Development and Validation of an IMU/GPS/Galileo Integration Navigation System for UAV

By

Maria de Fátima Alves Nunes Bento

April 2013

Universität der Bundeswehr München
Fakultät für Luft- und Raumfahrttechnik
Institut für Raumfahrttechnik und Weltraumnutzung

Development and Validation of an IMU/GPS/Galileo Integration Navigation System for UAV

Dipl.-Ing. Maria de Fatima Alves Nunes Bento

Vollständiger Abdruck der bei der
Fakultät für Luft- und Raumfahrttechnik
der Universität der Bundeswehr München
zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

Eingereichten Dissertation

Vorsitzender:	Univ.Professor. Dr.-Ing Axel Schulte
1.Berichterstatter:	Univ. Professor. Dr.-Ing Günter W. Hein
2.Berichterstatter:	Univ. Professor. Dr.-Ing Bernd Eissfeller

Diese Dissertation wurde am 18.10.2012 bei der Universität der Bundeswehr München eingereicht und durch die Fakultät für Luft- und Raumfahrttechnik am 14.11.2012 November 2012 angenommen. Die mündliche Prüfung fand am 19.04.2013 April 2013 statt.

Abstract

Several and distinct *Unmanned Aircraft Vehicle* (UAV) applications are emerging, demanding steps to be taken in order to allow those platforms to operate in an un-segregated airspace. The key risk component, hindering the widespread integration of UAV in an un-segregated airspace, is the autonomous component: the need for a high level of autonomy in the UAV that guarantees a safe and secure integration in an un-segregated airspace. At this point, the UAV accurate state estimation plays a fundamental role for autonomous UAV, being one of the main responsibilities of the onboard autopilot.

Given the 21st century global economic paradigm, academic projects based on inexpensive UAV platforms but on expensive commercial autopilots start to become a non-economic solution. Consequently, there is a pressing need to overcome this problem through, on one hand, the development of navigation systems using the high availability of low cost, low power consumption, and small size navigation sensors offered in the market, and, on the other hand, using *Global Navigation Satellite Systems* Software Receivers ($GNSS_{SR}$). Since the performance that is required for several applications in order to allow UAV to fly in an un-segregated airspace is not yet defined, for most UAV academic applications, the navigation system accuracy required should be at least the same as the one provided by the available commercial autopilots.

This research focuses on the investigation of the performance of an integrated navigation system composed by a low performance inertial measurement unit (IMU) and a $GNSS_{SR}$. A strapdown mechanization algorithm, to transform raw inertial data into navigation solution, was developed, implemented and evaluated. To fuse the data provided by the strapdown algorithm with the one provided by the $GNSS_{SR}$, an Extended Kalman Filter (EKF) was implemented in loose coupled closed-loop architecture, and then evaluated. Moreover, in order to improve the performance of the IMU raw data, the Allan variance and denoise techniques were considered for both studying the IMU error model and improving inertial sensors raw measurements.

In order to carry out the study, a starting question was made and then, based on it, eight questions were derived. These eight secondary questions led to five hypotheses, which have been successfully tested along the thesis.

This research provides a deliverable to the Project of Research and Technologies on Unmanned Air Vehicles (PITVANT) Group, consisting of a well-documented UAV

navigation algorithm, an implemented and evaluated navigation algorithm in the MatLab environment, and Allan variance and denoising algorithms to improve inertial raw data, enabling its full implementation in the existent Portuguese Air Force Academy (PAFA) UAV. The derivable provided by this thesis is the answer to the main research question, in such a way that it implements a step by step procedure on how the Strapdown IMU (SIMU)/ $GNSS_{SR}$ should be developed and implemented in order to replace the commercial autopilot.

The developed integrated SIMU/$GNSS_{SR}$ solution evaluated, in post-processing mode, through van-test scenario, using real data signals, at the Galileo Test and Development Environment (GATE) test area in Berchtesgaden, Germany, when confronted with the solution provided by the commercial autopilot, proved to be of better quality.

Although no centimetre-level of accuracy was obtained for the position and velocity, the results confirm that the integration strategy outperforms the Piccolo system performance, being this the ultimate goal of this research work.

This dissertation mainly recommends that, as soon as possible, steps be taken in order to overcome the restrictions associated with the feasibility of performing UAV flight tests in the GATE area, in order to test the developed algorithm under realistic flight conditions. If these flight tests can be performed, as soon as possible, the PAFA UAV will be the first UAV flying in the GATE.

Keywords: Strapdown, IMU, Allan Variance, EKF, WMRA.

Resumo

A variedade de aplicações, tanto militares como civis, dos Veículos Aéreos não Tripulados (UAV) está a aumentar, fazendo com que seja necessário tomar medidas que permitam a sua utilização em espaço aéreo não segregado. O factor de risco que tem atrasado a integração destes sistemas em espaço aéreo não segregado tem a ver com a necessidade de o UAV possuir um elevado grau de autonomia que lhe permita uma integração segura naquele tipo de espaço. Atualmente, o cálculo preciso associado à determinação tanto da posição e velocidade como da atitude do UAV é fundamental para o seu funcionamento em modo autónomo, sendo este cálculo uma das responsabilidades do piloto automático.

Na conjuntura actual, projetos académicos baseados em plataformas baratas mas por outro lado levando a bordo pilotos automáticos caros, torna esta opção um contrassenso. Consequentemente, existe a necessidade de ultrapassar este problema através do desenvolvimento de sistemas de navegação de baixo custo, disponíveis no mercado, baseados em sensores baratos, de baixo consumo, e de dimensões reduzidas. A este desenvolvimento, deverá ser associada a utilização de *Global Navigation Satellite Systems* Software Receivers ($GNSS_{SR}$). Como os requisitos associados às necessidades que visam permitir que um UAV voe em espaço aéreo não segregado ainda não estão definidas, considera-se, no mundo académico, que a precisão do sistema de navegação do UAV deverá, no mínimo, ser igual à das pilotos automáticos disponíveis no mercado.

Esta tese centra-se na investigação do desempenho de um sistema de navegação integrado, composto por uma unidade de medida inercial (IMU) de baixo desempenho e por um $GNSS_{SR}$. Foi desenvolvido, implementado e validado um algorithm de mecanização responsável por transformar os dados inercias brutos numa solução de navegação. O algoritmo de fusão sensorial escolhido para integrar os dados provenientes do IMU e do GNSSsr foi implementado utilizando um Extended Kalman Filter (EKF), que foi desenvolvido e avaliado com base numa arquitetura de baixo acoplamento. De modo a aumentar o desempenho dos dados provenientes do IMU, utilizou-se não só a técnica de Allan variance mas também técnicas de elminição de ruídos, quer para estudar o modelo, quer para melhorar as medições e ruidosas dos sensores de inércia.

Este estudo foi orientado por uma pergunta de partida, da qual derivaram oito perguntas secundárias. Estas conduziram a cinco hipóteses a testar durante o trabalho de investigação.

A investigação originou um software para o Projecto de Investigação e Tecnologias em Vehiculos Aéreos Não Tripulados (PITVANT), que inclui: *i)* um algoritmo de navegação de UAV documentado; *ii)* um algoritmo de navegação implementado e testado em ambiente Matlab; *iii)* algoritmos de Allan Variance e eliminação de ruído que visam o melhoramento dos dados inerciais. O objetivo último deste software consiste na sua fácil implementação nos UAV do projecto PITVANT, servindo assim de resposta à pergunta de partida no sentido em que apresenta quais os procedimentos a seguir para o desenvolvimento e implementação de um SIMU/GNSSsr que visa a substituituir as funções, associadas à navegação, de um piloto automático comercial.

A solução SIMU/GNSSsr testada, através da realização de testes numa viatura, em modo pós-processamento, utilizando sinais da dados obtidos na *Galileo Test and Development Environment* (GATE), em Berchtesgaden, Alemanha, provou ser uma solução com melhor qualidade relativamente áquela fornecida pelo piloto automático comercial.

Embora não tenha sido atingida uma precisão de posição ao nível do centímetro, os resultados obtidos provam que a estratégia de integração seguida é superior à do sistema Piccolo, tendo sido este o objectivo último deste trabalho.

Esta dissertação recomenda que, logo que possível, sejam tomadas as medidas necessárias conducentes ao levantamento das restrições associadas à possibilidade de efetuar testes com o UAV na GATE, de modo a testar o algoritmo obtido sob condições de voos reais. Caso estes testes de voo sejam efectuados, o UAV da Força Aérea Portuguesa será o primeiro a voar na zona da GATE, Alemanha.

Acknowledgements

People are human, sensors are instruments. They have one thing in common, though: Both are affected by errors!

If for humans such errors limit the relationships, for the instruments such errors limit the accuracy with which the observable can be measured.

For humans, the best we can do is to model our minds in order to be able to interact with whoever is around us! For instruments, the best we can do is to model our sensor in such a way that those errors can be minimized… no matter what type of sensor you use…

No matter what you do, perfection is never achieved but you can get pretty close if you try hard enough!

A PhD is an emotionally difficult time: throughout the research, I had moments of creativity, days of nostalgia, and I also felt moments of "I-am-ready-to-pack-it-all"!

At the beginning, you can´t take the first step because you fear it will not be good enough! At the middle, you do not feel able to make a decision because you fear it might be wrong, and then you spend too long planning until you run out of time. When do you finish the PhD? You realize that perfection really does not exist but you can, at least, try! If you are realistic in your goals, if you keep focused and make a consistent effort, than, you have a good chance of finishing you PhD. However, is takes a lot effort, discipline and, of course, sweat!

This thesis culminates five years of PhD research at the FAF Munich University, which would never have been achieved without the help and support of some persons. Those goals would never have been achieved without the help of some of you and without the trust that the Portuguese Air Force Academy has given me, inspiring me to do my best in all my research.

To the Portuguese Air Force Chief of Staff, General José Pinheiro, and Lieutenant-Colonel José Morgado for their efforts into making the agreement between the Institute of Geodesy and the Portuguese Air Force Academy, paving the way for my coming to Munich.

To the Portuguese Air Force Academy Commander, General João Cordeiro.

To the former Portuguese Air Force Academy Commander, General Serôdio Fernandes. Thank you!

I would, also, like to express my appreciation for part of the financial support provided by the *Fundação Calouste Gulbenkian*.

I would like to thank the Faculty of the Institute of Geodesy and Navigation; in particular, I would like to express my deepest and most sincere gratitude to my supervisors, Professor Guenter W. Hein and Professor Bernd Eissfeller with whom I have had the pleasure and luck of working with.

I would like to thank my fellow colleagues of the Institute with whom I had the privilege of working with.

Warm thanks also to my colleagues and friends from Portugal, whose contributions were of enormous value to me. In particular, many thanks to Carlos Vaz for revising the English of my entire PhD thesis.

Warm thanks also to Ana, Manuela and Carla.

A deep gratitude for my mother, Lucinda, and my father, António, who have taught me to be always honest in my work. I feel a deep gratitude to my sisters Alexandra, Maria do Céu, Natália and Isabel, and also to my cousin Maria José, for all their support in the last years.

Last, but not least, my special gratitude is to my husband, Paulo Bento, and to our beautiful daughter Lara Bento, who was born in the second year of this long journey and whose raising up seems to be harder than making a PhD!

Thank you Friends and Family for having put up with my bad mood of the last years!

Table of Contents

List of Figures

List of Tables

List of Acronyms

6DoF	6 Degrees of Freedom
ADU	Air Data Unit
AHRS	Attitude Heading and Reference System
AI	Artificial Intelligence
AGL	Above Ground Level
ASIC	Application Specific Integrated Circuit
ANFIS	Adaptive Neural Fuzzy Information Systems
ANN	Artificial Neural Networks
ARW	Angular Random Walk
C/A Code	Coarse-Acquisition Code
C3UV	Centre for Collaborative Control of Unmanned Vehicles
CCT	Cloud Cap Technology
CFMTFA	Military and Technical Training Center
CST	Cluster Sampling Technique
DASD	Direct Access Storage Devices
db	Daubechies
DCM	Direction Cosine Matrix
DGPS	Differential GPS
DKF	Discrete Kalman Filter
DMU	Digital Measurement Unit
DS&A	Detect, Sense & Avoid Systems
DSP	Digital Signal Processor
DWT	Discrete Wavelet Transform
ECEF	Earth Centred Earth Fixed
ECI	Earth Centred Inertial
EKF	Extended Kalman Filter
ELINT	Electronic Intelligence
EMB	Embraer, Empresas Brasileiras de Aeronáutica S.A

ENU	East North Up
FOI	Swedish Defence Research Agency
FDR	Flight Data Record
FOG	Fiber Optic Gyros
GATE	Galileo Test and Development Environment
GCS	Ground Control Station
GNC	Guidance, Navigation and Control
GNSS	Global Navigation Satellite System
$GNSS_{SR}$	Global Navigation Satellite System Software Receiver
GPS	Global Positioning System
HALE	High Altitude Long Endurance
HP	High Pass
IKF	Indirect Kalman Filter
IDWT	Indirect Discreet Wavelet Transformation
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
INU	Inertial Navigation Unit
ISA	Inertial Sensor Assembly
KF	Kalman Filter
LKF	Linearized Kalman Filter
LOD	Level Of Decomposition
LOS	Line Of Sigh
MALE	Medium Altitude Long Endurance
MEMS	Micro Electromechanical Systems
PAFA	Portuguese Air Force Academy
PITVANT	Project of Research and Technologies on Unmanned Air Vehicles
R&D	Research and Development
RGU	Rate Gyro Unit
RLG	Ring Laser Gyro
RNP	Required Navigation Performance
SDK	Standard Developer Kit

SIMU	Strapdown Inertial Measurement Unit
SINS	Strapdown Inertial Navigation System
SLAM	Simultaneous Localization And Mapping
SSD	Solid State Drive
STD	Standard Deviation
UART	Universal Asynchronous Receiver Transmitter
UAS	Unmanned Aircraft Systems
UAV	Unmanned Aircraft Vehicle
UCB	University of California at Berkeley
UGV	Unmanned Ground Vehicles
UKF	Unscented Kalman Filter
VGU	Vertical Gyro Unit
VTOL	Vertical Take Off and Landing
WMRA	Wavelet Multi-Resolution Analysis
WLAN	Widespread Wireless Local Area Network

PART I

INTRODUCTION AND BACKGROUND CONCEPTS

“Either then the earth is spherical or it is at least naturally spherical. And it is right to call anything that which nature intends it to be, and which belongs to it, rather than which it is by constraint and contrary to nature. The evidence of the senses further corroborates this. How else would eclipses of the moon show segments shaped as we see them? As it is, the shapes which the moon itself each month shows are of every kind-straight, gibbous, and concave-but in eclipses the outline is always curved: and, since it is the interposition of the earth that makes the eclipse, the form of this line will be caused by the form of the earth's surface, which is therefore spherical."

Works of Aristotle, vol., I, p. 389.

CHAPTER 1

INTRODUCTION

1 Introduction

This chapter starts with an overview of the background and literature review related to this research. Given the fact that this thesis does not have one central literature review, several topics related to Unmanned Aircraft Vehicle (UAV), Inertial Navigation System (INS), low performance inertial sensors, strapdown algorithms, and denoising techniques will be discussed. As next, the problem statement for this study is presented as well as the constraints, and the adopted methodology. The present chapter finalizes with the presentation of the thesis outline.

1.1 Background and Literature Review

UAS. It is a fact that the Unmanned Aircraft Systems (UAS) are changing the way military and civil operations are carried out. Associated to the development of high technology navigation and operation systems, it is possible, nowadays, to use the UAS platform (UAV) to perform several tasks such as data and image acquisition of affected areas, localization and tracking of specific targets, map building, communication relays, pipeline surveying, border patrolling, military operations, policing duties, search and rescue, and traffic surveillance.

UAV Research Groups. In particular, the scientific world uses unmanned vehicles for, among others: ocean and climate research; environment and Earth research; magnetic, radiological, gravimetric mapping, and geophysical monitoring of natural processes; platforms to test development of new technologies and operation concepts for UAV. As a matter of fact, several institutes and/or universities have been using commercial UAV as test platforms for their experiments, while others are focusing on developing their own UAV with personalized capabilities required for specific applications.

UAV Research in Portugal. Undertaken and led by the Portuguese Air Force Academy (PAFA), the professed Project of Research and Technologies on Unmanned Air

Vehicles (PITVANT) project has caught the attention of a few international entities such as the Institute of Geodesy and Navigation (IFEN), University FAF Munich, Germany; The University of California at Berkeley (UCB), Centre for Collaborative Control of Unmanned Vehicles (C3UV); Swedish Defence Research Agency (FOI); Embraer, Empresas Brasileiras de Aeronáutica S.A (EMB), and Honeywell. The main objective of the PITVANT project is to promote R&D activities in several aeronautics-related areas of interest not only for the PAFA but also for their partners in the PITVANT project. The ANTEX-M, from the Portuguese acronym for "Aeronave Não Tripulada Experimental-Militar", is the PAFA UAS platform. It provides us with the capability to test control systems, develop intelligent structures for detection of vehicle defects, and to test control algorithms for autonomous team vehicles.

UAV Airworthiness. It is important to realize the fact that although the UAV range of application is rising, the current legislation and standards only allow the UAV to fly in segregated airspace (CAA, 2010), limiting the use of UAV for civil purposes. Actually, parts of the civil and/or military applications imply that the UAV must be able to fly not only in a segregated airspace but also in an un-segregated airspace. However, currently, it appears that most UAV in Europe are confined to an airspace specifically assigned to that purpose (segregated airspace) or are operated over the sea. A good example of this is the PITVANT project that uses the Military and Technical Training Centre (CFMTFA) – former Air Base No. 2, located at Ota – to perform all the needed fly tests. This operational area creates privileged conditions for the UAS flight tests. Nonetheless, if it has been agreed that the UAV builders are responsible for testing their own products, the reality is that the requirements, subjacent to the need to operate such platforms in un-segregated airspace, will require certification and formal flight-testing (Ingham, 2008). Still, the operation of the UAV in un-segregated airspace can be unsafe for the manned aircraft, as well as, for the crews on the ground, being this factor a limitation to its operation in un-segregated airspace. As a matter of fact, due to the several existing restrictions, up to know, there are no clear rules regarding the law and specific requisites needed for the airworthiness certification of an UAV, so that it can operate in a un-segregated airspace (CAA, 2010).

UAV Regulation, Certification, and Integration. The growing national operation of such platforms demands that concrete steps be taken in order to implement a process for the airworthiness certification of the military and civil UAV. To do it so, it is necessary, first, to face three major challenges: regulation, certification and integration.

The regulation aims to establish standards and requirements for the certification of

UAV. It should be stressed that the regulation should take into account the type of UAV. After solving the issue of regulation, we can move to the stage dealing with the certification standards and requirements for each of the different types of UAV. Finally, the integration of UAV into un-segregated airspace demands that certain requirements are met to allow for a safe operation. This integration must be done to ensure an equivalent level of safety (at least) to the one that exists for manned aircraft (Carrasco, 2009). Taking into account the specificities of each type of UAV, this issue implies the use of security technology requirements; for instance, a UAV which aims to operate in un-segregated airspace needs to comply with more restrictive technological requirements than one which aims to operate in a segregated airspace – for example the use of Detect, Sense & Avoid Systems (DS&A). In fact, it is expected that those UAV will have to comply with similar operational requirements to the ones of manned aircraft. As a consequence, they must be subject to similar flight tests in order to prove that the UAV technology is mature enough to permit them to fly in the same airspace as manned aircraft (Ingham, 2008).

UAV Autonomy. Effectively, in order to fly in un-segregated airspace, it is crucial that the UAV has the capability of safe autonomous flight control. Concerning autonomy, the navigation system component plays a fundamental role. In fact, self-navigation makes the big difference between a remotely controlled vehicle and an autonomous UAV, permitting operation out of visual contact. Automated control of an aerial vehicle presumes that the UAV has accurate information about its position, velocity and orientation.

UAV Avionics. In order to have an autonomous flight, the UAV must hold avionics systems that enable it to maintain a stable attitude and to follow the desired trajectories. Consequently, one of the most important capabilities of an autonomous UAV is its state estimation or localization. As a matter of fact, the flight control algorithm performance of an autonomous or semi-autonomous UAV depends mostly on the information about its state. Neither the navigation nor the control of an autonomous vehicle can exist without the UAV state estimation.

Navigation and Control Measurements Sensors. The need to determine the vehicle state implies that some essential measurements be considered. Particularly, the following measurements are needed: 3D position; ground velocity; accelerations; attitude; heading; angular rates and time. Such measurements can then be obtained from several navigational units, such as the INS, Global Positioning System (GPS), Differential GPS (DGPS), and/or GNSS. For an autonomous or even semi-autonomous unmanned vehicle, attitude, linear accelerations and some air data must be provided to its flight control algorithm. Attitude

information, pitch, roll and heading angles, can be obtained in several ways. Pitch and roll angles can be obtained from the Inertial Navigation Unit (INU), from the Digital Measurement Unit (DMU) or from the Vertical Gyro Unit (VGU). Whereas, heading angle may be obtained from the INU or the compass. The angular rates, pitch, roll and yaw rate, can be obtained from the INU, DMU, or the Rate Gyro Unit (RGU). Also important are the linear accelerations (longitudinal, lateral and vertical) obtained from the INU, DMU or the accelerometers. Air data information, like airspeed and barometric altitude, can be extracted from the Air Data Unit (ADU), optical/laser sensors or Pitot tubes. An alternative to the individual gyroscope or Inertial Measurement Unit (IMU) is a self-contained Attitude Heading and Reference System (AHRS). Indeed, there are others forms for extracting attitude information, for instance, magnetometers are widely used to determine the attitude of UAV by measuring the Earth's magnetic field.

Low Cost Navigation System Component. The need for low cost autonomous UAV has led to and stimulated the development of the Micro Electromechanical Systems (MEMS). This technology has greatly influenced the highly integrated and light weight navigation systems, as well as, the development of miniature sensors, such as microcontrollers and autopilots. Nowadays, the challenge is to work with the current generation of low cost MEMS-Based sensors, providing powerful robust navigation capabilities that can deal with the large errors experienced with these low grade sensors. Achieving a low cost navigation solution with almost the same performance of that provided by more expensive and accurate inertial navigation systems in UAV applications is still a big challenge.

Inertial navigation systems, more precisely IMU, are widely used as sensors for position estimation. Current IMU technologies come in many shapes, sizes and cost, depending on the application and performance required. Although MEMS inertial sensors offer small, cheap, light and power-efficient units, they are not currently capable of providing high accuracy, precision navigation solutions due to the inherent measurement noise. In fact, compared to expensive and higher-grade navigation systems, a Strapdown IMU (SIMU[1]), using the MEMS accelerometer and gyroscope sensors, presents big position and attitude errors over very short time intervals. However, if this accuracy can be improved by integrating low cost IMU sensors with other sensors, using sensor fusion methods, new

[1] Position, velocity and orientation calculated numerically and integrating the acceleration and angular rate data measured by the low performance IMU sensors strapped-down to the object being tracked.

applications can emerge, with lower prices and higher accuracy. Although external sensors for INS/IMU aiding, such as vision/radar, are widely used in autonomous UAV, the GPS/INS integration is still the most commonly used option. In fact, the advantages and disadvantages of the INS/SIMU and the GPS make them complementary, being the best solution for estimating the UAV position, velocity and attitude. Additionally, the best estimate is obtained by combining both the INS/SIMU and GPS measurements with one of the existing integration methods.

State Estimation Methods. Regarding the estimation methods for integrated navigation data, the more common approaches are: the Kalman Filter (KF), the Linearized Kalman Filter (LKF) or the Extended Kalman Filter (EKF); sampling based filters, like the particle filters and the Unscented Kalman Filter (UKF), and Artificial Intelligence (AI) based methods, such as the Artificial Neural Networks (ANN) or the Adaptive Neural Fuzzy Information Systems (ANFIS).

The LKF and the EKF have an extensive use in the design of some navigation software. Given the simplicity and low computational demand of the LK filters, they have been very attractive for low cost UAV applications. Each has its advantages and disadvantages; the choice of which one to use will depend on the particular situation at hand. The EKF can provide a more accurate solution but it is more complex than the LKF and it also requires a higher computational overhead. The EKF has been successfully applied to helicopter state estimation problems, see (Jun, et al., 1999) and (Gavrilets, 2003) for more details. However, in 2004 a comparison analysis between the EKF and the UKF (St-Pierre, et al., 2004) showed that the EKF had some weaknesses. In fact, since the sensor model used in the filter is strongly nonlinear, the UKF is better for the performance estimations. Moreover, regarding the implementation of this sampling based filter, the UKF is simpler than the EKF, because no state equations derivatives need to be calculated. For application in small UAV, this point is very important given the limitations of the onboard computer. An interesting idea has been proposed by (Merwe, et al., 2004) related with the possibility of implementing the UKF in its square root form, since it has been proved to present improved numerical stability along with reduced computational complexity. A research group from Stanford University has also presented, in (Langelaan, et al., 2004), a method for the navigation of a small UAV in an unsurveyed environment. Considering the sensor limitations, the group addresses the problem of estimating the state of the air vehicle in such an environment. The results of the simulations conducted in two dimensions showed that while an EKF implementation diverges, the UKF implementation generates consistent estimates of the state of the vehicle.

Another sampling based filter is the particles filter, also known as sequential Monte Carlo filter, which has been widely developed for nonlinear/non-Gaussian processes based on the Bayesian filtering theory (Godsill, et al., 2000). These methods have been put aside, mainly due to the lack of computing capacity. However, recent research has been conducted in order to apply these methods to some practical problems such as communications, computer vision and radar target tracking (Taha, et al., 2003).

In addition, there are AI based estimation methods that are well differentiated from the two types presented before. They differ on the fact that they do not use any mathematical models in the system dynamics and measurements (Schumacher, 2000). Although these methods are considered simpler to implement in terms of design, they present some limitations such as the fact that no statistical information is needed as input. There are several approaches, based on AI methods, dealing with the navigation and control of the UAV. For instance, non-linear adaptive control is considered in (Schumacher, 2000), fuzzy logic is proposed by (Nikolos, et al., 2003), and a framework for fuzzy logic based UAV navigation and control is proposed in (Doitsidis, et al., 2004).

UAV Challenges. Nowadays, the typical topics that the major universities are working on are the development of autonomous Micro and Mini UAV; autonomous Vertical Take Off and Landing (VTOL) UAS platform applications and Vision-based systems for navigation and control of autonomous vehicles; collision avoidance; multi-vehicle systems integration and Simultaneous Localization and Mapping algorithm development (SLAM). Micro UAV are too small to be remotely controlled and, due to their physical constraints, they are unable to support the traditional stability-and-control or navigation aids. Thus, the flight autonomy of such vehicles is an issue of great importance. Basically, the key challenges, in the Micro and Mini UAV development are: aerodynamics at low Reynolds numbers; miniaturization of the airframe, components and payload; autonomy; collaborative control; robust flight Guidance, Navigation and Control (GNC), using technologies such as optic flow vision, integration of air traffic management, control, flight dynamics, image processing, and remote sensing; development of light-weight and low-noise propulsion systems capable of providing longer flight times and robust communications.

Sensors and algorithms creation and development for autonomous collision avoidance and localization without the use GPS for Micro and Mini UAV is also great challenge. It is a fact that the GPS, though small and light, does not work indoors and in enclosed near-Earth environments. Since flying insects utilize optic-flow to manoeuver through regions with dense obstacles fields, they have been topic of research. Flying insects do not have GPS or

IMU to perform tasks like collision avoidance, altitude control, take-off and landing. Thus, in the last few years, new technologies based on flying insects have been developed, modelling the Micro UAV flight patterns in such environments. Flying inside buildings, a very complex environment, is a very difficult task for an unmanned vehicle, but not for insects, which naturally avoid undesired collisions. Most UAV, especially Micro and Mini UAV, are not equipped with on-boar sensors to detect imminent collisions. In fact, the vision-based navigation for an autonomous UAV is a great challenge at this moment.

Sensing technologies, as laser range finders or radar, are available, but just for medium and large UAV. These technologies are unrealistic for Micro and Mini UAV due to their heavy weight and excessive power requirements. It is a fact that radar is an excellent candidate to detect obstacles; however, the antenna weight and, so, the weight of the associated power supply are prohibitive for small UAV. However, since most UAV carry an onboard camera, it makes all the sense to use, as an alternative, small and lightweight cameras. Now, the big problem and also the big challenge is to create a non-complex, efficient, robust and real time computer vision algorithm capable of seeing and avoiding obstacles practical to be used in small UAV. Moreover, this algorithm must be capable of detecting and avoiding collisions; real time UAV guidance (plan the new trajectory and guide the UAV around the obstacle) and manoeuvring the UAV (respond to guidance algorithm and manoeuver the UAV to a smooth and stable transition).

In many applications, the active cooperation of several different vehicles such as the UAV, the Unmanned Ground Vehicles (UGV), and airship vehicles, has important advantages. Indeed, over the last few years, research on the coordination and cooperation of multiple vehicles has been conducted (e.g. DARPA Grand Challenge). In (Brown, et al. 2004), multiple UAV have been used for cooperative vehicle tracking and mobile ad-hoc networking. In 2006, a research supported by the Office of Naval Research and Berkeley University, related with collaborative tasks in the air, was presented by Ryan, et al. (2006). They have produced UAV collaboration through in-the-air task allocation and conflict resolution. A complete system for cooperative UAV applications, with integrated capabilities including in-the-air task, was developed. Each UAV had an onboard software processes that provide low-level control, task selection and negotiation, vision-based control, and aircraft-to-aircraft communication. This combination allowed for a high degree of autonomy at both the individual and group level, requiring a minimum of human intervention.

SLAM algorithms are a landmark-based terrain aided navigation system with the capability for online map building, while simultaneously using the generated map to bind the

errors in the inertial navigation system. SLAM techniques have been widely used for ground robot navigation but only a few SLAM techniques are based on vision sensors. Usually these techniques are associated with GPS/INS systems, as it has been presented by Jonghyuk and Sukkarieh (2004). There, a new concept of autonomous UAV navigation, based on SLAM algorithm and applying a 6DoF (6 Degrees of Freedom) airborne platform, was presented. As a result, it was proved that the SLAM augmented the low cost GPS/INS systems when applied to various GPS denied situations, such as urban canyons, indoors, or even underwater. Another example is the perception system designed for the Karma airship (Lacroix, et al. 2002) which applies stereo vision, interest point matching and Kalman filtering techniques for simultaneous localization and mapping using only vision.

1.2 Motivation

Although, still incipient, several and distinct UAV applications are emerging demanding steps to be taken in order to allow those platforms to operate in an un-segregated airspace. The key risk component, intimidating the widespread integration of UAV in an un-segregated airspace, is the autonomous component: the need for a high level of autonomy in the UAS platforms that guarantees a safe and secure integration in an un-segregated airspace. Therefore, the UAV accurate state estimation (UAV position, velocity, and orientation) plays a fundamental role for autonomous UAV.

The ANTEX-M is the PAFA UAV that aims to develop UAV as systems and technology demonstrators in the areas concerning UAV development, defence, aeronautics and civil protection. One of the platforms developed for the project ANTEX-M is the ANTEX-M X02, which contains an onboard commercial autopilot responsible, among other tasks, for the UAV state estimation.

However, given the 21st century global economic paradigm, it can be stated that commercial autopilots start to become a non-economic solution for academic research in the field of UAV. In fact, the problem is of balance. Usually, the academic projects are based on inexpensive platforms but on expensive commercial autopilots. Most of the commercial autopilots available on the market are not open source so users are forced to accept the navigation solution performance provided by the navigation system inside the autopilot. There is a pressing need to overcome this problem one solution may be a navigation system with a compact, light, cheap, and precise navigation system, which is still a challenge.

Nowadays, UAV technology – specially the high availability of low cost, low power

consumption, and small size navigation sensors – has matured to the extent that it is widely used, not only by the military, civil and commercial market but also by the Academics world. With such a wide range of applications, the UAV payload is a restriction that must be taken into account, and so the associated advantages are of crucial importance.

At the same time, since the performance that is required for several applications in order to allow UAV to fly in an un-segregated airspace is not yet defined, it is believed that for most of UAV applications, the use of high-end INS does not make sense. In fact, high-end INS is generally confined to high accuracy navigation and geo-referencing applications. Therefore, for most UAV academic application the accuracy required will be at least the same as the one provided by the available commercial autopilots. In such a way, the challenge that arises is if it is feasible to develop a low-cost SIMU/ GNSS_Software-Receiver ($GNSS_{SR}$) integration algorithm for UAV state estimation – using low performance inertial sensors and a GPS/Galileo software receivers – which can in the near future, replace the commercial autopilot solution existent onboard the PAFA ANTEX-M UAV. Actually, it is believed that efficient and accurate state estimation algorithms using low performance sensors is crucial to make reliable, inexpensive unmanned vehicles feasible.

Equally important and motivating, is the fact that the navigation algorithm performance to be developed will have the opportunity to be evaluated and validated using real signals at the GATE[2] (Galileo Test and Development Environment) test area in Berchtesgaden, Germany.

1.3 Statement of the Problem

In the academic world, it starts to be quite usual for the institutes to develop their own inexpensive GNSS (GPS/Galileo/GLONASS) software receivers, able to be used in novel applications. The main idea, underlying the development of these software receivers, is to minimize the hardware by using software to do the majority of the signal processing. Thus, in the field of UAV, and due to both the payload and economic restrictions, it is believed that the choice will be for the development of this type of navigation systems. In fact, the available technology and the growing usage of software receivers offer the opportunity to

[2] GATE is an operational Galileo test range providing signals compliant with the Galileo SIS ICD, allowing for testing our navigation application under real world conditions.

replace the existing INS/GNSS navigation systems in emergent areas, such as the UAV applications. The development of navigation systems with a compact, light, cheap and precise navigation solution (depending on the application), using low cost IMU and software receivers to guarantee different levels of autonomy at lower prices is the big challenge.

However, the advantages of low performance sensors come at a cost. In fact, as size decreases, sensitivity decreases and noise increases decreasing overall sensor performance. According (Barbour, 2010), the current state of the art of the MEMS IMU has only reached tactical grade quality; even so, they are widely being used in UAV application. So first, it is necessary to select the correct IMU to use in a specific integrated navigation system. This task is not as easy as it may seem. Indeed, there is an enormous diversity of technologies applied to inertial navigation sensors making them different from one another; the large quality variety associated to the inertial sensor market leads to the need and importance of some type of inertial sensor classification. Although this is true, it should be stated that up to now there is no universally agreed definition of high, medium, and low grade INS/IMU. In fact, frequently, the lack of information on the reliability of the INS/IMU (MEMS-based) leads to its incorrect application. Equally important is the fact that, when faced with such a variety of inertial sensors, it is also quite hard to know which technical data one should analyse and compare before making a decision. Sometimes, it can be very difficult, especially for inexpert users of such a technology, to make the right decision about which one will meet the application requirements the best. Important to realize is also the fact that different sensor technologies offer a different range of advantages and disadvantages while offering a similar performance. To some extent, it is clear that all types of inertial sensors exhibit errors, such as bias, scale factor, and noise. The question that arises is “Do we really understand what those terms are?” In fact, to find a topic in the current literature that discusses all these points is not that easy. Thus, this thesis seeks to provide the theoretical background required to understand the concepts behind the inertial sensors.

In addition, low performance inertial sensor raw data present a high level of noise. The accelerometer and gyroscope raw signals are buried within a large window of high frequency measurement noise. Therefore, it is crucial to know more about the inertial sensor signals in both the frequency and the time domain, in order to make it possible to distinguish among the different types of errors to allow for their easy separation from the true signal. If the high frequency noise component is removed from the inertial sensor signal, it is expected that the performance of the signal measurements can be improved and, consequently, make it possible to construct an error model (mathematical model of the IMU measurements),

suitable to the sensor, improving the navigation solution.

Finally, the navigation solution accuracy also depends on the performance of the IMU/$GNSS_{SR}$ integration filter. The best architecture to integrate the SIMU data with the $GNSS_{SR}$ must be achieved in order to avoid that the combined navigation solution diverges, allowing for the mission accomplishment. If this is not the case, than the UAV can only operate under line of sight flight conditions.

1.4 Constraints

Due to the UAV payload constraints, the nature of the GATE hardware necessary to fly the UAV in the GATE area, and due to airworthiness restrictions, a van test has been performed in lieu of a flight tests. Therefore, the research was hindered, in part, by the lack of consolidated UAV flight tests.

1.5 Research Methodology

This thesis focuses on investigating the performance of an integrated navigation system composed by a low performance MEMS-based IMU and a $GNSS_{SR}$. Most compelling evidence and different from the common, in the field of integrated navigation, is the fact that no GNSS hardware receivers have been used. Instead, software receivers have been used in this thesis: a GPS and a Galileo Software receiver.

Besides this, the IMU sensor that has been used in this thesis is the low performance Systron Donner MMQ50. Therefore, in order to get an INS solution (position, velocity and orientation), a SIMU mechanization algorithm has been developed and implemented.

Additionally, an EKF has been implemented, to merge the SIMU data with the aiding information provided by the $GNSS_{SR}$, in loose-coupled closed-loop architecture. The IMU sensor (accelerometers and gyroscopes[3]) errors are modelled by biases, estimated in the EKF, according to the loose configuration, closed loop strategy and feedback to the IMU output to improve the raw data before mechanization, limiting error growth.

Moreover, and considering that it is possible to improve the performance of the IMU raw data prior to the estimation phase in the EKF, Allan variance and denoise techniques are considered for both studying the IMU error model and improving the inertial sensors raw

[3] From now on referred to as “gyros”.

measurements.

In the final analysis, this thesis aims to have a navigation solution with an accuracy, at least, equal to the one provided by the commercial Piccolo Autopilot, aiming to, in the near future, replace it onboard the UAV.

Research Objectives. The present thesis has the main objective of investigating the performance of an integrated navigation algorithm, developed and implemented with the purpose of fusing the data provided by a low performance IMU and a $GNSS_{SR}$, for UAV state determination.

Important to realize is that the ultimate goal of this research is to examine if the investigated solution reveals to be an appropriate solution, able to take up the role of the commercial autopilot onboard the PAFA UAV. The thesis will provide a deliverable to the PITVANT Project Group, consisting of a well-documented UAV navigation algorithm, enabling its full implementation in the existent UAS platforms.

To accomplish this main objective, we will attempt, with this work, to achieve the following specific objectives:

- To present a clear overview of the basic inertial navigation concepts, hoping to clarify what sometimes is a misunderstanding in the inertial navigation world;
- To present the reader with the theoretical background related to the coordinate systems, in order to ease the understanding the basic principles behind coordinate transformations;
- To develop and implement all the strapdown mechanization equations, in several coordinate frames, in order to get the SIMU solution;
- To develop an SIMU error model appropriate for the low performance sensor world;
- To present a clear picture of inertial sensor signals and errors, in order to clearly understand the output of those sensors;
- To present the current state of the art regarding the appropriate methods/techniques for analysing the field of low performance inertial navigation sensor signals;
- To provide the reader with a methodology for developing general inertial measurements models, error and compensation forms, for the low performance inertial sensor category;

- To investigate the general performance characteristics of the MMQ50, using the Allan variance and wavelet multi-resolution analysis techniques;
- To develop and present a simple MMQ50 measurement model, in the error and compensation forms;
- To prove that it is possible to improve the low performance inertial raw data before being fed to the strapdown mechanization equations;
- To present procedure guidelines, to develop all of the hardware and software required for this type of projects;
- To investigate the performance of an SIMU/ $GNSS_{SR}$ integrated navigation algorithm, in a loosely coupled closed-loop architecture;
- To investigate the performance when the GATE Galileo measurements are included;
- To confront the performance of the integrated SIMU/ $GNSS_{SR}$ solution obtained in the van-test scenario with the solution provided by the commercial autopilot.

Research Main and Secondary Questions.

The main question guiding this thesis is:

How should an integrated SIMU/$GNSS_{SR}$ *algorithm, working with low performance inertial navigation sensors (specifically, the Systron Donner MMQ50) and aiming to replace a commercial autopilot navigation solution, be developed and implemented?*

In order to answer the question, it has been broken into eight secondary questions, being each of these associated with the chapter in parenthesis.

Q1: What is the purpose of developing a SIMU and is it possible to provide a generic SIMU algorithm, suitable for different categories of IMU sensors, as a complete and comprehensive "recipe" able to be implemented in any computer? (Chapters 5 and 6)

Q2: Does the available literature provide easy, comprehensive and understandable information about inertial navigation sensor signals and errors? What are the most important properties of the inertial navigation sensor signals? (Chapter 8)

Q3: What is the current state of the art in regard to appropriate methods/techniques, in the field of low performance inertial navigation sensors signals in search for a higher accuracy navigation solution? (Chapter 8)

Q4: How to deal with low performance inertial navigation sensors in the sensor modelling process? (Chapter 9)

Q5: Is it possible to improve the MMQ50 sensor raw data before feeding it to the SIMU mechanization equations? (Chapter 10)

Q6: What are the more common estimation methods used for integrating inertial and GNSS data? How does one optimally estimate the interest parameters from noise corrupted data? (Chapters 12, 13, and 14)

Q7: What are the most prominent considerations for a platform design to be used in a van-test or a flight test environment? (Chapters 16 and 17)

Q8: Can we guarantee a navigation solution with at least the same accuracy as the one provided by a commercial autopilot navigation solution? Does the Galileo data improve the navigation solution? (Chapter 18)

Hypothesis. From the previous questions, the following hypotheses were formulated:

H1: *A complete strapdown IMU navigator algorithm can be validated using a trajectory generator, which gives perfect accelerometer and gyro measurements, proving that it is capable of being a "recipe" able to be used with any type of inertial navigation sensors and providing an accurate SIMU navigation solution.*

H2: *Low performance inertial navigation sensor signals can be improved, before being fed into the SIMU algorithm, using denoised techniques, useful to eliminate undesired high frequency components of the inertial signals.*

H3: *Allan variance analysis is a good method to identify main error sources in the inertial sensor measurements, helping in the determination of relevant numerical*

values for modelling the sensors.

