias-drift of the accelerometers (*i.e.*, WN, RW and first order GM process). Similarly, $\mathbf{q}_{g,n}$ and $\mathbf{q}_{g,bi}$ are the noise variance quantities that will be used within the KF to model the bias-drift of the gyros (*i.e.*, WN and first order GM process). All these quantities are computed based on the parameters that were obtained with the AV technique (see Tables 4.4 and 4.5). In order to describe the relationship between the parameters obtained from the experiments (*i.e.*, N, B and K), and the noise process that are modeled (*i.e.*, WN, first order GM process and RW), we take as an example the x-axis accelerometer, so the spectral density in discrete time of the process driving noises of a WN process can be expressed as:

$$q_{ax,n} = \sigma^2_{WN_{ax}}/\Delta t = N^2_{ax}/(3600 * \Delta t) \tag{A.7}$$

where $\sigma^2_{WN_{ax}}$ is the variance of the white noise process and N is the velocity random walk associated to the x-axis accelerometer from Table 4.4. The spectral density, $q_{ax,bi}$, in discrete time for the first order GM process is given by [62]:

$$q_{ax,bi} = \sigma^2_{GM_{ax}}\left(1 - e^{-2\Delta t/T_{c,ax}}\right) \tag{A.8}$$

where $T_{c,ax}$ is the correlation time from Table 4.5 and $\sigma^2_{GM_{ax}}$ is the covariance of the first GM process that can be determined by means of the bias instability parameter for the x-axis accelerometer (B_{ax}) from Table 4.4.

$$\sigma_{GM_{ax}} = B_{ax} * 0.664/3600 \tag{A.9}$$

The spectral density, $q_{ax,bi}$, in discrete time for the random walk process can be expressed as:

$$q_{ax,k} = \sigma^2_{RW_{ax}} * \Delta t = K^2_{ax} * \Delta t/(3600)^3 \tag{A.10}$$

where $\sigma^2_{RW_{ax}}$ is the noise covariance of the RW process and K_{ax} is the acceleration random walk for the x-axis accelerometer from Table 4.4.

A.2 IMU Error State-Space for the 27 States with Third Order AR Models

The transition matrix in the discrete time used to augment the KF with a bias-drift modeled as a third order AR process for each inertial sensors can be described by Eq. (A.11):

$$A_{k,18x18} = \begin{bmatrix} 0_{3\times3} & I_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & I_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ -diag(\boldsymbol{\alpha}_3^a) & -diag(\boldsymbol{\alpha}_2^a) & -diag(\boldsymbol{\alpha}_1^a) & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & I_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & I_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & -diag(\boldsymbol{\alpha}_3^g) & -diag(\boldsymbol{\alpha}_2^g) & -diag(\boldsymbol{\alpha}_1^g) \end{bmatrix} \tag{A.11}$$

where $\boldsymbol{\alpha}_1^a$, $\boldsymbol{\alpha}_2^a$ and $\boldsymbol{\alpha}_3^a$ are vectors with the coefficients for the three accelerometers, while $\boldsymbol{\alpha}_1^a$, $\boldsymbol{\alpha}_2^a$ and $\boldsymbol{\alpha}_3^a$ are the vectors with the coefficients for the three gyros. This transition matrix was described for one inertial sensor in Eq. (3.30); in the case of the three accelerometers, the expression, $diag(\boldsymbol{\alpha}_1^a)$, is given by:

$$diag(\boldsymbol{\alpha}_1^a) = \begin{bmatrix} \alpha_1^{ax} & 0 & 0 \\ 0 & \alpha_1^{ay} & 0 \\ 0 & 0 & \alpha_1^{az} \end{bmatrix} \tag{A.12}$$

The complete error states of the KF will have 27 states, which are given by:

$$\delta\mathbf{x} = \begin{bmatrix} \delta\mathbf{r}^n & \delta\mathbf{v}^n & \delta\boldsymbol{\psi}^n & \delta\mathbf{b}_{a,b1} & \delta\mathbf{b}_{a,b2} & \delta\mathbf{b}_{a,b3} & \delta\mathbf{b}_{g,b1} & \delta\mathbf{b}_{g,b2} & \delta\mathbf{b}_{g,b3} \end{bmatrix}^T \tag{A.13}$$

where $\delta\mathbf{b}_{a,b1}$, $\delta\mathbf{b}_{a,b2}$ and $\delta\mathbf{b}_{a,b3}$ are the nine states associated to the third order AR models of the three accelerometer, while $\delta\mathbf{b}_{g,b1}$, $\delta\mathbf{b}_{g,b2}$ and $\delta\mathbf{b}_{g,b3}$ are the nine states associated to the third order AR models of the three gyros.