H4: *A generic and satisfactory IMU sensor error model for the MMQ50 IMU can be obtained using specific procedures, applicable to different sensor manufacturers of the same category.*

H5: *The navigation solution provided by the developed and implemented SIMU/GNSS$_{SR}$ when confronted with the one provided by the commercial Autopilot produces better results.*

1.6 Bibliographical Study

The more pertinent bibliographic sources cited along this thesis are presented below.

To be able to present an inertial navigation synopsis, the work of (Barbour, 2009), (Britting, 1971), (Grewal, et al., 2001) and (Schmidt, 2009) have been consulted.

The SIMU navigator development and implementation is adapted from Savage (Savage, 2007). It should be stressed that the ECEF coordinate systems is the same as the one proposed by Savage. As a consequence, the several navigation equations will be different from the ones usually used in the field of inertial navigation.

The developed and implemented SIMU navigation error dynamic equations are also adapted from (Savage, 2007).

Low performance inertial navigation sensor analysis is done taking into account the work developed in (IEEE STD 647, 2006), (IEEE Std 528, 2001), (Nassar, 2005), (Zhong, et al., 2000), (El-Sheimy, et al., 2004), (Liu, et al., 2007), (Bruton, et al., 1999), and (Demoz, 2004).

Key concepts and guidelines related to Allan Variance are based on (IEEE STD 647, 2006), (IEEE Std. 952, 1997), (Allan, 1966), and (Lawrence C. Ng, 1993).

Finally, the topic related to the integrated navigation system architectures is a result of the work produced by (Savage, 2007), (Groves, 2008), and (Titterton, et al., 2004).

1.7 Contributions of this Thesis

It is the goal of this work to reach a straightforward algorithm, able to be implemented in any kind of UAV application, using low performance IMU sensors and a $GNSS_{SR}$. Thus, the major contributions of this thesis are:

- **The presentation** of a clear overview about inertial navigation basic concepts, hoping to clarify what sometimes is a misunderstanding in the inertial navigation world. As a matter of fact, differences among some basic concepts related to the inertial navigation are presented. For instance, it is quite usual to associate names such as IMU, or INS to just a triad of inertial sensors. And so, the differences between an ISA, IMU, and INS are shortly presented in this thesis;
- **The presentation** of the several coordinate frames and transformation among them taking into account a non-common E-Frame. In this thesis, the Earth-centred, Earth-fixed (ECEF) frame axis definitions differs from the familiar one, which implies different matrix transformations between the coordinate frames, according to Savage orientation (Savage, 2007);
- **The development and presentation** of the Strapdown navigation equations in several coordinates system;
- **The development of a single speed SIMU algorithm** – as a simplification of the two-speed algorithm proposed by reference (Savage, 2007) – SIMU algorithm using the Direction Cosine Matrix (DCM). The algorithm is proposed as a complete "recipe" in order to be easy to implement even from the point of view of a non-system analyst, and also to be applied with any type of IMU sensor in any desired coordinate frame;
- **The presentation** of a complete and comprehensive description about inertial sensors signals and errors.
- **The presentation** of the theory and methodology associated to the improvement of low performance IMU sensors measurements;
- **The development and presentation** of a general and simple inertial sensors measurements models in the error an compensation form for low performance IMU sensors, in particular for the MMQ50;

- **The development and presentation** of a methodology to be applied to low performance inertial navigation sensors, of the same category, in order to improve their measurements before being used in the SIMU mechanization equations;
- **The development** of the adopted MMQ50 error model;
- **The development** of data logger software in C++ to communicate with the several sensors on-board the vehicle as well as with the commercial autopilot. This data logger is able to be used in real time, and able to store and pre-process the sampled data.

1.8 Thesis Outline

In order to provide the reader a comprehensive overview about how this research has been done, this thesis has been organized into six parts and nineteen chapters, as follows:

Part I − Introduction and Background Concepts − which gives an introduction to the subject of this thesis includes four chapters which are organized as follows.

Chapter 1 introduces the research subject, giving an overview of the study as well. This chapter includes a background and literature review; provides the motivation and statement of the problem, as well as, the constraints of the thesis. This chapter also addresses the methodology adopted, including the research objectives, the research main and secondary questions, and the hypotheses that sustain this dissertation.

Chapter 2 provides a general overview of some basic concepts related to the Inertial Navigation Systems. For instance, it contains information such as the differences among an inertial sensor assembly (ISA), inertial measurement unit (IMU) and INS; fundamental INS concept; inertial navigation platforms types; IMU strapdown navigator process and inertial sensors trends. Finally, the IMU used in this thesis is presented.

Chapter 3 goes into details regarding the coordinate frames and the Earth model used along this thesis. At first, this chapter provides the definitions and some specifications of the coordinate frames. Next, it discusses the Earth model, as well as, some other concepts such as the transport rate and radii of curvature. The chapter finalizes with

the presentation of the transformations among the different coordinate frames used in the thesis.

Chapter 4 presents a summary of Part I.

Part II − Strapdown Navigator – It discusses the general INS kinematic equations, INS mechanization equations and INS digital form. This discussion is developed throughout the three chapters that are included in this Part.

Chapter 5 presents the continuous form of the navigation equations in the different coordinate frames. After this, the equivalent digital integration algorithm form based on the differential equations considered in the previous topic is presented. Finally, a validation of the developed and implemented strapdown algorithm is presented. It should be stressed that this algorithm is validated using a trajectory generator that has been developed to give perfect accelerometer and gyro measurements.

Chapter 6 moves on the INS error equation development.

Chapter 7 is a summary of Part II. The answer to the first secondary research question is given.

Part III − Low-Performance Sensor Analysis and MMQ50 IMU Modelling – this part consists of four chapters which are organized as follows:

Chapter 8 introduces some background information considered relevant for the subsequent two chapters. An overview of the IMU sensors signal, error sources, inertial sensor stochastic error identification and modelling techniques is introduced. Wavelet multi-resolution analysis and signal de-noising techniques to improve the performance of inertial sensors are introduced and discussed. Overall, this chapter presents the nature of those errors and the techniques for characterizing them are discussed. Chapter 8 finalizes with the presentation of the accelerometer and rate gyros measurements in the compensation form.

Chapter 9 The generic accelerometer and rate gyro measurements models in the error

form are presented, together with the derived adopted models.

Chapter 10 The different techniques discussed at Chapter 8 are applied to the low-performance MMQ50 IMU in order to identify the source of errors present in both accelerometers and rate gyros. The results of the experimental application of the Allan variance method, as well as, the wavelet multi-resolution analysis of the MMQ50 are, here, presented and discussed. Based in those experimental results and taking into account the adopted model presented at Chapter 9, the MMQ50 IMU sensors measurements in the compensation form is, finally, achieved.

Chapter 11 Part III summary. Answers the second, third, fourth and fifth secondary research questions.

Part IV – Filtering – completely dedicated to the IMU/GPS/Galileo integration algorithm development. This part includes four chapters.

Chapter 12 presents the estimation theory used throughout the thesis. In particular the Kalman Filter (KF) and Extended Kalman Filter (EKF) are discussed.

Chapter 13 provides a general overview about the main aspects according to INS/GNSS integration strategies. The loosely coupled close loop SIMU/$GNSS_{SR}$ strategy is here presented.

Chapter 14 goes into details related to the SIMU/$GNSS_{SR}$ specifications.

Chapter 15 summarizes Part IV. Answers the sixth research question.

Part V − Platform Design and Van-Tests − details the UAV instrumentation which includes both the hardware and software implementation. It also describes some of the tests carried out. This part is divided into three chapters described as follows:

Chapter 16 goes into details on the Hardware architecture, describing the overall setup to perform the tests.

Chapter 17 moves on the software design architecture, which is depicted with more details related to the developed onboard sensor fusion GUI. Particular attention is also paid to the data flow of the software, in order to provide the reader with an idea of how the software interacts with the sensors and the other systems.

Chapter 18 describes the van-tests that have been performed and presents the results and corresponding analysis after applying the aiding SIMU integration algorithm developed at Part IV. Summarizes Part V. Answers the seventh and eighth secondary research questions.

Part VI Thesis Summary: Chapter 19 – concludes the entire work and presents some recommendations to take into account for future works concerning this thesis.

In order to better understand the connection between the chapters, the organization of the thesis is illustrated in Fig. 1.1 with the chapter level dependency graph – block diagram roadmap − which shows how the subject of each chapter depends upon the material in other chapters. The arrows indicate the recommended order for reading this thesis. Boxes above another box and connected by arrows indicate that the material represented by the upper boxes is background material for the subject presented in the lower box. For instance, if the reader is interested in chapter 9, it is recommended that he reads chapter 8 first.

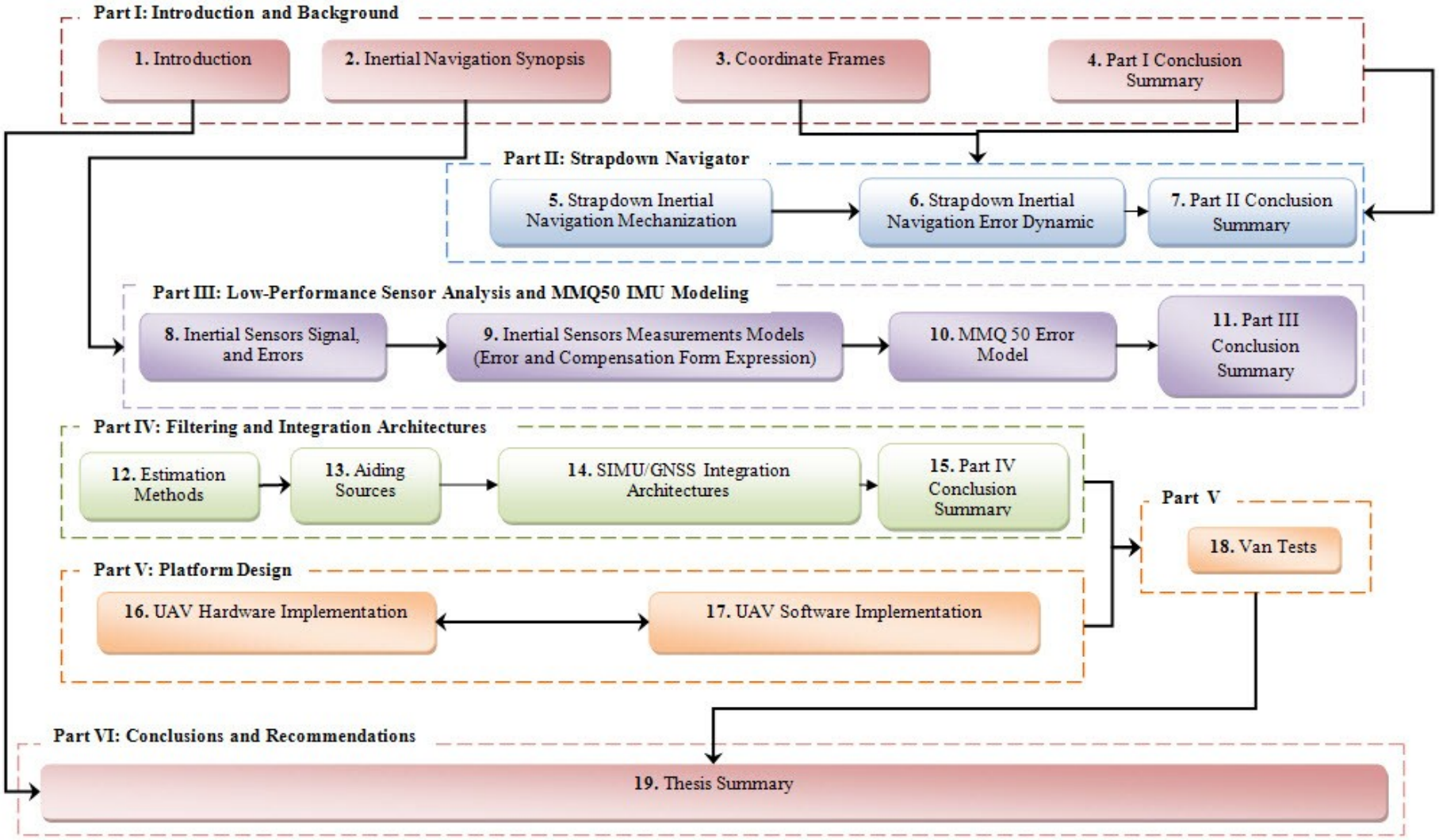

Figure 1.1 Thesis Block Diagram Roadmap

CHAPTER 2

INERTIAL NAVIGATION SYNOPSIS

2 Inertial Navigation Synopsis

Before the development of celestial navigation, sailors like Christopher Columbus navigated by "deduced" (or "dead") reckoning, hereafter called DR. Although Columbus had done some experiments with celestial navigation techniques in Portugal, he was a dead reckoning navigator. In DR, the navigator finds his position by measuring the course and distance he has sailed from some known point. So, starting from a known point such as a port, the navigator measures out his course and the distance from that point, on a chart, pricking the chart with a pin to mark the new position. To measure the course, the sailors used a magnetic compass and time, whereas the distance was determined by a time and speed calculation; the navigator multiplied the speed of the ship by the time travelled to get the distance.

Nowadays, inertial navigation sensors, as accelerometers, and rotation sensors, as gyroscopes, – components of an INS – are used to continuously calculate (via dead reckoning) the position, velocity, and orientation of a moving body without the need for external references. This is what the word "*navigate"* – derived from the Latin "*navigare*" – is all about. It means "to sail", so navigation is the act of navigating, which in turn is the art of determining the position, velocity, and orientation of a moving object in relation to a known initial position. Accelerometers and gyroscopes (henceforth abbreviated to gyros) are inertial sensors that are used for such a purpose.

Equally important is the fact that accelerometers and gyroscopes are sensors that measure proper acceleration (acceleration experienced relative to free-fall, the acceleration that is felt by people and objects) and angular rates relative to an inertial frame, thus being called inertial sensors. Keep in mind that all the reference frames in which the Newton's laws of motion are valid are called inertial frames. On the contrary, reference frames in which those laws are not valid are called non-inertial frames. Consequently, the sensors that measure the acceleration or angular rate with respect to features in the environment are not considered inertial sensors.

All in all, an INS is a dead-reckoning navigation system which uses accelerometers and gyroscopes to compute the position, velocity and attitude of a moving body navigating anywhere on the globe.

In order to make sure that the readers have a clear picture of the inertial navigation world, an overview of some basic concepts related to the inertial navigation systems is, here, presented. This chapter finalizes with the presentation of the Systron Donner IMU used throughout this thesis.

2.1 Inertial Navigation System Concept

Figure 2.1 illustrates the INS fundamental concept. Above all, the primary and basic idea of an INS is to compute the position location of a body from the accelerometer measurements. As it is illustrated, a set of three accelerometers is attached to the moving body in order to track its acceleration in three mutually orthogonal directions. The first thing to remember is the fact that the accelerometers are designed to measure the specific force component of the acceleration, thus they are only capable of measuring non-gravitational accelerations, and so the gravity acceleration must be added to the accelerometer measurements in order to get the total acceleration.

Moreover and according to Figure 2.1, it is possible to observe that the gravity component is calculated in the gravity computation block as a function of the INS computed position. In fact, another key point is that the calculation of this gravity is done using a model of the gravity vector field as a function of the position in space in which the INS is supposed to operate - topic discussed in Chapter 3.

Finally, the gravity vector is added to the specific force in order to get the three components of the total acceleration. Next, this total acceleration is integrated to give a continuous estimate of the corresponding velocity component of the moving body; a second integration is performed in order to get the body position in relation to a known starting point. It should be stressed that the initial conditions are known.

Although, generally speaking this is how it works, the INS operation is not as simple as it may seem. In fact, when navigating near the Earth surface, it does not make any sense to get the body position relative to an inertial frame. Indeed, the INS navigation parameters must be provided relative to a navigation frame. As a consequence, the inertial navigation system - specifically the inertial navigator - involves more than coming up with simple accelerometer measurement integrations.

At this stage, it is important to present the concept of INS platforms. Two classical INS mechanization approaches, directly related to the INS platform type, must be considered.

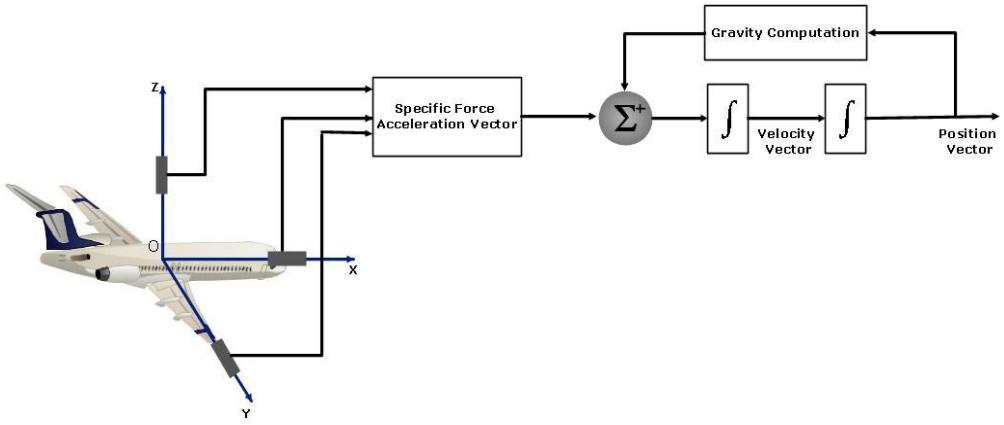

Figure 2.1Fundamental INS concept.

2.2 Inertial Navigation Platforms: Gimballed versus Strapdown Approach

According to the previous section, an INS provides position location related to an inertial frame based on acceleration measurements integrations. In order to have the same information related to a navigation frame, some more complex calculations are needed. Therefore, in order to determine navigation parameters such as position, velocity, and attitude in a specific navigation frame, the acceleration projections on this frame must be provided. It is now time to present the gyroscope's role.

The gyroscope's task is to maintain the accelerometers sensitive axes in a known orientation with respect to a specific coordinate frame in order to match them with the navigation frame axes. To achieve that, two different approaches can be considered: either a mechanical or a computational approach; the so-called navigation "gimballed" approach, and the "strapdown" approach. Both concepts are depicted in Figure 2.2 and discussed as follows.

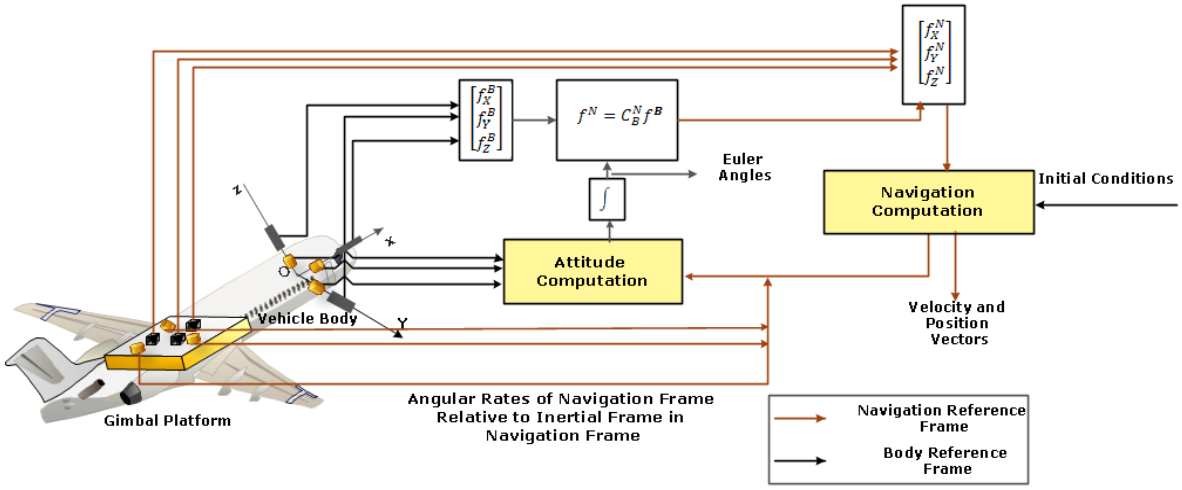

Figure 2.2 Navigation Gimballed versus Strapdown concept.

2.2.1 Navigation Gimballed Concept

The navigation gimballed approach deals with the physical implementation of the navigation frame using for that a three axis gyro-stabilized platform (gimbals' platform) with three orthogonally placed accelerometers. This platform is able to maintain a specific orientation relative to the navigation frame. In fact, the mounted gyroscopes are able to detect any platform rotations associated to the vehicle´s motion, and then these signals are fedback to the platform gimbal torques motors which rotates the gimbals in order to cancel out such rotations. As a consequence, the gyros will process at precisely the same rate as the navigation frame relative to the inertial frame in the navigation (Savage, 2007). As a result, the sensor axes are always aligned with the navigation frame; the measurements obtained directly from the accelerometers mounted in such platform are the specific force vectors in the navigation frame, see Figure 2.2.

2.2.2 Strapdown Concept

In the second approach - the professed strapdown INS (SINS) - the gimbal platform is eliminated by the inertial sensor platform, which is directly mounted within the INS chassis, i.e., "strapped down" to the INS and to the body (Grewal, et al., 2001). In this case, the system has no capability to mechanically determine the accelerometers orientation by itself; the strapdown INS is not isolated from the vehicle's movements as in the gimballed one; the accelerometers' orientation must be achieved analytically. The attitude computation block, shown in Figure 2.2, is responsible for such a computational task by processing the angular rate vector measured by the strapdown angular rate sensors. As a result, the gyro

measurements are mathematically integrated to provide changes in orientation of the nominally orthogonal accelerometer triad.

As it can be seen, both the strapdown and the navigation gimballed systems provide the same input - specific force acceleration vector - to the navigation computation block. Nonetheless, whereas in the gimballed system the specific force vector in navigation coordinates is measured directly by the accelerometers mounted on the gimballed platform, in the strapdown one the specific force vector is first measured by the strapdown accelerometers as a vector in the sensor coordinate frame, and then computationally rotated from the sensor frame into the desired navigation frame.

Besides that, in both configurations, initial design conditions must be provided to the system, specifically to the computation and navigation blocks.

2.3 ISA/IMU/INS

This section provides a description of the differences among an inertial sensor assembly (ISA), an inertial measurement unit (IMU), and the inertial navigation system (INS). To demonstrate it Figure 2.3 depicts the different concepts, whose brief description comes as follows:

- ISA - an ISA usually consists of three accelerometers and three gyros, which are generally rigid and assembled on a common sensor base; this sensors base is precisely mounted within a housing with the related electronics (sensors support electronics). The information provided by such units is usually: inertial raw data as well as time information.
- IMU - given the similarities between an IMU and an ISA, they are occasionally mistaken with one another. In fact, the IMU contains a cluster of sensors: usually three accelerometers and three gyros rigidly mounted on a common base in order to maintain the same relative orientations. This unit also outputs raw data, but unlike the ISA, this data is compensated; the inertial raw data is first compensated for the common sensor errors, such as temperature, bias, and scale factors. The IMU provides as outputs incremental angles and velocities in the ISA frame.
- INS - the inertial navigation system is an IMU whose outputs are fed to a navigation navigator capable of performing calculations such as the gravitational acceleration (not measured by accelerometers), integrate acceleration to get the velocity, double

integrate the acceleration to get the position, and calculate the vehicle´s orientation. It should be stated that, sometimes, this unit also provides compensated raw data which, for instance, can be used for stabilization purposes. Therefore, an INS outputs a set of navigation parameters, which are position, velocity, and attitude of the moving body in which it is mounted.

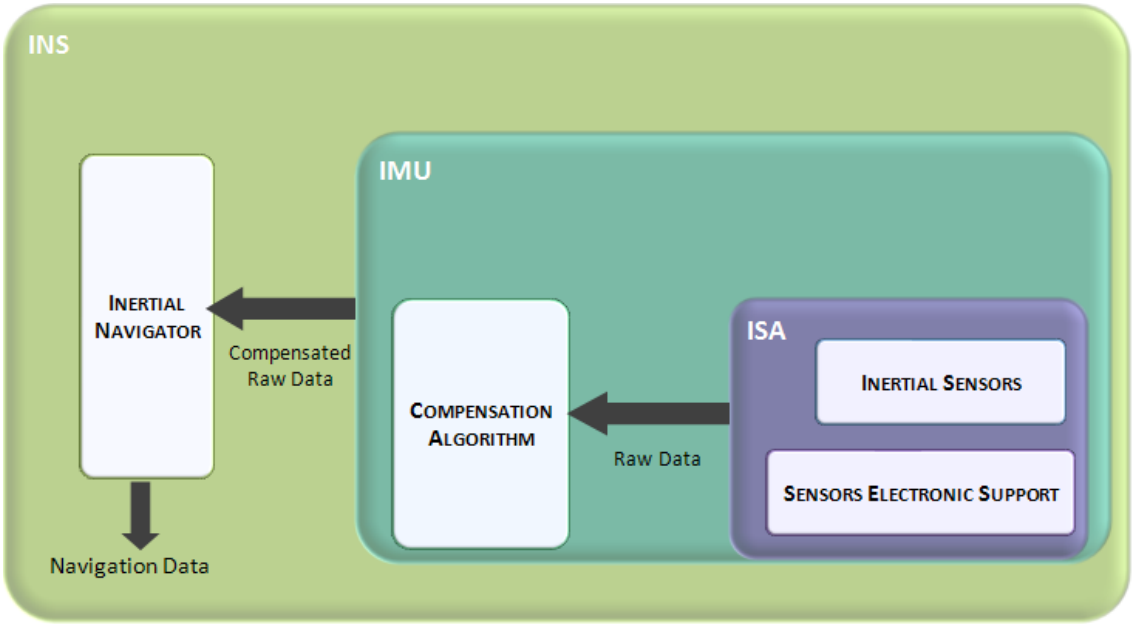

Figure 2.3 ISA, IMU and INS.

2.4 Strapdown Inertial Navigation System Mechanization

All things considered, the conception of the Strapdown Inertial Navigation System (SINS) appears to substitute the mechanical orientation by an analytical one, using some extra and complex onboard computation. True is the fact that, at the time of the patent of the idea, a lack of digital computers of reasonable size delayed the development of such a system till the early 1970's. However, as soon as they became available, a new page of modern navigation began. Nowadays, these systems can be built smaller in size, with low power consumption, and at reduced costs. As a matter of fact, those characteristics associated to the improved reliability of the new generations of computers are making them a first choice for UAV applications.

At the present time, the big challenge is to work with the current generation of low performance and low cost inertial sensors/systems based on MEMS technology, providing powerful robust navigation capabilities that can deal with the large errors experienced with these low grade sensors. The idea is to achieve a low cost navigation solution with the same

performance of that provided by more expensive and accurate inertial navigation systems in UAV applications.

The sensor used in this thesis is an IMU and so far, a Strapdown IMU (SIMU) algorithm (navigator) needs to be developed in order to get the vehicle state information from the IMU sensor measurements. In a strapdown configuration, the gyro data must be mathematically integrated to provide changes in orientation of the nominally orthogonal accelerometer triad.

According to Figure 2.4, the strapdown navigator process can be divided into two phases: the alignment phase, and the navigation phase. The first phase is related to the calculation of the initial conditions[4] for velocity, position, and orientation in order to make possible to start the integration process. With the awareness of the initial attitude, velocity and position, the trajectory of a moving body with respect to a reference frame can be determined. The second one is related to the navigation computer, which is responsible for performing five basic operations (Savage, 2007), displayed in Figure 2.4 and discussed as follows:

- **Attitude computation:** integration algorithm responsible for integrating the angular rate sensor input data providing attitude orientation;
- **Coordinate transformation:** sensor assembly attitude orientation computed previously is applied to the input accelerometer measurements; accelerometer data from the sensor axes (body axes, in this case) is transformed to accelerations in navigation coordinates;
- **Velocity computation:** at this stage, an integration algorithm is applied to the accelerations in the navigation frame in order to calculate the body´s velocity relative to earth - including the addition of the gravitational acceleration effects not measured by the accelerometers - in the navigation frame as well;
- **Position computation:** an integration algorithm is applied to the computed velocity in order to calculate the body´s position location;
- **Inertial navigation solution:** given the fact that body parameters can be represented in one of the several existent coordinate frames(see chapter 3 for more details on coordinate frames), a transformation computation is applied to the velocity, position,

[4] Important to realize is the fact that, independently of the platform approach being used in either design, the initial conditions must be provided to the navigator.

and attitude information in order to get the data in the forms required for the system output. Equally important is the fact that associated to the desired coordinate frame, different mechanizations for the strapdown equations can be considered. Nonetheless, it should be stated that the variations in the mechanizations are in the strapdown computational algorithms and not in the arrangement of the sensors or in the mechanical layout of the system. As a matter of fact, the choice of the mechanization is dependent on the application, and so, the five basic operations are always needed independently of which mechanizations used.

Transformation from the output of the IMU into the attitude, velocity and position information is usually described by both the inertial navigation equations and the mechanization equations in the navigation frame. The navigation equations describe the dynamics of body motion whereas the mechanization equations are used to derive the position, velocity and attitude increments by solving the equations of motion. Combined with the initial conditions of the system obtained from INS alignment and external sensors, these computed increments can then provide the attitude, velocity and position information needed for navigation.

Chapter 5, provides in-depth derivations for both the continuous (inertial navigation equations) and the digital (mechanization equations) computation algorithms used for the attitude/velocity/position integration and acceleration transformation.

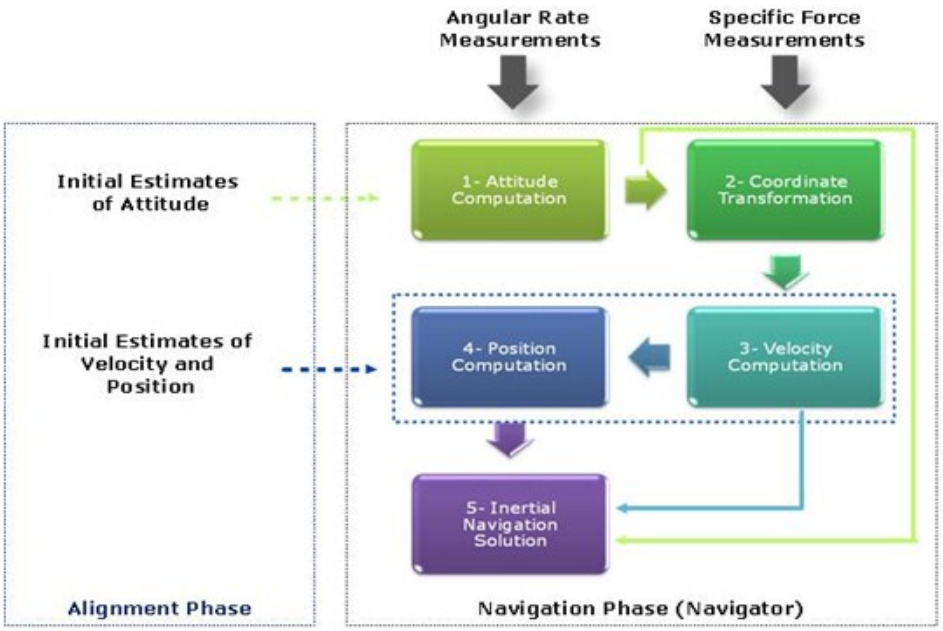

Figure 2.4 Strapdown inertial navigation system process.

2.5 Inertial Navigation Sensors Classification

The performance quality of an IMU is directly related to its sensors quality; the accuracy of the navigator solution is dependent of the sensors used; different applications call for different sensors qualities; there is a great variety of technologies applied to inertial navigation sensors making them different from one another; the wide quality spectrum associated to the inertial sensor market leads to the need and importance of some type of inertial sensor classification. Although this may be true, it should be stated, that up to now there is no universally agreed definition of high, medium, and, low grade IMU. Therefore, often, the lack of information on the reliability of the MEMS IMU leads to its incorrect application.

Equally important is the fact that, when faced with such a diversity of inertial sensors and systems, it is quite hard to know which technical data one should analyse and compare before making a decision. Sometimes it can be very difficult for inexpert users of such technology to make the right decision about which will meet the application requirements the best. Another key point is the fact that sometimes many different sensor technologies offer a different range of advantages and disadvantages while offering a similar performance.

Still, there are some questions that arise when trying to select the right sensor, such as: what makes an IMU to be considered low performance? Why are some low performance IMU often confused with medium accuracy ones? In fact, under those circumstances, it is important to be familiar with some accuracy concepts; it is crucial to be familiar with the performance parameters which make the INS/IMU/ISA different from one another.

Therefore, before the presentation of the IMU that has been used in this thesis, it is considered fundamental to give the reader an overview of the different types of IMU. Moreover, this classification will also help to situate the used IMU in the actual technology spectrum.

2.5.1 Inertial Sensors Adopted Classification

Although, inertial sensors can usually be classified according to their mode of operation; their method of detecting position of the mass, or according to their fabrication process as well, in this thesis the IMU are grouped according to their accuracy. Adapted from (Groves, 2008), ISA, INS and IMU, may be grouped into five different applications – consumer; tactical; intermediate; navigation, and marine – with increasing accuracy specifications corresponding to three levels of performance: low performance, medium

performance and high performance, as it is illustrated in Figure 2.5. For instance at the highest grade, we have INS widely used in marine applications, such as ships and submarines, and aircrafts as well. On the other extreme, we can find the lowest grade of inertial sensors (ISA and IMU) largely used in consumer/automotive applications. Important to realize is the fact that in the low performance category we do not yet find INS, but just inertial sensors (ISA) and/or IMU for strapdown applications. And the same happens in the high performance category where we do not find yet IMU.

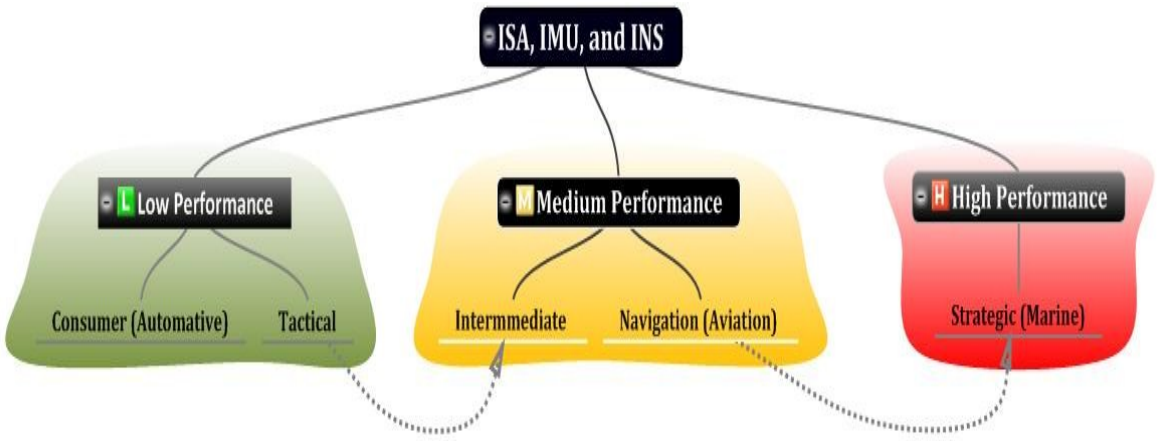

Figure 2.5 ISA, IMU and INS Categorization.

In this thesis, the parameters that have been taken into account in order to categorize the INS/IMU/ISA in the three different levels of performance presented are: bias, scale factor, and random walk (noise) (Schmidt, 2009). Therefore, Table 2.1 presents the reference values, range of sensing performances, based on such parameters, to be considered when classifying an INS/IMU or ISA.

Important to realize is that the smaller the inertial sensor errors are, the better the quality of the IMU and, consequently, the better the accuracy of the resulting navigation solution. For more details on inertial sensors errors - such as, bias, scale factor, noise - refer to Part III, chapter 8 where this issue is addressed in more detail.

Table 2.1 Performance grades for inertial navigation systems and IMU sensors. Source: Adapted from Grewal, et al. (2007) and Petovello (2003).

Category		**High Performance**	**Medium Performance**	**Low Performance**
Stand Alone Position Errors		< 30 (m/hr)	$1 - 4$ (km / hr)	20 (km/hr) (tactical) $-$ 2 (km/min) (Consumer)
Gyros	Bias (deg/h)	0.0001 – 0.005	0.01 – 0.1	0.1-10 (can be larger: 100-10 000 for consumer grade)
	Scale Factor (ppm)	-	5 - 50	200 -500 (N/A for consumer grade)
	Angular Random Walk (Noise) (deg/$hr\sqrt{Hz}$)	-	0.002 – 0.005	0.2 – 0.5 (N/A for consumer grade)
Accelerometers	Bias (mg)	0.001	0.05 – 0.1	2 – 4 (> 12 for consumer grade)
	Scale Factor (ppm)	-	10 – 20	400 – 1000 (N/A for consumer grade)
	Velocity Random Walk (Noise) (mg/h/$\sqrt{Hz}$)	-	0.05 – 0.1	2 -4 (N/A for consumer grade)

2.5.2 Accelerometers and Gyros Trends

Figures 2.6 and 2.7 (adapted from (Schmidt, 2009)) depict a perspective of the current gyroscope and accelerometer technologies. They provide a comprehensive view of the gyro and accelerometer bias (mg and deg/hr, respectively) and scale factor stability (ppm) requirements for various mission applications and what type of gyro is likely to be used in current applications.

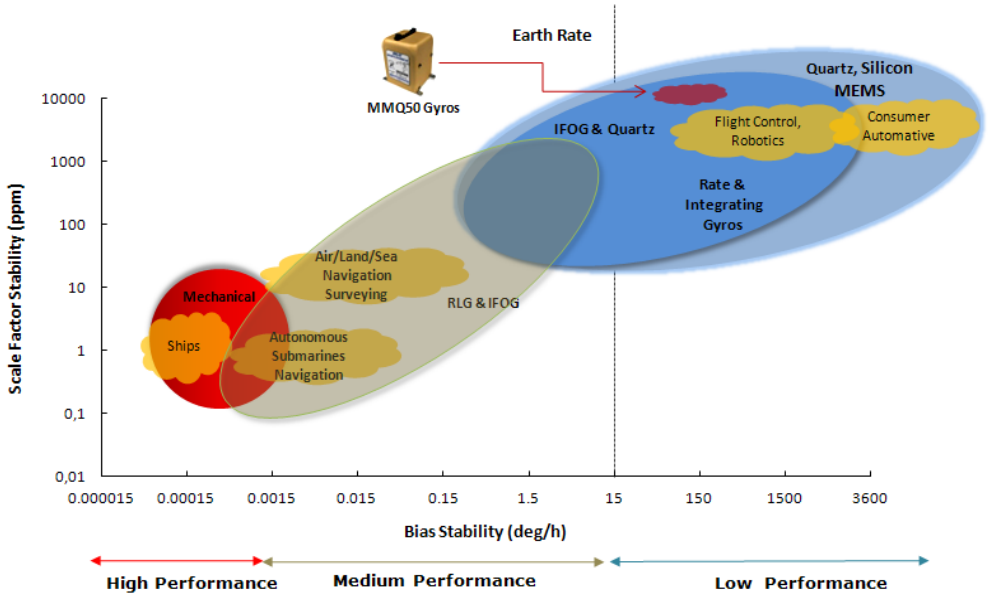

Figure 2.6 Current gyroscopes technology trends.

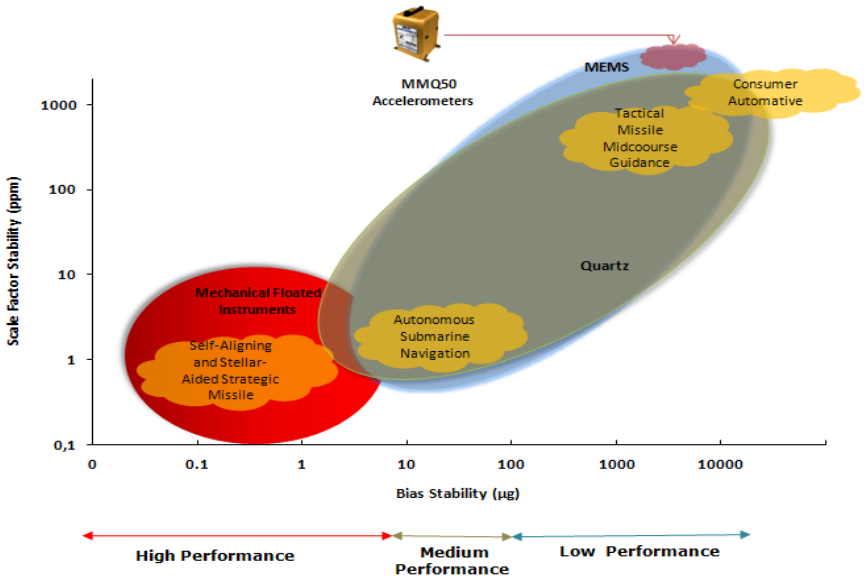

Figure 2.7 Current accelerometers technology Trends.