The design matrix, G, for the 27 error states based on the third AR models can be written as:

$$G = \begin{bmatrix} 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ C_b^n & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & -C_b^n & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & I_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & I_{3\times3} \end{bmatrix} \tag{A.14}$$

In this case, the noise covariance matrix, Q, of the model is:

$$Q = \begin{bmatrix} diag(\mathbf{q}_{a,n}) & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & diag(\mathbf{q}_{g,n}) & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & diag(\mathbf{q}_{a,b}) & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & diag(\mathbf{q}_{g,b}) \end{bmatrix} \tag{A.15}$$

where $\mathbf{q}_{a,n}$, $\mathbf{q}_{g,n}$ are the same spectral densities of the white noise processes described in the 18 states AV models, while $\mathbf{q}_{a,b}$ and $\mathbf{q}_{g,b}$ are the the spectral densities of the third order AR processes for each inertial sensor. In the case of the y-axis accelerometer, the spectral density in discrete time is by given by:

$$q_{ay,b} = \beta_0^2 \tag{A.16}$$

where β_0 is the standard deviation of the AR process.

Bibliography

[1] J.L. Weston and D.H. Titterton. Modern inertial navigation technology and its application. *Electronics & Communications Engineering Journal*, 12(2):49–64, 2000.

[2] D.K. Shaeffer. MEMS inertial sensors: A tutorial overview. *IEEE Commun. Mag.*, 51(4):100–109, 2013.

[3] T.B. Gabrielson. Mechanical-thermal noise in micromachined acoustic and vibration sensors. *IEEE Trans. Electron Devices*, 40(5):903–909, 1993.

[4] Alex G. Quinchia, Gianluca Falco, Emanuela Falletti, Fabio Dovis, and Carles Ferrer. A comparison between different error modeling of MEMS applied to GPS/INS integrated systems. *Sensors*, 13(8):9549–9588, 2013.

[5] P. Aggarwal, Z. Syed, A. Noureldin, and N. El-Sheimy. *MEMS-Based Integrated Navigation*. GNSS technology and applications series. Artech House, 2010.

[6] Y. Stebler, S. Guerrier, J. Skaloud, and M. Victoria-Feser. A framework for inertial sensor calibration using complex stochastic error models. In *Position Location and Navigation Symposium (PLANS), 2012 IEEE/ION*, pages 849–861, 2012.

[7] R.J. Vaccaro and A.S. Zaki. Statistical modeling of rate gyros. *IEEE Trans. Instrum. Meas.*, 61(3):673 –684, Mar. 2012.

[8] Stéphane Guerrier, Yannick Stebler, Jan Skaloud, Maria-Pia Victoria-Feser, et al. Generalized method of wavelet moments. 2011.

[9] IEEE standard specification format guide and test procedure for single-axis interferometric fiber optic gyros. *IEEE Std 952-1997*, page i, 1998.

[10] William J Riley and DA Howe. *Handbook of frequency stability analysis*. US Department of Commerce, National Institute of Standards and Technology, 2008.

[11] IEEE standard specification format guide and test procedure for linear, single-axis, non-gyroscopic accelerometers. *IEEE Std 1293-1998 (R2008)*, pages 1 –249, Nov. 2008.

[12] N. El-Sheimy, Haiying Hou, and Xiaoji Niu. Analysis and modeling of inertial sensors using Allan variance. *IEEE Trans. Instrum. Meas.*, 57(1):140–149, Jan. 2008.

[13] S. Moafipoor, L. Bock, J.A. Fayman, G. Mader, and P.J. de Jonge. Development and assessment of a low dynamic vehicle navigation system. In *Proc. International Technical Meeting of The Institute of Navigation*, pages 895–907, San Diego, CA, EEUU, Jan. 2011.

[14] Javier Hidalgo, Pantelis Poulakis, Johan Köhler, Jaime Del-Cerro, and Antonio Barrientos. Improving planetary rover attitude estimation via mems sensor characterization. *Sensors*, 12(2):2219–2235, 2012.

[15] Mohammed El-Diasty and Spiros Pagiatakis. Calibration and stochastic modelling of inertial navigation sensor errors. *Journal of Global Positioning Systems*, 7(2):170–182, 2008.

[16] X. Zhang, Y. Li, P. Mumford, and C. Rizos. Allan variance analysis of error characteristics of MEMS inertial sensors for an FPGA- based GPS/INS system. In *Proc. Int. Symp. on GPS/GNSS*, pages 127–133, Yokohama, Japan, Nov. 2008.

[17] Claudia C. Naranjo. Analysis and modeling of MEMS based inertial sensors. Master's thesis, School of Electrical Engineering Kungliga Tekniska Hgskolan, Sweden, 2008.

[18] Haiying Hou. Modeling inertial sensors errors using Allan variance. Master's thesis, University of Calgary, Alberta, Canada, september 2004.

[19] J. Georgy, A. Noureldin, M.J. Korenberg, and M.M. Bayoumi. Modeling the stochastic drift of a MEMS-based gyroscope in gyro/odometer/GPS integrated navigation. *IEEE Trans. Intell. Transport. Syst.*, 11(4):856 –872, Dec. 2010.

[20] Naser El-Sheimy, Sameh Nassar, Klaus-Peter Schwarz, and A. Noureldin. Modeling inertial sensor errors using autoregressive (AR) models. *Navigation*, 51(4):259–268, 2004.

[21] Yannick Stebler, Stéphane Guerrier, Jan Skaloud, and Maria-Pia Victoria-Feser.

Constrained expectation-maximization algorithm for stochastic inertial error modeling: Study of feasibility. *Measurement Science and Technology*, 22, 2011.

[22] N. El-Sheimy, S. Nassar, and A. Noureldin. Wavelet de-noising for IMU alignment. *IEEE Aerosp. Electron. Syst. Mag.*, 19(10):32 – 39, Oct. 2004.

[23] J. Skaloud, A. M. Bruton, and K. P. Schwarz. Detection and filtering of short-term $(1/f)$ noise in inertial sensors. *Journal of Navigation*, 46(2):97 – 107, 1999.

[24] R. Ramalingam, G. Anitha, and J. Shanmugam. Microelectromechnical systems inertial measurement unit error modelling and error analysis for low-cost strapdown inertial navigation system. *Defence Science Journal*, 59(6), 2009.

[25] Aboelmagd Noureldin, Justin Armstrong, Ahmed El-Shafie, Tashfeen Karamat, Don McGaughey, Michael Korenberg, and Aini Hussain. Accuracy enhancement of inertial sensors utilizing high resolution spectral analysis. *Sensors*, 12(9):11638–11660, 2012.

[26] Zhiqiang Xing and D. Gebre-Egziabher. Modeling and bounding low cost inertial sensor errors. In *Position, Location and Navigation Symposium, 2008 IEEE/ION*, pages 1122 –1132, may 2008.

[27] Yigiter Yuksel. Notes on stochastic errors of MEMS inertial units.

[28] Yigiter Yuksel, Naser El-Sheimy, and Aboelmagd Noureldin. Error modeling and characterization of environmental effects for low cost inertial MEMS units. In *Position Location and Navigation Symposium (PLANS), 2010 IEEE/ION*, pages 598 –612, May. 2010.

[29] Mohammed El-Diasty and Spiros Pagiatakis. A rigorous temperature-dependent stochastic modelling and testing for MEMS-based inertial sensor errors. *Sensors*, 9(11):8473–8489, 2009.

[30] B.E. Grantham and M.A. Bailey. A least-squares normalized error regression algorithm with application to the allan variance noise analysis method. In *Position, Location, And Navigation Symposium, 2006 IEEE/ION*, pages 750–756, 2006.

[31] F. Vernotte, E. Lantz, J. Groslambert, and J.J. Gagnepain. Oscillator noise analysis: multivariance measurement. *IEEE Trans. Instrum. Meas.*, 42(2):342–350, 1993.

[32] M. Jew, A. El-Osery, and S. Bruder. Implementation of an FPGA-based aided IMU on a low-cost autonomous outdoor robot. In *Position Location and Navigation Symposium (PLANS), 2010 IEEE/ION*, pages 1043–1051, 2010.

[33] A. G. Quinchia and C. Ferrer. A low-cost GPS&INS integrated system based on a FPGA platform. In *Proc. International Conference on Localization and GNSS*, pages 152–157, Tampere, Finland, Jun. 2011.

[34] M. Miller. Mav-2015 afrl's vision for bird sized UAV. In *Proc. International Technical Meeting of The Institute of Navigation*, pages 1–33, San Diego, CA, EEUU, Jan. 2011.

[35] V. Agarwal, H. Arya, and S. Bhaktavatsala. Design and development of a real-time DSP and FPGA-based integrated GPS-INS system for compact and low power applications. *IEEE Trans. Aerosp. Electron. Syst.*, 45(2):443 –454, Apr. 2009.

[36] Yong Li, P. Mumford, and C. Rizos. Performance of a low-cost field re-configurable real-time GPS/INS integrated system in urban navigation. In *Position, Location and Navigation Symposium, 2008 IEEE/ION*, pages 878 –885, May. 2008.

[37] Zhi-Jian Sun and Xue-Mei Liu. Application of floating point DSP and FPGA in integration navigation system. In *Proc. International Conference on Computer Science and Software Engineering*, volume 4, pages 58 –61, Dec. 2008.

[38] El-Osery, Aly I., Michael Jew, and Stephen Bruder. An inertial navigation system for autonomous outdoor robots. In *Proc. International Technical Meeting of The Institute of Navigation*, pages 57–66, San Diego, CA, Jan. 2010.

[39] Abdelfatah W.F., J. Georgy, U. Iqbal, and A. Noureldin. Real-time realization of 2D RISS/GPS integrated navigation on Xilinx's Microblaze soft-core processor for land applications. In *Proc. International Technical Meeting of The Institute of Navigation*, pages 414–421, Jan. 2011.