In the past, increasingly accurate rotating, fibber optic and ring laser gyros have been widely used; present trends is towards small, low-cost, micro-machined angular rate sensors of medium accuracy such as the Coriolis vibratory gyros. For more details about accelerometer and gyroscope technology the reader should consult (Titterton, et al., 2004).

Referring to Figure 2.6, and Figure 2.7, three major technologies have enabled advances in both the military and commercial worlds: ring laser gyros (RLG), fibber optic gyros (FOG), and micro-electro-mechanical systems (MEMS) gyroscopes and accelerometers. RLG and FOGs are now mature technologies, whereas MEMS is still a very active development area (Barbour, 2009). As a matter of fact, the significant cost, size, low power consumption, and weight advantages of MEMS based sensors have resulted in a tremendous increase of the number of applications where such devices can be used, particularly at the lower performance end of the spectrum presented in both corresponding figures − as it is the case of automotive and consumer applications.

MEMS based inertial sensors and IFOG technologies are expected to replace, in the near-term, many of the current systems using ring laser gyros (RLG) and mechanical instruments (Schmidt, 2009). As a matter of fact, the exhaustive demand for sensors belonging to the tactical lower performance end of the spectrum, i.e. for consumer/automotive applications, has led to an incessantly performance improvement of such MEMS based sensors. It is believed that MEMS instruments will come to dominate the entire inertial sensor application spectrum, in the far-term (Schmidt, 2009).

Although this may be true, the fact is that there are still some goals to meet before this can be achieved. Indeed, in both Figures, it is evidently visible how MEMS has struggled to reach the medium grade quality or the tactical grade quality (higher part of the low performance of the spectrum, 1 deg/h), and it is only now reaching that level of performance. In fact for many applications, the idea is to meet the desired performance at a reduced cost, low power consumption and size, and this can be achieved with some tactical grade sensors.

Keep in mind that, as size decreases, sensitivity (scale factor) decreases, and noise increases. At the present time, the performance of the MEMS IMU continues to be limited by gyro performance, which are now around 10 - 30 deg/h, rather than by accelerometer performance, which has demonstrated tens of micro g or better (Weinberg, et al., 2006). For this reason, until gyroscope performance is improved to the required goal, we will still have to work in integration algorithms when working with the tactical grade systems (IMU and ISA). This is the case of the tactical grade IMU used in this thesis that is described as follows.

It should be noted, that in order to locate the used IMU in the current technology, the

gyro and accelerometer performance is represented in both Figure 2.6, and Figure 2.7 respectively.

2.5.3 Tactical Grade IMU: MMQ50

The MEMS based IMU Systron Donner MMQ50, Figure 2.8, is a miniature MEMS quartz IMU (MMQ). This IMU sensor incorporates solid-state quartz micro machined inertial rate sensors and uses high performance, low cost silicon MEMS accelerometers, being ideal and attractive for embedded applications where extremely small size, low cost, and low power consumption are requirements.

The MMQ50 is an integrate unit which combines the inertial sensor assembly (silicon accelerometers and quartz rate gyros), the inertial sensor assembly electronics and a processor in which digital filtering and compensation algorithms (for scale factor, bias and temperature) are implemented. Technical data for this sensor are presented at Table 2.2.

Figure 2.8 Systron Donner MMQ50.

In addition, this IMU consumes less that 5 watts of power from plus and minus 12 volts supplies, and provides temperature compensated angular rate and acceleration data or change in attitude (delta Theta, $\Delta\theta$) and change in velocity (delta V, ΔV) at a 450 Hz output rate. Equally important is the fact, that this sensor, also, includes a compensation routine for other calibration parameters such as: bias, scale factor and alignment.

Table 2.2 MMQ50 performance summary.

Tactical Grade Systron Donner MMQ50			
Technology	MEMS Quartz Gyros and Silicon Accelerometers		
Parameter	Units	Gyros	Accelerometers
Bias In-run Stability 5	deg/h or mg	50-200 (1σ)	3 (1σ)
Bias Instability (Noise Floor) 6	deg/h or mg	5-15(1σ)	0.2, 1σ
Scale Factor In-run Stability	ppm	5000 (1σ)	5000 (1σ)
Angle Random Walk/Velocity Random Walk (White noise)	deg/√h or mg/√HZ	0.15	0.5

[5] Post-compensation over -40ºC, 25ºC, 40ºC, and 71°C.
[6] This is the measure of bias stability, or noise, and not of bias offset

CHAPTER 3

COORDINATE FRAMES

3 Coordinate Frames

The main topics under discussion in this chapter are the coordinate frames and the earth model. Section 3.1 provides the definitions and some specifications of the coordinate frames that are used within this thesis. Section 3.2 discusses the Earth model as well as some other concepts such as transport rate, radii of curvature, which have been selected for implementation in the strapdown algorithm. The exact analytical equations for the Earth surface shape, gravity, and other Earth related parameters based on reference (DoD, 4 July 1997) are developed. We will first describe the Earth's geometry and then the gravity model that has been selected. For more details about the topic the reader should read (Savage, 2007) and (Britting, 1971).

Important to realize is the fact that in this thesis the Earth-centred, Earth-fixed (ECEF) frame axis definitions differs from the familiar one which will imply a different matrix transformation between coordinate frames. As a matter of fact, the y axis is parallel to and aligned with the direction of the Earth's rotation instead of the z axis, the typical in the common literature. Therefore, special attention has to be paid to Section 3.1 so that the reader does not get confused with the different DCM obtained from and with the here proposed E-Frame definition. For this chapter and for more details about nomenclature and mathematical techniques used along this thesis the reader should consult Appendix A.

3.1 Definitions

Inertial sensors measure their motion with respect to an inertial frame; the GPS measures position and velocity of a receiver's antenna with respect to a constellation of satellites providing estimates of the GPS antenna position in the ECEF coordinate system; the INS provides us a navigation solution in a specific navigation frame.

In fact, navigating with different sensors can be considered a multiple coordinate

frame problem. Moreover, if the goal is to develop an integrated navigation algorithm – where measurements from several sensors need to be compared – it is crucial to have a convenient means to transfer the measurements among the coordinate systems, otherwise we will get a big chaos.

To define a coordinate frame, its origin and a set of axes, in terms of which the motion of objects may be described, (i.e., a reference), must be provided. In this section, the definition and properties of the several reference frames used along the thesis are discussed. In general, the navigation frames here defined can be divided into the following frames: Inertial; Earth-centred; wander azimuth, and local geodetic. Important to realize is the fact that usually certain local geodetic frames are associated with specific Earth-centred frames. In the same way, a variety of the body frames are also associated with specific local geodetic frames. Again, and with this in mind, it is considered crucial to take into account that the transformation matrices, i.e. the DCM, used to convert navigation variables from one frame to another will be based on such axis definitions, which can be somehow different from the usual, as it is the case of the Earth-Frame here defined.

Table 3.1 presents a summary of some of the properties and some considerations related to each coordinate system used along this thesis as illustrated at Figure 3.1.

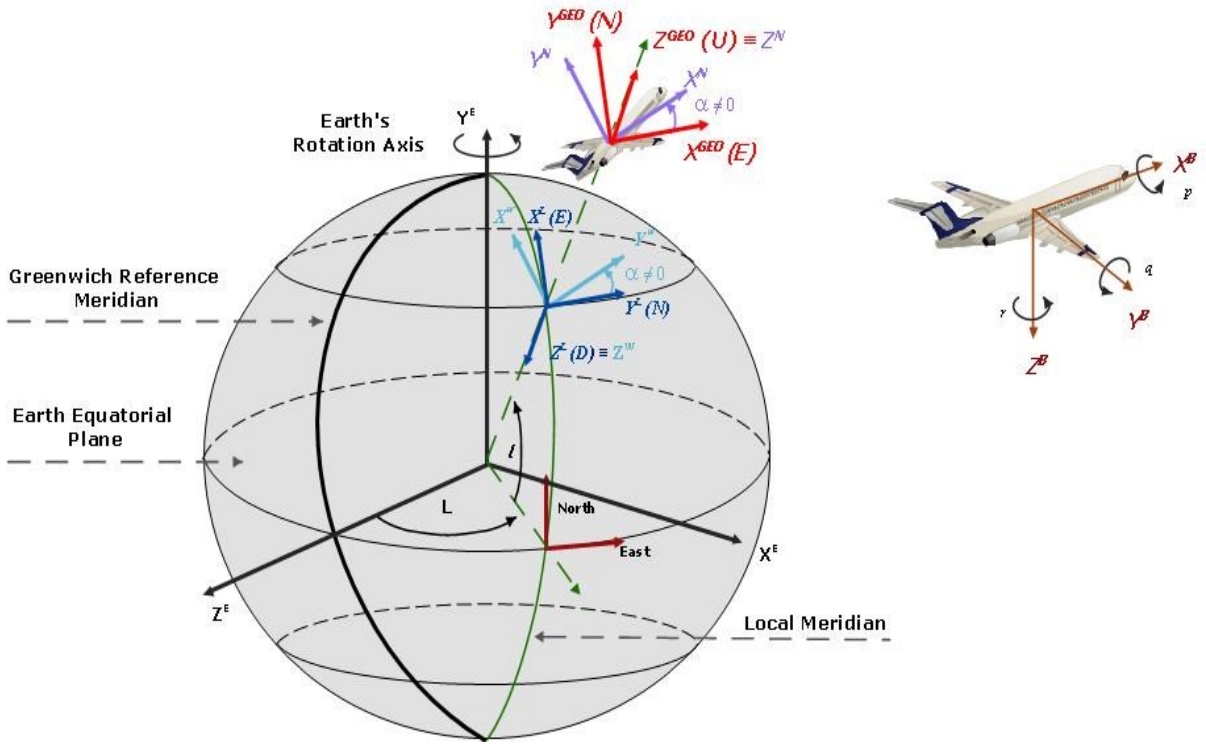

Figure 3.1 Coordinate Frames.

According to Figure 3.1, above, the navigation frames – N, L, and GEO – correspond to the E-frame rotated around the Y axis through the longitude angle (L), and, thereafter, rotated around the new X axis (negatively) through the latitude angle (*l*).

Usually, it is preferred to model body motion in the North pointing N-Frames: the L-Frame or GEO-Frame due to the fact that its axes are aligned with the local North, East and Down (NED) or East, North, Up (ENU) directions, respectively (Schwarz, et al., 1997). Therefore, the attitude angles relative to the local level frame – roll, pitch and heading – can be obtained directly as an output of the mechanization equations. And also, because the definition of the L/Geo Frame is based on the normal reference ellipsoid, we can directly obtain the geographic coordinates latitude (*l*), longitude (L) and height (h), and also as it will be verified at Section 3.2, the gravity model associated to those two reference frames is also simplified.

The B-Frame corresponds to the "Body" coordinate frame parallel to strapdown inertial sensors axis. It should be stressed, that in this thesis the IMU sensors sensitive axes are considered coincident with the body axes and so, the projections of specific force and angular rates on the body frame are available directly from the output of the accelerometers and gyros respectively.

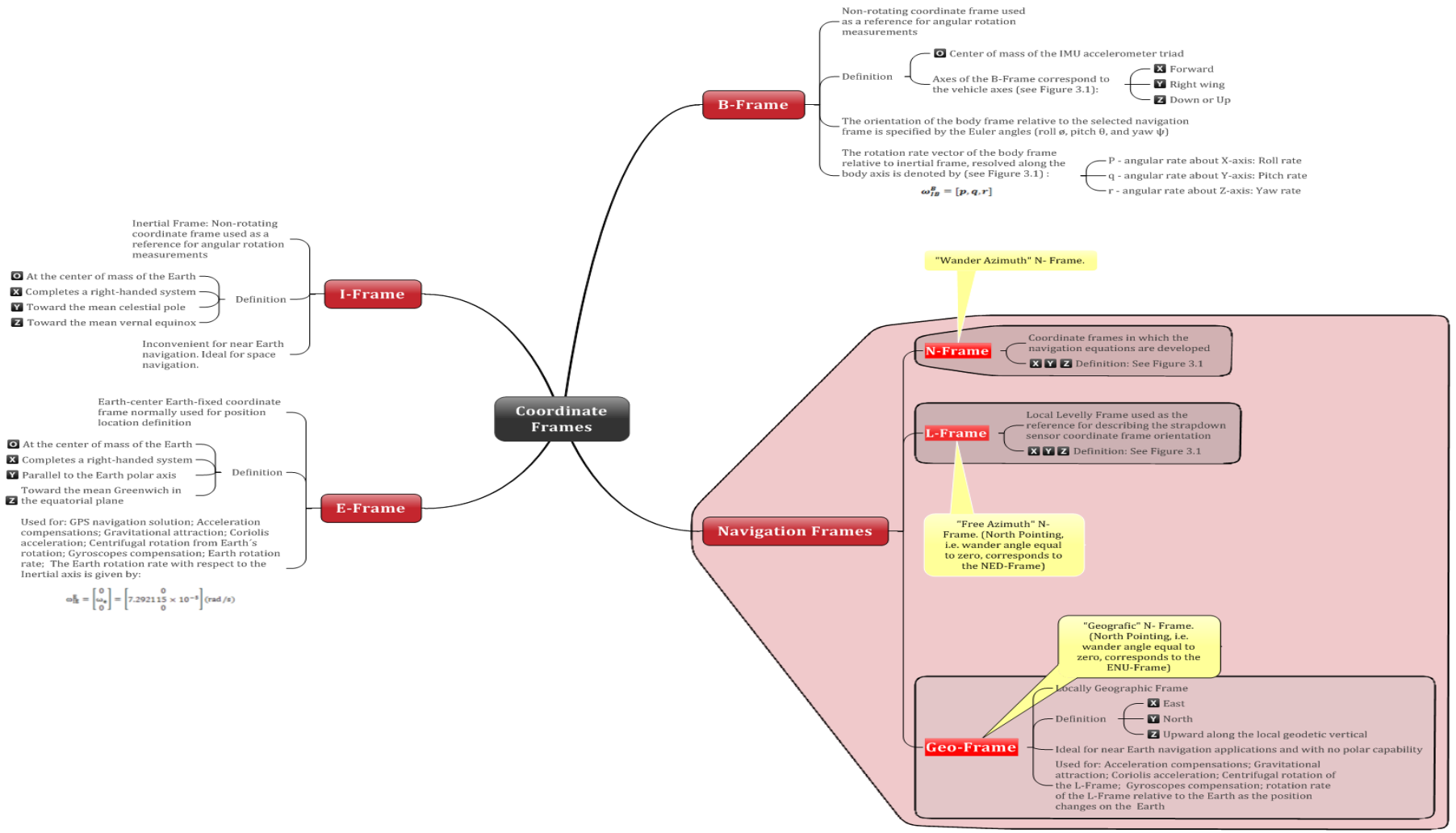

Figure 3.2 Coordinate frames definitions and characterization. Source: Adapted from (Savage, 2007).

3.2 Transformation between Reference Frames

Important to realize is the fact that any navigation problem thus involves at least two coordinate frames (Bekir, 2007): an object frame and a reference frame. The object frame describes the body whose position and/or orientation is desired, while the reference frames describe a known body, such as the Earth, relative to which the object position and /or orientation is desired.

In general, it is clear that, along this thesis, several coordinate frames can be used, for instance inertial measurements are in a specific coordinate frame, whereas the navigation solution is in another coordinate frame. And so, some transformations need to be done to transform the vector components from one frame to the other. All the transformations that will be made in this thesis are through the use of the direction cosine matrix (DCM). See Appendix A for more details on notation and DCM equations and properties.

The main objective of this section is to give the reader first, the corresponding transformation matrices between all those different frames; and second, the relative angular rate correspondent to the two frames. In addition, the position rate vector (time of changes of the coordinates in the selected frame) associated at each coordinate frames that relates the curvilinear coordinates with the respective frame velocity components is achieved. It should be stressed that a subsection of Appendix A discusses how to achieve the angular rates between the several coordinate frames and the I-Frame, and between the several coordinate frames and the E-Frame as well. As a matter of fact, all these equations here presented will sustain part of Chapter 5, dedicated to the strapdown navigator. Therefore, it was considered relevant to present here the final forms of the essential transformations needed to implement the strapdown navigator algorithm.

Tables 3-1 and 3-2, is a summary of the different transformations and angular rates considered crucial for the navigator development.

Table 3.1 Summary of DCM between coordinate frames.

Transformations Between Coordinate Frames	
From B to Computational Frame K	$\mathbf{C_B^K}$
K = L (NED)	$C_B^L = \begin{bmatrix} c\theta c\psi & -c\phi s\psi + s\phi s\theta c\psi & s\phi s\psi + c\phi s\theta c\psi \\ c\theta s\psi & c\phi c\psi + s\phi s\theta s\psi & -s\phi c\psi + c\phi s\theta s\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix}$
K = GEO (ENU)	$C_B^{GEO} = \begin{bmatrix} c\theta s\psi & c\phi c\psi + s\phi s\theta s\psi & -s\phi c\psi + c\phi s\theta s\psi \\ c\theta c\psi & -c\phi s\psi + s\phi s\theta c\psi & s\phi s\psi + c\phi s\theta c\psi \\ s\theta & -s\phi c\theta & -c\phi c\theta \end{bmatrix}$
K = N	Equation (A.20)
From E to Computational Frame K	$\mathbf{C_E^K}$
K = L (NED)	$C_E^L = \begin{bmatrix} -sLsl & cl & cLsl \\ cL & 0 & -sL \\ -sLcl & -sl & -cLcl \end{bmatrix}$
K = GEO (ENU)	$C_E^{GEO} = \begin{bmatrix} cL & 0 & -sL \\ -slsL & cl & -slcL \\ clsL & sl & clcL \end{bmatrix}$
K = N	$C_E^N = \begin{bmatrix} cLc\alpha - sLslsc\alpha & clsc\alpha & -sLc\alpha - cLslsc\alpha \\ -cLsc\alpha - sLslcc\alpha & clcc\alpha & sLsc\alpha - cLslcc\alpha \\ sLcl & sl & cLcl \end{bmatrix}$

Table 3.2 Summary of angular rates between coordinate frames.

Angular Rates	
From I to Computational Frame K	$\underline{\boldsymbol{\omega}}_{\mathbf{IK}}^{\mathbf{K}}$
K = E	$\underline{\omega}_{IE}^{E} = [0 \quad \omega_{IE} \quad 0]^{T}$
K = L (NED)	$\underline{\omega}_{iL}^{L} = [cl(\omega_{IE} + \dot{L}) \quad -\dot{l} \quad -sl(\omega_{IE} + \dot{L})]^{T}$
K = GEO (ENU)	$\underline{\omega}_{IGEO}^{GEO} = [-\dot{l} \quad cl(\omega_{IE} + \dot{L}) \quad sl(\omega_{IE} + \dot{L}]^{T}$
K = N	$\underline{\omega}_{IN}^{N} = [-\dot{l} + cls\alpha\omega_{IE} \quad \dot{L}cl + clc\alpha\omega_{IE} \quad sl]^{T}$
From E to Computational Frame K	$\underline{\boldsymbol{\omega}}_{\mathbf{EK}}^{\mathbf{K}}$
K = L (NED)	$\underline{\omega}_{EL}^{L} = [\dot{L}cl \quad -\dot{l} \quad -\dot{L}sl]^{T}$
K = GEO (ENU)	$\underline{\omega}_{EGEO}^{GEO} = [-\dot{l} \quad \dot{L}cl \quad \dot{L}sl]^{T}$
K = N	$\underline{\omega}_{EN}^{N} = [-\dot{l} \quad \dot{L}cl \quad 0]^{T}$

3.3 Earth Model

3.3.1 Earth Shape Model and Related Parameters

In general, and for navigation purposes, to approximate the true figure of the Earth to a sphere is not satisfactory at all. In fact, precise measurements require that an ellipsoidal shape must be fit to the Earth´s surface. Because the shape of the Earth may be described as slightly flattened at the poles, the geometrical figure, see Figure 3.3, that is mostly used to nearly approximate the shape of the earth is an ellipsoid of revolution. It should be stressed that the altitude of a position location over the Earth, means the height (h) above the reference ellipsoid, measured along a line that passes through the position point, and is perpendicular to the surface of the reference ellipsoid below $(h > 0)$ or above $(h < 0)$ the position location (Savage, 2007).

Figure 3.3 illustrates the geometry involved with the selected ellipsoid of revolution. The relative geometry position and the definition of position variables that will be used in

subsequent analytic developments – as part of the strapdown navigator to be developed – are described. Note that the ellipticity, which represents how closely an ellipsoid approaches a spherical shape, is intentionally exaggerated (for illustrative purposes).

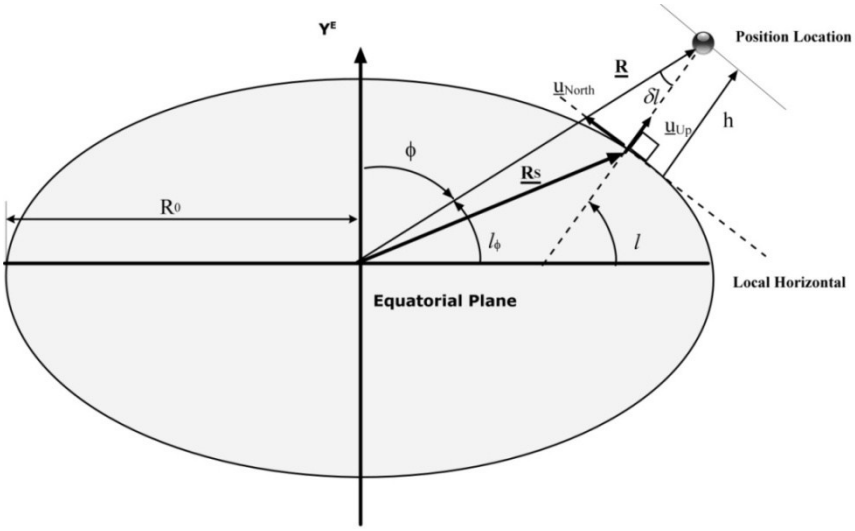

Figure 3.3 Earth Surface Analytical Model. Position Relative to Earth in Local Meridian Plane. Source: Adapted from (Savage, 2007).

Where:

$\underline{R}_S$	Distance vector from the centre of the ellipsoid to a point on the ellipsoid surface. Note that although the point in the figure is above the earth surface, the equations that will be developed will take into account both situations: above and below the earth surface.
R_0	Semi-major axis of the reference ellipse, i.e., corresponds to the earth´s equatorial radius.
e	Ellipticity of the reference ellipse, also called as flattening.
u_{UP}	Unit vector perpendicular (plum-bob line) to the ellipsoid surface at $\underline{R}_S$.
h	Altitude above the Earth reference ellipsoid (from Earth´s surface along the Earth surface normal unit vector $u_{UP}$to the current position location).
l	Geodetic latitude defined as the angle from the Earth's equatorial plane to u_{UP}.
$l_\emptyset$	Geocentric latitude defined as the angle from the Earth´s equatorial plane to $\underline{R}$.
∂l	Deflection of vertical: difference between geodetic and geocentric latitudes.

ϕ Angle from Y^E to $\underline{R}$. Note that this angle lies in the range of zero to π, depending on the position location (Northern or Southern hemisphere).

The Geodetic Reference System 80 (GRS80) (Moritz, 1980), and the World Geodetic System (WGS84) (DoD, 4 July 1997) are the two closely related standards for defining the reference ellipsoids. The reference ellipsoid that has been selected is the WGS-84. Table 3-3 is a listing of the principal numerical values associated to the Earth related navigation constants (DoD, 4 July 1997) that will be used to calculate the gravity model parameters according to the above equation, to be implemented in the strapdown navigator in order to compensate the specific force for the gravity component.

Table 3.3Earth shape, gravity, and angular rates WGS-84 ellipsoid constants. Source: WGS84 (DoD, 4 July 1997).

Constant	WGS-84	Units
μ	3986005	$\times 10^8$ m^3/s^2
J_2	0.001082627	-
J_3	-2.5327 E^{-6}	-
R_0	6 378 124	m
w_e	7.2921150	$\times 10^{-5}$ rad/s
e = f	1 / 297.257223	-

Taking into account Figure 3.3 and the constants values presented at Table 3-3, Table 3-4 presents now the formulas that have been used to implement the several Earth related navigation parameters. In this table we can find the needed input parameters, the desired output parameters and related equations. Special attention goes to the two parameters in the last two rows (output column), called the two radii of curvature: the first sensed when the craft moves North-South along a meridian (local navigation point radius of curvature in latitude direction, latitude change), rl; and the other sensed when the craft moves East-West (in longitude direction, longitude change), r_L (Savage, 2007). It should be stated that, in common literature, this two curvature radii are usually called: the meridian radii of curvature (R_m), and prime radius of curvature (R_P), respectively (Bekir, 2007). It should be stated that

the radii of curvature at the Earth surface point, r_{LS}, is equal to the modified radial distance to the Earth surface location, $R_S^{'}$. For more details about this topic the reader is invite to read (Savage, 2007).

Table 3.4 Formulas to be used to calculate ellipsoidal earth referenced navigation parameters.

Input	Output	Equation
$\mathbf{C_E^N}$	$u_{UP_{YE}}$	$u_{UP_{YE}} = D_{23}$(Table 3)
$\mathbf{R_0, e, u_{UP_{YE}}}$	R'_S	$R'_S = R_0 / \sqrt{1 + u^2_{UP_{YE}}[(1-e)^2 - 1]}$
$\mathbf{R'_S, e, u_{UP_{YE}}}$	R_s	$R_s = R'_S / \sqrt{1 + u^2_{UP_{YE}}[(1-e)^4 - 1]}$
$\mathbf{R_0, R_S, R'_S, h}$	R	$R^2 = R_S^2 + 2hR_0/R'_S + h^2$
$\mathbf{R'_S, e, R, h, u_{UP_{YE}}}$	$cos\emptyset$	$cos\emptyset = u_{UP_{YE}}[(1-e)^2 R'_S + h]/R$
$\mathbf{R'_S, R, h}$	$\left(\frac{sin\emptyset}{\sqrt{1-u^2_{UP_{YE}}}}\right)$	$\left(\frac{sin\emptyset}{\sqrt{1-u^2_{UP_{YE}}}}\right) = (R'_S + h)/R$
$\mathbf{R'_S, R, h, e, u_{UP_{YE}}}$	$cosl, \left(\frac{sinl}{\sqrt{1-u^2_{UP_{YE}}}}\right)$	$cosl = \{(1 - u^2_{UP_{YE}}[1-(1-e)^2])R'_S + h\}/R \left(\frac{sinl}{\sqrt{1-u^2_{UP_{YE}}}}\right)$ $= 1 - u^2_{UP_{YE}}[1-(1-e)^2]R'_S/R$
$\mathbf{R'_S, R_0, e}$	r_{ls}	$r_{ls} = (1-e)^2 R_S'^3 / R_0^2$
$\mathbf{r_{ls}, h}$	r_l	$r_l = r_{ls} + h$
$\mathbf{r_{Ls} = R'_S, h}$	r_L	$r_L = r_{LS} + h$

3.4 Curvature Matrix

Given the equations for the two radii of curvature, r_l and r_L, it is now possible to construct the curvature matrix for a body (vehicle) at altitude, h, expressed in the GEO-Frame, L-Frame and N-Frame, according to the reference (Savage, 2007):

$$F_C^{GEO} = F_C^{ENU} = \begin{bmatrix} \frac{1}{r_l} & 0 & 0 \\ 0 & \frac{1}{r_L} & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.1}$$

The equivalent curvature matrix for the L-Frame North point mechanization will be:

$$F_C^L = F_C^{NED} = C_{ENU}^{NED} \begin{bmatrix} \frac{1}{r_l} & 0 & 0 \\ 0 & \frac{1}{r_L} & 0 \\ 0 & 0 & 0 \end{bmatrix} C_{NED}^{ENU} = \begin{bmatrix} \frac{1}{r_L} & 0 & 0 \\ 0 & \frac{1}{r_l} & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.2}$$

The curvature matrix in the N-Frame is obtained from the GEO in the same way. In fact, as it has been presented at Section 3.1, the N-Frame is rotated from the GEO-Frame about the local vertical by the wander angle α (see Section 3.1, Figure 3.1). Thus, and in the same manner, in order to get the curvature matrix in the N-Frame, one has to proceed as follows:

$$\begin{aligned} F_C^N &= C_{GEO}^N F_C^{GEO} C_N^{GEO} = \\ &= \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{r_l} & 0 & 0 \\ 0 & \frac{1}{r_L} & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} F_{C_{11}} & F_{C_{12}} & 0 \\ F_{C_{21}} & F_{C_{22}} & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{aligned} \tag{3.3}$$

Where the curvature terms are given as follows:

$$\begin{aligned} F_{C_{11}} &= \frac{1}{r_l}(1 + D_{21}^2 f_{eh}) \qquad F_{C_{22}} = \frac{1}{r_l}(1 + D_{22}^2 f_{eh}) \\ F_{C_{12}} &= \frac{1}{r_l} D_{21} D_{22} f_{eh} \qquad F_{C_{21}} = \frac{1}{r_l} D_{21} D_{22} f_{eh} \\ f_e &= \frac{(1-e)^2 - 1}{1 + D_{23}^2} \qquad f_h = \frac{1}{1 + h/R'_S} \\ f_{eh} &= f_e f_h \end{aligned} \tag{3.4}$$

Where:

D_{2j} Element j in the second row of the C_N^E.

The elements D_{21}, D_{22}, and D_{23} that can be extracted from the C_E^N matrix presented at Table 3, after apllying the relation $C_N^E = \left(C_E^N\right)^T$. Thus we have:

$$D_{21} = cls\alpha, D_{22} = clc\alpha \text{ and } D_{23} = sl \tag{3.5}$$

Therefore, the elements of the curvature matrix will be given by:

$$\begin{aligned} F_{C_{11}} &= \frac{1}{r_l}\left(1 + (cls\alpha)^2 \frac{(1-e)^2 - 1}{1 + s^2 l} \frac{1}{1 + h/R'_S}\right) \\ F_{C_{22}} &= \frac{1}{r_l}\left(1 + (clc\alpha)^2 \frac{(1-e)^2 - 1}{1 + s^2 l} \frac{1}{1 + h/R'_S}\right) \\ F_{C_{12}} &= \frac{1}{r_l}(cl)^2 s\alpha c\alpha \frac{(1-e)^2 - 1}{1 + s^2 l} \frac{1}{1 + h/R'_S} = F_{C_{21}} \end{aligned} \tag{3.6}$$

3.5 Transport Rate

The transport rate $\omega_{EN}^N = \underline{\rho}^N$, is defined as the angular rate of the locally level N-Frame relative to the E-Frame. This vector can be decomposed into two components: the horizontal component (the N-Frame X, Y components), and the vertical component (Z component). The horizontal components rotate the N-Frame Z axis to remain parallel to the geodetic vertical (i.e., the u_{UP} vector). The vertical transport rate component is somewhat arbitrary, and depends on the type of frame mechanization selected.

According reference (Savage, 2007), the general equation for the transport rate in the N-Frame is given by:

$$\begin{aligned}\underline{\omega}_{EN}^{N} &= (\text{Horizontal Component} + \text{Vertical Component}) \\ &= F_C^N\left(\underline{u}_{ZN}^N \times \underline{v}^N\right) + \rho_{ZN}\underline{u}_{ZN}^N\end{aligned} \tag{3.7}$$

Where:

F_C^N Curvature matrix in the N-Frame given by (3.3).

$\underline{v}^N$ Velocity vector relative to the earth in the N-Frame.

ρ_{ZN} Vertical (Z axis) transport rate component, which value depends on the selected mechanization (see Table 3-5 for different options).

$\underline{u}_{ZN}^N$ Unit vector along the N-Frame Z axis, see (3.9).

Important is the fact that if the vertical mechanization selected is the wander azimuth with free azimuth vertical mechanization, then the unit vector along the N-Frame Z axis is given by:

$$\underline{u}_{ZN}^N = [0 \quad 0 \quad 1]^T \tag{3.8}$$

If, for another hand, the vertical mechanization selected is the North pointing, i.e., GEO, then the vertical component will be given by:

$$\underline{u}_{UP}^{GEO} = [0 \quad 0 \quad 1]^T \tag{3.9}$$

Finally, if the L-Frame is selected with vertical mechanization north pointing as well, i.e., NED mechanization, then the unit vector is given by:

$$\underline{u}_{UP}^{NED} = [0 \quad 0 \quad -1]^T \tag{3.10}$$

Note that the equivalent curvature matrices for the GEO, L, and N frames, are given by equations (3.1), (3.2), and (3.3), respectively. And so, in order to achieve the transport rate in each reference frame, one has to select the vertical component, ρ_{ZN}, from Table 3-5.

Table 3.5 N-Frame mechanization (effect in the transport rate vertical component).

	Vertical Mechanization	**Wander Angle rate** $\dot{\alpha}$	$\boldsymbol{\rho_{ZN}}$	**Vertical Inertial Angular Rate** $\omega_{IB_Z}^B = \dot{\alpha} + (\dot{L} + \omega)\sin l$	**Polar Capability** $l = \pm 90°$
North Pointing Mechanization ($\alpha = 0$)	North Pointing	0	$\dot{L}\sin l$	$(\dot{L} + \omega_{IE})\sin l$	No polar capability
Wander Azimuth Mechanization ($\alpha \neq 0$)	Free Azimuth	$-\dot{L}\sin l$	0	$\omega_{IE}\sin l$	Full Polar Capability
	Foucault	$-(\dot{L} + \omega_{IE})\sin l$	$-\omega_{IE}\sin l$	0	Full polar capability
	Unipolar Northern	$-\dot{L}$	$-\dot{L}(\sin l - 1)$	$\dot{L}(\sin l - 1) + \omega_{IE}\sin l$	Northern Hemisphere Polar Capability

3.5.1 Transport Rate in the Geo-Frame

The transport rate for each coordinate frame is now presented. First, the transport rate in the GEO-Frame is derived and then the transport rate in the L and N frames.

Taking into account the general equation for the transport rate given by (3.8), it is possible to derive the transport rate equation in the GEO-Frame in the following manner:

$$\underline{\rho}^{GEO} = F_C^{GEO}\left(\underline{u}_{UP}^{GEO} \times \underline{v}^{GEO}\right) + \rho_Z{}^{GEO}\underline{u}_{UP}^{GEO} \tag{3.11}$$

Where:

$\underline{v}^{GEO}$ Velocity vector in the GEO frame which components are $(v_{East}, v_{North}, v_{Up})$ henceforth denoted as (v^E, v^N, v^U).

According to Table 7, the vertical component for the North pointing mechanization is given by:

$$\underline{\rho}_Z^{GEO} = \underline{\rho}_Z^{NED} = \dot{L}\sin l \tag{3.12}$$

Thus, substituting (3.1), (3.10), and (3.13) in (3.12) we obtain:

$$\underline{\rho}^{GEO} = \begin{bmatrix} \rho^E \\ \rho^N \\ \rho^U \end{bmatrix} = \begin{bmatrix} \frac{1}{r_l} & 0 & 0 \\ 0 & \frac{1}{r_L} & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} v^E \\ v^N \\ v^U \end{bmatrix} + \dot{L}\sin l \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$
$$= \begin{bmatrix} \frac{-v^N}{r_l} \\ \frac{v^E}{r_L} \\ \dot{L}\sin l \end{bmatrix} \tag{3.13}$$

3.5.2 Transport Rate in the L-Frame

The transport rate in the L-Frame can be derived directly from (3.14) as follow:

$$\underline{\rho}^{L} = \underline{\rho}^{NED} = \begin{bmatrix} \rho^N \\ \rho^E \\ \rho^D \end{bmatrix} = C_{ENU}^{NED} \underline{\rho}^{GEO}$$
$$= \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \frac{-v^N}{r_l} \\ \frac{v^E}{r_L} \\ \dot{L}\sin l \end{bmatrix} \tag{3.14}$$
$$= \begin{bmatrix} \frac{v^E}{r_L} \\ \frac{-v^N}{r_l} \\ -\dot{L}\sin l \end{bmatrix}$$

3.5.3 Transport Rate in the N-Frame

In the same manner, and using an uncomplicated way to get the transport rate in the N-Frame is to do it from the GEO-Frame transport rate setting the Z axis component to zero, according to Table 3-5, and using (3.3) for C_{GEO}^{N} with (3.14), we get:

$$\underline{\rho}^{N} = \begin{bmatrix} \rho_{XN} \\ \rho_{YN} \\ \rho_{ZN} \end{bmatrix} = C^{N}_{GEO}\underline{\rho}^{GEO}$$

$$= \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{-v^{N}}{r_{l}} \\ \frac{v^{E}}{r_{L}} \\ 0 \end{bmatrix} \tag{3.15}$$

$$= \begin{bmatrix} \frac{-v^{N}}{r_{l}}\cos\alpha + \frac{v^{E}}{r_{L}}\sin\alpha \\ \frac{v^{N}}{r_{l}}\sin\alpha + \frac{v^{E}}{r_{L}}\cos\alpha \\ 0 \end{bmatrix}$$

Table 3-6, is a summary of the transport rate equations developed for the three different reference frames. The table lists the algorithm function: input parameters, output parameters and related equations.

Table 3.6 Transport rate for North pointing and free azimuth vertical mechanization summary.

Input	Output	Equation
$\mathbf{r_l}, \mathbf{r_L}, \mathbf{u_{UP}^{GEO}}, \underline{\mathbf{V}}^{\mathbf{GEO}}, \boldsymbol{\rho_Z}$	$\underline{\rho}^{GEO(ENU)}$	$\underline{\rho}^{GEO} = \begin{bmatrix} \frac{-v^{N}}{r_{l}} & \frac{v^{E}}{r_{L}} & \dot{L}\sin l \end{bmatrix}^{T}$
$\mathbf{r_l}, \mathbf{r_L}, \mathbf{u_{UP}^{NED}}, \underline{\mathbf{V}}^{\mathbf{L}}, \boldsymbol{\rho_Z}$	$\underline{\rho}^{L(NED)}$	$\underline{\rho}^{L} = \begin{bmatrix} \frac{v^{E}}{r_{L}} & \frac{-v^{N}}{r_{l}} & -\dot{L}\sin l \end{bmatrix}^{T}$
$\mathbf{C^{N}_{GEO}}, \underline{\boldsymbol{\rho}}^{\mathbf{GEO}}$	$\underline{\rho}^{N}$(Wander Azimuth)	$\underline{\rho}^{N} = \begin{bmatrix} \frac{-v^{N}}{r_{l}}\cos\alpha + \frac{v^{E}}{r_{L}}\sin\alpha & \frac{v^{N}}{r_{l}}\sin\alpha + \frac{v^{E}}{r_{L}}\cos\alpha & 0 \end{bmatrix}^{T}$

Another key point is that the position rate vector in the GEO, L, and N frames: $\dot{\underline{R}}^{GEO}$, $\dot{\underline{R}}^{L}$, and $\dot{\underline{R}}^{L}$ that relate the curvilinear coordinates with the different frame velocity components can be achieved by relating equations on Table 3-7 with the respective angular rates given in Table 3-5. Therefore the position rate vector in the GEO-Frame is given by:

Considering $\dot{h} = \underline{u}_{ZGEO}\underline{v}^{GEO}$ (Savage, 2007):

$$\underline{\dot{R}}^{GEO} = \begin{bmatrix} \dot{l} \\ \dot{L} \\ \dot{h} \end{bmatrix} = \begin{bmatrix} v^N/(r_{ls} + h) \\ v^E/((r_{Ls} + h)\cos l) \\ v^U \end{bmatrix} \tag{3.16}$$

The position rate vector in the L-Frame is given by:

$$\underline{\dot{R}}^{L} = \begin{bmatrix} \dot{l} \\ \dot{L} \\ \dot{h} \end{bmatrix} = \begin{bmatrix} v^N/(r_{ls} + h) \\ v^E/((r_{Ls} + h)\cos l) \\ -v^D \end{bmatrix} \tag{3.17}$$

In this case, the altitude rate is given by $\dot{h} = \underline{u}_{UP}^{NED}\underline{v}^{L}$ (Savage, 2007). Finally, the position rate vector in the N-Frame is given by:

$$\underline{\dot{R}}^{N} = \begin{bmatrix} \dot{l} \\ \dot{L} \\ \dot{h} \end{bmatrix} = \begin{bmatrix} \frac{1}{(r_{ls} + h)} v_{YN} \\ \frac{1}{r_l \cos l}(1 + f_{eh}\cos^2 l) v_{XN} \\ v_{ZN} \end{bmatrix} \tag{3.18}$$

3.6 Plumb-Bob gravity Model

In the development of the strapdown navigator algorithm, in locally level geodetic vertical coordinates, gravity will appear in conjunction with an Earth rate based centripetal acceleration term (Savage, 2007). The sum of these two terms is what is called the plumb-bob gravity, given the fact that it acts along the line a plumb-bob will take when stationary relative to the Earth at the same position location. For instance, when the INS is at rest (zero velocity relative to the Earth); the plumb-bob gravity is equivalent to the negative of the specific force acceleration. Therefore, according to reference (Savage, 2007) the plumb-bob gravity (gravity vector) is defined as the difference between the gravitation and centripetal accelerations (due to the Earth´s rotation), given by the following equation:

$$\underline{g}_P = \underline{g} - \omega_{IE} \times (\omega_{IE} \times \underline{R}) \tag{3.19}$$

Where:

$\underline{g}_P$	Plumb-bob gravity vector (commonly called the local gravity vector).
$\underline{R}$	Position vector from the earth´s centre (geocentric position vector of the mass).
$\underline{g}$	True gravity (gravitational vector/force of attraction between the Earth and mass).
ω_{IE}	Earth´s rotation rate vector (see Table 3.1).