[40] I. Skog and P. Händel. In-car positioning and navigation technologies - a survey. *IEEE Trans. Intell. Transport. Syst.*, 10(1):4–21, 2009.

[41] Jay A Farrell and Matthew Barth. *The Global Positioning System & Inertial Navigation*. Number 4. McGraw-Hill, 1999.

[42] AD King. Inertial navigation-forty years of evolution. *GEC review*, 13(3):140–149, 1998.

[43] D. Titterton and J. Weston. *Strapdown Inertial Navigation Technology.* The American Institute of Aeronautics and Astronautics, second edition, 2004.

[44] Mohinder S. Grewal, Lawrence R. Weill, and Angus P. Andrews. *Global Positioning Systems, Inertial Navigation, and Integration.* John Wiley & Sons, 2007.

[45] H. Goldstein, C. Poole, and J. Safko. *Classical Mechanics.* 2002.

[46] J. B. Marion and S. T. Thornton. *Classical Dynamics of Particles and Systems.* 1995.

[47] E.-H. Shin. Accuracy improvement of low cost INS-GPS for land application. Master's thesis, Dept. of Geomatics Eng., University of Calgary, Calgary, Canada, Alberta, Canada, Dec 2001.

[48] *World Geodetic System 1984 (WGS-84) - Its Definition and Relationships with Local Geodetic Systems. NIMA TR8350.2, 3rd ed., National Imagery and Mapping Agency, St. Louis, MO.*

[49] René Forsberg. Gravity field terrain effect computations by fft. *Bulletin géodésique*, 59(4):342–360, 1985.

[50] M. G. Petovello. *Real-Time Integration of a Tactical-Grade IMU and GPS for High-Accuracy Positioning and Navigation.* PhD thesis, Dept. of Geomatics Eng., University of Calgary, Calgary, Canada, 2003.

[51] Eun-Hwan Shin. *Estimation Techniques for Low-Cost Inertial Navigation.* PhD thesis, University of Calgary, Alberta, Canada, may 2005.

[52] Stefan Knedlik, Junchuan Zhou, Zhen Dai, Ezzaldeen Edwan, and Otmar Loffeld. Analysis of low-cost GPS/INS integration for bistatic SAR experiments. In *Proceedings of the 21st International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2008)*, pages 2115–2122, 2001.

[53] S. Godha. Performance evaluation of low-cost MEMS-based IMU integrated with GPS for land vehicle navigation application. Master's thesis, Dept of Geomatics Eng.,University of Calgary, Canada, 2007.

[54] M Kayton and W. Fried. *Avionics Navigation Systems.* 1997.

[55] C. E. Schultz. INS and GPS integration. Master's thesis, Technical University

of Denmark, Lingby, Denmark, 2006.

[56] George Schmidt and R Phillips. INS/GPS integration architectures. *NATO RTO Lecture Series, EN-SET-064-05*, 2011.

[57] Jacques Georgy. *Advanced nonlinear techniques for low cost land vehicle navigation.* PhD thesis, Queenś University, Ontario, Canada, Jul 2010.

[58] Ravindra Babu and Jinling Wang. Real-time data analysis of ultra-tight GPS/INS integration, 2007.

[59] Gianluca Falco, Garry A. Einicke, John T. Malos, and Fabio Dovis. Performance analysis of constrained loosely coupled GPS/INS integration solutions. *Sensors*, 12(11):15983–16007, Nov. 2012.

[60] Y. Li, J. Wang, C. Rizos, Mumford, and W. P., Ding. Low-cost tightly coupled GPS/INS integration based on a nonlinear kalman filtering design. In *Proc. International Technical Meeting of The Institute of Navigation*, pages 958–966, Monterey, CA, EEUU, Jan. 2006.

[61] Gary Minkler and Jing Minkler. *Theory and application of Kalman filtering.* Magellan Book Company Palm Bay, 1993.

[62] Robert Grover Brown and Patrick Y C Hwang. *Introduction to Random Signals and Applied Kalman Filtering*, volume 3rd ed. John Wiley & Sons, Inc., 1997.

[63] P. Maybeck. *Stochastic Models, Estimation, and Control*, volume 1. 1979.

[64] A. Analytic Sciences Corporation Technical Staff. Gelb. *Applied optimal estimation.* MIT Press, Cambridge, Massachusetts, 1974.

[65] R. E. Kalman. New approach to linear filtering and prediction problems. *IEEE/ASME Trans. Mechatron.*, 82:34–45, 1960.

[66] Zainab Fatima Syed. *Design and Implementation Issues of a Portable Navigation System.* PhD thesis, University of Calgary, Alberta, Canada, Feb 2009.

[67] R.M. Rogers. *Applied Mathematics in Integrated Navigation Systems.* AIAA Education Series. Amer. Inst. of Aeronautics &, 2000.

[68] P.G Savage. *Introduction to Strapdown Inertial Navigation Systems.* Strapdown Associates, 2nd edition, 1996.

[69] Charles S. Smith. Piezoresistance effect in germanium and silicon. *Phys. Rev.*, 94:42–49, Apr 1954.

[70] A.M. Madni and L.A. Wan. Microelectromechanical systems (MEMS): an

overview of current state-of-the-art. In *Aerospace Conference, 1998 IEEE*, volume 1, pages 421–427 vol.1, 1998.

[71] R. Payne and K. Dinsmore. Surface micromachined accelerometer: A technology update. In *SAE*, pages 127–135, Detroit, MI, Mar 1991.

[72] Mohamed Gad-el Hak. *MEMS: introduction and fundamentals*. CRC press, 2010.

[73] K. Maenaka. MEMS inertial sensors and their applications. In *Networked Sensing Systems, 2008. INSS 2008. 5th International Conference on*, pages 71–73, 2008.

[74] RTO EDUCATIONAL NOTES EN-SET. Advances in navigation sensors and integration technology. Feb, 2004.

[75] Naser El-Sheimy and Xiaoji Niu. The promise of MEMS to the navigation community. *InsideGNSS*, 2(2):46–56, 2007.

[76] Andrei. Shkel and Cenk Acar. *MEMS Vibratory Gyroscopes*. Springer US, Boston, MA, 2009.

[77] Vibration rejecting yaw rate gyro scope ADXRS642 data sheet, 2011.

[78] Jonathan Bernstein et al. An overview of MEMS inertial sensing technology. 2003.

[79] J. Bernstein, Raanan Miller, W. Kelley, and P. Ward. Low-noise MEMS vibration sensor for geophysical applications. *Microelectromechanical Systems, Journal of*, 8(4):433–438, 1999.

[80] Ran Fang, Wengao Lu, Chang Liu, Zhongjian Chen, Yuan Ju, Guannan Wang, Lijiu Ji, and Dunshan Yu. A low-noise interface circuit for MEMS vibratory gyroscope. In *Solid-State and Integrated Circuit Technology (ICSICT), 2010 10th IEEE International Conference on*, pages 1456–1458, 2010.

[81] M.D. Pottenger and W.J. Kaiser. Noise modeling methodology for full-system inertial microsensor codesign. In *Nanotech*, volume 1, pages 140–143, 2001.

[82] IEEE standard for inertial sensor terminology. *IEEE Std 528-2001*, pages 1–20, 2001.

[83] Mohinder Grewal and Angus Andrews. How good is your gyro [ask the experts]. *Control Systems, IEEE*, 30(1):12–86, 2010.

[84] P. Aggarwal, Z. Syed, X. Niu, and N. El-Sheimy. A standard testing and

calibration procedure for low cost MEMS inertial sensors and units. *Journal of Navigation*, 61:323–336, 4 2008.

[85] VectorNav Technologies. VECTORNAV embedded navigation solutions. http://www.vectornav.com/, Jul. 2011.

[86] Walter Stockwell. Bias stability measurement: Allan variance. *Crossbow Technology, Inc. http://www.xbow.com*, 2004.

[87] Harvey Weinberg. Gyro mechanical performance: The most important parameter. Technical report, Analog Devices, Inc, Sep. 2011.

[88] I. Skog and P. Händel. Calibration of a MEMS inertial measurement unit. In *Proc. XVII IMEKO WORLD CONGRESS*, Rio de Janeiro, Brazil, Sep. 2006.

[89] 3DM-GX3-25 Technical Product Overview data sheet.

[90] Atomic IMU 6 degrees of freedom data sheet.

[91] Ismael Colomina and Maria Eulália Parés. Sensor orientation: Precise trajectory and attitude determination with INS, Jan 2012.

[92] J Allen. Micro-system inertial sensing technology overview. *Sandia Report*, 2009.

[93] Haiyang Chao, Calvin Coopmans, Long Di, and YangQuan Chen. A comparative evaluation of low-cost IMUs for unmanned autonomous systems. In *Multisensor Fusion and Integration for Intelligent Systems (MFI), 2010 IEEE Conference on*, pages 211–216. IEEE, 2010.

[94] Tye Brady. Expanding the spacecraft application base with MEMS gyros. In *SPIE MOEMS-MEMS*, pages 79280J–79280J. International Society for Optics and Photonics, 2011.

[95] Giorgio De Pasquale and Aurelio Somá. Reliability testing procedure for MEMS IMUs applied to vibrating environments. *Sensors*, 10(1):456–474, 2010.

[96] Walid Abdel-Hamid. *Accuracy Enhancement of Integrated MEMS-IMU/GPS Systems for Land Vehicular Navigation Applications.* PhD thesis, University of Calgary, Alberta, Canada, Jan 2005.

[97] A.B. Chatfield. Fundamentals of high accuracy inertial navigation. *Progress in Astronautics and Aeronautics, P. Zarchan, Ed. American Institute of Aeronautics and Astronautics*, 174, 1997.