The term $\omega_e \times (\omega_e \times \underline{R})$ corresponds to the centripetal acceleration experienced by a vehicle navigating in the vicinity of the Earth due to its rotation, not being separately distinguishable from true gravity (gravitational acceleration).

In order that a compensation term for the plumb-bob gravity can be made in the strapdown algorithm, it is crucial that we first model the gravity vector and then achieve the desirable plumb-bob gravity vector. Therefore, the standard gravity model that is used to model the gravity vector, $\underline{g}$, in the strapdown algorithm is based on the generalized model given in (Britting, 1971) which correctly divides the gravity components for positive and negative altitudes (note: at h = 0 the model for $h \geq 0$ equals the model for $h < 0$). Figure 3.4 illustrates this model, whose gravity components in polar coordinates are given by:

For $h \geq 0$:

$$\begin{aligned} g_r &= -\frac{\mu}{R^2}\left[1 - \frac{3}{2}J_2\left(\frac{R_0}{R_S}\right)^2(3\cos^2\phi - 1) - 2J_3\left(\frac{R_0}{R_S}\right)^3\cos\emptyset(5\cos^2\phi - 3) - \cdots\right] \\ \frac{g_\emptyset}{\sin\emptyset} &= 3\frac{\mu}{R^2}\left(\frac{R_0}{R}\right)^2\left[J_2\cos\phi + \frac{1}{2}J_3\frac{R_0}{R}(5\cos^2\phi - 1) + \cdots\right] \\ g_\theta &\approx 0 \end{aligned} \tag{3.20}$$

For $h < 0$:

$$\begin{aligned} g_r &= \frac{R}{R_S}g_{rs} \\ \frac{g_\phi}{\sin\phi} &= \frac{R}{R_S}\left(\frac{g_\phi}{\sin\phi}\right)_S \\ g_\theta &\approx 0 \end{aligned} \tag{3.21}$$

Where (see Figure 3.4):

g_r	Gravity component along the $\underline{R}$ direction.
$g_\emptyset$	Gravity component perpendicular to $\underline{R}$ in the local meridian plane.
g_θ	Gravity component perpendicular to $\underline{R}$ and to the local meridian plane.
R, R_0	Magnitude of $\underline{R}$ and Earth´s equatorial radius, respectively.
ϕ	Angle from Y^E to $\underline{R}$.
μ	Gm (G is the universal gravitational constant; m is mass of the Earth).
J_2, J_3 ...	Small empirical constants function of the mass distribution of the Earth. Numerical values for μ, J_2, J_3 are presented at Table 3.1.
g_{rs}	Corresponds to g_r calculated using (3.12) with R set to R_S
$\left(\frac{g_\emptyset}{\sin\emptyset}\right)_S$	Corresponds to $\frac{g_\emptyset}{\sin\emptyset}$ calculated using (3.12) with R set to R_S.

Important to realize is the fact that when a spherical Earth of constant density is considered, those small constants (J_2, J_3 ...) would be zero and so different equations are obtained for the gravity components in polar coordinates and, consequently, for the plumb-bob gravity which is calculated from this components.

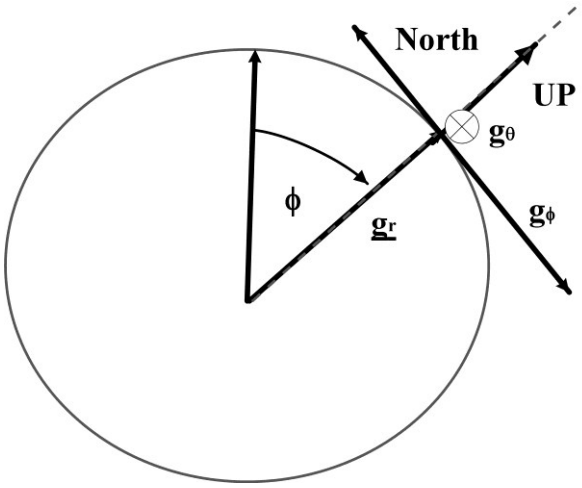

Figure 3.4 Gravity Model. Source: (Britting, 1971).

However, it does not make any sense to implement the gravity vector in polar coordinates. In fact, gravity components along navigation coordinates − aligned with the local horizontal North, u_{North}, and geodetic vertical, i.e., u_{Up} − are desired. According to Figure 3.3, the gravity components along $\underline{u}_{UP}$ and $\underline{u}_{North}$, can be achieved through the deflection of the vertical angle ∂l as follows:

$$\underline{g}^{GEO} = \begin{bmatrix} g^E \\ g^N \\ g^U \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\partial l & -sin\partial l \\ 0 & sin\partial l & cos\partial l \end{bmatrix} \begin{bmatrix} 0 \\ -g_\phi \\ g_r \end{bmatrix} = \begin{bmatrix} 0 \\ -g_\phi cos\partial l - g_r sin\partial l \\ -g_\phi sin\partial l + g_r cos\partial l \end{bmatrix} \quad (3.22)$$

Where:

g^N Gravity component along $\underline{u}_{North}$.

g^U Gravity component along $\underline{u}_{UP}$.

Table 3.8 is a summary of the gravity components along East, North, Up directions, needed to calculate the gravity model that will be used to calculate the plumb-bob gravity. This will, then, be added to the specific force obtained from the accelerometer measurements in the navigator algorithm. It should be stated that the reference gravity model that is used is explained with more detail at references (Savage, 2007) and (Britting, 1971).

Table 3.7 Gravity components in the GEO-Frame to be used in the gravity model.

Input	Output	Equation
-	$g^{E} = g_{East}$	$g_{East} = 0$
$\mathbf{g_{\phi}, g_{r}, \partial l}$	$g^{N} = g_{North}$	$g_{North} = -g_{\phi}\cos\partial l - g_{r}\sin\partial l$
$\mathbf{g_{\phi}, g_{r}, \partial l}$	$g^{U} = g_{Up}$	$g_{Up} = -g_{\phi}\sin\partial l + g_{r}\cos\partial l$

To finish this section, the plumb-bob gravity components in different coordinate frame, being used in this thesis for implementation of the strapdown algorithm, are presented at Table 3.9. It should be noted that the components g_{North} and g_{UP}can be found in Table 3.8, whereas the needed elements D_{21}, and D_{22} are given by (3.6). Important to bear in mind, is the fact that those two elements depend on the selected vertical mechanization.

Table 3.8 Plumb-Bob gravity in the GEO, L, and N frames.

Input	Output	Equation
$\mathbf{g_{North}, g_{Up}, u_{UP_{YE}}, D_{21}, D_{22}, R'_S, h, \omega}$	$\underline{g}_P^{GEO}$	$\underline{g}_P^{GEO} = \begin{bmatrix} g_P^E \\ g_P^N \\ g_P^U \end{bmatrix} = \begin{bmatrix} 0 \\ \left(\frac{g_{North}}{\sqrt{1-u_{UP_{YE}}^2}}\right) - (R'_S + h)\omega_e^2 u_{Up_{YE}} \\ g_{Up} + (R'_S + h)\omega_e^2(1-u_{Up_{YE}}^2) \end{bmatrix}$
$\mathbf{C_{GEO}^{NED}, \underline{g}_P^{GEO}}$	$\underline{g}_P^{L(NED)}$	$\underline{g}_P^{L} = \begin{bmatrix} g_P^N \\ g_P^E \\ g_P^D \end{bmatrix} = \begin{bmatrix} g_P^N \\ g_P^E \\ -g_P^U \end{bmatrix}$
$\mathbf{C_{GEO}^{N}, \underline{g}_P^{GEO}}$	$\underline{g}_P^{N}$	$\underline{g}_P^{N} = \begin{bmatrix} g_{PX}^N \\ g_{PY}^N \\ g_{PZ}^N \end{bmatrix} = \begin{bmatrix} \left(\frac{g_P^N}{\sqrt{1-u_{UP_{YE}}^2}}\right) D_{21} \\ \left(\frac{g_P^N}{\sqrt{1-u_{UP_{YE}}^2}}\right) D_{22} \\ g_P^N \end{bmatrix}$

CHAPTER 4

PART I CONCLUSION SUMMARY

4 Part I Conclusion Summary

The **specific objectives** associated to Part I were: *i)* to present a clear overview of the inertial navigation basic concepts, hoping to clarify what sometimes is a misunderstanding in the inertial navigation world, and *ii)* to give the reader the theoretical background related to coordinate systems in order to clearly understand the basic principles behind coordinate transformations.

In Chapter 1, the motivation and statement of the problem as well as the constraints of the thesis were presented. The adopted research methodology used along the thesis was described and the research main question that guides this thesis was identified. Finally, the contributions to the study were identified.

Chapter 2 confirms that the first major contribution of this study has been satisfactorily addressed. As a matter of fact, a clear overview on the inertial navigation basic concepts was presented. Differences between an ISA, IMU, and INS as well as other relevant information related to inertial navigation sensors were identified.

In the same manner, details provided in Chapter 3 prove that the second major contribution has been adequately addressed, as well. This chapter presented a summary of all the needed DCM between the several coordinate's frames used along this thesis. The Earth model that will be used in Part II, as well as some other concepts such as transport rate, radii of curvature, were described.

PART II

STRAPDOWN NAVIGATOR

"The equations of motion in a non-inertial system differ from the equations in an inertial system by additional terms called inertial forces. This allows us to detect experimentally the non-inertial nature of a system. ...

V. I. Arnold: Mathematical Methods of Classical Mechanics Second Edition, p. 129"

CHAPTER 5

STRAPDOWN INERTIAL NAVIGATION MECHANIZATION

5 Strapdown Inertial Navigation Mechanization

The first secondary research question of this thesis is concerned with "*what is the purpose of developing a SIMU, and is that possible to provide a generic SIMU algorithm as a complete and comprehensive "recipe" able to be implemented in any computer being suitable for different categories of IMU sensors?"*. This question is investigated in this chapter.

The strapdown INS computer has the responsibility of providing as its output the navigation parameters – position, velocity and attitude – of the body state in which the INS is installed, having as inputs the inertial measurements – specific force and angular rates – supported by the strapdown inertial sensors. The set of differential equations implemented and executed in the computer, describing the relationship between those measurements – attitude integration, velocity integration, and acceleration integration – are what is usually called the strapdown mechanization equations.

In fact, the mechanization equations are no more than a digital algorithm – based in a continuous form of differential navigation equations – designed to generate the identical solution at the update time to that of a continuous one.

The implementation and execution, with digital algorithms operating at a specified repetition rate, of the associated navigation equations in the computer is what is here professed the **strapdown navigator**, whose implementation is complex due to the need to navigate along the curved surface of a rotating Earth. As a consequence, some compensation related to the Earth´s gravity field, instability in the vertical direction, sensor errors must be considered during the implementation. As a matter of fact, all these "obstacles" have to be known in order to be converted into corrections to the navigation equations.

According Savage (Savage, 2007) describes the two-speed INS mechanization algorithms that are the result of over 20 years of worldwide developments in strapdown navigation algorithms. The high speed part (1 – 4 kHz) computes the coning and sculling

motion, and the medium speed part (100 - 400 Hz) generates the navigation states, i.e., position, velocity and attitude. However, an advance in modern computer technology proposes the use of the single-speed algorithm. For this reason and also given the fact that the algorithm is to be used, at first glance, in post-processing mode, in this project a single speed algorithm has been developed as a simplification of the two-speed algorithm proposed in (Savage, 2007).

In order to develop the strapdown navigator, this part is divided into three chapters. In the first chapter, the continuous form of the navigation equations is described in the different coordinate frames. Then, the equivalent digital integration algorithm form based on the differential equations presented at first step is derived in this chapter as well.

Given the fact that the main objective of this thesis is to investigate the performance of an integrated navigation algorithm using a low performance IMU, rises the need for error mitigation techniques, and so not only the navigation equations in both continuous and digital form are here presented and also the associated error equations. Thus, Chapter 6 describes the error behaviour associated to the implemented inertial navigation equations.

Finally, and in order to validate the implemented algorithm, a trajectory generator has been developed to give perfect accelerometer and gyro measurements. Important to realized, that it was crucial at this point to test the algorithm in order to find out if all the transformations and equations are well implemented and executed. Therefore, Chapter 7 summarizes Part II answering to the proposed first secondary question and validating Hypothesis 1.

Due to the fact that different coordinate systems are associated to different parts of the strapdown navigator here implemented and given the fact that the DCM will be used extensively in this chapter, Chapters 3 must be considered as a support chapter for this chapter.

5.1 General Mechanization Equations: Continuous Form

The continuous form of the navigation equations depends on which coordinate frame the navigation solution is expressed in. And so, this section describes, first, the general equations for the I-Frame implementation and, next, how they are modified (due to the curved and rotating Earth) for implementation in the GEO, L, and N frames respectively.

As usual, we start with Newton´s second law – the fundamental equation for the motion of a particle in the Earth´s gravitational field – and finally we finish with the equations

in the several coordinate frames.

Thus, starting with Newton´s law, the fundamental equation for a navigating body in the gravitational field of the earth, expressed in an inertial frame, is (Savage, 2007).

$$\ddot{\underline{R}}^{I} = \underline{a}^{I}_{SF} + \underline{g}^{I} \tag{5.1}$$

Where:

$\underline{R}$	Column vector representing the position vector from the origin of the I-Frame to the moving body.
$\ddot{\underline{R}}^{I}$	Total inertial acceleration acting on the moving body.
$\underline{a}^{I}_{SF}$	Specific force vector.
$\underline{g}^{I}$	Gravitational acceleration vector in I-Frame coordinates.

Equation (5.1) seems to be a good starting point to develop the navigation equations in the other coordinates frames. Let's start with the same equation – which is a set of three-second-order differential equations – and transform it into a set of six first-order differential equations (Schwarz, et al., 2006). To achieve that, we start with the time derivative of the position vector which is equal to the velocity vector, and the time derivative of the velocity, which is equal to the acceleration vector and are given respectively by:

$$\dot{\underline{R}} = \underline{v}^{I} \tag{5.2}$$

$$\dot{\underline{v}}^{I} = \underline{a}^{I}_{SF} + \underline{g}^{I} \tag{5.3}$$

Where:

$\underline{v}^{I}$	Column vector representing the velocity vector projected along the I-Frame axes.

Bearing in mind some considerations related to the specific force vector, it can be stated that the specific force vector provided by the strapdown accelerometers is related to the body frame, $\underline{a}^{B}_{SF}$, and not to the inertial frame. Therefore, in order to have those measurements into the inertial frame, we have to transform such measurements into the I-Frame as follows:

$$\underline{a}_{SF}^{I} = C_{B}^{I}\underline{a}_{SF}^{B} \tag{5.4}$$

Where the rotation matrix C_{B}^{I} is used as the transformation matrix from B-Frame to the I-Frame, which according to Appendix A can be obtained as follows:

$$\dot{C}_{B}^{I} = C_{B}^{I}(\omega_{IB}^{B}\times) \tag{5.5}$$

Where $\omega_{IB}^{B}$are the angular rates provided by the strapdown gyros.

On the other hand, the gravitational vector is usually not given in the I-Frame, but either in the E-frame or N-frame. Another rotation matrix has to be used in order to transform the gravitational vector into the I-Frame.

$$\underline{g}^{I} = C_{E}^{I}\underline{g}^{E} \tag{5.6}$$

Using Equations (5.2) to (5.6), we get the set of the first-order differential equations of the form (Schwarz, et al., 2006):

$$\dot{X}^{I} = \begin{bmatrix} \dot{\underline{R}}^{I} \\ \dot{\underline{v}}^{I} \\ \dot{C}_{B}^{I} \end{bmatrix} = \begin{bmatrix} \underline{v}^{I} \\ C_{B}^{I}\underline{a}_{SF}^{B} + C_{E}^{I}\underline{g}^{E} \\ C_{B}^{I}(\omega_{IB}^{B}\times) \end{bmatrix} \tag{5.7}$$

Where:

$\dot{\underline{R}}^{I}$	Position rate equation expressed in the I-Frame
$\dot{\underline{v}}^{I}$	Velocity rate equation expressed in the I-Frame.
$\dot{C}_{B}^{I}$	Attitude rate equation expressed in the I-Frame.

Equations (5.7) describe the set of inertial navigation equations: the **Mechanization Equations** of a strapdown inertial navigation system, expressed in the I-Frame, which can be interpreted as a dynamic system representing the body – in which the strapdown inertial system is installed – motion (Schwarz, et al., 2006).

As follows, and using as support equations (5.7) the strapdown mechanization equations for the GEO, L, and N frames are developed.

It should be stated that the final equations are presented and so if there is a need for

more details about how to achieve those results, the reader should consult (Schwarz, et al., 2006) and (Savage, 2007).

To explain how to derive the mechanization equations in the different local level frames from (5.7), some basic considerations about each equation must be considered:

- Position Rate Equation
- For the position rate equation, curvilinear coordinates (l, L, h) are used as coordinate states of the position equation. Therefore, the time derivative of the curvilinear coordinates has to be related to the velocity components in order to achieve a three dimensional differential equation. Consequently equations (3.17), (3.18), and (3.19) of Chapter 3 must here be considered.
- Velocity Rate Equation
- Due to the Earth rotation, the gravitation model (5.6) must be substituted by the local gravity model which includes effects of the gravitation and centripetal component of gravity. In other words, the centripetal acceleration is also included in the gravity model differing from the inertial frame mechanization. And so, a compensation for the apparent gravitational acceleration acting on the body, which includes the gravitational acceleration caused by mass attraction of the Earth and centripetal acceleration of the body resulting from the Earth´s rotation must be done. To put it differently, the plumb-bob gravity model presented at Table 3.9, Chapter 3 must be here considered.
- A correction for the Coriolis acceleration, caused by the body´s velocity over the surface of a rotating Earth, must be here considered as well.
- A correction for the centripetal acceleration of the body resulting from its motion over the Earth´s surface must also be considered.
- Attitude Rate Equation
- The fact that the Earth is neither flat nor moving must be compensated as well in the attitude rate equation (5.7). Therefore, two new terms appear: the one due to the inertial referenced angular rate measured by the gyros will appear as corrections in the third row of Equation (5.7). Note that those two new terms are included into the angular rate of the local level with respect to the inertial frame vector – which are (see Table 3.1 of Chapter 3 how to get this vector for each specific frame):

- A correction term due to the rotation of the Earth with respect to an inertial frame.
- A correction term, usually called transport rate, due to the rotation of the considered navigation frame (GEO, L, or N) with respect to the Earth.

According to the exposed above, it is now possible to construct the new equations in the different local level frames, based in the format presented at (5.7).

Table 5.1 emerges from the exposed, presenting the system of differential equations accomplished for the desired computational frame k (where k can be the GEO, L or N frame). It should be noted that the required initial conditions are also presented.

Table 5.1Inertial navigation equations (Strapdown mechanization equations) in several coordinate frames.

Computational Frame (k)	**Initial Conditions**	**Equations (Vector state)**
K = GEO (ENU)	$\underline{\mathrm{R}}^{\mathrm{GEO}}(0) = \begin{bmatrix} l(0) \\ \mathrm{L}(0) \\ \mathrm{h}(0) \end{bmatrix}$, $\underline{\mathrm{v}}^{\mathrm{GEO}}(0)$, and $\mathrm{C}_{\mathrm{B}}^{\mathrm{GEO}}(0)$	$\dot{\underline{\mathrm{x}}}^{\mathrm{GEO}} = \begin{bmatrix} \dot{\underline{\mathrm{R}}}^{\mathrm{GEO}} \\ \dot{\underline{\mathrm{v}}}^{\mathrm{GEO}} \\ \dot{\mathrm{C}}_{\mathrm{B}}^{\mathrm{GEO}} \end{bmatrix} = \begin{bmatrix} \mathrm{R}_{\mathrm{GEO}}^{-1}\underline{\mathrm{v}}^{\mathrm{GEO}} \\ \mathrm{C}_{\mathrm{B}}^{\mathrm{GEO}}\underline{\mathrm{a}}_{\mathrm{SF}}^{\mathrm{B}} + \underline{\mathrm{g}}_{\mathrm{P}}^{\mathrm{GEO}} - \left(2\omega_{\mathrm{IE}}^{\mathrm{GEO}} + \omega_{\mathrm{EGEO}}^{\mathrm{GEO}}\right) \times \underline{\mathrm{v}}^{\mathrm{GEO}} \\ \mathrm{C}_{\mathrm{B}}^{\mathrm{GEO}}\left(\omega_{\mathrm{IB}}^{\mathrm{B}} \times\right) - \left(\omega_{\mathrm{IGEO}}^{\mathrm{GEO}} \times\right)\mathrm{C}_{\mathrm{B}}^{\mathrm{GEO}} \end{bmatrix}$
K = L (NED)	$\underline{\mathrm{R}}^{\mathrm{L}}(0) = \begin{bmatrix} l(0) \\ \mathrm{L}(0) \\ \mathrm{h}(0) \end{bmatrix}$, $\underline{\mathrm{v}}^{\mathrm{L}}(0)$, and $\mathrm{C}_{\mathrm{B}}^{\mathrm{L}}(0)$	$\dot{\underline{\mathrm{x}}}^{\mathrm{L}} = \begin{bmatrix} \dot{\underline{\mathrm{R}}}^{\mathrm{L}} \\ \dot{\underline{\mathrm{v}}}^{\mathrm{L}} \\ \dot{\mathrm{C}}_{\mathrm{B}}^{\mathrm{L}} \end{bmatrix} = \begin{bmatrix} \mathrm{R}_{\mathrm{L}}^{-1}\underline{\mathrm{v}}^{\mathrm{L}} \\ \mathrm{C}_{\mathrm{B}}^{\mathrm{L}}\underline{\mathrm{a}}_{\mathrm{SF}}^{\mathrm{B}} + \underline{\mathrm{g}}_{\mathrm{P}}^{\mathrm{L}} - (2\omega_{\mathrm{IE}}^{\mathrm{L}} + \omega_{\mathrm{EL}}^{\mathrm{L}}) \times \underline{\mathrm{v}}^{\mathrm{L}} \\ \mathrm{C}_{\mathrm{B}}^{\mathrm{L}}\left(\omega_{\mathrm{IB}}^{\mathrm{B}} \times\right) - \left(\omega_{\mathrm{IL}}^{\mathrm{L}} \times\right)\mathrm{C}_{\mathrm{B}}^{\mathrm{L}} \end{bmatrix}$
K = N	$\underline{\mathrm{R}}^{\mathrm{N}}(0) = \begin{bmatrix} l(0) \\ \mathrm{L}(0) \\ \mathrm{h}(0) \end{bmatrix}$, $\underline{\mathrm{v}}^{\mathrm{N}}(0)$, and $\mathrm{C}_{\mathrm{B}}^{\mathrm{N}}(0)$	$\dot{\underline{\mathrm{x}}}^{\mathrm{N}} = \begin{bmatrix} \dot{\underline{\mathrm{R}}}^{\mathrm{N}} \\ \dot{\underline{\mathrm{v}}}^{\mathrm{N}} \\ \dot{\mathrm{C}}_{\mathrm{B}}^{\mathrm{N}} \end{bmatrix} = \begin{bmatrix} \mathrm{R}_{\mathrm{GEO}}^{-1}\underline{\mathrm{v}}^{\mathrm{L}} \\ \mathrm{C}_{\mathrm{B}}^{\mathrm{N}}\underline{\mathrm{a}}_{\mathrm{SF}}^{\mathrm{B}} + \underline{\mathrm{g}}_{\mathrm{P}}^{\mathrm{N}} - (2\omega_{\mathrm{IE}}^{\mathrm{N}} + \omega_{\mathrm{EN}}^{\mathrm{N}}) \times \mathrm{v}^{\mathrm{N}} \\ \mathrm{C}_{\mathrm{B}}^{\mathrm{N}}\left(\omega_{\mathrm{IB}}^{\mathrm{B}} \times\right) - \left(\omega_{\mathrm{IN}}^{\mathrm{N}} \times\right)\mathrm{C}_{\mathrm{B}}^{\mathrm{N}} \end{bmatrix}$

5.2 Mechanization Equations Selected

In the previous section, the continuous form for the mechanization equations in different coordinate frames has been presented. In this section it is selected the one that will be implemented to be used with the low performance IMU presented in Section 2.5.3.

To some extent, the navigator algorithm developed in this thesis uses the wander frame to mechanize the inertial navigation equations, i.e. the N-Frame mechanization. Although the UAV is not supposed to traverse the poles to make the wander frame mechanization a requirement, it is a very widely used and a standard way of defining the navigation equations. In fact, with the wander frame as a basis, it is easier to compare the algorithm development and resulting performance with many systems already in use (Groves, 2008). So, both the velocity and position integrated functions are developed in the N-Frame. However, the same is not done to the attitude integrated function, as suggested by Savage (Savage, 2007), which is developed in the L-Frame (or more commonly called the NED frame).

Therefore, the system of differential equations in the continuous form – equations that have at inputs accelerometers measurements ($\mathrm{a}_{\mathrm{SF}}^{\mathrm{B}}$), and gyros measurements ($\omega_{\mathrm{IB}}^{\mathrm{B}}$) to provide a navigation solution at its output – selected for the implementation is given by:

$$\begin{aligned}\left[\dot{\underline{\mathrm{x}}}^{\mathrm{N/L}}\right] &= \begin{bmatrix}\dot{\underline{\mathrm{R}}}^{\mathrm{N}}\\ \dot{\underline{\mathrm{v}}}^{\mathrm{N}}\\ \dot{\mathrm{C}}_{\mathrm{B}}^{\mathrm{L}}\end{bmatrix}\\ &= \begin{bmatrix}\underline{\mathrm{R}}_{\mathrm{N}}^{-1}\underline{\mathrm{v}}^{\mathrm{N}}\\ \mathrm{C}_{\mathrm{B}}^{\mathrm{N}}\underline{\mathrm{a}}_{\mathrm{SF}}^{\mathrm{B}} + \underline{\mathrm{g}}_{\mathrm{p}}^{\mathrm{N}} - \left(2\underline{\omega}_{\mathrm{IE}}^{\mathrm{N}} + \underline{\omega}_{\mathrm{EN}}^{\mathrm{N}}\right)\times\underline{\mathrm{v}}^{\mathrm{N}}\\ \mathrm{C}_{\mathrm{B}}^{\mathrm{L}}\left(\underline{\omega}_{\mathrm{IB}}^{\mathrm{B}}\times\right) - \left(\underline{\omega}_{\mathrm{IL}}^{\mathrm{L}}\times\right)\mathrm{C}_{\mathrm{B}}^{\mathrm{L}}\end{bmatrix}\end{aligned} \tag{5.8}$$

Where the initial conditions are given by:

$$\underline{\mathrm{R}}^{\mathrm{N}}(0) = \begin{bmatrix}l(0)\\ \mathrm{L}(0)\\ \mathrm{h}(0)\end{bmatrix}, \underline{\mathrm{v}}^{\mathrm{N}}(0), \text{and } \mathrm{C}_{\mathrm{B}}^{\mathrm{L}}(0) \tag{5.9}$$

All in all, the five basic operations to be performed by the strapdown navigator in computer are (Savage, 2007):

- C_B^N – Attitude computation using the angular rate sensor data, $\underline{\omega}_{IB}^B$;
- $\underline{a}_{SF}^N$ –Transforming Accelerometer data from B Frame to N Frame using attitude from step one, $C_B^N \underline{a}_{SF}^B$;
- Integrating accelerometer data from step two, $\underline{a}_{SF}^N$ to get velocity relative to earth in N Frame. Adding gravity and Coriolis contribution;
- Integrating velocity to get position;
- Converting Attitude, Velocity, and Position data to desired format for output.

Table 5-2 presents the related equations required to implement Equations (5.8).

It should be stressed that the algorithm for calculating the position relative to the Earth is in the form of altitude (h) above the Earth surface and the C_N^E direction cosine matrix defining the angular attitude between the local level N-Frame and the E-Frame, according to the next section, from which latitude and longitude can be extracted. Therefore, in the strapdown algorithm implemented, according to the next section, the altitude integration is made separately. Thus, in addition to Equations (5.8), a vertical channel control algorithm has been also implemented in order to eliminate the error in vertical velocity and altitude. The algorithm implemented is the one suggest from (Savage, 2007). This algorithm is suggested for applications that exceed 10 minutes of navigation, given the fact that this is the time when the vertical position error becomes unacceptable.

And so, this algorithm is based on an external input for altitude, which can be provided by a GPS receiver, from a barometric pressure sensor, or from another type of sensor onboard the UAV. The algorithm proposed is the following:

Consider that the altitude obtained from the strapdown algorithm without aiding is h, and the one provided by the GPS or aiding sensor is h_{sensor} then the altitude error signal will be:

$$\partial h = h - h_{sensor} \tag{5.10}$$

Now let´s consider the following vertical channel control signals e_{vc1}, e_{vc2} and e_{vc3} and gains,C_1, C_2 and C_3:

$$\begin{aligned} \mathrm{e_{vc1}} &= \mathrm{e_{vc3}} + \mathrm{C_2}\,\partial\mathrm{h} \\ \mathrm{e_{vc2}} &= \mathrm{C_3}\,\partial\mathrm{h} \\ \dot{\mathrm{e}}_{\mathrm{vc3}} &= \mathrm{C_1}\,\partial\mathrm{h} \end{aligned} \tag{5.11}$$

As a consequence, the velocity and altitude rate equations are, starting from the original ones given by (5.8), modified to include those signals in the following manner (Savage, 2007).

$$\begin{aligned} \dot{\underline{\mathrm{v}}}^{\mathrm{N}} &= \underline{\mathrm{a}}^{\mathrm{N}}_{\mathrm{SF}} + \underline{\mathrm{g}}^{\mathrm{N}}_{\mathrm{p}} - \left(\underline{\omega}^{\mathrm{N}}_{\mathrm{EN}} + 2\underline{\omega}^{\mathrm{N}}_{\mathrm{IE}}\right) \times \underline{\mathrm{v}}^{\mathrm{N}} - \mathrm{e_{vc1}}\underline{\mathrm{u}}^{\mathrm{N}}_{\mathrm{ZN}} \\ \dot{\mathrm{h}} &= \underline{\mathrm{u}}^{\mathrm{N}}_{\mathrm{ZN}}\underline{\mathrm{v}}^{\mathrm{N}} - \mathrm{e_{vc2}} \end{aligned} \tag{5.12}$$

Equations (5.8) and (5.12) are presented in the continuous form.

Table 5.2 Input parameters and related equations for the inertial mechanization equations selected (5.8).

Input	Equation
$\underline{\boldsymbol{\omega}}_{\mathbf{IB}}^{\mathbf{B}} = \left[\boldsymbol{\omega}_{\mathbf{x}}^{\mathbf{B}} \quad \boldsymbol{\omega}_{\mathbf{y}}^{\mathbf{B}} \quad \boldsymbol{\omega}_{\mathbf{z}}^{\mathbf{B}}\right]^{\mathbf{T}}$	Gyros measurements (IMU (MMQ50) gyros outputs)
$\underline{\mathbf{a}}_{\mathbf{SF}}^{\mathbf{B}} = \left[\mathbf{a}_{\mathbf{x}}^{\mathbf{B}} \quad \mathbf{a}_{\mathbf{y}}^{\mathbf{B}} \quad \mathbf{a}_{\mathbf{z}}^{\mathbf{B}}\right]^{\mathbf{T}}$	Accelerometers measurements (IMU (MMQ50) accelerometers outputs)
$\underline{\mathbf{R}}^{\mathbf{N}}(\mathbf{0}), \underline{\mathbf{V}}^{\mathbf{N}}(\mathbf{0})$ and $\mathbf{C}_{\mathbf{B}}^{\mathbf{L}}(\mathbf{0})$	Initialization Phase (initial conditions)
$\mathbf{C}_{\mathbf{B}}^{\mathbf{L}}$	Chapter 3;Table 3-1
$\mathbf{C}_{\mathbf{N}}^{\mathbf{L}}$	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix}$
$\underline{\boldsymbol{\omega}}_{\mathbf{IE}}^{\mathbf{N}}$	$\omega_{\mathrm{IE}}^{\mathrm{N}} = \mathrm{C}_{\mathrm{E}}^{\mathrm{N}} \omega_{\mathrm{IE}}^{\mathrm{E}} = \begin{bmatrix} cl s\alpha \\ cl c\alpha \\ sl \end{bmatrix} \omega_{\mathrm{e}}$
$\underline{\boldsymbol{\omega}}_{\mathbf{EN}}^{\mathbf{N}}$	Chapter 3; Table 3-2
$\underline{\mathbf{g}}_{\mathbf{p}}^{\mathbf{N}} = [\mathbf{g}_{\mathbf{P_{XN}}} \quad \mathbf{g}_{\mathbf{P_{YN}}} \quad \mathbf{g}_{\mathbf{P_{ZN}}}]^{\mathbf{T}}$	Chapter 3; Table 3-9
$\underline{\mathbf{u}}_{\mathbf{ZN}}^{\mathbf{N}}$	$[0 \quad 0 \quad 1]^{\mathrm{T}}$
$\underline{\boldsymbol{\omega}}_{\mathbf{IL}}^{\mathbf{L}} = \mathbf{C}_{\mathbf{N}}^{\mathbf{L}}\left(\boldsymbol{\omega}_{\mathbf{IE}}^{\mathbf{N}} + \boldsymbol{\omega}_{\mathbf{EN}}^{\mathbf{N}}\right)$	Equations above
$\underline{\mathbf{v}}^{\mathbf{N}} = \left[\mathbf{v}_{\mathbf{x}}^{\mathbf{N}} \quad \mathbf{v}_{\mathbf{y}}^{\mathbf{N}} \quad \mathbf{v}_{\mathbf{z}}^{\mathbf{N}}\right]$	Output
$\underline{\mathbf{R}}_{\mathbf{N}}^{-\mathbf{1}}$	$\begin{bmatrix} 0 & \frac{1}{(\mathrm{r}_l + \mathrm{h})} & 0 \\ \frac{1}{\mathrm{r}_l \mathrm{cosl}}(1 + \mathrm{f}_{\mathrm{eh}} \cos^2 l) & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

5.3 Inertial Navigation Equations: Digital Form

Section 5.2 presented the general inertial navigation equations to be implemented with the IMU in order to get the SIMU able of providing a navigation solution.

In this section, the desirable single speed algorithm in the digital form will be developed by simplifying and modifying the multispeed algorithm proposed by (Savage, 2007).

It should be noted that given the fact that a single speed algorithm is developed means that one will just have an update time for all calculations, here considered as the m[7] cycle.

It is important to realize that different from the outputs of high grade IMU, which are incremental angles and velocities due to precise digitization, most low cost IMU output specific forces and angular rates. Therefore, the mechanization equation proposed in (Savage, 2007) is operated on incremental velocities and incremental angles. And so, the strapdown navigator here developed uses **data increments as IMU output data**.

The IMU being used in this thesis is the Systron MMQ50, presented at Chapter 2, that outputs data at 450Hz (once every 0.0022 seconds). Therefore, the gyro output angular rates in degrees per second, and accelerometer output velocity rates in meters per second (more details are given in Part V, Chapter 17, must be converted, before integration, into angular increments in radians, and velocity increments in meters per second, respectively. This conversion is achieved using Equations (5.13).

$$\begin{cases} \Delta\underline{\theta}_{IB}^{B} = \int_{t}^{t+\Delta t} \omega_{IB}^{B} dt \\ \Delta\underline{V}_{IB}^{B} = \int_{t}^{t+\Delta t} a_{SF}^{B} dt \end{cases} \tag{5.13}$$

Where $\Delta\theta_{IB}^{B}$ represents the incremental angle (from gyros) and ΔV_{IB}^{B} the incremental velocity (from accelerometers). The parameter Δt corresponds to the IMU data sampling rate.

Next, the digital form of the SIMU algorithm, whose continuous form was presented in the previous section, is developed and presented. Given the fact that the development of such digital equations is considered too exhaustive, only the relevant equations are here presented. On the other hand, the methodology adopted and developed algorithm will be

[7] The m cycle corresponds to the rate at which the inertial sensors output the inertial measurements.

presented in the form of a "recipe" in order to make it both easily understandable and implementable. For more details about the algorithm the reader should consult (Savage, 2007).

This "recipe" aims to be a step by step methodology able to be implemented in the eyes of a non-INS analyst, as well as to be able to be implemented with any type of IMU sensors.

As next the proposed "recipe" is presented.

5.3.1 Strapdown IMU Recipe

"Recipe": SIMU

As follows, the step by step methodology associated to the proposed "recipe" for a SIMU implementation is presented.

Ingredients:

After initialization (see Table 5.3 for the initialization process), each IMU data record is processed in post processing mode using the following steps:

- IMU raw data correction
- Attitude Update
- Velocity Update
- Position Update

IMU Raw Data Correction

Equations for the correction of inertial sensors raw data is provided at Part IV, Chapter 14, Equations 14.20 and 14.21.

Extra calculation: Compute Increments

Given the fact that in this work we are working with a low performance inertial unit, some extra needed calculation on the received inertial measurements are needed. Therefore, the parameters that must be first determined are the incremental angles, coning, sculling and rotation increments.

To compute the incremental velocity and rotations of the body frame with respect to the I-Frame expressed in body frame,$(\Delta V_{IB}^{B}\ ,\ \Delta\theta_{IB}^{B})$ we should use (5.14) respectively. Thus, for a single speed algorithm with cycle rate equal to m, the integrated B-Frame angular rate

increments are given by:

$$\underline{\alpha}_{m} = \Delta\underline{\alpha}_{m} = \Delta\underline{\theta}_{IB}^{B} = \int_{t_{m-1}}^{t_m} \omega_{IB}^{B} dt \qquad (5.14)$$

On the other hand, the integrated B-Frame acceleration increments are given by:

$$\underline{\upsilon}_{m} = \Delta\underline{\upsilon}_{m} = \Delta\underline{V}_{IB}^{B} = \int_{t_{m-1}}^{t_m} a_{SF}^{B} dt \qquad (5.15)$$

Where:

$\Delta\underline{\alpha}_m$ Summation of integrated angular rate output increments from angular rate sensors.

$\Delta\underline{\upsilon}_m$ Summation of integrated acceleration output increments from accelerometers sensors.

Taking into account Equations (5.14) and (5.15), coning and sculling, and rotation increments can now be calculated as follows:

Coning increment $\underline{\beta}_m$

$$\underline{\beta}_{m} = (\frac{7}{12}\Delta\underline{\alpha}_{m-1} \times)\Delta\underline{\alpha}_{m} \qquad (5.16)$$

Sculling increment $\Delta\underline{\upsilon}_{Scul_m}$

$$\Delta\underline{\upsilon}_{Scul_m} = \delta\underline{\upsilon}_{Scul_m} \qquad (5.17)$$

Where $\delta\underline{\upsilon}_{Scul_m}$ is given by:

$$\delta\underline{\upsilon}_{Scul_m} = \frac{7}{12}\Delta\underline{\alpha}_{m-1} \times \Delta\underline{\upsilon}_{m} + \frac{7}{12}\Delta\underline{\upsilon}_{m-1} \times \Delta\underline{\alpha}_{m} \qquad (5.18)$$

Rotation increment $\Delta \underline{V}_{\mathrm{rot_m}}$

The first order approximation of the rotation increment is given by (Savage, 2007):

$$\Delta \underline{V}_{\mathrm{rot_m}} \approx \frac{1}{2}\left(\underline{\alpha}_{\mathrm{m}} \times \underline{v}_{\mathrm{m}}\right) \tag{5.19}$$

Table 5.3 Initialization process.