[98] Mohammed El-Diasty, Ahmed El-Rabbany, and Spiros Pagiatakis. An accurate nonlinear stochastic model for MEMS-based inertial sensor error with wavelet

networks. *Journal of Applied Geodesy*, 1(4):201, 2007.

[99] W Flenniken, J Wall, and D Bevly. Characterization of various IMU error sources and the effect on navigation performance. In *ION GNSS*, pages 13–16, 2005.

[100] John H Wall and David M Bevly. Characterization of inertial sensor measurements for navigation performance analysis. In *Proceedings of the 19th International Technical Meeting of the Satellite Division of The Institute of Navigation ION GNSS*, pages 2678–2685, 2006.

[101] Nitaigour Premchand Mahalik. *MEMS*. Tata McGraw-Hill Education, 2008.

[102] Lawrence Ward. *Dynamical cognitive science*. The MIT Press, 2002.

[103] L. M. Ward and P. E Greenwood. 1/f noise. 2(12):1537, 2007.

[104] M.S. Keshner. 1/f noise. *Proceedings of the IEEE*, 70(3):212–218, 1982.

[105] Lawrence M. Ward and Priscilla E. Greenwood. The mathematical genesis of the phenomenon called $1/f$ noise. Technical report, University of British Columbia and University of North Carolina, Jun 2010.

[106] Yudan Yi. On improving the accuracy and reliability of GPS/INS-based direct sensor georeferencing. Technical Report 484, The Ohio State University, 2007.

[107] Sameh Nassar. *Improving the Inertial Navigation System (INS) Error Model for INS and INS/DGPS Applications*. PhD thesis, University of Calgary, Alberta, Canada, nov 2003.

[108] Ahmed H. Mohamed. *Optimizing the Estimation Procedure in INS/GPS Integration for Kinematic Applications*. PhD thesis, University of Calgary, Alberta, Canada, april 1999.

[109] Richard F. Voss. $1/f$ (flicker) noise: A brief review. In *33rd Annual Symposium on Frequency Control. 1979*, pages 40–46, 1979.

[110] J. B. Johnson. The schottky effect in low frequency circuits. *Phys. Rev.*, 26:71–85, Jul 1925.

[111] W. Schottky. Small-shot effect and flicker effect. *Phys. Rev.*, 28:74–103, Jul 1926.

[112] J. Bernamont. *Fluctuations de potential aux bornes d'un conducteur métallique de faible volume parcouru par un courant*. Annalen del Physik, 1937.

[113] Sveinung Erland and Priscilla E. Greenwood. Constructing $1/\omega^{\alpha}$ noise from reversible markov chains. *Phys. Rev. E*, 76:031114, Sep 2007.

[114] A. Noureldin, T.B. Karamat, M.D. Eberts, and A. El-Shafie. Performance enhancement of MEMS-based INS/GPS integration for low-cost navigation applications. *IEEE Trans. Veh. Technol.*, 58(3):1077 –1096, Mar. 2009.

[115] Vaibhav Saini, S. C. Rana, and MM Kuber. Online estimation of state space error model for MEMS IMU. *Journal of Modelling and Simulation of Systems*, 1:219–225, Aug. 2010.

[116] A Waegli and Jan Skaloud. Assessment of GPS/MEMS-IMU integration performance in ski racing. In *Proc. of ENC-GNSS (TimeNav'07)*, Geneva, Switzerland, Jun. 2007.

[117] Minha Park and Yang Gao. Error and performance analysis of MEMS-based inertial sensors with a low-cost GPS receiver. *Sensors*, 8(4):2240–2261, 2008.

[118] R. V. C. Wong, K. P. Schwarz, and M. E. Cannon. High-accuracy kinematic positioning by GPS-INS. *Navigation*, 35(2):275–288, 1998.

[119] Chul Woo. Kang, Chang Ho. Kang, and Chan Gook Park. Wavelet de-noising technique for improvement of the low cost MEMS-GPS integrated system. In *International Symposium on GPS/GNSS*, Oct. 2010.

[120] Xingming Wu, Li Duan, and Weihai Chen. A Kalman Filter approach based on random drift data of fiber optic gyro. In *Industrial Electronics and Applications (ICIEA), 2011 6th IEEE Conference on*, pages 1933 –1937, Jun. 2011.

[121] MicroStrain, Williston VT, USA. *3DM-GX3-25 Data Communications Protocol*, 2011. Release 1.14.

[122] Navsas. Navigation signal analysis and simulation, sat-surfer training board & software suite for GNSS training. http://www.navsas.eu/, May. 2013. Accessed: 12/03/2013.

[123] G. Heinzel, A. Rudiger, and R. Schilling. Spectrum and spectral density estimation by the discrete fourier transform (DFT), including a comprehensive list of window functions and some new at-top windows. Technical report, Albert-Einstein-Institut, 2002.