Inputs	Values	Initialization
$\mathbf{\underline{R}^N(0)}$	$l(0)$ and L(0) Set the initial wander angle α(0) = 0	$C_N^E(0)$ (Table 3-1 of Chapter3)
$\mathbf{C_B^L(0)}$	$\emptyset(0), \theta(0), \psi(0)$	$C_B^L(0)$ (Table 3-1 of Chapter3)
$\mathbf{\underline{R}^N(0)}$ and $\mathbf{\underline{v}^N(0)}$	$l(0)$, h(0), α(0), v_X^N, v_Y^N	$\underline{g}_P^N(0)$ (Table 3-9 of Chapter 3) $r_l(0)$ and $r_L(0)$ (Table 3-4 of Chapter 3) Transport rate vector $\omega_{EN}^N(0)$ (Table 3-2 of Chapter 3)
$\mathbf{C_N^E(0)}$	$D_{12}D_{22}D_{32}$ of $\left(C_N^E(0)\right)^T$	ω_{IE}^N (Table 3-2 of Chapter 3)
$\boldsymbol{\omega_{IE}^N}$, $\boldsymbol{\omega_{EN}^N}$	-	$\underline{\omega}_{IL}^L = C_N^L\left(\omega_{IE}^N + \omega_{EN}^N\right)$

The SIMU step by step methodology "recipe" starts here:

Step 1: Attitude Update

According to (5.8), the attitude rate equation is given by:

$$\dot{C}_B^L = C_B^L\left(\underline{\omega}_{IB}^B \times\right) - \left(\underline{\omega}_{IL}^L \times\right)C_B^L \tag{5.20}$$

From (5.20) to update the attitude, one has to account for the angular rotation rate of the body frame, ω_{IB}^{B} relative to I-Frame and also for the angular rotation rate of the wander azimuth L-Frame, ω_{IL}^{L} relative to I-Frame. This is done according to the following equation (Savage, 2007):

$$C_{B_I(m)}^{L_I(m)} = C_{L_I(m-1)}^{L_I(m)} C_{B_I(m-1)}^{L_I(m-1)} C_{B_I(m)}^{B_I(m-1)} \tag{5.21}$$

Where:

$B_I(m), L_I(m)$	Discrete orientation of the B-Frame and L-frame, respectively, in no rotating inertial space at time tm.
$C_{L(m-1)}^{L(m)}$	Direction cosine matrix that accounts for L-Frame rotation relative to inertial space from its attitude at time t_{m-1} to its attitude at time tm.
$C_{B(m)}^{B(m-1)}$	Direction cosine matrix that accounts for B-Frame rotation relative to inertial space from its attitude at time tm to its attitude at time t_{m-1}.
$C_{B_I(m-1)}^{L_I(m-1)}$	DCM relating the body and local-level frames at time t_{m-1}.

Equation (5.21) relates the B-Frame and L-Frame attitudes at the same time. Although it is true that the B-Frame can be rotated dynamically at higher rates (200-300 degrees per second) than the inertial angular rotation rate of the L-Frame – which is generally small and equal to the Earth's rotation plus the transport rate (L-Frame rate relative to the Earth), which is typically never larger than a few Earth rates – it is common that the L-Frame update is generally performed at a lower rate than the B-Frame update. However, this is not the case of this implementation, which is a single speed algorithm taking into account that our UAV is not supposed to fly at high dynamics also.

The B-Frame and L-Frame motion updates to C_B^L are performed by the terms: $C_{B(m)}^{B(m-1)}$ and $C_{L(m-1)}^{L(m)}$, respectively, in the following manner: First, the B-Frame and L-Frame rotation vectors associated to each angular rate rotation are achieved; then the associated DCM are updated. More details about the rotation vector are presented in Appendix A.

B-Frame Rotation Matrix

The update of the attitude in (5.21) accounting for the angular rotation rate of the body frame relative to I-Frame, ω_{IB}^{B}, is performed by the $C_{B(m)}^{B(m-1)}$. In order to achieve the associated DCM, first the respective B-Frame rotation vector is determined using the following equation (Savage, 2007):

$$\underline{\phi}_m = \underline{\alpha}_m + \underline{\beta}_m \tag{5.22}$$

Where $\underline{\alpha}_m$ is the angular rate increments vector calculated at (5.14), and $\underline{\beta}_m$ the coning increment calculated at (5.16); the rotation vector $\underline{\phi}_m$ defines the $B_I(m)$ frame attitude relative to $B_{I_{(m-1)}}$ frame. It should be noted that the coning increments measure the effect of coning motion components present in the gyros measurements (Savage, 2007), where "coning motion" is defined as the condition when an angular rate vector is itself rotating. Important to realize is the fact that when the angular rate vector, ω_{IB}^{B} is not rotating (absence of coning motion), the rotation vector will just be equal to the angular rate increment vector $\underline{\alpha}_m$. Given that, the same will happen if working with a sensor that has a sampling rate not high enough to sense this coning motion and, for this reason, this coning correction is not included in the common strapdown algorithm for the low performance and low rate inertial sensors.

As a result, the B-Frame rotation matrix can be obtained by substituting (5.22) into Equation (A.13) of Appendix A, resulting in:

$$C_{B(m)}^{B(m-1)} = I + \frac{\sin\phi_m}{\phi_m}\left(\underline{\emptyset}_m \times\right) + \frac{1-\cos\phi_m}{(\phi_m)^2}\left(\underline{\phi}_m \times\right)\left(\underline{\phi}_m \times\right) \tag{5.23}$$

L-Frame rotation Matrix

In a similar form as for the B-Frame rotation matrix, first the L-Frame rotation vector is determined using (Savage, 2007):

$$\underline{\zeta}_m \approx \int_{t_{m-1}}^{t_m} \underline{\omega}_{IL}^{L} dt \tag{5.24}$$

Where the L-Frame rate vector $\underline{\omega}_{IL}^{L}$ (see Table 5-2) is given by:

$$\underline{\omega}_{IL}^{L} = \underline{\omega}_{IE}^{L} + \underline{\omega}_{EL}^{L} = C_{N}^{L}(\underline{\omega}_{IE}^{N} + \underline{\omega}_{EN}^{N}) \tag{5.25}$$

Using (5.25) in (5.24) we obtain the L-Frame rotation vector as:

$$\underline{\zeta}_{m} \approx C_{N}^{L}\left[\underline{\omega}_{IE_{m-1/2}}^{N}\,\Delta t_{m} + \rho_{Z_{m-1/2}}^{N}\underline{u}_{ZN}^{N}\,\Delta t_{m} + F_{C_{m-\frac{1}{2}}}^{N}(\underline{u}_{ZN}^{N} \times \sum \Delta\underline{R}_{m}^{N})\right] \tag{5.26}$$

Using Equation (A.12) of Appendix A, the L-Frame rotation matrix is given by:

$$C_{L(m)}^{L(m-1)} = I - \frac{\sin\zeta_{m}}{\zeta_{m}}\left(\underline{\zeta}_{m}\times\right) + \frac{1-\cos\zeta_{m}}{(\zeta_{m})^{2}}(\zeta_{m}\times)\left(\underline{\zeta}_{m}\times\right) \tag{5.27}$$

Finally, applying (5.27), (5.23) and $C_{B(m-1)}^{L(m-1)}$ in (5.21), we obtain the attitude update. Important to realize is the fact that the first attitude update obtained is achieve with the $C_{B(m-1)}^{L(m-1)} = C_{B(0)}^{L(0)}$ which is obtained from the initialization phase.

Normalization and Orthogonalization

After the attitude update matrix has been updated, we have to proceed to its normalization and orthogonalization corrections. This is done according to the following equation:

$$\hat{C}_{B_{+}}^{L} = (I - E_{Sym})\hat{C}_{B_{-}}^{L} \tag{5.28}$$

With E_{Sym} matrix calculated as follows:

$$E_{Sym} = \frac{1}{2}\left(\hat{C}_{B}^{L}(\hat{C}_{B}^{L})^{T} - I\right) \tag{5.29}$$

Where:

$\widehat{\ }$	Designation for parameter calculated by the strapdown algorithm here developed, hence, containing error.
E_{Sym}	Error in the rows of $\hat{C}_B^L$ characterized by symmetry about the $\widehat{C}_B^L$ diagonal. For more details consult (Savage, 2007).
$\hat{C}_{B_+}^L$	The rows of $\hat{C}_B^L$ matrix are orthogonalized and normalized (columns will consequently also be orthogonalized and normalized).
$\hat{C}_{B_+}^L$	Direction cosine matrix before corrections are applied (the one directly obtained at point 5).

Step 2: Velocity Update

The N-Frame velocity update equation presented in the previous section is given by:

$$\underline{v}_m^N = v_{m-1}^N + C_L^N \Delta \underline{v}_{SF_m}^{\ L} + \Delta \underline{v}_{grav/cor_m}^N - e_{vc1} \underline{u}_{ZN}^N \Delta t \quad (5.30)$$

Where:

$\underline{v}_{m-1}^N$	Velocity vector at the previous epoch.
C_L^N	DCM form L-Frame to the N-Frame (see Table 5.2).
$\Delta \underline{v}_{SF_m}^{\ L}$	Velocity increment due to the specific force calculated using Equation (5.31).
$\Delta \underline{V}_{grav/cor_m}^N$	Correction for Coriolis force and gravity effects using Equation (5.33).
$\underline{u}_{ZN}^N$	$[0 \quad 0 \quad 1]^T$ (See Chapter 3 for more details).

With:

$$\Delta \underline{v}_{SF_m}^{\ L} \approx C_{B_{m-1}}^{L_{m-1}} \Delta \underline{v}_{SF_m}^{\ B_{m-1}} \quad (5.31)$$

$$\Delta \underline{v}_{SF_m}^{\ B_{m-1}} = \underline{v}_m + \Delta \underline{V}_{rot_m} + \Delta \underline{\upsilon}_{Scul_m} \quad (5.32)$$

The rotation and sculling increments are obtained from (5.19) and (5.17) respectively. It is considered important to account for the rotational and sculling motion given the fact that it is not possible to integrate angular rate and linear acceleration concurrently in this digital algorithm implementation (Savage, 2007). The $C_{B_{m-1}}^{L_{m-1}}$ matrix is obtained from the attitude update algorithm, at the previous cycle. The correction for the gravity and Coriolis can be computed as follows:

$$\Delta \underline{v}_{grav/cor_m}^{N} \approx \left\{ \underline{g}_{p}^{N}{}_{m-1/2} - \left[2\underline{\omega}_{IE}^{N}{}_{m-\frac{1}{2}} + \rho_{ZN}{}_{m-\frac{1}{2}} \underline{u}_{ZN}^{N} + F_{C}^{N}{}_{m-\frac{1}{2}} (\underline{u}_{ZN}^{N} \times \underline{v}_{m-\frac{1}{2}}) \right] \times \underline{v}_{m-\frac{1}{2}} \right\} \Delta t_m \tag{5.33}$$

Where the subscript '$m-\frac{1}{2}$' denotes values at the midway in the interval $[t_{m-1}, t_m]$ which will be for all parameters given by (remember that we have a single speed algorithm):

$$(.)_{m-\frac{1}{2}} = \frac{3}{2}()_{m-1} - \frac{1}{2}()_{m-2} \tag{5.34}$$

It should be noted that an IMU is intrinsically unstable in the altitude channel and so to limit altitude errors, an independent altitude reference is used in order to implement the vertical control algorithm proposed at the previous section. In fact, the error control will be done using height information provided by the GPS.

Step 3: Position Update

The digital integration algorithm for propagating the position states is made separately for the different position states. The algorithm is divided into two parts: a trapezoidal integration is used to achieve the altitude and then the latitude and longitude are based in the DCM update. In fact, the position is commonly represented as altitude above the Earth's surface plus angular position over the Earth´s surface, and then it can be described by the third column of the C_N^E matrix, which represents the projection of a locally vertical unit vector on the E-Frame axes. Starting with the altitude update, we have:

Altitude Update

First by trapezoidal integration we get the N-Frame position increment:

$$\Delta \underline{R}_m^N = \int_{t_{m-1}}^{t_m} \underline{v}^N dt \equiv \frac{1}{2}\left(\underline{v}_m^N + \underline{v}_{m-1}^N\right)\Delta t \tag{5.35}$$

The altitude will be updated using the numerical integration of the velocity vector´s vertical components as follows:

$$\begin{aligned} h_m &= h_{m-1} + \int_{t_{m-1}}^{t_m} \underline{u}_{ZN}^N \cdot \underline{v}^N dt \\ &= h_{m-1} + \underline{u}_{ZN}^N \cdot \Delta \underline{R}_m^N \end{aligned} \tag{5.36}$$

Adding the vertical channel control algorithm presented at the previous section, the final altitude, after correction, will be:

$$h_{m_+} = h_{m_-} - e_{vc2_m} \Delta t_m \tag{5.37}$$

Where:

-, +	Indicates the value for h_m before and after the vertical stabilization addition, respectively.
e_{vc2_m}	Altitude control signal calculated according to (5.11).

Position Update

Position update is achieved by updating the position direction cosine matrix from E-Frame to the N-Frame as follows:

$$C_{NE_m}^E = C_{NE_{m-1}}^E C_{NE_m}^{NE_{m-1}} \tag{5.38}$$

Again, the direction cosine matrix is updated using the relevant rotation vector. Thus, the position rotation vector defining the N-Frame attitude at time t_m relative to E-Frame attitude at time t_{m-1} is given by (Savage, 2007):

$$\begin{aligned}\underline{\xi}_{\mathrm{m}} &= \int_{\mathrm{t_{m-1}}}^{\mathrm{t_m}} \underline{\rho}^{\mathrm{N}} \mathrm{dt} \\ &= \rho^{\mathrm{N}}_{\mathrm{Z_{m-1/2}}} \mathrm{u}^{\mathrm{N}}_{\mathrm{ZN}} \Delta \mathrm{t_m} + \mathrm{F}^{\mathrm{N}}_{\mathrm{C_{m-\frac{1}{2}}}} (\mathrm{u}^{\mathrm{N}}_{\mathrm{ZN}} \times \sum \Delta \mathrm{R}^{\mathrm{N}}_{\mathrm{m}})\end{aligned} \tag{5.39}$$

Therefore, the position rotation change matrix is obtained from ξ_m as follows:

$$\mathrm{C}^{\mathrm{NE_{m-1}}}_{\mathrm{N_{Em}}} \approx \mathrm{I} + \left(\underline{\xi} \quad \times\right) + \frac{1}{2}\left(\underline{\xi}_m \times\right)\left(\underline{\xi}_m \times\right) \tag{5.40}$$

Applying (5.40) in (5.38), we get the updated position. Note that here $\mathrm{C}^{\mathrm{E}}_{\mathrm{NE_{m-1}}}$ represents the anterior (previous cycle) DCM position matrix.

Step 4: Output Data

Whenever the output data is required, one prepares the output data according to Section 5.4. The desired and implemented outputs of the system are the curvilinear coordinates$(l, \mathrm{L}, \mathrm{h})$; Earth referenced velocities$(\mathrm{v^N}, \mathrm{v^E}, \mathrm{v^D})$; attitude angle*s* (ϕ, θ, ψ) and wander azimuth angle (α).

Step 5: Compute/Update Variables

The variables initialized at the beginning of the navigation, presented in Table 5-3, third column, must now be update in order to be used in the next cycle.

Step 6: Correct IMU Data and Return to Step 1

5.4 Attitude, Velocity and Position Output

Although the two wander azimuth frames, N and L, are used for the implementation of the mechanization equations, the navigation parameters are transformed from the N-Frame to the NED and GEO frame according to Section 5.2, as outputs of the system. Therefore, the outputs of the strapdown navigator here presented are: geodetic coordinates$(l, \mathrm{L}, \mathrm{h})$; Earth referenced velocities $(\mathrm{v^E}, \mathrm{v^N}, \mathrm{v^U})$ and/or $(\mathrm{v^N}, \mathrm{v^E}, \mathrm{v^D})$, and attitude angles of the body frame with respect to the L-Frame, Euler angles(ϕ, θ, ψ).

5.4.1 Attitude Output

The typical output information that is required for attitude is the Euler angles which can be extracted from the C_B^L by application of the Equation (A.19) of Annex A. Important to realize is the fact that the C_B^L matrix is related to the L-Frame free azimuth mode. This implies that the obtained heading is not true heading, i.e., heading relative to the true North. Due to the fact that heading is also typically required in relation to true North for output purposes, it can be calculated as follows:

$$\psi_{True} = \psi_{Platform} - \alpha \tag{5.41}$$

Important also is the fact that if the L-Frame North pointing situation, considered, α is set to zero in (5.41) then the true heading angle is directly obtained: $[\phi\ \theta\ \psi]^T$.

5.4.2 Velocity Output

The velocity components in the N-Frame have no sense for navigation output. In fact, they are just used for acceleration integration and then transformed to another coordinate frame for output purposes. One of the possibilities for output might be along the locally level East (E), North (N), and Upward (U), the so called ENU coordinates denoted as the GEO-Frame, whose relation with the N-Frame is given by the rotation matrix presented at Chapter 3 and here remembered:

$$C_N^{GEO} = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.42}$$

Therefore, in order to get the velocity components in the GEO-Frame, we just have to calculate it as follows:

$$\underline{v}^{GEO} = \begin{bmatrix} v_E \\ v_N \\ v_U \end{bmatrix} = C_N^{GEO}\underline{v}^N \tag{5.43}$$

If the L-Frame North pointing is desired, we just have to use the C_{GEO}^{L}matrix in the following manner:

$$\underline{v}^{NED} = \begin{bmatrix} v_N \\ v_E \\ v_D \end{bmatrix} = C_{GEO}^{L}\underline{v}^{GEO} \tag{5.44}$$

5.4.3 Position Output

Latitude, longitude, altitude, and wander angle are the typical outputs from position determination. Latitude, longitude and wander angle are calculated using Equation (A.30) of Annex A. Altitude is directly calculated using the velocity as shown in Equation (5.43), for instance.

CHAPTER 6

STRAPDOWN INERTIAL NAVIGATION ERROR DYNAMIC EQUATIONS

6 Strapdown Inertial Navigation Error Dynamic Equations

Due to the uncertainties presented in the used sensors as well as in the gravity model being used, the truth is that the navigation parameters obtained at the navigator´s output (SIMU solution) contain errors. As a matter of fact, many models have been developed to describe the time dependent behaviour of such errors, and the choice of which one to use is mainly dependent of the application in cause. Which method to select is a trade-off involving considerations of vehicle operating environment, computer capability, and accuracy requirements.

The behaviours of the errors in inertial navigation system can be modelled with error propagation models. The propagate model estimates the total error in the system as a result of different kinds of errors in the sensors or the initial alignment phase. Errors models are developed by disturbing the nominal differential equations presented at the chapter strapdown navigator, which have been optimized at the previous section.

Therefore, in this chapter, the time rate differential equations describing the propagation of position, velocity and attitude errors in the strapdown navigator implemented at the previous chapter are derived. Important to realize is that the objective of those equations is not to provide information about the errors of the IMU sensor, but instead to illustrate how errors propagate in time. On the other hand, if we want to know about those errors, i.e., and we want to estimate them in order to improve de performance of the overall system, they need to be estimated in a KF, for instance.

In order to define new equations related to the errors of the navigation parameters output by Equation (6.1), first the continuous form of the selected navigation equations are here remembered.

$$\left[\dot{\underline{x}}^{N/L}\right] = \begin{bmatrix} \dot{\underline{R}}^{N} \\ \dot{\underline{v}}^{N} \\ \dot{C}_{B}^{L} \end{bmatrix} = \begin{bmatrix} \underline{R}_{N}^{-1}\underline{v}^{N} \\ C_{B}^{N}\underline{a}_{SF}^{B} + \underline{g}_{p}^{N} - \left(2\underline{\omega}_{IE}^{N} + \underline{\omega}_{EN}^{N}\right) \times \underline{v}^{N} \\ C_{B}^{L}\left(\underline{\omega}_{IB}^{B} \times\right) - \left(\underline{\omega}_{IL}^{L} \times\right)C_{B}^{L} \end{bmatrix} \quad (6.1)$$

According to Equation (6.1), it is possible to observe that the mechanization equations are a function of input sensor measurements $\underline{a}_{SF}^{B}$ and $\underline{\omega}_{IB}^{B}$. As a consequence, it is clear that sensor raw data errors and computational errors, for instance present in the gravity model utilized and presented at Chapter 3, will affect the accuracy of the navigation solution provided by (6.1).

Therefore, as next, we developed the rate equations for position, velocity and attitude errors as a function of input sensor and gravity model errors (Savage, 2007).

In order to achieve that the following steps must be taken:

- It is necessary to refine basic navigation equations developed at Chapter 5 for easy error equation development. Section 6.1;
- Linearize some earth related parameters and write navigation equations in N-Frame. Section 6.2;
- Define attitude, velocity and position error parameters and show their relationship to navigation parameters. Section 6.3;
- Develop a select the error parameter rate equations. Navigation system error equations. Section 6.4;
- Model inertial sensor errors. Sensor error terms. Part IV Chapter 14.

6.1 Refinement of the Earth related parameters

For the error analysis, it is advantageous to recognize that the Earth related parameter (e, J_2, J_3, etc…) equations provided from Table 3-3 Chapter 3, are small compared to the dominant Earth mass attraction term, and also that the operating altitude for the navigation system is generally small compared to the radius of the Earth, R_0 (Savage, 2007).

Therefore, linearization has been applied to the parameters containing those terms. As follows, a summary of the refinement equations for the earth related parameters from Savage (Savage, 2007) is proposed:

$$R'_S \approx \left(1 + u^2_{ZN_{YE}} e\right) R_0 \tag{6.2}$$

$$R_S \approx \left(1 - u^2_{ZN_{YE}} e\right) R_0, R \approx R_S + h \tag{6.3}$$

$$r_{ls} = \left[1 + 2\left(2u^2_{ZN_{YE}} - 1\right)e\right] R_S \tag{6.4}$$

$$\partial G^N_C = -\frac{2e}{(1 + h/R'_S)} \begin{bmatrix} D^2_{21} & D_{21}D_{22} & 0 \\ D_{21}D_{22} & D^2_{22} & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{6.5}$$

$$\omega^N_{EN} = \rho_{ZN} \underline{u}^N_{ZN} + \frac{1}{r_l}\left(\underline{u}^N_{ZN} \times \underline{v}^N\right) + \frac{1}{r_l} \partial G^N_C \left(\underline{u}^N_{ZN} \times \underline{v}^N\right) \tag{6.6}$$

The new plum-bob gravity equation is:

$$\underline{g}^N_p = -H(R)\underline{u}^N_{ZN} + \partial gp_{up} \underline{u}^N_{ZN} + \partial gp_{North} C^N_E \underline{u}^E_{North} \tag{6.7}$$

Where:

$H(R)$	Gravity magnitude parameter that characterizes, the fundamental difference between gravity values above ($h \geq 0$) and below ($h < 0$) the Eart´s surface.
∂g_{Pup}	Small variation in the vertical component of plumb-bob gravity from the nominal spherical Earth uniform density approximation.
∂g_{PNorth}	Small variation in the North component of plumb-bob gravity from the nominal spherical Earth uniform density approximation.

And so:

For $h \geq 0$:

$$\begin{aligned} H(R) &= \frac{\mu}{R^2} \\ \partial gp_{up} &\approx \frac{3}{2} J_2 \frac{\mu}{R^2}\left(\frac{R_0}{R}\right)^2 \left(3u^2_{ZN_{YE}} - 1\right) + \left(1 - u^2_{ZN_{YE}}\right) R\omega^2_e \\ \partial gp_{North} &\approx u_{ZN_{YE}} \sqrt{1 - u^2_{ZN_{YE}}} \left[6\frac{\mu}{R^3_0}(2J_2 - e) - w^2_e\right] h \end{aligned} \tag{6.8}$$

And for h< 0:

$$\begin{aligned} H(R) &= \frac{R}{R_S}\frac{\mu}{R_S^2} \\ \partial gp_{up} &\approx \frac{R}{R_S}\left[\frac{3}{2}J_2\frac{\mu}{R_S^2}\left(\frac{R_0}{R_S}\right)^2\left(3u_{ZN_{YE}}^2-1\right)\right. \\ &\quad \left.+\left(1-u_{ZN_{YE}}^2\right)R_S w_e^2\right] \\ \partial gp_{North} &\approx 0 \end{aligned} \tag{6.9}$$

Where:

$$\underline{u}_{ZN}^{E} = \begin{bmatrix} u_{ZN_{XE}} \\ u_{ZN_{YE}} \\ u_{ZN_{ZE}} \end{bmatrix} C_N^E \underline{u}_{ZN}^{N} = \begin{bmatrix} D_{13} \\ D_{23} \\ D_{33} \end{bmatrix} \tag{6.10}$$

$$\underline{u}_{North}^{E} = \frac{1}{\sqrt{1-u_{ZN_{YE}}^2}} \begin{bmatrix} -u_{ZN_{XE}}u_{ZN_{YE}} \\ 1-u_{ZN_{YE}}^2 \\ -u_{ZN_{ZE}}u_{ZN_{YE}} \end{bmatrix} \tag{6.11}$$

6.2 Navigation Equations for the N-Frame Error Analysis (after linearization)

A summary of the navigation equations for the N frame error analysis after refinement (linearization) of the Earth related parameters is presented. It should be noted that the navigation equations always include the vertical channel control. First, the navigation mechanization equations, derived in Chapter 5, are presented. For more details read (Savage, 2007).

6.2.1 Position Rate Equations

Given the fact that the Position equation can be defined in terms of $\underline{R}^N$or C_N^E along with h separately, two different position rate equations are presented.

$$\dot{\underline{R}} = \underline{R}_N^{-1}\underline{v}^N \tag{6.12}$$

Where:

$$\underline{R}_N^{-1} = \begin{bmatrix} 0 & \frac{1}{r_{ls}+h} & 0 \\ \frac{1}{r_{ls}+h}(1+f_{eh}\cos^2 l) & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.13}$$

Or

$$\dot{C}_N^E = C_N^E(\omega_{EN}^N \times) \tag{6.14}$$

$$\dot{h} = \underline{v}^N.\underline{u}_{ZN}^N - e_{vc2} \tag{6.15}$$

$$\partial h = h - h_{\text{aiding sensor}} \tag{6.16}$$

$$e_{vc1} = e_{vc3} + C_2\,\partial h \;;\; e_{vc2} = C_3\,\partial h\,,\; \dot{e}_{vc3} = C_1\,\partial h \tag{6.17}$$

6.2.2 Velocity Rate Equations

$$\dot{\underline{v}}^N = C_B^N a_{SF}^B + \underline{g}_P^N - \left(\omega_{EN}^N + 2\omega_{IE}^N\right) \times \underline{v}^N - e_{vc1}\underline{u}_{ZN}^N \tag{6.18}$$

$$\underline{u}_{ZN}^N = [0\ 0\ 1]^T \tag{6.19}$$

6.2.3 Attitude Rate Equations

It should be noted that for the error analysis the attitude equation that is used given by (5.8) differs from the one here used by the fact that the L-Frame is eliminated and since the L-Frame is fixed relative to the N-Frame we can write $\omega_{IL} = \omega_{IN}$ and therefore the attitude rate equation is of the form:

$$\dot{C}_B^N = C_B^N(\underline{\omega}_{IB}^B \times) - (\underline{\omega}_{IN}^N \times)C_B^N \tag{6.20}$$

$$\underline{\omega}_{IN}^N = \underline{\omega}_{IE}^N + \underline{\omega}_{EN}^N \tag{6.21}$$

$$\underline{\omega}_{IE}^N = \left(C_N^E\right)^T \underline{\omega}_{IE}^E \tag{6.22}$$

$$\underline{\omega}_{IE}^E = [0 \quad \omega_e \quad 0]^T \tag{6.23}$$

Now, for all equations from (6.12) to (6.23) we will apply the refinement presented at Section 6.1. As a consequence the new parameters after linearization are given by:

rls	Equation (6.4).
f_{eh}	$-\frac{2e}{1+{}^{h}/_{R'_S}}$
ω_{EN}^{N}	Obtained from Equation (6.6). Note that the curvature matrix is different from the one considered in the strapdown navigator.
$\underline{g}_{P}^{N}$	Equation (6.7).

6.3 Navigation Error Parameters

Before presentation of the SIMU error dynamic equations it is first considered relevant to present the navigation error parameters separately. Therefore, this section will define the attitude, velocity and position error parameters and show their relationship to navigation parameters.

In order to achieve the equations related to the errors in DCM, it is important at this point to define two new reference frames: the true (T) frame, which is, in our case, the wander azimuth N-Frame system that represents the true position of the system; and the computed (C) frame, which is also the wander azimuth N-Frame where all the calculation are made taking into account that it is the T-Frame. The designation $\widehat{\ }$ will be used for parameter calculated in the strapdown navigator developed at Chapter 5 that may contain errors, commonly used as the C-Frame. Therefore, the parameter without the $\widehat{\ }$ is defined to be the idealized error free value commonly called value in the T-Frame.

There is also a set of axes that specify the actual platform alignment that differs from the T-Frame by any unknown misalignment errors of the inertial sensors. Note that if there are no misalignment errors, this frame commonly called the platform frame, will be our N-Frame (the considered T-Frame).

Important to realize is that these three reference frames differ by a series of small angle rotations, called rotation angle error vectors which are presented before the definition of navigation error parameters selected for the attitude/velocity/position equations.

6.3.1 Angular Error Parameters (error parameters for the attitude)

Angular error parameters are the error parameters important to consider for the attitude equation.

As it has been seen at the previous section angular data are calculated that relate the B-Frame to the L-Frame $\left(C_B^L\right)$ and the N-Frame to the E-Frame $\left(C_N^E\right)$.

A simple way of relating the B-Frame and E-Frame is:

$$C_B^N = C_L^N C_B^L \tag{6.24}$$

$$C_B^E = C_N^E C_B^N \tag{6.25}$$

The DCM's C_B^E, C_N^E, C_B^N are calculated in the strapdown, and so containing errors being therefore not the true DCM but the computed one denoted respectively as: $\hat{C}_B^E$, $\hat{C}_N^E$, $\hat{C}_B^N$.

As follows, we present the rotation angle error vectors associated with each of those DCM.

Rotation angle error vectors:

The relevant rotation angle error vectors are, according to (Savage, 2007):

$\underline{\gamma}^N$	Rotation angle error vector associated with C_B^N, considering the N-Frame to be misaligned, projected on N-Frame axes.
$\underline{\varepsilon}^N$	Rotation angle error vector associated with C_N^E, considering the N-Frame to be misaligned, projected on N-Frame axes.
$\underline{\psi}^N$	Rotation angle error vector associated with C_B^E, considering the E-Frame to be misaligned, projected on N-Frame axes.
$\underline{\gamma}^N \equiv \underline{\phi}$	Platform angle error: angle error from the true axes to the platform axes.
$\underline{\varepsilon}^N \equiv \delta\underline{\theta}$	Computer angle error: angle error from the true axes to computed axes.
$\underline{\psi}^N \equiv \underline{\psi}$	Angle error from the computed axes to the platform axes.

The three angle error vectors in the N-Frame are related as:

$$\underline{\psi}^{N} = \underline{\gamma}^{N} - \underline{\varepsilon}^{N} \tag{6.26}$$

Or as in the common literature:

$$\underline{\psi} = \underline{\phi} - \delta\underline{\theta} \tag{6.27}$$

As next, the equations for the different errors in the DCM and angle error vectors can be derived.

$\underline{\boldsymbol{\gamma}}^{\mathbf{N}}$

According to reference (Savage, 2007) and considering the N-Frame to be misaligned, the error in the $\hat{C}_B^N$ matrix, projected on N-Frame axes can be achieved as follows:

$$\begin{aligned} \hat{C}_B^N = C_B^{\hat{N}} = C_N^{\hat{N}} C_B^N &\equiv \left[I - \left(\underline{\gamma}^{N} \times\right)\right] C_B^N \\ &= C_B^N - \left(\underline{\gamma}^{N} \times\right) C_B^N \\ &= C_B^N + \delta C_B^N \end{aligned} \tag{6.28}$$

Where the δ term represents a perdurbation from the true value (computed minus truth), and δC_B^N representes the error in $\hat{C}_B^N$ caused by misalignment of the N-Frame, which is given by:

$$\delta C_B^N = -\left(\underline{\gamma}^{N} \times\right) C_B^N \tag{6.29}$$

Therefore the rotation angle error vector associated with C_B^N , considering the N-Frame to be misaligned, projected on the N-Frame axes is given by:

$$\left(\underline{\gamma}^{N} \times\right) = I - \hat{C}_B^N (C_B^N)^T \tag{6.30}$$

$\underline{\boldsymbol{\varepsilon}}^{\mathbf{N}}$

In the same manner, considering the N-Frame to be misaligned, the error in the $\hat{C}_N^E$ matrix, projected on N-Frame axes can be achieved as follows:

$$\begin{aligned}\hat{C}_E^N = C_E^{\hat{N}} = C_N^{\hat{N}} C_E^N &\equiv \left[I - \left(\underline{\varepsilon}^N \times\right)\right] C_E^N \\ &= C_B^N - \left(\underline{\varepsilon}^N \times\right) C_E^N \\ &= C_E^N + \delta C_E^N\end{aligned} \tag{6.31}$$

Where δC_E^N represents the error in $\hat{C}_E^N$ caused by misalignment of the N-Frame given by:

$$\delta C_E^N = -\left(\underline{\varepsilon}^N \times\right) C_E^N \tag{6.32}$$

As a consequence of (6.30) the error in $\hat{C}_N^E$ will be:

$$\delta C_N^E = C_N^E \left(\underline{\varepsilon}^N \times\right) \tag{6.33}$$

Therefore the rotation angle error vector associated with C_N^E , considering the N-Frame to be misaligned, projected on the N-Frame axes is given by:

$$\left(\underline{\varepsilon}^N \times\right) = (C_N^E)^T \hat{C}_N^E - I \tag{6.34}$$

$\underline{\boldsymbol{\psi}}^{\mathbf{N}}$

Finally applying the same procedure to achieve the error in the C_B^E matrix and to achieve the respective rotation angle error vector we have:

$$\begin{aligned}\hat{C}_B^E = C_B^{\hat{E}} = C_{\hat{E}}^E C_B^E &\equiv \left[I - \left(\underline{\psi}^E \times\right)\right] C_B^E \\ &= C_B^E - \left(\underline{\psi}^E \times\right) C_B^E \\ &= C_B^E + \delta C_B^E\end{aligned} \tag{6.35}$$

Where δC_B^E representes the error in $\hat{C}_B^E$ caused by misalignment of the E-Frame given by:

$$\delta C_B^E = -\left(\underline{\psi}^E \times\right) C_B^E \tag{6.36}$$

Therefore the rotation angle error vector associated with C_B^E , considering the E-Frame to be misaligned, projected on the N-Frame axes is given by:

$$\left(\underline{\psi}^E \times\right) = I - \hat{C}_B^E (C_B^E)^T \tag{6.37}$$

6.3.2 Velocity Error Parameters

In this section the velocity error parameters are defined in the N-Frame. Thus according to the definition of error (computed value - truth value) we can say that:

$$\delta\underline{V}^E \equiv \underline{\hat{v}}^E - \underline{v}^E \tag{6.38}$$

$$\delta\underline{v}^N \equiv \underline{\hat{v}}^N - \underline{v}^N \tag{6.39}$$

Where:

$\delta\underline{V}^E$	Error in the velocity relative to the Earth measured in the E-Frame, projected on E-Frame axes.
$\delta\underline{v}^N$	Error in the velocity relative to the Earth measured in the N-Frame, projected on N-Frame axes.
$\underline{\hat{v}}^N$	Velocity vector computed in the strapdown navigator.
$\underline{v}^N$	True velocity vector.

Considering $\delta\underline{V}$ as the error in velocity relative to the Earth measured in the E-Frame, the relation between $\delta\underline{V}$ and $\delta\underline{v}$ is:

$$\delta\underline{V}^N = \delta\underline{v}^N + \underline{\varepsilon}^N \times \underline{v}^N \tag{6.40}$$

Where:

$$\delta\underline{V}^{N} = C_{E}^{N}\delta\underline{V}^{E} \tag{6.41}$$

If the error in velocity relative to the Earth measure in the GEO-Frame is desired, than we have:

$$\delta\underline{V}^{GEO} \equiv \hat{\underline{v}}^{GEO} - \underline{v}^{GEO} \tag{6.42}$$

Where the relation between $\delta\underline{V}$ and $\delta\underline{v}$ is given by:

$$\delta\underline{V}^{GEO} = \delta\underline{v}^{GEO} + \delta\alpha\, \underline{u}_{ZGEO}^{GEO} \times \underline{v}^{GEO} \tag{6.43}$$

Where the term $\delta\alpha$ corresponds to the wander angle error, and $\delta\underline{v}^{GEO}$ is calculated from the N-Frame components using:

$$\delta\underline{v}^{GEO} = C_{N}^{GEO}\delta\underline{v}^{N} \tag{6.44}$$

6.3.3 Position Error Parameters

Position error can be defined in terms of error in $\underline{R}^{E}$or error in the angular position matrix in C_{E}^{N} along with error in h.

The position error if defined as the error in the position vector that describes the system position location relative to the Earth then applying Equation (computed-true) we have (Savage, 2007):

$$\delta\underline{R}^{E} \equiv \hat{\underline{R}}^{E} - \underline{R}^{E} \tag{6.45}$$

Where:

$\underline{R}^{E}$	Position vector from Earth´s centre to the strapdown IMU as described in E-Frame axes.
$\hat{\underline{R}}^{E}$	$\underline{R}^{E}$ computed in the navigator.
$\delta\underline{R}^{E}$	Error in $\hat{\underline{R}}^{E}$ as projected on the E-Frame axes.

Important to realize is that the error in $\underline{\widehat{R}}^{E}$as projected in the N-Frame is given by:

$$\delta\underline{R}^{N} \equiv C_{E}^{N}\ \delta\underline{R}^{E} \tag{6.46}$$

An alternative to position location relative to the Earth can be described by the angular orientation of the N-Frame relative to the E-Frame and the altitude above the Earth´s surface (Savage, 2007). And so using the C_{E}^{N} matrix the angular position error definition is defined by ε^{N} . The associated position errors will be of the form:

$$\delta C_{N}^{E} = C_{N}^{E}\left(\underline{\varepsilon}^{N}\times\right) \tag{6.47}$$

$$\left(\underline{\varepsilon}^{N}\times\right) = \left(\delta C_{N}^{E}\right)^{T}\hat{C}_{N}^{E} - I \tag{6.48}$$

$$\delta h = \hat{h} - h \tag{6.49}$$

Where:

$\underline{\varepsilon}^{N}$	Rotation angle error vector associated with the computed matrix C_{N}^{C} according to figure, which is the same as $\hat{C}_{N}^{E}$.
δh	Error in the navigator computed altitude $\hat{h}$.

The equivalency between the two forms of position errors is given by (Savage, 2007):

$$\delta\underline{R}^{N} = R\left(\underline{\varepsilon}^{N}\times\underline{u}_{ZN}^{N}\right) + \delta h\ \underline{u}_{ZN}^{N} \tag{6.50}$$

Therefore:

$$\begin{aligned}\underline{\varepsilon}^{N} &= \frac{1}{R}\left(\underline{u}_{ZN}^{N}\times\ \delta\underline{R}^{N}\right) + \varepsilon_{ZN}\ \underline{u}_{ZN}^{N}\\ \delta h &= \ \underline{u}_{ZN}^{N}.\,\delta\underline{R}^{N}\end{aligned} \tag{6.51}$$

As a consequence the relationship between $\underline{\varepsilon}^{N}$, $\delta\underline{R}^{N}$ and latitude, longitude, wander angle errors is given by:

From (6.50):

$$\begin{aligned}\delta R_{XN} &= R\,(\delta L cos\mathit{l} cos\alpha\ +\ \delta \mathit{l} sin\alpha)\\ \delta R_{YN} &= -R\,(\delta L cos\mathit{l} sin\alpha -\ \delta \mathit{l} cos\alpha)\\ \delta R_{ZN} &\text{ not related to } (\delta \mathit{l}, \delta L, \delta\alpha)\end{aligned} \tag{6.52}$$

And from (6.51):

$$\begin{aligned}\varepsilon_{XN} &= \delta L cos\mathit{l} sin\alpha\ -\ \delta \mathit{l} cos\alpha\\ \varepsilon_{YN} &= \delta L cos\mathit{l} cos\alpha\ +\ \delta \mathit{l} sin\alpha\\ \varepsilon_{XN} &= \delta L sin\mathit{l} + \delta\alpha\end{aligned} \tag{6.53}$$

6.3.4 Gravity Errors

Using the gravity model from (6.7), the plumb-bob gravity error in the N-Frame is as follows:

$$\begin{aligned}\delta \underline{g}^{N}_{P} &\approx F(h)\frac{g}{R}\,\underline{u}^{N}_{ZN}\delta h +\ \delta \underline{g}^{N}_{Mdl}\\ F(h) &=\ 2\ for\ h \geq 0\\ F(h) &=\ -1\ for\ h < 0\end{aligned} \tag{6.54}$$

Where:

$\delta \underline{g}^{N}_{P}$	Corresponds to the error in the plumb-bob gravity vector $\underline{g}^{N}_{P}$.
g	Magnitude of gravity at position $\underline{R}^{N}$.
$\delta \underline{g}^{N}_{Mdl}$	Modelling error in $\underline{g}^{N}_{P}$ from differences in model from true Earth model, and differences caused by first order approximations.

6.3.5 Transport Rate Error

The transport rate vector error is (Savage, 2007):

$$\delta\omega^{N}_{EN} = \frac{1}{r_{\mathit{l}}}\left(\underline{u}^{N}_{ZN} \times \delta\underline{v}^{N}_{ZN}\right) - \frac{1}{r^{2}_{\mathit{l}}}\left(\underline{u}^{N}_{ZN} \times \underline{v}^{N}\right)\delta h + \delta\rho_{ZN}\,\underline{u}^{N}_{ZN} \tag{6.55}$$

Where $\delta\rho_{ZN}$ will depend on the type of mechanization (see Chapter 3) that has been selected.

6.4 Navigation Error Equations

Finally, this section presents the navigation error equations associated to the SIMU developed at Chapter 5, taking into account all the development made at the previous 3 sections. In order to develop the error model associated to the implemented SIMU navigation equations, there are, in general, two approaches: the formal approach and the direct approach.

In the formal approach, two sets of navigation parameter differential equations are first defined: an idealized error free set and a set of identical structure to the error free set which is implemented in the SIMU computer. It is a well-known fact that the developed and implemented SIMU algorithm contains errors in the navigation parameters, mainly due to sensor raw data errors, and due to initialization errors. Thus, the difference is then taken from the SIMU implemented differential equations (solution provided by the computer) and the equivalent idealized error free equations. The difference is identified as the error equations for the computer navigation parameters. In the differencing operation, error parameter products are dropped as second order, i.e., negligible. The resulting linearized differential error equation set is then reformatted to convert the derived navigation error parameters to the basic error parameters selected for error analysis.

On the other hand, in the direct method, the analytical differential of the theoretical idealized error free navigation parameter differential equation is taken directly to obtain the linearized error parameter differential equations. As with the formal method, the differential error equations are then reformatted to convert the error parameters to the basic error parameters selected for error analysis.

Several error models have been developed to describe the time-dependent behaviours of these errors, the choice of which to adopt is mainly dependent on the application. In any case, it should be noted that it is recommended that the model has to have into account the type of IMU one is working with. In fact, the basic assumption behind the error model discussed so far is that all the attitude errors are small. This is true given the fact that we are working with low performance IMU, where the heading will be completely unknown even in stationary mode, due to the large errors associated with the gyros. It important to realize that we are working with gyros that cannot even feel the Earth's rotation rate. When using such sensors, it is expected that the heading error can grow fast in a very short time.

All things considered, the selection of a particular set of basic error parameters depends on the unique requirements associated to the application being considered. A major consideration is whether the associated error parameter differential equations are to be applied

analytically or by numerical integration techniques, as it the case in this thesis, where a Kalman filter application will be applied.

The common set of navigation error parameters is: $\underline{\psi}^{N}, \delta\underline{V}^{N}, \delta\underline{R}^{N}$. In fact, this set presents a minimum number of components required to describe attitude/velocity/position errors (9); there are no singularities in differential equations and direct equivalencies exist between this set and many other desired forms (Savage, 2007).

Given the definitions for the rotation angular error angles, applying a perturbation to the set of navigation equations of Equations (6.12) to (6.20), the complete set of error models for the SIMU outputs of position, velocity, and attitude are derived and presented as follows (Savage, 2007).

6.4.1 Derivation of Error Equations

Therefore, using the set of parameters: $\underline{\psi}^{N}, \delta\underline{V}^{N}, \delta\underline{R}^{N}$the position, velocity and attitude error with vertical channel control dynamics are described by:

$$\begin{aligned}
\delta\dot{\underline{R}}^{N} &= \delta\underline{V}^{N} - \underline{\omega}_{EN}^{N} \times \delta\underline{R}^{N} - C_3(\delta R - \delta h_{Prsr})\underline{u}_{ZN}^{N} \\
\delta\dot{\underline{V}}^{N} &= C_B^N \delta\underline{a}_{SF}^{B} + \underline{a}_{SF}^{N} \times \underline{\psi}^{N} - \frac{g}{R}\delta\underline{R}_{H}^{N} - \left(2\underline{\omega}_{iE}^{N} + \underline{\omega}_{iN}^{N}\right) \times \delta\underline{V}^{N} + \delta\underline{g}_{Mdl}^{N} \\
&\quad + \left[\left(F(h)\frac{g}{R} - C_2\right)\delta R + C_2\delta h_{Prsr} - \delta e_{VC3}\right]\underline{u}_{ZN}^{N} \\
\dot{\underline{\psi}}^{N} &= -C_B^N \delta\underline{\omega}_{IB}^{B} - \left(\underline{\omega}_{iE}^{N} + \underline{\omega}_{EN}^{N}\right) \times \underline{\psi}^{N}
\end{aligned} \tag{6.56}$$

Considering vertical channel control is considered then we can also split the position and the velocity error rate equations into vertical and horizontal:

$$\begin{aligned}
\delta\dot{\underline{R}}_{H}^{N} + \delta\dot{R}^{N}\underline{u}_{ZN}^{N} &= \delta\underline{V}^{N} - \underline{\omega}_{EN}^{N} \times \delta\underline{R}^{N} - C_3(\delta R - \delta h_{Prsr})\underline{u}_{ZN}^{N} \\
\delta\dot{\underline{V}}^{N} &= C_B^N \delta\underline{a}_{SF}^{B} + \underline{a}_{SF}^{N} \times \underline{\psi}^{N} - \frac{g}{R}\delta\underline{R}_{H}^{N} - \left(2\underline{\omega}_{iE}^{N} + \underline{\omega}_{iN}^{N}\right) \times \delta\underline{V}^{N} + \delta\underline{g}_{Mdl}^{N} \\
&\quad + \left[\left(F(h)\frac{g}{R} - C_2\right)\delta R + C_2\delta h_{Prsr} - \delta e_{VC3}\right]\underline{u}_{ZN}^{N} \\
\dot{\underline{\psi}}^{N} &= -C_B^N \delta\underline{\omega}_{IB}^{B} - \left(\underline{\omega}_{iE}^{N} + \underline{\omega}_{EN}^{N}\right) \times \underline{\psi}^{N}
\end{aligned} \tag{6.57}$$

With:

$$\delta\underline{R}_H^N = \delta\underline{R}^N - \delta R\underline{u}_{ZN}^N$$
$$\delta R = \underline{u}_{ZN}^N \delta\underline{R}^N$$
$$\delta\dot{e}_{VC3} = C_1(\delta R - \delta h_{Prsr})$$
$$F(h) = 2 \text{ for } h \geq 0 \quad F(h) = -1 \text{ for } h < 0$$

Where $\delta\underline{a}_{SF}^B$ corresponds to the accelerometer measurements errors expressed in the B-Frame, and $\delta\underline{\omega}_{IB}^B$ to the Gyros measurements errors expressed in the B-Frame as well.

All in all, using the set of parameters: $\underline{\psi}^N, \delta\underline{V}^N, \delta\underline{R}^N$ the position, velocity and attitude error – without vertical channel control dynamics and considereing F(h) = 2 – are described by:

$$\begin{aligned}
\delta\dot{R}^N &= \delta\underline{V}^N - \underline{\omega}_{EN}^N \times \delta\underline{R}^N \\
\delta\dot{\underline{V}}^N &= C_B^N \delta\underline{a}_{SF}^B + \underline{a}_{SF}^N \times \underline{\psi}^N - \frac{g}{R}\delta\underline{R}_H^N - \left(2\underline{\omega}_{IE}^N + \underline{\omega}_{IN}^N\right) \times \delta\underline{V}^N \\
&\quad + \delta\underline{g}_{Mdl}^N + 2\frac{g}{R}\delta R\underline{u}_{ZN}^N \\
\dot{\underline{\psi}}^N &= -C_B^N \delta\underline{\omega}_{IB}^B - \left(\underline{\omega}_{IE}^N + \underline{\omega}_{EN}^N\right) \times \underline{\psi}^N
\end{aligned} \tag{6.58}$$

As a consequence, the $\boldsymbol{\psi}$ - error model for the N-Frame in matrix form given by:

$$\delta\dot{\underline{x}}^{N} = \begin{bmatrix} \dot{\underline{\psi}}^{N} \\ \delta\dot{\underline{v}}^{N} \\ \delta\dot{\underline{R}}^{N} \end{bmatrix} = \begin{bmatrix} \dot{\psi}_x \\ \dot{\psi}_y \\ \dot{\psi}_z \\ \delta\dot{v}_x \\ \delta\dot{v}_y \\ \delta\dot{v}_z \\ \delta\dot{R}_x \\ \delta\dot{R}_y \\ \delta\dot{R}_z \end{bmatrix} = \begin{bmatrix} F_{\psi\psi} & F_{\psi v} & F_{\psi r} \\ F_{v\psi} & F_{vv} & F_{vr} \\ F_{r\psi} & F_{rv} & F_{rr} \end{bmatrix} . \begin{bmatrix} \underline{\psi}^{N} \\ \delta\underline{v}^{N} \\ \delta\underline{R}^{N} \end{bmatrix} + \begin{bmatrix} 0_{3\times3} \\ I_{3\times3} \\ 0_{3\times3} \end{bmatrix} \delta\underline{g}^{N}_{Mdl} + C^{N}_{B} \begin{bmatrix} \delta\omega^{B}_{IB} \\ \delta a^{B}_{SF} \\ 0_{3\times1} \end{bmatrix} \qquad (6.59)$$

Where:

$$F_{\psi\psi} = \begin{bmatrix} 0 & \sin l\omega_{IE} & -\left(\frac{V^N}{r_l}\sin\alpha + \frac{V^E}{r_L}\cos\alpha + \cos l\cos\alpha\omega_{IE}\right) \\ -\sin l\omega_{IE} & 0 & -\frac{V^N}{r_l}\cos\alpha + \frac{V^E}{r_L}\sin\alpha + \cos l\sin\alpha\omega_{IE} \\ \frac{V^N}{r_l}\sin\alpha + \frac{V^E}{r_L}\cos\alpha + \cos l\cos\alpha\omega_{IE} & \frac{V^N}{r_l}\cos\alpha - \frac{V^E}{r_L}\sin\alpha - \cos l\sin\alpha\omega_{IE} & 0 \end{bmatrix}$$

$F_{\psi v} = [0_{3\times3}]$; $F_{\psi r} = [0_{3\times3}]$; $F_{r\psi} = [0_{3\times1}]$; $F_{rv} = [I_{3\times3}]$

$$F_{rr} = \begin{bmatrix} 0 & 0 & -\left(\frac{V^N}{r_l}\sin\alpha + \frac{V^E}{r_L}\cos\alpha\right) \\ 0 & 0 & -\frac{V^N}{r_l}\cos\alpha + \frac{V^E}{r_L}\sin\alpha \\ \left(\frac{V^N}{r_l}\sin\alpha + \frac{V^E}{r_L}\cos\alpha\right) & \frac{V^N}{r_l}\cos\alpha - \frac{V^E}{r_L}\sin\alpha & 0 \end{bmatrix}; \quad F_{vr} = \begin{bmatrix} -\frac{g}{R} & 0 & 0 \\ 0 & -\frac{g}{R} & 0 \\ 0 & 0 & \frac{2g}{R} \end{bmatrix}$$

$$F_{vv} = \begin{bmatrix} 0 & 2\sin l\omega_{IE} & -\left(\frac{V^N}{r_l}\sin\alpha + \frac{V^E}{r_L}\cos\alpha + 2\cos l\cos\alpha\omega_{IE}\right) \\ -2\sin l\omega_{IE} & 0 & -\frac{V^N}{r_l}\cos\alpha + \frac{V^E}{r_L}\sin\alpha + 2\cos l\sin\alpha\omega_{IE} \\ \frac{V^N}{r_l}\sin\alpha + \frac{V^E}{r_L}\cos\alpha + 2\cos l\cos\alpha\omega_{IE} & \frac{V^N}{r_l}\cos\alpha - \frac{V^E}{r_L}\sin\alpha - 2\cos l\sin\alpha\omega_{IE} & 0 \end{bmatrix}$$

$$F_{v\psi} = \begin{bmatrix} 0 & -a_{SF}^U & a_{SF}^N \\ a_{SF}^U & 0 & -a_{SF}^E \\ -a_{SF}^N & a_{SF}^E & 0 \end{bmatrix}$$

According to Equation (6.60) and considering the N-Frame being the GEO-Frame (that is the N-Frame North pointing, corresponding to the ENU-Frame where $\underline{u}_{ZN}^N = [0\ 0\ 1]$ and $\alpha = 0$) we have for the position, velocity, and attitude errors, the following equations:

Position Error Dynamic ENU-Frame:

$$\delta\dot{\underline{R}}^{N} = \delta\underline{V}^{N} - \underline{\omega}^{N}_{EN} \times \delta\underline{R}^{N} = F_{rv}\delta\underline{V}^{N} - F_{rr}\delta\underline{R}^{N}$$

$$\delta\dot{\underline{R}}^{ENU} = \begin{bmatrix} \delta\dot{l} \\ \delta\dot{L} \\ \delta\dot{h} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \delta\underline{V}^{E} \\ \delta\underline{V}^{N} \\ \delta\underline{V}^{U} \end{bmatrix} - \begin{bmatrix} 0 & 0 & -\frac{V^{E}}{r_{L}} \\ 0 & 0 & -\frac{V^{N}}{r_{l}} \\ \frac{V^{E}}{r_{L}} & \frac{V^{N}}{r_{l}} & 0 \end{bmatrix} \begin{bmatrix} \delta l \\ \delta L \\ \delta h \end{bmatrix} \tag{6.60}$$

Velocity Error Dynamic GEO-ENU-Frame:

$$\delta\dot{\underline{V}}^{N} = C^{N}_{B}\delta\underline{a}^{B}_{SF} + \underline{a}^{N}_{SF} \times \underline{\psi}^{N} - \frac{g}{R}\delta\underline{R}^{N}_{H} - \left(2\underline{\omega}^{N}_{IE} + \underline{\omega}^{N}_{IN}\right) \times \delta\underline{V}^{N} + \delta\underline{g}^{N}_{Mdl} + 2\frac{g}{R}\delta R\underline{u}^{N}_{ZN}$$

$$\delta\dot{\underline{V}}^{ENU} = C^{GEO}_{B}\delta\underline{a}^{B}_{SF} + F_{v\psi}\underline{\psi}^{N} - F_{vv}\delta\underline{V}^{N} + F_{vr}\delta\underline{R}^{N}$$

$$\delta\dot{\underline{V}}^{ENU} = \begin{bmatrix} \delta\dot{V}^{E} \\ \delta\dot{V}^{N} \\ \delta\dot{V}^{U} \end{bmatrix} = C^{ENU}_{B}\delta\underline{a}^{B}_{SF} + \begin{bmatrix} 0 & -a^{U}_{SF} & a^{N}_{SF} \\ a^{U}_{SF} & 0 & -a^{E}_{SF} \\ -a^{N}_{SF} & a^{E}_{SF} & 0 \end{bmatrix} \underline{\psi}^{ENU} - \begin{bmatrix} 0 & 2\sin l\omega_{IE} & -\frac{V^{E}}{r_{L}} + 2\cos l\omega_{IE} \\ -2\sin l\omega_{IE} & 0 & -\frac{V^{N}}{r_{l}} \\ \frac{V^{E}}{r_{L}} + 2\cos l\omega_{IE} & \frac{V^{N}}{r_{l}} & 0 \end{bmatrix} \delta\underline{V}^{ENU} + \begin{bmatrix} -\frac{g}{R} & 0 & 0 \\ 0 & -\frac{g}{R} & 0 \\ 0 & 0 & \frac{2g}{R} \end{bmatrix} \delta\underline{R}^{ENU} \tag{6.61}$$

Attitude Error Dynamic GEO-ENU-Frame:

$$\dot{\underline{\psi}}^{N} = -C_B^N \delta\underline{\omega}_{IB}^B - \left(\underline{\omega}_{IE}^N + \underline{\omega}_{EN}^N\right) \times \underline{\psi}^N$$

$$\dot{\underline{\psi}}^{ENU} = -C_B^{ENU} \delta\underline{\omega}_{IB}^B - F_{\psi\psi}\underline{\psi}^N$$

$$\dot{\underline{\psi}}^{ENU} = -C_B^{ENU} \delta\underline{\omega}_{IB}^B - \begin{bmatrix} 0 & \sin l\omega_{IE} & -\left(\frac{V^E}{r_L} + \cos l\omega_{IE}\right) \\ -\sin l\omega_{IE} & 0 & -\frac{V^N}{r_l} \\ \frac{V^E}{r_L} s\alpha + \cos l\omega_{IE} & \frac{V^N}{r_l} & 0 \end{bmatrix} \underline{\psi}^{ENU} \quad (6.62)$$

According to Equations (6.61), (6.62) and (6.63) it can be stated:

- Position errors depend on velocity and position errors;
- Velocity errors depend on accelerometer biases, attitude errors, velocity and position errors;
- Attitude errors depend on gyros bias (drift) and attitude errors;
- Looking at matrix $F_{v\psi}$ in Equation (6.62) in general a_{SF}^E and a_{SF}^N are, at constant speed, very small when compared to a_{SF}^U whic is close to the local gravity value. As a consequence it is say that we may have a strong coupling between $\delta V^E$and ψ_y and between $\delta V^N$and ψ_x. In the same way we will have a weak coupling between δV^E, or $\delta V^N$and ψ_z;
- When two parameters are strongly coupled that means that any change in one will automatically strongly effect the other one;

As a result if one of those parameters, i.e. error state, is accurately estimated then the other state can also be accurately estimated. In such way, and considering the example, if the velocity error states are accurately estimated, they will not only improve the accuracy of the velocity, but also the accuracy of the computed pitch and roll

6.5 Strapdown Navigator Algorithm Validation

After implementation of the digital inertial navigation algorithm presented at the previous section, in this section, a general navigator algorithm (true trajectory generator) has been developed in order to validate the presented algorithm. The navigation parameter solution generated with the developed algorithm is compared numerically against the equivalent navigation profile parameters using the true trajectory generator. Therefore, a simulation was run to validate the algorithms presented in the "recipe" of Section 6.2. The main objective of this trajectory generator (error free: perfect accelerometers and Gyros raw data) is to confirm that the selected state space model is able to calculate the UAV's state, i.e., the UAV position, velocity and attitude. This is accomplished by comparing $\underline{\mathrm{v}}^{\mathrm{GEO}}, l, \mathrm{L}, \mathrm{h}$ and $\mathrm{C}_{\mathrm{B}}^{\mathrm{L(NED)}}$ simulator outputs whit those from the digital algorithm presented at Section 5.3.

The trajectory generated was a simple flight path with the following characteristics:

- Constant velocity.
- Constant altitude equal to 10 km.
- Spherical and rotating earth effects and gravity included.
- Trajectory developed in the N-Frame coordinates.
- Time duration approximately 10 hours.

Figures 5.1 and 5.2, confront the obtained SIMU latitude and longitude in the N-Frame with the true latitude and longitude simulated using a trajectory generator developed with the presented characteristics. In both figures, the red trajectory corresponds to the true trajectory while the green corresponds to the SIMU solution. Both latitude and longitude are presented in degrees. As it is possible to observe, the SIMU solution shows no difference between the true trajectory and the solution obtained using the developed SIMU.

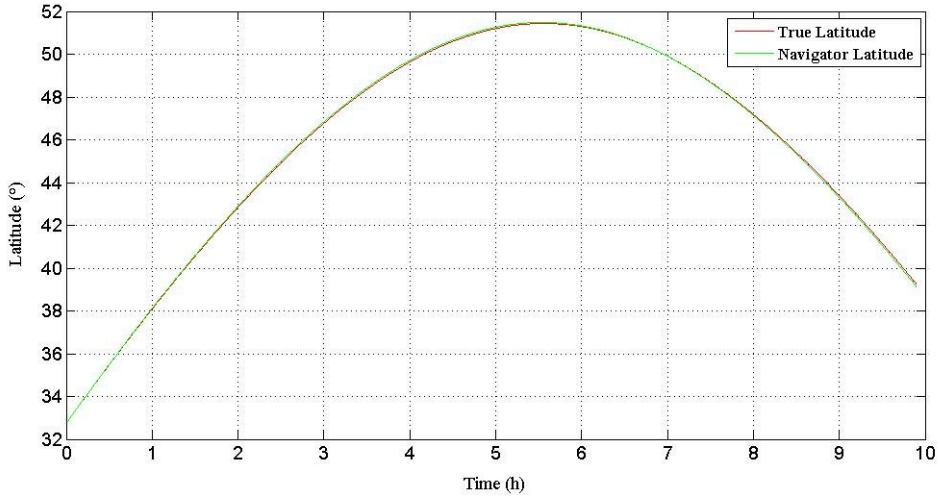

Figure 6.1 Latitude graph from the strapdown navigator given simulated IMU measurements.

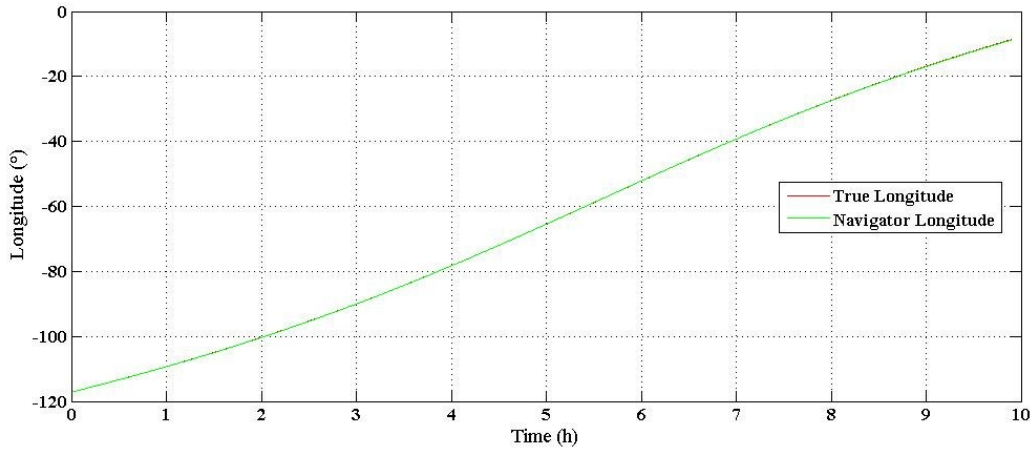

Figure 6.2 Longitude graph from the strapdown navigator given simulated IMU measurements.

Here presented are the graphics relative to the altitude results, without vertical channel control and with vertical channel control.

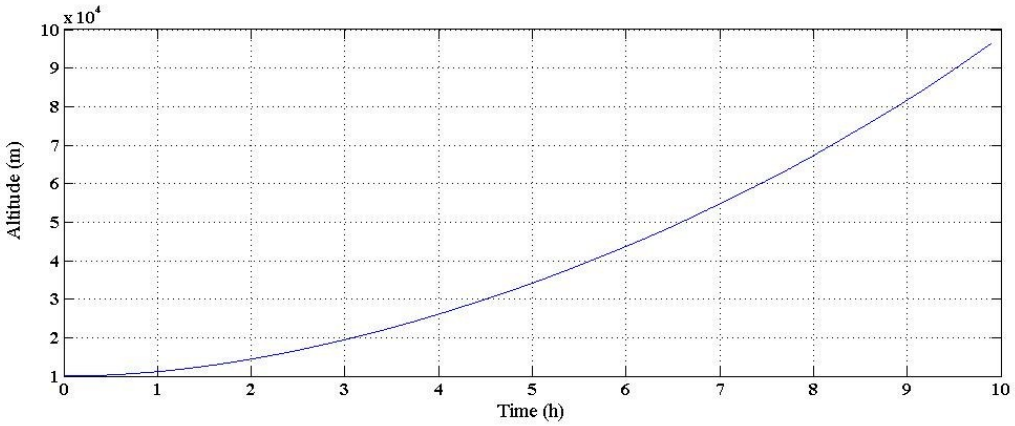

Figure 6.3 Altitude graph without vertical channel control (note that the aircraft is flying at a constant altitude).

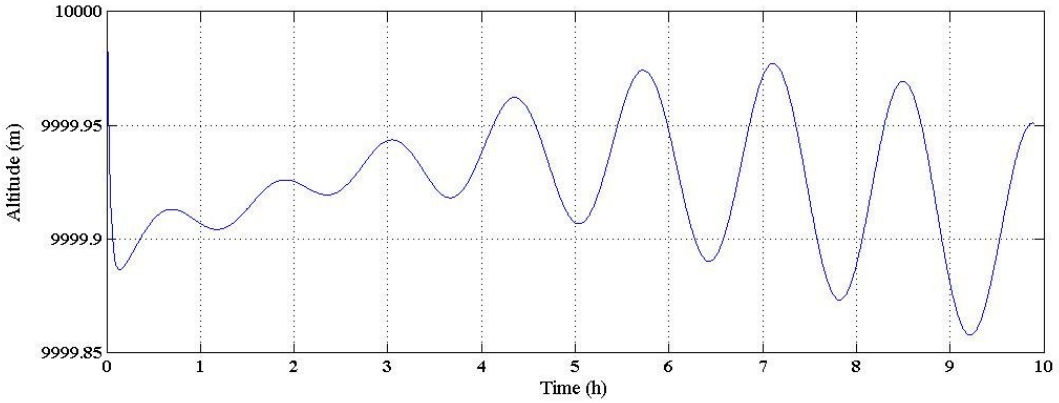

Figure 6.4 Altitude graph with vertical channel control.

Taking into account that the trajectory has been designed to be at constant altitude (10 km), the comparison between graphs 5.3 and 5.4, shows that without vertical control the altitude diverges.

The curves seem to follow the system well enough to justify a successful validation of the implemented SIMU algorithm.

CHAPTER 7

PART II CONCLUSION SUMMARY

7 Part II Conclusion Summary

The **specific objectives** associated to Part II were: *i)* to develop and implement all the strapdown mechanization equations, in several coordinate frames, in order to get the SIMU solution, and *ii)* to develop a SIMU error model appropriate in the low performance sensors world.

The first secondary research question associated to both specific objectives of Part II has been studied in Chapters 5 and 6.

More and more applications using low cost IMU are emerging in the various sectors. However, when working with such units instead of inertial navigation systems, we must be conscious of the need of deriving the navigation algorithm that is responsible for converting the IMU measurements into position, velocity, and attitude. The development and implementation of such navigation equations in a computer is what is called the SIMU.

In Chapter 5, a SIMU was investigated. In this chapter, the general navigation mechanization equations in the continuous form, and in several coordinate frames, taking into account Savage's theory were presented (Savage, 2007). It should be stressed that, although the algorithm takes into account coning and sculling motion, when working with some inertial sensors categories, with low output frequency; it does not make any sense to consider both in the developed algorithm. Anyway, given the fact that the objective was to develop and present a complete "recipe" able to be used with any type of inertial measurement unit, coning and sculling have also been included.

Next, the digital form of the selected mechanization equations is presented in the form of a complete "recipe" able to be directly implemented in a computer. Orthogonalization of the new attitude DCM has also been implemented in order to prevent numerical errors in the attitude update step.

As a result of diverse sources of errors, namely inertial sensors errors, initial alignment data errors, and computational errors, the accuracy of the navigation solution provided by the developed and presented SIMU is limited. As a consequence, the SIMU associated error dynamic equations describing the propagation of those errors is developed and presented at Chapter 6. The reason to develop this error model is due to the fact that it will be required as a model for the Kalman filter to be developed at Part IV. The error model here proposed is the $\underline{\psi}^{N}$error model.

Finally, and in order to validate the proposed SIMU, a trajectory generator able to provide perfect accelerometers and gyros raw data has been developed and implemented. The analysis, presented at Section 6.6, showed that the developed strapdown algorithm was able to follow the reference trajectory to a satisfactory degree, thus validating the implemented navigation equations.

Regarding **Q1** of this study, it can be concluded that, first, the purpose of developing a SIMU is associated to the fact that the IMU does not provide at its output a navigation solution but instead, inertial measurements, i.e., accelerations and angular rates. Therefore, in order to convert those data into navigation parameters, position, velocity and attitude, a set of mechanization equations must be developed and implemented. On the other hand, an easy and complete SIMU "recipe" has been identified, developed, implemented and tested, proving to be easily implemented in a computer, for different categories of IMU sensors, with the proviso that the differences that may exist will be on the correction of the IMU raw data. In fact, more details about the IMU measurement modelling are provided in Part III. Therefore, we can conclude that a generic SIMU algorithm can be provided as a complete and comprehensive "recipe" able to be implemented in any computer being suitable for different categories of IMU sensors if we take into account some specificity related to the inertial measurements models.

In light of the above, we are able to confirm that the specific objectives related to the development and implementation of all the strapdown mechanization equations in several coordinate frames in order to get the SIMU solution, and to the development of a SIMU error model appropriate in the low performance sensors world have been satisfactorily achieved.

Having the first secondary question, **Q1**, of this study been answered, we are now ready to test the first hypothesis of this study.

The **Hypothesis** under investigation in this Part is:

H1 *A trajectory generator, which gives perfect accelerometer and gyro measurements, can be developed to validate a single speed Strapdown Navigator algorithm capable of being a "recipe" able to be used with any type of IMU sensor providing an accurate SIMU solution.*

To achieve the specific objectives of Part II, one secondary research question was defined: **Q1** and **H1** were formulated.

The first secondary research question was answered successfully and so the hypotheses of Part II is confirmed and the results, proving that the mechanization equations digital form were suitable for a computational implementation using the presented "recipe", are graphically presented at Section 7.1.

PART III

LOW- PERFORMANCE SENSOR ANALYSIS AND MMQ 50 IMU MODELLING

"ARMS and the Heroes, who from Lisbon's shore, Thro' seas where sail was never spread before, Beyond where Ceylon lifts her spicy breast, And waves her woods above the wat'ry waste, With prowess more than human forc'd their way To the fair kingdoms of the rising day (...)"

Camões, Lusíadas, Canto I.

CHAPTER 8

INERTIAL SENSORS SIGNALS AND ERRORS

8 Inertial Sensors Signals and Errors

A person who works with raw data, obtained from the real world, knows that signals without noise are unfeasible. So far, inertial sensors signals are not an exception of this fact. Moreover and unfortunately, the noise corrupting the signal provided by low-performance MEMS-based inertial sensors, which is the case of the one used for this study, may increase to such significant levels that for all practical purposes de-noising techniques must be applied in order to recover the signal and proceed with further analysis (Bruton, et al., 1999), (El-Sheimy, et al., 2004), and (Liu, et al., 2007).

In this context, it is considered crucial to know more about inertial sensors signals in both the frequency and time domain in order to make it possible to distinguish between the different types of errors allowing for their easy separation from the true signal.

The second and third secondary research questions of this study are, generally speaking, concerned with whether the common literature provides both easy and comprehensive understandable information about inertial navigation sensors signals and errors and is concerned with the current state of the art related to methods and techniques that can be used to improve inertial raw data. Both questions are a subject of study in this chapter.

8.1 Inertial Sensors Signals

The great variety of technologies applied to inertial navigation sensors make them different from one another, and the wide quality spectrum associated to the inertial sensor market, leads to the need and importance of some type of inertial sensor classification. It should be stated, that up to now there is no universally agreed definition of high, medium, and, low grade IMU. Consequently, frequently, the lack of information on the reliability of the IMU (MEMS-based)

leads to its incorrect application.

Equally important is the fact that, when faced with such a diversity of inertial sensors, it is also quite hard to know which technical data one should analyse and compare before making a decision. Sometimes it can be very difficult, especially for inexpert users of such technology, to make the right decision about which will meet the application requirements the best. Important to realize is also the fact that sometimes many different sensor technologies offer a different range of advantages and disadvantages while offering a similar performance.

To some extent, it is clear that all types of inertial sensors exhibit errors such as bias, scale factor, and noise, among others. The question that arises is – do we really understand what those terms are? The secret, when working with low-performance inertial sensors, is to know more details about the inertial raw data, about the noise component in the sensor measurements. In fact, this is the way to improve the performance of the IMU output data prior to using it as a part of the navigation system.

Given the fact that it was considered difficult to find literature both easy and comprehensively understandable about inertial navigation sensors signals and errors, the next three subsections seek to overcome this gap.

IMU sensors data accuracy is mainly limited by the level of noise present in the respective IMU sensors (Nassar, 2005). According to Figure 8.1, inertial measurements are composed by the true signal plus errors. In detail, those errors can be grouped into deterministic errors and stochastic errors (include both correlated errors and measurement noise), see left side of Figure 8.1 (El-Sheimy, et al., 2004).

Next, after removing the deterministic parameters – according to the calibration data (see Chapter 9 for more details) – from the measurements, the required sensor signal (true motion signal) is buried into a large window of high frequency noise (Bruton, et al., 1999).

Figure 8.1 right side is a conceptual plot of the frequency spectrum of the inertial raw data obtained at the IMU output, which is composed, by the true signal plus errors. Analysing the inertial signal in the frequency domain, it can be stated that the noise affecting that signal contains two components (Skaloud, 1999): the low-frequency component (long-term errors), and the high-frequency component (short-term errors). The low-frequency component is characterized by correlated noise, whilst the high-frequency component has white (uncorrelated) noise characteristics (often called wide-band noise). In the time domain, both errors are combined

affecting the accelerometers/gyros measurements accuracy. Overall, the useful/needed sensor signals, the true signals (specific force and angular rates), are hidden in a combined low and high-frequency noise components.

As follows, more details about that division into long-term and short-term errors are provided:

- Long-term errors (rapidly changing errors) appear at the low-frequency band of the sensor output signal, and can be normally determined after a sufficiently long observation period. Individual error sources that belong to this category would include the main components of the gyro drift, accelerometer bias, scale factors, and errors due to non-perfect initialization (misalignment errors). In the case of low-performance sensors, usually, just gyro drift and accelerometer bias are considered and then modelled in the integration process if wanted, or advantageous. In fact, according to Figure 8.1, the long-term errors are reduced by updating the filter with the information coming from a GNSS system (GPS for example).
- Short-term errors complement the long-term inertial error spectrum up to half of the inertial sensor-sampling rate, see Figure 8.1. The prevailing error sources include errors in quantization, intermittent drifts, scale factor errors, and noise due to vehicle vibrations. It must be stated that this type of errors are not directly observable and would only appear as aliased frequencies in the lower part of the spectrum. However, if they can be suppressed prior to the mechanization of the equations in the SIMU algorithm, the short-term noise aliasing is prevented and the accuracy of an inertial system is expected to improve, especially over short periods (Bruton, et al., 1999). According to the same figure, the short-term errors are reduced by a filtering process (El-Sheimy, et al., 2004), wavelet de-noise techniques for example. Short-term errors depend, certainly, on the performance of the inertial sensors being used.

Finally, both the very-long-term and very-short-term not reduced by those techniques will be included into the residual part, as well as, some residuals coming from the deterministic part. For more details about how to reduce those two components the reader should consult (Bruton, et al., 1999).

Therefore, and taking into account what has been stated and Figure 8.1, if the

deterministic part is accurately removed; if the high amplitude noise can be successfully removed from the sensor measurements; if the correlated noise can be identified and accurately modelled, then the quality of the inertial raw data prior mechanization can surely be improved, and as a consequence the quality of the final estimates will be higher and will depend clearly on the quality of both the models being used in the filters to compensate the low-term errors and the measurements being made to compensate the high-term errors.

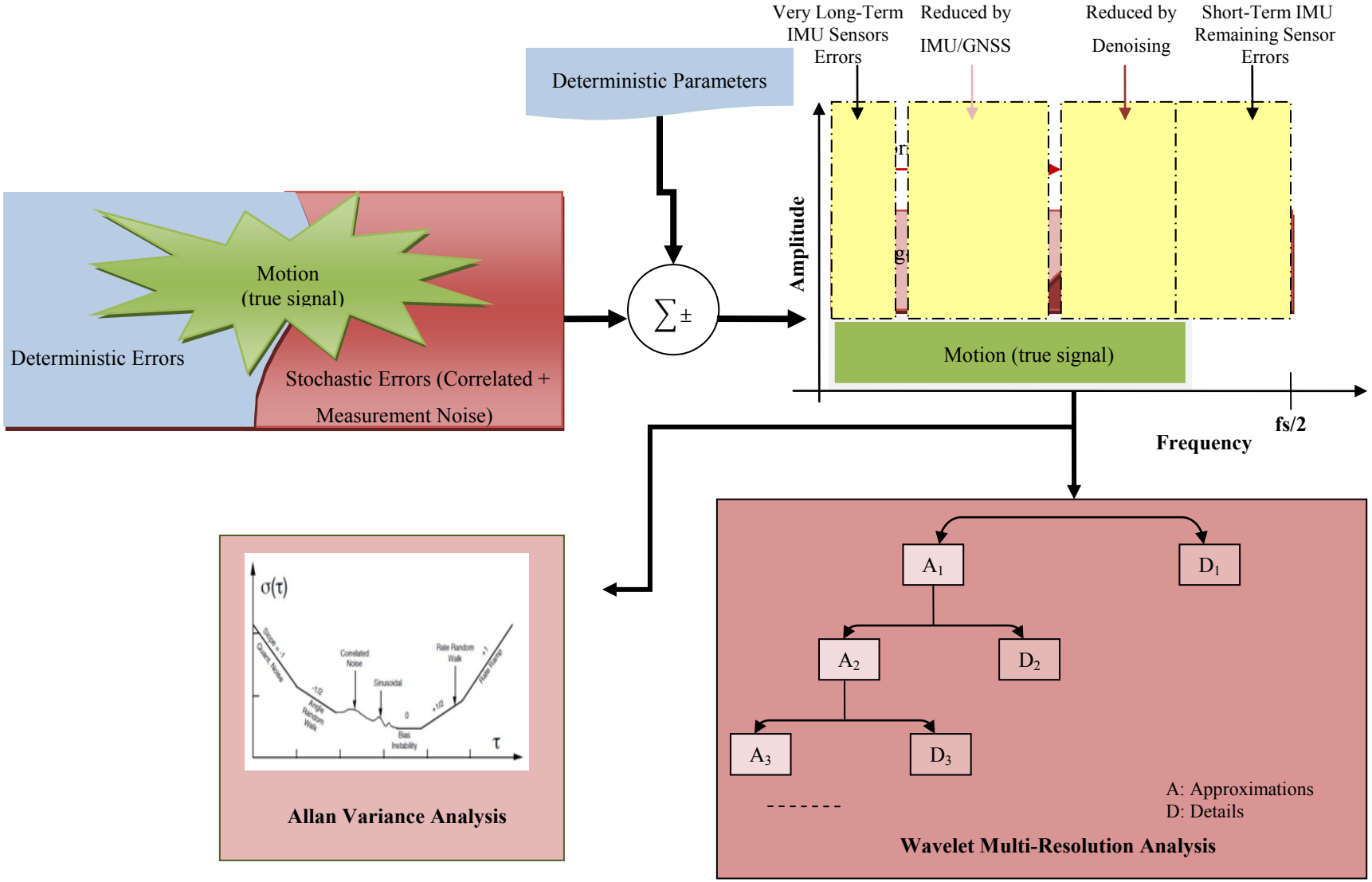

Figure 8.1 Schematic plot of INS signal in the frequency domain and associated wavelet decomposition. Source: Adapted from (Skaloud, 1999).

8.2 Inertial Sensor Errors

It is a well-known that the major error sources in an inertial navigation system are due to inertial sensors imperfections; incorrect navigation alignment (initialization), and imperfections in the SIMU algorithm (mainly due to the gravity model being used). Still, the largest errors are associated with the accelerometer and rate gyro sensors; for both sensors, the major errors are in the measurement outputs (specific force and angular rates, respectively), caused by internal mechanical imperfections, electronics errors or other sources (Groves, 2008).

Inertial sensors contain many possible error sources, with new errors and error models being a popular topic of research. Different grades of inertial navigation units contain different levels of these errors, but in fact, no sensor is immune to any of them.

Due to the complexity and diversity of the inertial sensor instrumental errors, it is hard to present all the deterministic and stochastic errors present in the inertial sensor measurements. Therefore, the definition and classification of deterministic and stochastic errors present in the low-performance class of IMU sensors used by the author of this dissertation throughout his doctoral studies, in accordance with some references as (Groves, 2008), (IEEE STD 647, 2006), (Woodman, 2007), (IEEE STD 647, 2006), and (Ring Laser Gyro, 1983) is given as follows. Important to realize is that all error classifications are adopted by the author due to the fact that there is no universal agreement, according to some of them.

According to what was stated at the previous section, the IMU sensors raw data are contaminated by errors, which can be grouped into deterministic errors and stochastic errors, see Figure 8.2.

Commonly, the deterministic errors include scale factor and misalignment errors; bias offset (constant component part of the bias); non-linearity errors, etc… (Groves, 2008). Stochastic errors, also called random errors, include random bias, random drifts, and quantization noise.

Moreover, an according to (Groves, 2008), each deterministic error source has four components: a fixed contribution, a variation over temperature (temperature component), a run-to-run variation, and an in-run variation. The four components detailed and illustrated in Figure 8.3, are, in this thesis, respectively referenced as the constant component; temperature component; turn-on component (turn-on to turn-on offset), and drift component (time varying).

According to Figure 8.3, it can be stated that inertial sensors measurements errors can be

classified into two categories based on their ability to be determined or modelled: deterministic and random. As illustrated at the same figure, the first three components can be deterministically determined whereas the drift component stochastically modelled due to its random nature in terms of determination. Important to realize is the fact that although the in-run variation (drift component/in-run bias variation) is considered a deterministic error component, because it is a component hard to be model and to be analysed, it will be stochastically determined and so considered as a stochastic correlated noise.

Usually, the IMU processor using laboratory calibration data corrects both the constant and temperature components. The turn-on component results in a contribution to the error source that is different each time the sensor is used remaining, however, constant within the run. Unlike the constant and temperature components, the IMU processor does not compensate this component and so, it must be calibrated in the alignment phase and/or integration algorithms each time the sensor is used. Finally, there is the drift (in-run variation) contribution, which is quite difficult to observe in practice (Groves, 2008). Usually this component is determined by a stochastic process and so compensated through integration with other sensors. It should be noted that the output stability of inertial sensors usually refers to the variation of their output either during the same run (in-run variation, drift component in Figure 8.3), or from turn-on to turn-on (constant component in Figure 8.3). Whereas the in-run variation is treated as a random process, being modelled using different algorithms such as the Gauss-Markov model, random walk, autoregressive (AR) models; the turn-on error is considered a random constant during a single run and, hence, can be expressed by the mean output of the run.