[124] Xiaoji Niu, Sameh Nasser, Chris Goodall, and Naser El-Sheimy. A universal approach for processing any MEMS inertial sensor configuration for land-vehicle navigation. *Journal of Navigation*, 60:233–245, Apr. 2007.

[125] Michel Misiti, Yves Misiti, Georges Oppenheim, and Jean-Michel Poggi. *Wavelet*

Toolbox: for use with MATLAB; [User's Guide]. MathWorks, 1996.

[126] G.A. Einicke, J.T. Malos, D.C. Reid, and D.W. Hainsworth. Riccati equation and EM algorithm convergence for inertial navigation alignment. *IEEE Trans. Signal Processing*, 57(1):370–375, Jan. 2009.

[127] R H Shumway and D S Stoffer. An approach to time series smoothing and forecasting using the EM algorithm. *Journal of Time Series Analysis*, 3(4):253–264, 1982.

[128] CF Wu. On the convergence properties of the EM algorithm. *The Annals of Statistics*, 11(1):95–103, 1983.

[129] Dongliang Huang, H. Leung, and N. El-Sheimy. Expectation maximization based GPS/INS integration for land-vehicle navigation. *IEEE Trans. Aerosp. Electron. Syst.*, 43(3):1168–1177, Jul. 2007.

[130] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38, 1977.

[131] Geoffrey J McLachlan and Thriyambakam Krishnan. *The EM algorithm and extensions*, volume 382. Wiley-Interscience, 2007.

[132] Elizabeth Eli Holmes. Derivation of an EM algorithm for constrained and unconstrained multivariate autoregressive state-space (MARSS) models. *Northwest Fisheries Science Center, NOAA Fisheries*, 2725, 2012.

[133] Petko Petkov and Tsonyo Slavov. Stochastic modeling of MEMS inertial sensors. *Cybernetics and information technologies*, 10(2), 2010.

[134] A.G. Quinchia, C. Ferrer, G. Falco, and F. Dovis. Constrained non-linear fitting for stochastic modeling of inertial sensors. In *Design and Architectures for Signal and Image Processing (DASIP), 2013 Conference on*, pages 119–125, Oct 2013.

[135] C.A. Greenhall. Recipes for degrees of freedom of frequency stability estimators. *IEEE Trans. Instrum. Meas.*, 40(6):994–999, Dec 1991.

[136] D.A. Howe, D.W. Allan, and J.A. Barnes. Properties of signal sources and measurement methods. In *Thirty Fifth Annual Frequency Control Symposium. 1981*, pages 669–716, May 1981.

[137] Abdeslam Serroukh, AT Walden, and DB Percival. Statistical properties and uses of the wavelet variance estimator for the scale analysis of time series. *Journal of*

the American Statistical Association, 95(449):184–196, 2000.

[138] Donald P Percival. On estimation of the wavelet variance. *Biometrika*, 82(3):619–631, 1995.

[139] Donald B Percival and Peter Guttorp. Long-memory processes, the Allan variance and wavelets. *Wavelets in Geophysics*, 4:325–344, 1994.

[140] F. Vernotte and E. Lantz. Statistical biases and very-long-term time stability analysis. *IEEE Trans. Ultrason., Ferroelect., Freq. Contr.*, 59(3):523–530, 2012.

[141] *Global Optimization Toolbox: For Use with MATLAB;[user's Guide]*. MathWorks, 2011.

[142] Tamara G Kolda, Robert Michael Lewis, and Virginia Torczon. A generating set direct search augmented lagrangian algorithm for optimization with a combination of general and linear constraints. *Sandia National Laboratories*, 2006.

[143] Robert Michael Lewis and Virginia Torczon. A globally convergent augmented lagrangian pattern search algorithm for optimization with general constraints and simple bounds. *SIAM Journal on Optimization*, 12(4):1075–1089, 2002.

[144] Samuel Prívara, Jiří Cigler, Zdeněk Váňa, Frauke Oldewurtel, Carina Sagerschnig, and Eva Žáčeková. Building modeling as a crucial part for building predictive control. *Energy and Buildings*, 56:8–22, 2013.

[145] M El-Diasty, A El-Rabbany, and S Pagiatakis. Temperature variation effects on stochastic characteristics for low-cost MEMS-based inertial sensor error. *Measurement Science and Technology*, 18(11):3321, 2007.

[146] M El-Diasty, A El-Rabbany, and S Pagiatakis. Stochastic characteristics of temperature-dependent MEMS-based inertial sensor error. In *The Institute of Navigation National Technical Meeting*, pages 18–20, 2006.

[147] Vivek Agarwal, Hemendra Arya, Biswanath Nayak, and Lalit R Saptarshi. Extended Kalman filter based loosely coupled INS/GPS integration scheme using FPGA and DSP. *International Journal of Intelligent Defence Support Systems*, 1(1):5–26, 2008.