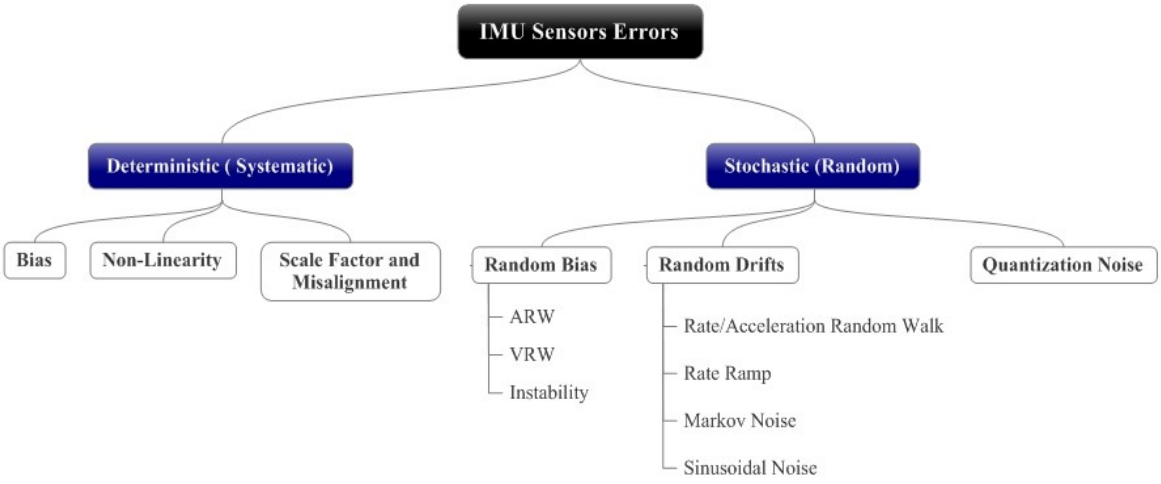

Figure 8.2 General IMU Sensor Errors.

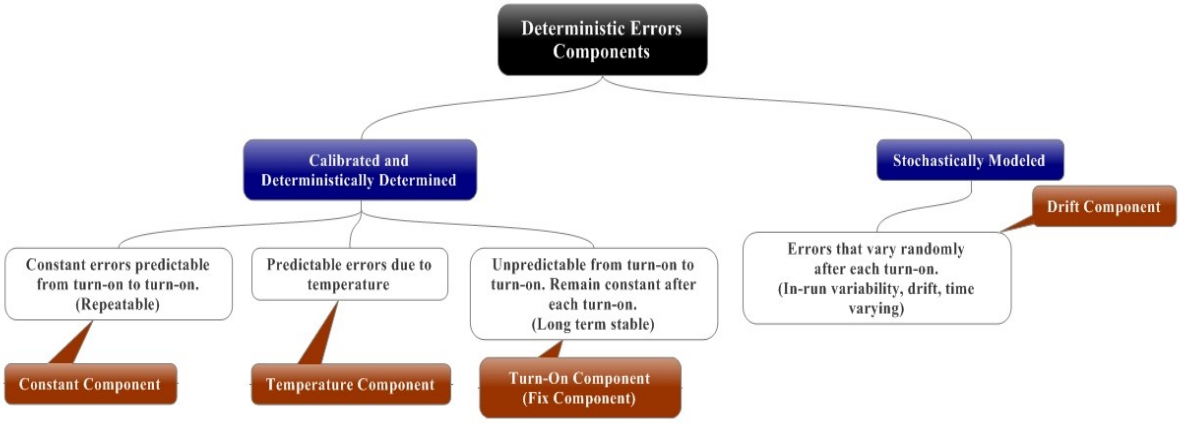

Figure 8.3Inertial sensors errors deterministic errors components.

8.3 Common Inertial Sensors Error Sources

A brief overview of the most important and more common error sources presented in low-performance MEMS based sensors is given hereafter.

8.3.1 Bias

Accelerometer and gyro bias is the dominant error source term in the overall error of an inertial instrument. Ideally, when there is no motion, when the input signal is zero, the output signal of an inertial sensor should be zero. However, what is observed is that this output signal is never zero. This is a consequence of some offset caused by manufacturing imperfections, being called the bias or zero offset. Thus, the bias of an inertial sensor is no more than the output signal that it provides when there is no input (IEEE Std 528, 2001).

According to the previous section, and according to (Groves, 2008), the bias may be divided into four parts, see Figure 8.3. Given the fact that the constant and temperature part are compensated, just the turn-on bias and the in-run bias are considered. The first, usually, comprises the turn-on-to-turn-on variation plus the residual fixed bias (instability), remaining after the sensor calibration. The bias drift (in-run bias variation) refers to the rate at which the error in an inertial sensor accumulates with time and it should be modelled as a stochastic process. More details about those components are given as follows:

- Turn-on Bias (also called turn-on-to-turn-on or offset bias): It is an indication of the stability at some specific temperatures; bias stability remains constant during a run, being the fixed bias component associated at each run. Typical units are m/s^2 for accelerometers and °/h for gyros.
- Bias Drift: The accelerometer or gyro bias is an indication of the increase of acceleration or angular error over time. As time goes by, the error accumulates. Thus, the smaller the accelerometer bias, the better the acceleration solution and the smaller the gyro bias, often called gyro drift, the better the angular solution. Typical units are m/s^2 for accelerometers and °/h for gyros.
- Bias Instability (bias stability) (Noise Floor or Flicker Floor/In-run variation): In general, this is another measure of the bias performance of the accelerometer or gyro; it is the noise in the bias offset. This is the minimum point on the Allan variance (Section 8.6). In general, this is the best stability that is possible to achieve with a fully modelled sensor and active bias estimation (Stockwell, 2010). It represents the lowest possible drift attainable in the system under test. Typical units are m/s^2 for accelerometers and °/h for gyros.

8.3.2 Scale Factor

Both accelerometers and gyroscopes provide an output signal in response to either the acceleration or the rotation of the body where they are installed. Normally, this output signal is analogue, such as a voltage proportional to the input signal, or digital, such as frequency, or a binary number. The scale factor, sometimes referred as sensitivity, is defined as the ratio between the output signal provided by the inertial sensor and the physical quantity to be measured (angular rate and/or acceleration); the scale factor is the ratio between a change in the output signal and the change in the input (IEEE Std 528, 2001). Although, it has been stated before that each error source is composed of four components, most of the scale factor errors contribution are deterministic in nature, and so, they can be determined by calibration and, then, compensated in the navigator. Moreover, usually the specification sheets just refer to the scale factor error defined as:

- Gyro scale factor error: The scale factor error indicates the angular error, which occurs during rotation. Therefore, the more you rotate the more error you accumulate. Units are part per million (ppm).
- Accelerometer scale factor error: It is the meter per second error for every meter per second sensed (the more specific force acceleration you encounter, and the more error you accumulate). Units are also parts per million.

8.3.3 Misalignment

Axes misalignment is an error that results from the imperfect mounting of the sensors. As a result, each axis is affected by the measurements of the other two axes in the B-frame. This error is often modelled in the INS error equation.

8.3.4 Noise

According to what it was stated before, noise can be separated into measurement noise (short-term noise) and correlated noise (long-term noise). The first one includes rate white noise often called angle/velocity random walk, due to white noise in the angular rates and accelerations

respectively; and instability, which is no more than the noise in the calculated deterministic error, source offset (stability part of the considered error source). We have the correlated noise, which includes the rate random walk, rate ramp, exponentially correlated noise, sinusoidal noise. Another noise often considered is the quantization noise. Rate random walk is considered a limiting case of an exponentially correlated noise with a very long correlation time. The presence of this error can be concluded from a slope of ½ in the Allan variance log-log plot.

Important to realize is the fact that correlated noise is normally modelled by some random process such as: random constant, random walk and the first order Gauss-Markov assumption (Gelb, 1974) through the stochastic error identification procedures done with the sensor data (for instance with Allan variance analysis) or in accordance with the manufacturer error sheet specifications.

Next, some considerations about zero mean random errors and random walk are presented.

8.3.4.1 Zero-Mean Random Errors

Zero-mean noise contributions in typical inertial sensors are white noise; correlated random noise; bias instability (bias offset noise); and angle random walk (IEEE STD 647, 2006). Some of these noises are high frequency components at the output of the inertial sensors and some of them are low frequency components. For instance, white noise is a high frequency component. The correlated random noise is, usually, characterized by an exponentially decaying autocorrelation function with a finite correlation time (El-Sheimy, et al., 2004). According to (IEEE STD 647, 2006), this noise appears at the low frequency part of the inertial sensors signal being therefore characterized as a long-term error (see Figure 8.1). The bias instability error is also considered as a long-term error due to its low frequency nature. Likewise, random walk is a long-term component. As a matter of fact and taking into account Figure 8.1, white noise is considered as a short-term noise (measurement noise) that can be reduced applying de-noise techniques to the output signal however, the long-term noise components are not so easy to separate from the signal. Correlated noise, bias instability, and random walk, as well as other errors, are all combined within a very small frequency band. The best that we can do is to analyse the output signal using Allan variance method in order to identify what type of errors are present in our sensor, in order to try to better model it according to its characteristics. Separating the white noise component will enhance the performance of the inertial sensors because it will limit the measurement uncertainties.

8.3.4.2 Random Walk

Gyro and accelerometer outputs, especially MEMS based sensors outputs, are perturbed by some thermo-mechanical noise, which fluctuates at a rate much greater than the sampling rate of the sensor, originating a perturbation in the samples obtained from the gyro (Woodman, 2007). This perturbation is a white noise sequence; the higher this value, the higher the noise measured on the angular rates and consequently on the angles. In fact, white noise is the highest frequency noise (short-term variation in the output) present in low-performance sensors such as the one used in this thesis. Low performance sensors outputs are mostly composed of angle/velocity quantization noise; angle/velocity white noise, and of rate/acceleration white noise.

For instance, angle/velocity white noises are the so-called angle random walk (ARW) for gyros and velocity random walk (VRW) for accelerometers. In general, random walk represents the average error that will occur when integrating the inertial sensors signal. Therefore, it will increase the longer the integration process is, providing a fundamental limitation to any angle measurements or velocity measurements that rely exclusively on the integration of rates or accelerations, respectively. It should be noted that the ARW from a rate gyro is equivalent to white noise in the angular rate outputs. In similar fashion, the integral of white noise in accelerometer outputs would be equivalent to VRW (Grewal, et al., 2001). Units are deg/$\sqrt{h}$ for gyros and mg/$\sqrt{Hz}$ for accelerometers. Sometimes, some manufacturers specify the ARW value as the noise density in deg/$\sqrt{h}/\sqrt{Hz}$. If the second one is provided by dividing it by 60, we get the first one.

For example, considering the MMQ50 IMU that is used in this thesis, the ARW value provided by the manufacturer is of 0.15 deg/$\sqrt{h}$. This means that, after 1 hour, the standard deviation of the orientation error will be 0.15 deg, after 2 hours, it will be $0.15\times\sqrt{2}$ deg, and so on.

Random walk errors are characterized by variances that grow linearly with time and power spectral densities that fall off as $1/f^2$. Usually, a random walk error model has the form:

$$\begin{aligned} \epsilon_k &= \epsilon_{k-1} + w_{k-1} \\ \sigma_k^2 &\stackrel{\text{def}}{=} E\langle \epsilon_k{}^2 \rangle = \sigma_{k-1}^2 + E\langle w_{k-1}^2 \rangle \\ &= \sigma_0^2 + k\, Q_w \\ \text{with, } Q_w &\stackrel{\text{def}}{=} E_k\langle w_k^2 \rangle \end{aligned} \tag{8.1}$$

The value of Q_w will be in units of squared-error per discrete time step Δt. Random walk error sources are usually specified in terms of standard deviations, that is, errors units per square-root of time unit.

8.4 Stochastic Error Reduction, Identification and Modelling

The third research question of this study asks: What is the current state of the art in regard to the appropriate methods/techniques, in the field of low performance inertial navigation sensors applied to low performance inertial sensors signals in the search of a navigation solution with a higher accuracy?

In order to find an answer to this question, it is necessary, first, to answer other questions, namely: Can we remove the high-frequency noise from the measurements, using de-noising techniques? Which technique should we use? Is the navigation solution improved when applying that technique to the inertial measurements prior to using them in the mechanization equations? Which methods can we use in order to identify and model noise terms present in the IMU sensors measurements?

As a matter of fact, inertial measurements being a sequence of data points measured at successive times, spaced at uniform time intervals, are considered time series. Thus, as with all-time series, methods for its analysis maybe divided into two classes: frequency-domain methods and time-domain methods. The former include spectral analysis and recently wavelet analysis; the latter include auto-correlation, cross-correlation, and recently Allan variance analysis.

As in most other analysis, in time series analysis, it is assumed that the data consist of a systematic pattern and random noise (error), which usually makes the pattern difficult to identify; most time series analysis techniques involve some form of filtering out noise in order to make the pattern more salient and consequently to model it if possible.

Therefore, the idea is to figure out which technique can be used for de-noising part of the measurements in order to make more salient the true signal and other errors that, although not possible to be de-noised, can be easier modelled. Wavelet multi-resolution analysis is the technique that has been identified as the one able to be applied to de-noise the sensors measurements. The Allan variance technique has the task to identify the source of the sensor errors in order to develop mathematical models to describe them.

Section 8.5 provides the methodology that have been implemented and used to de-noise the MMQ50 signal, being the results presented at Chapter 10.

The theory and methodology to implement the Allan variance are presented at Section 8.6 and the results obtained when applied to the MMQ50 are presented at Chapter 10.

8.5 Wavelet Multi-Resolution Analysis with Threshold Techniques for de-Noising Low Performance Inertial Sensors Data

In the inertial navigation field, scientists are faced with the problem of recovering a true signal from noisy data. According to Figure 8.1, the amplitude of noise present in the inertial sensors measurements is relatively high, and differing from one sensor to another; however, with low-performance sensors the tendency is to be higher and higher.

Important to remember also is the fact, that low-performance IMU usually uses gyroscopes with noise levels larger than the Earth´s rotation rate, as it is the case of the MMQ50 rate gyros, used along this work (See Figure 2.6 of Chapter 2). Hence, it is extremely important to consider de-noise techniques when working with such category of sensors. According to what was stated at section 8.1, the measurement noise (white noise) cannot be modelled but instead it can be reduced by using signal de-noising techniques. It should be stressed that, in addition to this noise, we have also vehicle motion vibrations that can also be reduced by applying de-noising techniques.

However, we must be careful in selecting and applying de-noising techniques to the inertial data. In fact, if it is believed that we can obtain a good approximation of the original signal by filtering out the high-frequency content of the measurement; but by doing so, we may be at risk of losing relevant information. We cannot forget that the low-frequency content of the

signal is filtered out too.

Recently wavelet de-noising techniques are the state of the art to smooth raw inertial measurements in order to improve the performance of the strapdown algorithm, see for example (Nassar, 2003). The main advantage associated with the use of wavelet analysis is that it allows for the use of long-time wavelet intervals where more precise low-frequency information is needed, and shorter intervals where high-frequency information is required (El-Sheimy, et al., 2004). Therefore, wavelet analysis is capable of revealing aspects of the data that other signal analysis techniques miss; wavelet analysis has the facility of compressing or de-noising a signal without appreciable degradation of the original signal.

Bear in mind that the main objective of this section is not to give details about wavelet analysis but instead to provide the reader with the basic ideas behind this concept when applied to inertial sensors data in order to reduce its short-term errors. The methodology that has been implemented to de-noise the MMQ50 sensors raw data prior to the mechanization equations is here presented.

For more details about this topic, the reader should consult (Graps, 1995), (Bruton, et al., 1999), and (Mallat, 1989).

8.5.1 Methodology

Figure 8.4 represents the structure of the wavelet multi-resolution de-noising method developed, implemented, and then applied to the MMQ50 inertial measurements prior to be feed into the mechanization equations. As the figure shows, the wavelet multi-resolution de-noising algorithm works in the following manner:

Step 1: Wavelet multi-resolution analysis (WMRA): discrete wavelet transform (DWT) is applied to the original signal in order to decompose it. A wavelet base is selected and then the approximation and detail wavelet coefficients are computed for each level of decomposition;

Step 2: Threshold filtering is applied to the details coefficients obtained at step 1;

Step 3: Signal reconstruction using the wavelet de-noised coefficients obtained at step 2;

Step 4: Output signal de-noised is obtained.

Some details associated to each step are discussed afterwards.

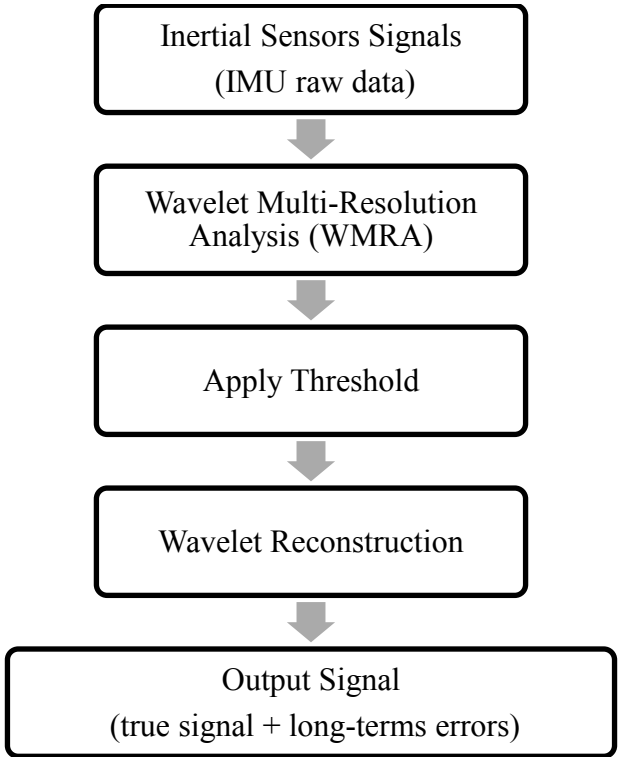

Figure 8.4 Wavelet multi-resolution de-noising method structure.

8.5.1.1 Step 1: WMRA

To transform a signal means to get another way of representing the same signal; the transformation does not change the information content present in that signal, so a wavelet transform does not change it. Consequently, the wavelet transform can be used as a tool to provide a time-frequency representation of the inertial sensors signal being used for its analysis and providing the possibility of de-noising the short-term error inherent to it. In fact, the wavelet transform is acknowledged to give good time resolution and poor frequency resolution at high frequencies, while giving good frequency resolution and poor time resolution at low frequencies. Therefore, with this capability of providing a time resolution representation of a signal, it is possible to use it to analyse the error characteristics of the IMU signal at different frequencies and to localize them in time.

Consider that we apply the discrete wavelet transform (DWT) to a discrete-time sequence x (k) with N samples. According to Figure 8.5, the signal will pass through two filters – a low-pass filter and a high-pass filter – being decomposed into two parts: the first part correspondent to the output of the low-pass filter, called the approximation part; the second part correspondent to the output of the high-pass filter, called the details part. It should be noted that the outputs of the filters represent the DWT approximation and detail coefficients, respectively. However, if the original signal has N samples that mean that each filter will output N samples, as well as, providing a total of 2N samples. To avoid this situation both the output filters are down sampled by 2 to provide approximation and details parts with N/2 samples each one. Applying the DWT successively, i.e., after the first wavelet decomposition the next wavelet decomposition (blue box that include the band-pass filter and down-sampling) will be performed on the approximation part obtained from the previous wavelet decomposition and so on. To obtain finer resolution frequency components of a specific signal; the signal is broken down into many lower-resolution components by repeating the DWT decomposition of the successive obtained approximation parts (El-Sheimy, et al., 2004). This procedure is the so-called multi-resolution analysis; it is known as the Mallat's pyramid algorithm (Mallat, 1989) or as the decomposition algorithm. According to the same figure, it is possible to see that each level of decomposition (LOD) produces a low-frequency component (approximation part) and a high-frequency component (detail part) of the signal. Therefore, as the detail coefficient contains high frequency components, most of the high frequency noise will lie in the detail part whereas the rotation rate and gravity information will lie in the approximation part.

In order to better understand this algorithm, a brief overview of the discrete wavelet transform (DWT) is given and, then, the methodology that has been followed to implement the WMRA algorithm is presented.

According to (Walker, 1999) the DWT of a discrete time real sequence x (k) is defined as:

$$X_{n,m}^{D} = \frac{1}{\sqrt{2^n}} \sum_{k} x(k)\, \psi(k2^{-n} - m) \qquad (8.2)$$

Where:

$X^{D}_{n,m}$	Wavelet coefficients of the signal x(k).
$\psi(k)$	Basis wavelet function utilized in the wavelet transform usually called the "mother" wavelet (ex: Daubechies (db), Haar, and so on).
n, m	Integer numbers for different scaling and shifting version of (k) . The value n is the scaling coefficient, the value m the shifting coefficient. Note that each value of n corresponds to analysing a different resolution level of the signal.

Therefore, the original signal x (k) can be obtained from the corresponding wavelet function using the equation above as follows:

$$x(k) = \sum_{n}\sum_{m} X_{n,m}\,\psi_{n,m}(k) \qquad (8.3)$$

Where:

$\psi_{n,m}(k)$	Scaled and shifted version of $\psi(k) = \frac{1}{\sqrt{2^n}}\psi(2^{-n} - m)$.

Equation (8.2) is called the analysis equation while the wavelet coefficients are computed by passing the signal through two complementary half-band filters. There are many variants of mother wavelets in literature and its selection will depend on the application.

According to Equation (8.2), each value of n corresponds to analysing a different resolution level of the signal.

The methodology followed and associated to Figure 8.5 of WMRA is the following (Bruton, et al., 1999): Based on the DWT equation compute the approximation coefficient $a_{n,m}$ at the nth resolution level using:

$$a_{n,m} = \frac{1}{\sqrt{2^n}}\sum_{k} x(k)\,\phi(2^{-n} - m) \qquad (8.4)$$

Where:

$\phi(k)$ Scaling function, which is similar to wavelet functions except that they have only positive values. It is important to realize that this function operates in a manner equivalent to low pass filtering which rejects high frequency components of the signal.

1: The approximation signal at the correspondent n^{th} resolution level is computed as:

$$x_n(k) = \sum_{K=-\infty}^{\infty} a_{n,m}\phi_{n,m}(k) \qquad (8.5)$$

2: Then, the correspondent detail coefficients $d_{n,m}$, at the same level of resolution, are computed as:

$$d_{n,m} = \sum_{k} x(k)\,\psi_{n,m}(k) \qquad (8.6)$$

Where $\psi_{n,m}(k)$ corresponds to the Wavelet base function.

3: Compute the detail signal as:

$$g_n(k) = \sum_{K=-\infty}^{\infty} d_{n,m}\psi_{n,m}(k) \qquad (8.7)$$

4: Return to step 1 and continue the wavelet decomposition process until an appropriate LOD is reached. It is important to realize that the next wavelet decomposition process is performed using the approximation signal obtained at step 2, instead of the original one.

Regarding the presented methodology, there are some relevant points that must be considered. It is important to realize that given the fact that the maximum frequency of the IMU signal is one-half of the inertial sensors sampling frequency, f_s, (according to the Nyquist

theorem), the filter will have a cut-off frequency of $f_s/4$ (exactly one-half of the maximum frequency of the measurements signal). Consequently, the first wavelet decomposition applied to the original signal will have the approximation part which include the signal components that have frequencies of less than $f_s/4$, whereas the details part will include the components of frequencies between $f_s/4$ and $f_s/2$, and so on.

Another key point is the fact that when several levels of decomposition are done, we have the risk of losing the desired signal (motion), consequently the LOD must be chosen according to the application and nature of the signal (El-Sheimy, et al., 2004). Equally important is the fact that the LOD is different from one IMU to another and different for static and dynamic data. Therefore, for both operation modes (static and dynamic), the selection of an appropriate LOD must be done carefully; the LOD selection must be always based on the objective of removing high frequency noise but keeping all the useful information contained in the static or dynamic data.

As an illustration, consider that we have a sequence of static data that is decomposed using the previous algorithm. What is expected to be observed in the approximation and detail parts? According to the stated before, we expect that the approximation part contains the Earth´s rotation rate or gravity frequencies, as well as, long-term inertial sensor errors, and some highly attenuated short-term noise components (El-Sheimy, et al., 2004). Therefore, since both effects are mixed within a very small frequency band at low frequencies, wavelet decomposition may not be able to separate the Earth´s rotation rate or gravity components from the long-term sensor errors. However, they can be modelled using stochastic processes. Also, we expect that the detail part contains high frequency noise components plus long-term errors, which can be easily separate from the true signal by de-noising techniques, and, consequently, facilitate the selection of an appropriate LOD.

For a sequence of dynamic data, the selection of the appropriate LOD is not so easy. In fact, when applying wavelet decomposition to the dynamic data, the output of the sensors contains both effects of the vehicle´s motion dynamics and sensor noise as well as some other undesirable effects. In this situation, it is not that easy to separate the vehicle's motion from the high frequency noise and so it is recommended (El-Sheimy, et al., 2004) to do a spectral analysis of the used kinematic data, in order to avoid the loss of motion information. Most compelling evidence is the fact that in fixed wings UAV applications, the vehicle motion dynamics are

usually in the low frequency portion of the spectrum and so, by analysing the raw data in the frequency domain, the low frequency range of the actual vehicle motion can be detected. As next, the respective LOD can be selected taking into account this range, assuring that the decomposition process will remove only the components that have frequencies higher than the detected motion frequency range.

Regarding the LOD selection, a simple and easy approach to do it is in post-processing mode; we apply several decomposition levels to the signal and compute the correspondent standard deviation (STD) for each approximation component; when the correspondent STD reaches its minimum value, the LOD is achieved. Another approach consists in computing, at each level, the mean value of the detail wavelet coefficient, and taking into account that white noise has zero mean, then the maximum level of decomposition will correspond to the level up to which mean value of detail coefficient is zero (Samrat, et al., 2009).

Altogether, associated to the fact that the detail part contains high-frequency noise and other disturbances, after several levels of decomposition applied to the signal, the white noise component can be separated and consequently the measurement uncertainty can be reduced. In addition, long-term sensor errors can be accurately modelled after being separated from the white noise component. As a consequence, the de-noising technique explained next (the thresholding technique) is not applied to the approximation part but just to the detail part.

8.5.1.2 Step 2: Threshold Filtering

After the decomposition of the inertial signal into as many levels as possible, we are now able to remove the noise from the high-frequency coefficients (detail part). To clarify, the act of using a threshold on the detail coefficients at each level of decomposition in order to decide which coefficients should passed through is what is called the de-noising process. It is a procedure that aims to reject noise by thresholding the wavelet coefficients of the noisy signal before the signal reconstruction.

Two methods can be used to change those coefficients which are the scaled approximation method, and the wavelet threshold method (Liu, et al., 2007). In the first one, after the signal decomposition to a certain level, the details part is set to zero and the left coefficients are used to reconstruct the signal. In terms of implementation, it can be considered an easy method, and the rebuild signal can keep the original signal's low frequency information

perfectly. In turn, the second method corresponds to the application of a threshold to the DWT coefficients. The signal is first decomposed into as many levels as possible and, then, we let the wavelet coefficients pass a threshold filter where the coefficients with an absolute value below the threshold are set to zero. Finally, the filtered coefficients are used to rebuild the signal.

According to the type of threshold filter that can be used, we consider one of two kinds: the soft and the hard threshold filter. In the case of the soft threshold filter, the coefficients with the magnitude above the threshold are replaced by another value which is equal to the difference between original coefficients and threshold value (T_n), whereas for the hard threshold filter case, the coefficients with an absolute value above the threshold are kept the same. More details about wavelet threshold techniques can be obtained from (Zhong, et al., 2000). Therefore, to de-noise the inertial sensors measurements and after decomposition of the signals (Step 1: WMRA) into approximation and detail wavelet coefficients at each LOD, the following procedure must be done:

1: Compute the noise variance at each level using Matlab facilities and then compute threshold T_n at each level n as:

$$T_n = \sigma_n \sqrt{2 \ln 2^n} \tag{8.8}$$

2: Compute level dependent soft thresholding using:

$$d_n^S = \begin{cases} \text{sgn}(d_n)(|d_n| - T_n) & \text{if } |d_n| \geq T_n \\ 0 & \text{if } |d_n| < T_n \end{cases} \tag{8.9}$$

Where:

sgn(.)	Sign of the detail coefficient.
Tn	Threshold calculated according to Equation (8.8), which is different at each level n.

3: Compute level dependent hard thresholding using:

$$d_n^H = \begin{cases} d_n & \text{if } |d_n| \geq T_n \\ 0 & \text{if } |d_n| < T_n \end{cases} \tag{8.10}$$

Note that S and H in Equations (8.9) and (8.10) denote soft and hard thresholding respectively.

Finally, from the modified detail wavelet coefficient, de-noised signal is reconstructed according to the next subsection step 3.

8.5.1.3 Step 3: Signal Reconstruction

In wavelets, the reconstruction of the signal is obtained by applying the inverse discrete wavelet transform (IDWT) on the previously computed wavelet coefficients. This is usually called the reconstruction or synthesis process.

According to the WMRA description and to Figure 8.5, the original signal can be represented as:

$$\begin{aligned} \text{Signal} &= A_1 + D_1 \\ &= A_2 + D_2 + D_1 \\ &= A_3 + D_3 + D_2 + D_1 \\ &= A_n + D_n + D_{n-1} + \cdots \ldots \ldots + D_1 \end{aligned} \tag{8.11}$$

The denoised signal is reconstructed using Mallat's algorithm (Mallat, 1989). In fact, the reconstruction will be done not using the original detail coefficients but the one that have been modified according to the procedure presented at the previous subsection step 2. Therefore, the de-noised signal is reconstructed using all the details obtained during the decomposition process up to the n^{th} resolution level, and using the approximation at those levels as well. The reconstruction will be done using the following equation:

$$x(k) = \sum_{n=-\infty}^{\infty} a_{n,m}\phi_{n,m}(k) + \sum_{n=-\infty}^{\infty}\sum_{m=-\infty}^{\infty} d_{n,m}\psi_{n,m}(k) \quad (8.12)$$

In the above equation, the first term represents the approximation at level n; the second term represents the details at all resolution levels up to n after the threshold process.

Important to realize is the fact that, for de-noising purposes, the choice of the threshold in the second step is crucial to the quality of the de-noising process and consequently of the rebuilt signal. Moreover, if the scaled method is used instead of the threshold filtering, then the de-noised signal can be obtained by passing the coefficients of the selected approximation level through the indirect discreet wavelet transformation low pass (IDWT LP) filter and resetting the coefficients of all subsequent details to zero before passing them through the IDWT high pass (HP) filters. If thresholding filtering is applied then the reconstruction will be done using the de-noised wavelet coefficients after applied the soft or hard threshold.

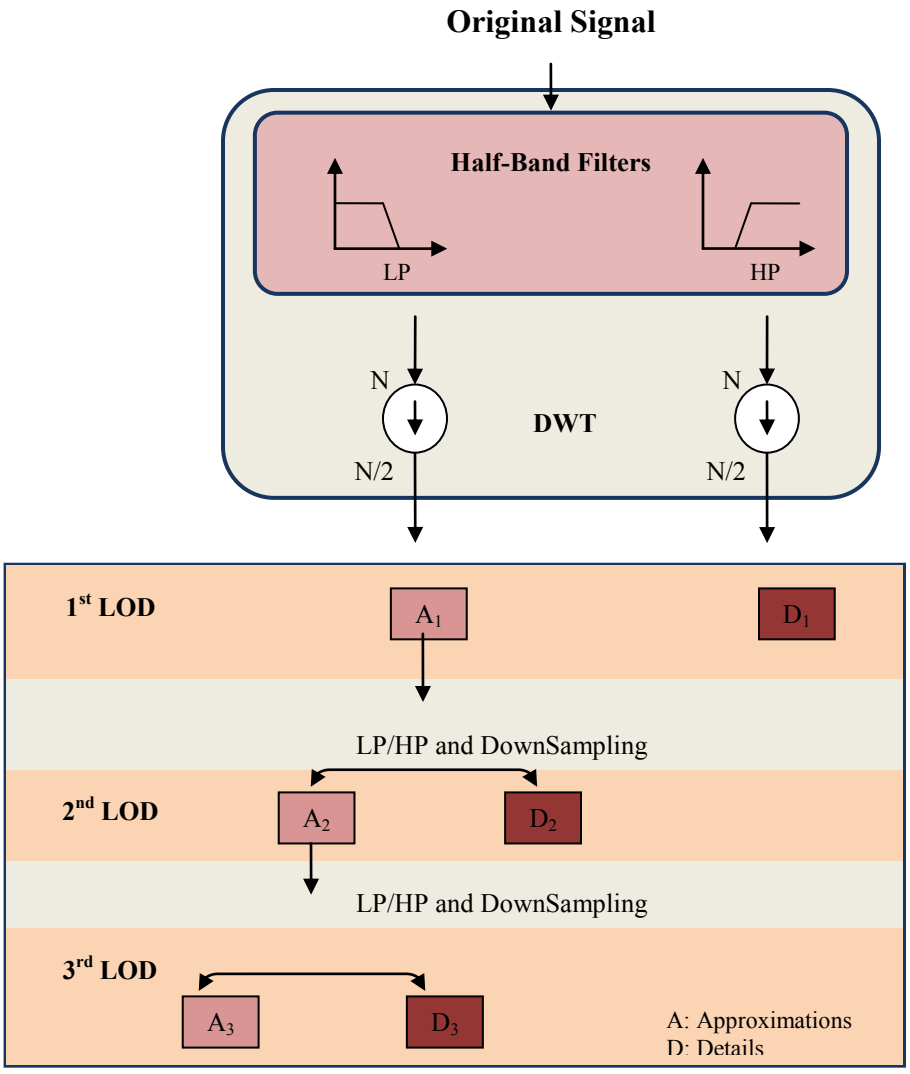

Figure 8.5 Wavelet Multi-Resolution Analysis Diagram.

8.6 Allan Variance as a Tool for Modelling Inertial sensor Long-Terms Errors

The Allan variance method is a time domain analysis technique that has been widely used in studies related to frequency stability of precision oscillators (Allan, 1966). However, given the close similarities between the types of noise presented on both the signals of clock systems and inertial sensors signals, this method has been successfully adapted and accepted as an IEEE

standard for gyro specifications and consequently to the identification of stochastic error sources in inertial sensors (IEEE STD 647, 2006).

One requirement when using Allan variance analysis is to assume that the noise present in the respective signal is generated by noise sources of specific character, see Figure 8.7. The magnitude of each noise source covariance, presented in the data, is then estimated from the data. Consequently, the differential equation descriptions associated with each stochastic error are derived.

In this thesis, the Allan variance method will be used to identify and obtain the variances for most of the random errors underlying in the low-performance sensors of the MMQ50 IMU. As next, the exact algorithm methodology that has been used is presented.

8.6.1 Methodology

The Allan variance method is based on the method of Cluster Sampling Technique (CST) analysis (Tehrani, 1983). In CST, data clusters of specific length are created out of the entire data. For more details in such equations and methodology, readers are referred to (IEEE STD 647, 2006).

According to Figure 8.6, the procedures to be applied to the MMQ50 measurements in order to analyse the different errors underlying the inertial measurements are given afterwards.

Step 1: Assume that there are N samples of data from either accelerometers or gyroscopes taken at a rate of f_s (sensor sampling frequency) samples per second, so the corresponding sampling time will be: $\tau_0 = {}^{1}/_{f_s}$.

Step 2: Form groups named clusters of n data points: τ_0, $2\tau_0$ … $n\tau_0$, with $1 \leq n < \frac{N}{2}$. As a result, we form $K = \frac{N}{n}$ clusters where n corresponds to the number of samples per second.

Step 3: Calculate time averages of the individual data points contained in each cluster over the length of that cluster, using the following expression:

$$\bar{w}_k(n) = \frac{1}{n}\sum_{i=1}^{n} w_{ki,} \quad k = 1, \dots, K \tag{8.13}$$

Step 4: Obtain differences between each two adjoining clusters and then compute the Allan variance from the clusters averages. The Allan variance of the sequence w_k at time step τ_n is defined by:

$$\sigma_{AV}^2(\tau_n) \cong \frac{1}{2(K-1)}\sum_{k=1}^{K-1}(\bar{w}_{k+1}(n) - \bar{w}_k(n))^2 \tag{8.14}$$

If we plot the obtained root Allan variance by applying the proposed algorithm, from step 1 to step 4, to the entire data, we obtain a characteristic curve which must be similar to Figure 8.7. Analysing the curve, it is possible to identify what types and magnitudes of possible error sources residing in the data are being analysed. Indeed, the random errors can be identified according to Table 8.1, and consequently modelled.

The accuracy in the estimate of the Allan deviation (root Allan variance) is given by:

$$\epsilon = \frac{1}{\sqrt{2(K-1)}} \tag{8.15}$$

According to the above equation, it is clear that the confidence of the estimation improves as the number of independent clusters (K) is increased. According to the same equation, it can be stated that the estimation errors in the region of short (long) τ are small (large) as the number of independent clusters in these regions is large (small).

8.6.1.1 Allan Variance Application Example

As an example, consider 2 hour saved data at a rate of 100Hz (sample time of 0.01 s), which provides us a sequence of 720000 data points. Thus, using the algorithm presented the 720000 clusters of size 1 with length equal to 0.01s can be formed and two clusters of size 360000 with length 3600s are formed. For instance, for the shorter cluster (containing just one point) the estimation error obtained is:

$$\epsilon = \frac{1}{\sqrt{2 * (\frac{720000}{1} - 1)}} \approx 0.08\% \tag{8.16}$$

On the other hand, for cluster containing 360000 points the estimation error will be:

$$\epsilon = \frac{1}{\sqrt{2 * (\frac{720000}{1} - 1)}} \approx 70\% \tag{8.17}$$

8.6.2 Allan Standard Deviation Plot

Once plotted the root Allan variance (standard deviation) in logarithmic scale, it is possible to discriminate between different contributing error sources by simply examining the varying slopes on the Allan variance plot.

Figure 8.7 is no more than a typical variance log-log plot of the square root of the Allan variance, $\sigma(\tau)$, versus τ, providing a mean of identifying and quantifying various noise terms that exist in the inertial sensor data under study. As it is possible to observe, different noise terms appear in different regions of τ allowing the easy identification of various random process existent in the sensor data being analysed. Note that Figure 8.7 is an ideal plot, in fact with real data; gradual transitions would exist between the different Allan variances slopes. In addition, a certain amount of noise would exist in the plot curve associated to the uncertainty of measured Allan variance (IEEE Std. 952, 1997).

The common stochastic error sources existing in inertial sensors are five: (1) Quantization noise; (2) Angle/Velocity random walk (A/VRW) or rate white noise (wide band noise); (3) Flicker noise (bias instability/noise floor); (4) Rate/Acceleration random walk, and (5) Rate ramp (IEEE STD 647, 2006).

The stochastic error coefficients associated to those errors are respectively: Q, N, B, K, and R, which can be determined by Allan variance analysis according to Table 8.1. Important to realize is the fact that those five noise terms are the most common. However, sometime other noise terms, such as sinusoidal noise, are also present in the data. Additionally, other noise terms can be present in the data or even can be confounded with one of the five terms presented before

and so the reader should be aware about it.

Typically, all of the mentioned error sources may be presented in the real measurements output by both the inertial sensors. The Allan variance log-log plot for those real measurements will include the combined effects of those existing noise terms, which are in most of the cases statistically independent.

Finally, the determined parameters Q, N, N, K and R can be used to determine the power spectral density (PSD) of the corresponding noises according to Equation (8.20), which relates the PSD of the noise terms in the original data to the obtained Allan variance. To summarize, the noises, their Allan variances and PSD are listed in Table 8.1.

In addition, if the stochastic errors listed in Table 8.1 are statistically independent to each other, then the total Allan variance for gyro or accelerometer can be expressed as:

$$\sigma_{\mathrm{Total}}^2 = \sigma_{\mathrm{Q}}^2 + \sigma_{\mathrm{N}}^2 + \sigma_{\mathrm{B}}^2 + \sigma_{\mathrm{K}}^2 + \sigma_{\mathrm{R}}^2 \tag{8.18}$$

Therefore, the root Allan variance, i.e., the Allan standard deviation is given by:

$$\begin{aligned} \sigma_{\mathrm{Total}}(\tau) &\cong \sum_{\mathrm{n}=-2}^{2} \mathrm{A_n}\tau^{\mathrm{n}/2} \\ \sigma_{\mathrm{Total}}(\tau) &\approx \frac{\mathrm{R}}{\sqrt{2}}\tau^{\frac{2}{2}} + \frac{\mathrm{K}}{\sqrt{3}}\tau^{\frac{1}{2}} + \mathrm{B}\sqrt{\frac{2\ln 2}{\pi}}\tau^{0} + \mathrm{N}\tau^{-\frac{1}{2}} + \mathrm{Q}\sqrt{3}\tau^{-\frac{2}{2}} \end{aligned} \tag{8.19}$$

Where:

$\sigma_{\mathrm{Total}}(\tau)$	Allan standard deviation.
R	$\mathrm{unit}\cdot \mathrm{s}^{-1}$.
K	$\mathrm{unit}\cdot \mathrm{s}^{-1/2}$.
B	Unit.
N	$\mathrm{unit}\cdot \mathrm{s}^{1/2}$.
Q	$\mathrm{unit}\cdot \mathrm{s}$.