[148] Zong-Tao Li, Tie-Jun Wu, Can-Long Lin, and Long-Hua Ma. Field programmable gate array based parallel strapdown algorithm design for strapdown inertial navigation systems. *Sensors*, 11(8):7993–8017, 2011.

[149] N Shantha Kumar and T Jann. Estimation of attitudes from a low-cost miniaturized inertial platform using Kalman filter-based sensor fusion algorithm. *Sadhana*, 29(2):217–235, 2004.

[150] G. Stringham. *Hardware/Firmware Interface Design: Best Practices for Improving Embedded Systems Development*. Elsevier Science, 2009.

[151] Jung Soon Jang and Darren Liccardo. Automation of small UAVs using a low-cost MEMS sensor and embedded computing platform. In *25th Digital Avionics Systems Conference, 2006 IEEE/AIAA*, pages 1–9. IEEE, 2006.

[152] Walid Farid Abdelfatah, Jacques Georgy, Umar Iqbal, and Aboelmagd Noureldin. FPGA-based real-time embedded system for RISS/GPS integrated navigation. *Sensors*, 12(1):115–147, 2011.

[153] Syed M Qasim, Ahmed A Telba, and Abdulhameed Y AlMazroo. FPGA design and implementation of matrix multiplier architectures for image and signal processing applications. *IJCSNS International Journal of Computer Science and Network Security*, 10(2):168–177, 2010.

[154] Clive Maxfield. *The design warrior's guide to FPGAs: devices, tools and flows*. Access Online via Elsevier, 2004.

[155] Fernando Pardo Carpio and José A Boluda Grau. *VHDL: lenguaje para síntesis y modelado de circuitos*. Ra-ma, 1999.

[156] Bluemore200-bluetooth Uart/RS232 Module data sheet, 2011.

[157] Rod Jesman, Fernando Martinez Vallina, and Jafar Saniie. *MicroBlaze Tutorial Creating a Simple Embedded System and Adding Custom Peripherals Using Xilinx EDK Software Tools*. Embedded Computing and Signal Processing Laboratory-Illinois Institute of Technology, Illinois, USA, 2006.

[158] Xilinx, Inc. *Microblaze processor reference guide*, 2009. Release 9.0.

[159] LM35 Precision Centigrade Temperature Sensors data sheet, 1999.

[160] Weidong Ding, Jinling Wang, Yong Li, Peter Mumford, and Chris Rizos. Time synchronization error and calibration in integrated GPS/INS systems. *ETRI journal*, 30(1):59, 2008.

[161] Jan Skaloud and Patrick Viret. GPS/INS integration: From modern methods of data acquisition fo new applications. *European Journal of Navigation*, 2(4):40–44, 2004.

[162] I. Skog and P. Händel. Time synchronization errors in loosely coupled GPS-aided inertial navigation systems. *IEEE Trans. Intell. Transport. Syst.*, 12(4):1014–1023, 2011.

[163] u-blox, Thalwil, Switzerland. *u-blox 6 Receiver Description Including Protocol Specification*, 2011. Revision for FW 7.03.

[164] A. Sundararajan and C. Cain. Using and creating flash files for the microblaze development kit-spartan-3a DSP 1800a starter platform, 2008.

[165] Bryan H. Fletcher. Using serial flash on the xilinx spartan-3e starter board, Oct 2007.

[166] Ju-wook Jang, Seonil Choi, and V.K. Prasanna. Area and time efficient implementations of matrix multiplication on |FPGAs. In *Field-Programmable Technology, 2002. (FPT). Proceedings. 2002 IEEE International Conference on*, pages 93–100, 2002.

[167] A. Amira and F. Bensaali. An FPGA based parameterizable system for matrix product implementation. In *Signal Processing Systems, 2002. (SIPS '02). IEEE Workshop on*, pages 75–79, Oct 2002.

[168] Ling Zhuo and Viktor K. Prasanna. Scalable and modular algorithms for floating-point matrix multiplication on FPGAs. In *In Proc. of The 18th International Parallel & Distributed Processing Symposium*, 2004.

[169] Scott J. Campbell and Sunil P. Khatri. Resource and delay efficient matrix multiplication using newer FPGA devices. In *Proceedings of the 16th ACM Great Lakes symposium on VLSI*, GLSVLSI '06, pages 308–311, New York, NY, USA, 2006. ACM.

[170] XILINX. *Virtex-5 FPGA XtremeDSP Design Considerations; [User's Guide]*. MathWorks, Jun 2010.

[171] Pololu Corporation. Pololu robotics and electronics. http://www.pololu.com/, May. 2013. Accessed: 20/05/2013.

[172] Curtis L. Olson. Flightgear flight simulator. http://www.flightgear.org/, Dec. 2013.

[173] A. Solimeno. Low-cost INS/GPS data fusion with extended Kalman filter for airborne applications. Master's thesis, Universidad Tecnica de Lisboa, Portugal, 2007.