Unit $\equiv \mathrm{rad/s}$ for the angular rates and Unit $\equiv \mathrm{m/s^2}$ for the accelerations.

In fact, after obtaining the Allan variance for different correlation times, applying the methodology presented before, the coefficients A_n (linked with the noise parameters listed in Table 8.1) can be computed using the least squares method. Finally, using Equation (8.20), the correspondent PSD can be obtained for each of the stochastic noises underlying the sensor signals.

It should be noted that one noise can predominate more than another one, making it a nonsense to model all five errors. Important to realize is that the Allan variance method will be used to study the random error of the MMQ50 in order to model it. Usually, the most dominant error in low-performance MEMS-based sensors has random walk characteristics.

The identification and characterization of the common five noise processes are given in Table 8.1, in both the frequency and time domains. Note that the names have been adapted to the inertial sensors according to (IEEE STD 647, 2006).

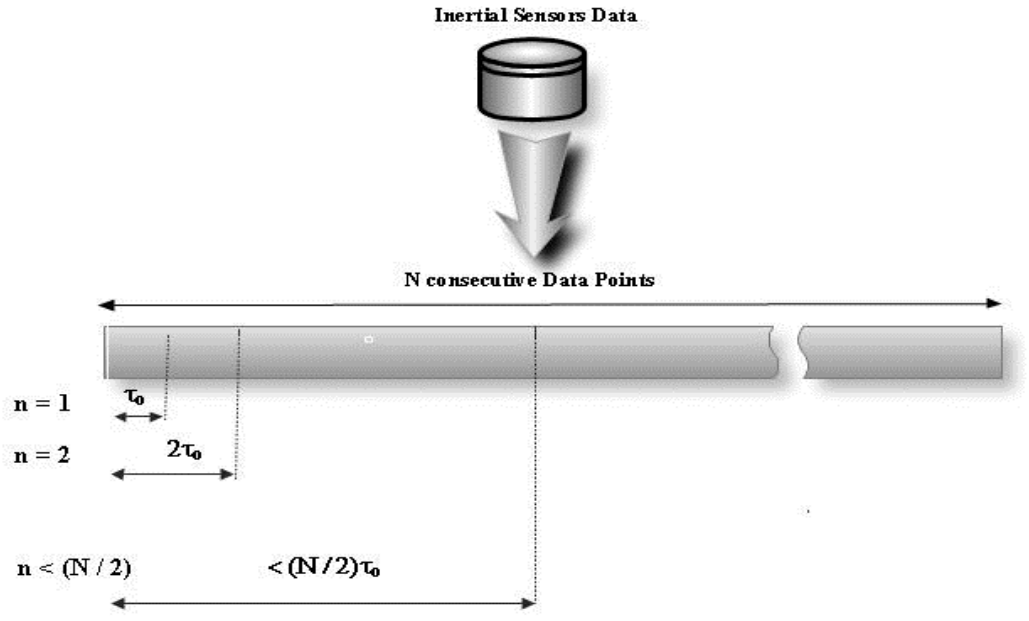

Figure 8.6 Allan Variance method.

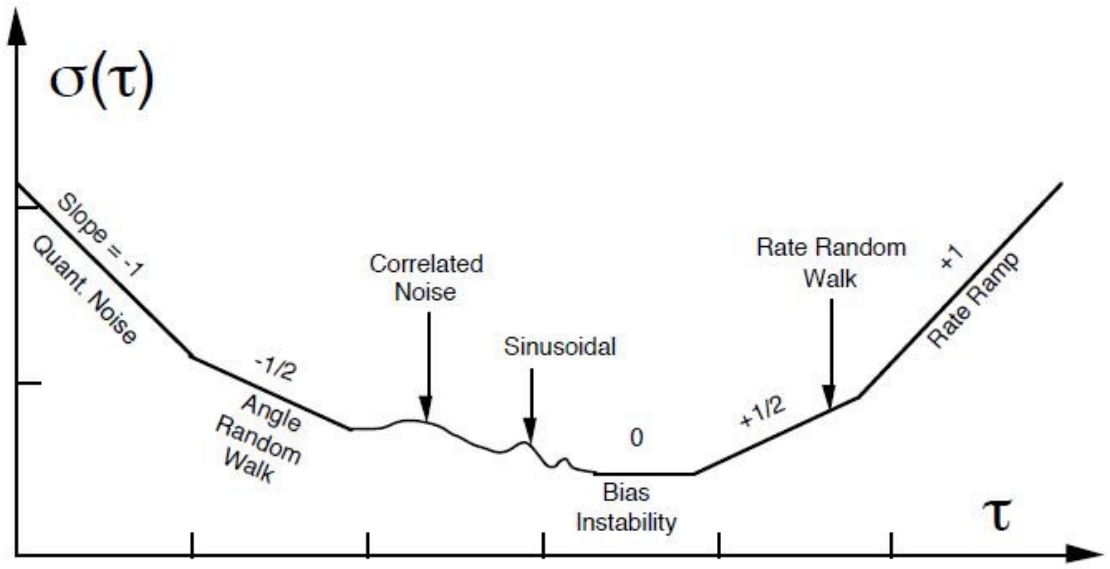

Figure 8.7 Sampled plot of the Allan Variance analysis results. Source: adapted from (IEEE Std. 952, 1997).

8.6.3 PSD and Allan Variance Affiliation (Conversion between frequency and time domains)

Random errors can be modelled using either time domain techniques or frequency domain techniques. The PSD of a random process describes how the power of a time series measurements is distributed in the frequency domain. The Allan variance, on the other hand, is an example of a time domain technique. The main advantage of this method is its affiliation expressed in the frequency domain with the noise power spectral density (PSD) of an intrinsic random process. Mathematically, the relation between the Allan variance and the two sided rate noise PSD is given by Equation (8.20). More details are given in (Papoulis, 1991).

$$\sigma^2(\tau) = 4\int_0^{\infty} S_w(f)\frac{\sin^4(\pi f\tau)}{(\pi f\tau)^2}\,df \qquad (8.20)$$

Where:

$\sigma^2(\tau)$	Is the computed Allan variance at a specific τ cluster correlation time.
$S_w(f)$	Is the respective PSD of the random process, in this case of the IMU measured data

(inertial sensor output) from which the Allan variance was computed.

According to the above equation, it can be stated that the Allan variance is proportional to the total power output of the random process when passed through a filter with the transfer function of the form $\sin^4(x)/x^2$. Using such a relationship between the Allan variance and the PSD, it is possible to show that the computed Allan variance of different error sources can be expressed in various powers of correlation time. Therefore, by assigning a different value of τ, different types of random process can be confirmed. In fact, by varying number of samples per cluster from one until the maximum considered cluster length ($< N / 2$), and repeatedly evaluating Equation (8.14) at step 4, variances at different cluster length $\tau_n$are computed. To determine the characteristics of the principal noise processes, the Allan deviation $AD(\tau) = \sqrt{\sigma_{AV}^2(\tau)}$ is computed and plotted with respect to the cluster length on log-log scale. As shown in Figure 8.7, different types of random processes cause slopes with different gradients. Therefore, different random errors present in the sensor data being evaluated are extracted by calculating the slopes of the Allan variance. Substituting noise PSD of individual noises in Equation (8.20), the relation between the noise coefficients and the Allan variance is calculated and presented in Table 8.1. For more details, see ref. (IEEE STD 647, 2006).

Table 8.2 presents the units for the different noise parameters for both inertial sensors. It should be noted that the same equations and definitions presented in this section are valid for both accelerometers and gyros.

Table 8.1 Affiliation between Allan variance and PSD for different error sources.

Description of the noise process	PSD ($S_w(f)$)	AD (σ (τ))	Log (σ) vs. log(τ)	Curve Slope	Noise Coefficient Value (A_n)
Quantization	$(2\pi f)^2\, Q^2 T$	$\frac{\sqrt{3}Q}{\tau}$	$\log(\sigma) = \log(\sqrt{3}Q) - \log(\tau)$ $(y = -x + b)$	**-1**	$Q = \sigma(\sqrt{3}), (\tau = \sqrt{3}hr)$
Angle/Velocity Random Walk (Rate/Acceleration White Noise)	N^2	$\frac{N}{\sqrt{\tau}}$	$\log(\sigma) = \log(N) - \frac{1}{2}\log(\tau)$ $(y = -\frac{1}{2}x + b)$	$-\frac{1}{2}$	$N = \sigma(1), (\tau = 1hr)$
Bias Instability	$\left(\frac{B^2}{2\pi}\right)\frac{1}{f}, f \le f_0$	$\sqrt{\frac{2ln2}{\pi}}B$	$\log(\sigma(f_0)) \cong \log(0.664B)$ $(y = b)$	**0**	$B = \frac{\sigma(f_0)}{0.664}$
Rate/Acceleration Random Walk	$\left(\frac{K}{2\pi}\right)^2 \frac{1}{f^2}$	$K\sqrt{\frac{\tau}{3}}$	$\log(\sigma) = \log\left(\frac{K}{\sqrt{3}}\right) + \frac{1}{2}\log(\tau)$ $(y = \frac{1}{2}x + b)$	$+\frac{1}{2}$	$K = \sigma(3), (\tau = 3hr)$
Rate Ramp	$\frac{R^2}{(2\pi f)^3}$	$\frac{R\tau}{\sqrt{2}}$	$\log(\sigma) = \log\left(\frac{R}{\sqrt{2}}\right) + \log(\tau)$ $(y = x + b)$	**+1**	$R = \sigma(\sqrt{2}), (\tau = \sqrt{2}hr)$

Table 8.2 Noise parameters units' identification.

Noise Types	Units	
	Gyroscope Raw Data (°/h)	**Accelerometers Raw Data (m/s^2)**
	Parameter of Interest	**Parameter of Interest**
Quantization	$Q = \frac{\sigma(\sqrt{3})}{3600}^{\circ} = \frac{\sigma(\sqrt{3})}{3600} \times \frac{10^6 \times \pi}{180}$ μrad	$Q = \sigma(\sqrt{3})$ m/s
Angle/Velocity Random Walk	$N = \frac{\sigma(1)}{60}$ °/√h	$N = \sigma(1) \times 60$ m/s /√h
Bias Instability	$B = \frac{\sigma(f_0)}{0.664}$ °/h	$B = \frac{\sigma(f_0)}{0.664} m/s^2$
Rate/Acceleration Random Walk	$K = \sigma(3) \times 60°/h/\sqrt{h}$	$K = \sigma(3) m/s^2/\sqrt{s}$
Rate Ramp	$R = \sigma(\sqrt{2})$ ×3600deg/h/h	$R = \sigma(\sqrt{2})$ m. Hz/s^2

CHAPTER 9

INERTIAL SENSOR MEASUREMENTS MODEL

9 Inertial Sensors Measurement Model (Error and Compensation Form Expression)

The main objective of this chapter is to present the procedures that were applied to answer the fourth research question of this study, which asks: How to deal with low performance inertial navigation sensors in the processing of sensor modelling?

We have tried to develop a strategy able to be used as an adequate and simple error model, feasible to be applied to any IMU belonging to the low/medium grade category of the inertial sensors. The suggestion is, from the general error model, and based on some sensor specifications, to construct a simple error model that includes just the dominant error sources usually present on such sensor category. In fact, if for one hand, a general error model has to cover several effects such as environmental influences, mounting uncertainties, input/output nonlinear behaviour, stochastic errors, etc.; on the other hand, a simple error model normally just includes a constant bias error, a moving bias error, and noise.

Important to realize is the fact that the error model to be developed is directly dependent of the navigation system architecture (i.e. integrated navigation system or stand-alone system) and/or the inertial sensors being used. A general error model may include several errors that are significant when inertial navigation systems are used for high or very high precision navigation, for long periods of time and without external aiding. With low-performance sensors, it does not make any sense to consider the several error contributions when the output signal is mainly contaminated by white noise. In this thesis as well as in similar works related to UAV navigation, usually, low-performance inertial sensors are used in an aided sense and/or for short periods. Thus, the general error models associated to the two inertial sensors are reduced to simpler error models where just the most dominant error sources should be modelled after its

correct identification.

In this study, the strategy that has been followed to develop the MMQ50 IMU measurements models expressed in the error form, as well as, the associated compensation forms are presented next, involving two steps. First, based on the standard measurement models and taking into account inertial sensors manufacturer specifications the adopted mathematical form of the error model is achieved. Second, being the topic of discussion of the next chapter, the nature of the dominant error sources present and affecting the performance of the MMQ50 is studied, using for that the mathematical tools presented at Chapter 8. In fact, these techniques will be used not only to identify the dominant errors but also to achieve the numerical values for the time constants, means, and variances for the various parameters that compose the adopted model developed at step 1.

9.1 Accelerometers and Gyros Standard Measurement Model

Gyroscopes and accelerometers are the two different sensors that are used in inertial navigation, and although they exhibit many similarities in their error model forms, two models, associated to each of them, will be developed. As follows, the general/standard measurement models expressed in the error form for both accelerometers and rate gyros are presented.

9.1.1 Accelerometers Standard Measurement Model in the Error Form and Associated Compensation Form

The sensor input vector is defined as containing the sensed quantities (accelerations in [m/s^2]). In accordance, the output vector is defined as containing the delivered sensor output values (accelerometer outputs in Volts [V]). Thus, the theoretical output measurement model for the accelerometers (Dorobantu, et al., 2004), expressed in the error form, in the B-Frame can be given as:

$$f_{out}^{B} = (Scf_{acc}.I + \delta Scf_{acc} + \delta Ma_{acc})f_{in}^{B} + \delta nl_{acc} + \delta b_{acc} + \tilde{v}_{acc} \quad (9.1)$$

Where δ denotes a perturbation, and the components are define and described in Table 9. 1.

Table 9.1 Accelerometers general error model.

Components	Definitions
$\mathbf{f}_{\mathbf{out}}^{\mathbf{B}} = \begin{bmatrix} \mathbf{f}_{\mathbf{out}_x}^{\mathbf{B}} \\ \mathbf{f}_{\mathbf{out}_y}^{\mathbf{B}} \\ \mathbf{f}_{\mathbf{out}_z}^{\mathbf{B}} \end{bmatrix}$	Vector of Specific force measurement delivered by the accelerometer triad. This vector must be converted from Volts (V) to respective engineering units according to Part IV of this thesis.
$\mathbf{f}_{\mathbf{in}}^{\mathbf{B}} = \begin{bmatrix} \mathbf{f}_{\mathbf{in}_x}^{\mathbf{B}} \\ \mathbf{f}_{\mathbf{in}_y}^{\mathbf{B}} \\ \mathbf{f}_{\mathbf{in}_z}^{\mathbf{B}} \end{bmatrix}$	True input specific force vector sensed by the accelerometers (given in [m/s2]).
$\mathbf{Scf}_{\mathbf{acc}} = \begin{bmatrix} \mathbf{Scf}_{\mathbf{acc}_x} \\ \mathbf{Scf}_{\mathbf{acc}_y} \\ \mathbf{Scf}_{\mathbf{acc}_z} \end{bmatrix}$	Constant part of the accelerometer scale factor vector. According to Chapter 8 each error source as four components, this is the constant part and the other three parts are included in the next parameter.
$\boldsymbol{\delta}\mathbf{Scf}_{\mathbf{acc}} = \boldsymbol{\delta}\mathbf{Scf}_{\mathbf{stability}} + \boldsymbol{\delta}\mathbf{Scf}_{\mathbf{temperature}} + \boldsymbol{\delta}\mathbf{Scf}_{\mathbf{drift}}$	Is the diagonal matrix expanded as a sum of the three scale factor remaining components: the turn-on-to-turn-on offset $\delta\mathrm{Scf}_{\mathrm{turn-on}}$, the temperature dependent component $\delta\mathrm{Scf}_{\mathrm{temperature}}$, and the time dependent component $\delta\mathrm{Scf}_{\mathrm{drift}}$.
$\boldsymbol{\delta}\mathbf{Scf}_{\mathbf{acc}} = \begin{bmatrix} \boldsymbol{\delta}\mathbf{Scf}_{\mathbf{stability}_x} + \boldsymbol{\delta}\mathbf{Scf}_{\mathbf{temperature}_x} + \boldsymbol{\delta}\mathbf{Scf}_{\mathbf{drift}_x} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\delta}\mathbf{Scf}_{\mathbf{stability}_y} + \boldsymbol{\delta}\mathbf{Scf}_{\mathbf{temperature}_y} + \boldsymbol{\delta}\mathbf{Scf}_{\mathbf{drift}_y} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{\delta}\mathbf{Scf}_{\mathbf{stability}_z} + \boldsymbol{\delta}\mathbf{Scf}_{\mathbf{temperature}_z} + \boldsymbol{\delta}\mathbf{Scf}_{\mathbf{drift}_z} \end{bmatrix}$	

$\boldsymbol{\delta Ma_{acc}} = \begin{bmatrix} \mathbf{0} & \boldsymbol{\delta Ma_{acc_{xy}}} & \boldsymbol{\delta Ma_{acc_{xz}}} \\ \boldsymbol{\delta Ma_{acc_{yx}}} & \mathbf{0} & \boldsymbol{\delta Ma_{acc_{yz}}} \\ \boldsymbol{\delta Ma_{acc_{zx}}} & \boldsymbol{\delta Ma_{acc_{zy}}} & \mathbf{0} \end{bmatrix}$	Off-diagonal matrix that accounts for the misalignments between the accelerometers reference system and the IMU reference frame.	
$\boldsymbol{\delta nl_{acc}}$	Matrix of accelerometers nonlinearities and also cross-coupling accelerometer errors.	
$\boldsymbol{\delta b_{acc}} = \boldsymbol{\delta b_{constant_{acc}}} + \boldsymbol{\delta b_{temperature_{acc}}} + \boldsymbol{\delta b_{stability_{acc}}} + \boldsymbol{\delta b_{drift_{acc}}}$ (Bias vector, which is defined as the sum of all perturbations that are resulting by zero inputs ($\mathbf{f_{in}^B} = \mathbf{0}$))	$\delta b_{constant_{acc}}$	Constant acceleration offset. It should be noted that normally the principal part of this error is eliminated through the IMU calibration (hardware compensation/laboratory calibration/coarse/fine alignment).
	$\delta b_{temperature_{acc}}$	Temperature dependent component (also eliminated through calibration or modelled).
	$\delta b_{stability_{acc}}$	The turn-on to turn-on offset (stability) normally modelled as a random constant: $\dot{\delta} b_{turn-on_{acc}} = 0$
	$\delta b_{drift_{acc}}$	Bias drift with random nature (see Chapter 8 for more details). This term has more additive components being the principal ones detailed as follows.
$\boldsymbol{\delta b_{drift_{acc}}} = \boldsymbol{\delta VRW} + \boldsymbol{\delta GM_{acc}} + \boldsymbol{\delta Q_{acc}} + \boldsymbol{\delta BI_{acc}} + \boldsymbol{\delta AccRW} + \ldots$	δVRW	Velocity random walk error modelled as a stochastic random walk process. The process differential equation is given by: $\dot{\delta} VRW = wn$, with wn ~ (N, 0), stochastic Gaussian white noise process of amplitude N and zero mean value.
	δGM_{acc}	Time-correlated acceleration drift, modelled as a Gauss-

$\boldsymbol{\delta b_{drift_{acc}}}$		Markov first order stochastic noise process. The process differential equation is given by: $\dot{\delta}GM_{acc} = -\beta.\delta GM_{acc} + \sqrt{2\beta\sigma^2_{\delta GM_{acc}}}$, with β the inverse of the correlation time.
	δQ_{acc}	Quantization error.
	δBI_{acc}	Bias instability term (IEEE STD 647, 2006) is the drift of low-frequency nature. Is often called the 1/f random process that has a slope of 0 in the Allan variance log-log diagram of Chapter 8 (Figure 8.7).
	$\delta AccRW$	This is a very low frequency random drift, with a $1/f^2$ variation in the PSD diagram (slope of 1/2 in the Allan variance log-log diagram of Chapter 8 (Figure 8.7)).
	δQ_{acc}	Quantization error.
$\boldsymbol{\tilde{v}_{acc}}$	White noise vector present in the accelerometer data.	

For more details about the general accelerometer error model the reader should consult (Dorobantu, et al., 2004). Finally, being the error dependence Equation invertible, the general compensation form of the error model to be used directly by the navigation algorithm implemented in Part II of this thesis is given by:

$$f_{in}^{B} = (Scf_{acc}.I + \delta Scf_{acc} + \delta Ma_{acc})^{-1}.(f_{out}^{B} - \delta nl_{acc} - \delta b_{acc} - \tilde{v}_{acc}) \quad (9.2)$$

9.1.2 Gyros Standard Measurement Model in the Error Form and Associated Compensation Form

The difference between rate and integrating gyros should be noted. The first ones measure rotation rate while the second ones measure incremental rotation. In this thesis, the IMU that has been used includes three rate gyros and, so, the general model presented is related to rate gyros.

The gyros error model will be similar to the accelerometer one, with the difference that, now, the input vector is formed from the sensed quantities angular rates in (deg/s) and the output vector formed from the sensor outputs (counts/s in this case).

In light of the above, the measurement model of the gyros expressed in the error form (Dorobantu, et al., 2004) in the B-Frame can be given as:

$$\omega_{out}^{B} = (Scf_{gyro}.I + \delta Scf_{gyro} + \delta Ma_{gyro})\omega_{in}^{B} + \delta nl_{gyro} + \delta b_{gyro} + \tilde{v}_{gyro} \quad (9.3)$$

Where the symbols have, corresponding to the gyro sensors, similar significations as presented in the table above for the accelerometers. Important to realize is that the gyro biases δb_{gyro} are usually denoted as drifts instead of bias and, so, it is common to use the term bias for accelerometers and drift for gyros.

Therefore, the general compensation form for the gyros measurement is given by:

$$\omega_{in}^{B} = (Scf_{gyro}.I + \delta Scf_{gyro} + \delta Ma_{gyro})^{-1} + (\omega_{out}^{B} - \delta nl_{gyro} - \delta b_{gyro} - \tilde{v}_{gyro}) \quad (9.4)$$

9.2 Accelerometers and Gyros adopted measurements models

At this point and given the nature of this thesis application, is important to realize that it is impossible and expensive to account, model and estimate all the errors sources present in both the accelerometer and gyro measurements. Moreover, as it has been stated before, low-performance sensors are contaminated with high amplitude noise that makes it impossible to distinguish between the different types of errors and to allow their easy separation from the true signal. Thus, in such conditions where the low performance inertial sensors are used to be integrated with other sensors, usually, a simple error model is used instead of a more complex one.

Overall, before the present, the simplified and adopted error model for the accelerometers and gyros is considered important to study which deterministic and stochastic errors should we take into account for our model. Remember that according to Section 8.1, the sensor outputs are affected by deterministic and stochastic errors. Deterministic errors are sometimes determined prior to the navigation process and so they do not need to be included in the developed model. Thus, in order to achieve the best error model associated to both accelerometers and gyros of the MMQ50, let's take a look into Figure 9.1, which represents a simplified block diagram of the hardware component of this specific low-performance IMU.

According to this figure, there is a block in the processing chain called the compensation routine. This block is responsible for the calculation of the thermal compensation for bias, scale factor, and alignments. A polynomial compensation, with respect to the internal temperature sensors, is used to correct the filtered sensor outputs. However, even though the sensor has the compensation included for the deterministic errors (hardware level), there is always a residual that remains, which will be included into the noise component part of our model.

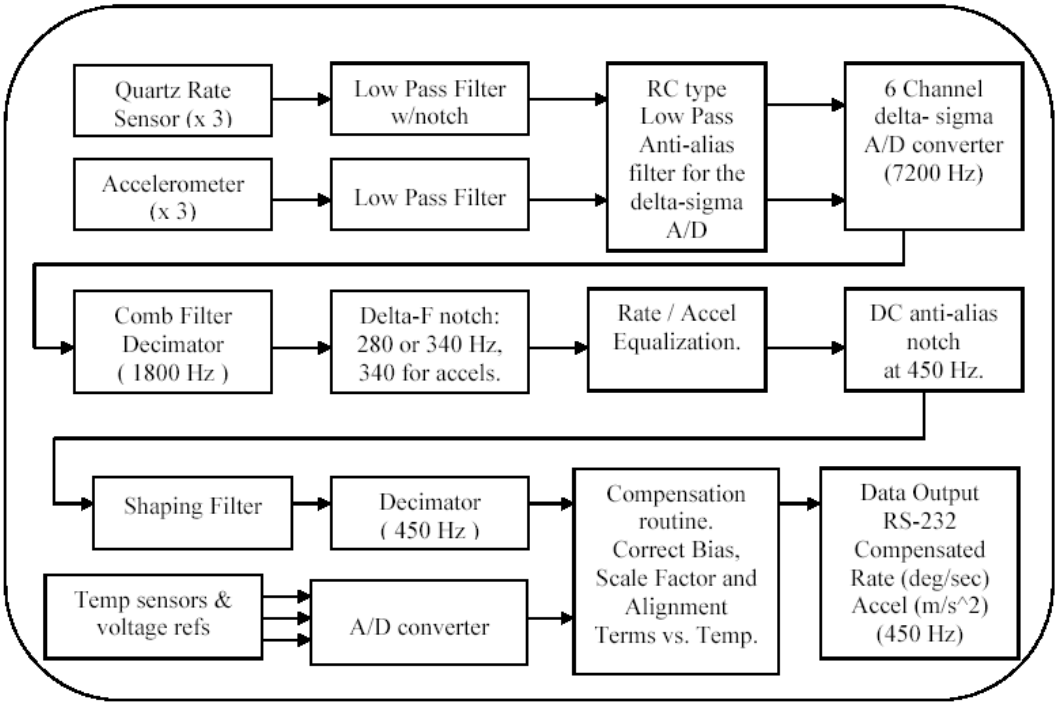

Figure 9.1 MMQ simplified system block diagram (Maximizing Bandwith in the Low Cost Miniature MEMS Quartz IMU, 2004).

Taking into account Figure 9.1 and both the accelerometer and gyros standard error models presented before, Figure 9.2 arises as the adopted accelerometer and gyro measurement models expressed in the error from.

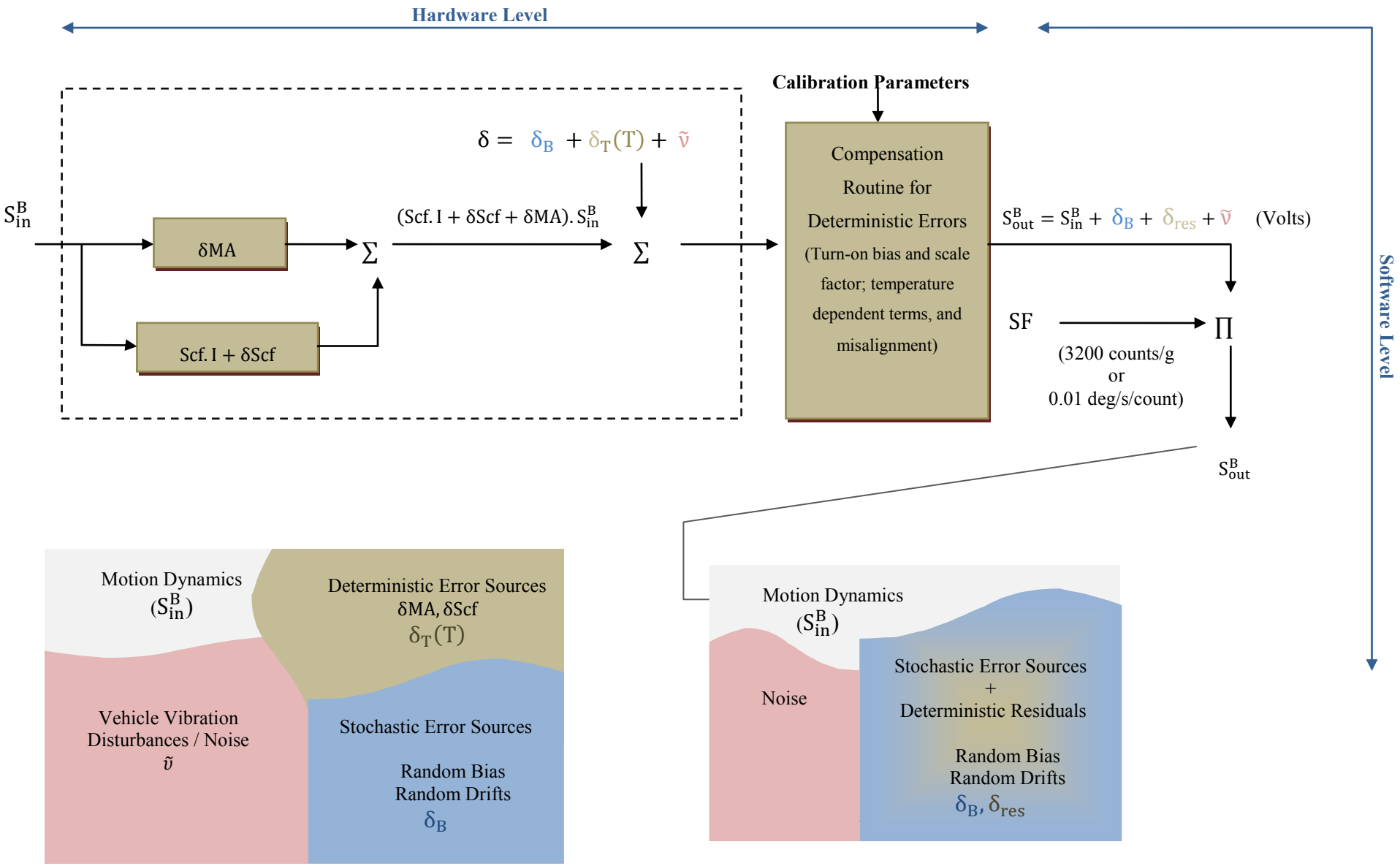

Figure 9.2 Low Performance MMQ50 IMU measurement model expressed in the error form development.

The input vector S_{IN}^{B} is the vector that contains the sensed quantities (true values), which can be either accelerations, more precisely specific force f_{IN}^{B} (m/s^2), or angular rates, ω_{IN}^{B} in (º/s). Therefore, taking into account, the standard models previously presented, we have come up with the simplified measurement model for both the accelerometer triad and gyroscope triad expressed in the error form as:

$$f_{out}^{B} \approx (1 + Scf_{acce}) f_{in}^{B} + \delta b_{acce} + \tilde{v}_{acce} \tag{9.5}$$

$$\omega_{out}^{B} \approx (1 + Scf_{gyro}) \omega_{in}^{B} + \delta b_{gyro} + \tilde{v}_{gyro} \tag{9.6}$$

Taking into account that the scale factor is compensated at the hardware level and that its drift component, when compared to the bias drift component, is negligible, then the adopted models will be:

$$f_{out}^{B} \approx f_{in}^{B} + \delta b_{acce} + \tilde{v}_{acce} \tag{9.7}$$

$$\omega_{out}^{B} \approx \omega_{in}^{B} + \delta b_{gyro} + \tilde{v}_{gyro} \tag{9.8}$$

According to Equations (9.7) and (9.4), the simplest error model to be considered with such type of sensors includes a bias and a random error. The bias term is divided into a constant offset error source called bias turn-on and a moving bias error source called the bias drift. The second one is related to the noise in the measurements, being considered wide-band noise.

Tables 9.2 and 9.3 summarize the adopted measurement model for the accelerometers and gyros, as well as, the procedures to remove or reduce the different error sources considered. The statistics of each error sources are topic of discussion in the next chapter.

Table 9.2 Accelerometers adopted measurement model expressed in the error form

Accelerometers Adopted Measurement Model:

$$\mathbf{f_{out}^B = (1 + Scf_{acc})f_{in}^B + \delta b_{acc} + \tilde{v}_{acc}} \approx \mathbf{f_{in}^B + \delta b_{acc} + \tilde{v}_{acc}}$$

Error Source	**Error Source Components**				**Procedures to Remove/Diminish**	
	Deterministic			Stochastic	Hardware Level/ Compensation	Software Level/Stochastic Modelling
	Constant	Temperature	Turn-on	Drift		
Scale Factor	Scf_{acc}	-	-	-	Scf_{acc}	-
Bias	$\delta b_{acc} = \delta b_{constant_{acc}} + \delta b_{temperature_{acc}} + \delta b_{turn-on_{acc}} + \delta b_{drift_{acc}}$					
	$\delta b_{constant_{acc}}$	$\delta b_{temperature_{acc}}$	$\delta b_{turn-on_{acc}}$	$\delta b_{drift_{acc}} = \delta VRW + \delta GM_{acc} + \delta Q_{acc} + \delta BI_{acc} + \delta AccRW + \dots$	$\dot{\delta} b_{constant_{acc}}$ $\dot{\delta} b_{temperature_{acc}}$	$\dot{\delta} b_{turn-on_{acc}} = 0$ $\dot{\delta} GM_{acc} = -\beta . \delta GM_{acc} + \sqrt{2\beta\sigma^2_{\delta GM_{acc}}}$ $\dot{\delta} VRW = wn, \quad wn \sim (N, 0)$
White Noise	-			$\tilde{v}_{acc}$	-	$E[\tilde{v}_{acc}{}^2] = \sigma^2_{acc} f_s$

Table 9.3 Gyro adopted measurement model expressed in the error form.

Gyros Adopted Measurement Model:

$$\boldsymbol{\omega}_{\mathbf{out}}^{\mathbf{B}} = (\mathbf{1} + \mathbf{Scf}_{\mathbf{gyro}})\boldsymbol{\omega}_{\mathbf{in}}^{\mathbf{B}} + \boldsymbol{\delta}\mathbf{b}_{\mathbf{gyro}} + \tilde{\boldsymbol{\nu}}_{\mathbf{gyro}} \approx \boldsymbol{\omega}_{\mathbf{in}}^{\mathbf{B}} + \boldsymbol{\delta}\mathbf{b}_{\mathbf{gyro}} + \tilde{\boldsymbol{\nu}}_{\mathbf{gyro}}$$

<table>
<tr><th rowspan="3">Error Source</th><th colspan="4">Error Source Components</th><th colspan="2">Procedures to Remove/Diminish</th></tr>
<tr><th colspan="3">Deterministic</th><th>Stochastic</th><th rowspan="2">Hardware Level/ Compensation</th><th rowspan="2">Software Level/Stochastic Modelling</th></tr>
<tr><th>Constant</th><th>Temperature</th><th>Turn-on</th><th>Drift</th></tr>
<tr><td>Scale Factor</td><td>Scf_{gyroc}</td><td>-</td><td>-</td><td>-</td><td rowspan="2">Scf_{gyro}</td><td rowspan="2">-</td></tr>
<tr><td rowspan="2">Bias</td><td colspan="4">$\delta b_{gyro} = \delta b_{constant_{gyro}} + \delta b_{temperature_{gyro}} + \delta b_{turn-on_{gyro}} + \delta b_{drift_{gyro}}$</td></tr>
<tr><td>$\delta b_{constant_{gyro}}$</td><td>$\delta b_{temperature_{gyro}}$</td><td>$\delta b_{turn-on_{gyro}}$</td><td>$\delta b_{drift_{gyro}} = \delta ARW + \delta GM_{gyro} + \delta Q_{gyro} + \delta BI_{gyro} + \delta RRW + \ldots$</td><td>$\dot{\delta} b_{constant_{gyro}}$
$\delta b_{temperature_{gyro}}$</td><td>$\dot{\delta} b_{turn-on_{gyro}} = 0$
$\dot{\delta} GM_{gyro} = -\beta . \delta GM_{gyro} + \sqrt{2\beta\sigma^2_{\delta GM_{gyro}}}$
$\delta ARW = wn, \quad wn \sim (N, 0)$</td></tr>
<tr><td>White Noise</td><td colspan="3">-</td><td>$\tilde{v}_{gyro}$</td><td>-</td><td>$E[\tilde{v}_{gyro}{}^2] = \sigma^2_{gyro} f_s$</td></tr>
</table>

According to both tables, it can be stated that after the hardware level compensation, the remaining errors to be considered affecting the bias are the $\delta b_{Turn-on}$, and the δb_{drift}. Therefore, the inertial sensors models, expressed in the error form, can be given by:

$$f_{out}^{B} \approx f_{in}^{B} + \delta b_{Turn-on_{acc}} + \delta b_{dritf_{acc}} + \tilde{v}_{acc} \quad (9.9)$$

$$\omega_{out}^{B} \approx \omega_{in}^{B} + \delta b_{Turn-on_{gyro}} + \delta b_{dritf_{gyro}} + \tilde{v}_{gyro} \quad (9.10)$$

for the accelerometers and gyros, respectively.

For both models, a study must be done in order to analyse which dominant error (bias drift terms presented at Table 9.2 and 9.3) affects the accelerometer and gyro bias drift in order to be included in the model.

According to the models given at those tables, there are six noise terms need to be evaluated: $\delta \mathbf{b}_{\mathbf{Turn-on_{acc}}}, \delta \mathbf{b}_{\mathbf{Turn-on_{gyro}}}, \delta \mathbf{b}_{\mathbf{drift_{acc}}}, \delta \mathbf{b}_{\mathbf{drift_{gyro}}}, \tilde{\boldsymbol{v}}_{\mathbf{acc}}, \tilde{\boldsymbol{v}}_{\mathbf{gyro}}$. The challenge of constructing inertial sensor error models consists in identifying the value of those variables. De-noise techniques, Allan variance and autocorrelation functions are the mathematical tools that will be used to get these values. The objective of Chapter 10 is to analyse, statically, the biases and random errors sources associated to the MMQ50.

9.3 Accelerometers and gyros measurements adopted compensation form

Base on Equations (9.9) and (9.10), respectively, the corresponding compensation forms for the accelerometers and gyros measurements that will be used to compensate inertial data before going through the strapdown navigator, are:

$$f_{in}^{B} \approx (f_{out}^{B} - \delta b_{turn-on_{acc}} - \delta b_{drift_{acc}} - \tilde{v}_{acc}) \quad (9.11)$$

$$\omega_{in}^{B} \approx (\omega_{out}^{B} - \delta b_{turn-on_{gyro}} - \delta b_{drift_{gyro}} - \tilde{v}_{gyro}) \quad (9.12)$$

Overall, the true values f_{in}^{B}, ω_{in}^{B} are corrupted by a bias decomposed into two components, the turn-on and bias components, and the wide-band noise ($\tilde{v}_{acc}, \tilde{v}_{gyro}$). The wide-band noise is assumed to be normally distributed with a zero mean and sampled covariance (Demoz, 2004).

$$E\left[\tilde{\upsilon}_{acc/gyro}^{2}\right] = \sigma_{acc/gyro}^{2} f_s \qquad (9.13)$$

According to the above equation, it can be stated that the wide-band noise increase as the sample rate of the sensor increases (Demoz, 2004).

The compensation forms, as well, as the procedures to achieve the values that compose the same are presented at Table 9.4 for the accelerometers and Table 9.5 for rate gyros, respectively.

According to both tables, the estimating model parameter values needed to be determined are related to bias turn-on, bias drift, and wide band noise. The procedures used to achieve those values are presented in the same tables. A brief overview about how it will be done is provided. It should be noted that all the experimental results and analysis are done at Chapter 10.

Table 9.4 Accelerometer adopted compensation model parameters determination.

$\mathbf{f_{in}^B \approx (f_{out}^B - \delta b_{turn-on_{acc}} - \delta b_{drift_{acc}} - \tilde{v}_{acc})}$					
Terms	**Variables**	**Model**	**Knows**	**Unknowns**	**Procedure to Achieve Unknowns**
Measurements output corrupted	$f_{out}^B = \begin{bmatrix} f_{out_x}^B \\ f_{out_y}^B \\ f_{out_z}^B \end{bmatrix}$	-	Sensors outputs	-	-
Bias Turn-on	$\delta b_{turn-on_{acc}}$	$\delta b_{turn-on_{acc}} =$ Constant	-	Constant	Average of long term static data
Bias Drift	$\delta b_{drift_{acc}}$	$\delta b_{drift_{acc}} = \delta VRW + \delta GM_{acc} + \delta Q_{acc} + \delta BI_{acc} + \delta AccRW + \ldots$	-	δVRW, δGM_{acc}, ...	Allan Variance and Autocorrelation Function
Wide-band Noise	$\tilde{v}_{acc}$	$E[\tilde{v}_{acc}^2] = \sigma_{acc}^2 f_s$	f_s = Sensor sampling frequency	σ_{acc}^2	Allan Variance Analysis and De-noise Technique

Table 9.5 Gyros adopted compensation model parameters determination.

$\boldsymbol{\omega_{in}^{B} \approx (\omega_{out}^{B} - \delta b_{turn-on_{gyro}} - \delta b_{drift_{gyro}} - \tilde{v}_{gyro})}$					
Terms	**Variables**	**Model**	**Knows**	**Unknowns**	**Procedure to Achieve Unknowns**
Measurements output corrupted	$\omega_{out}^{B} = \begin{bmatrix} \omega_{out_x}^{B} \\ \omega_{out_y}^{B} \\ \omega_{out_z}^{B} \end{bmatrix}$	-	Sensors outputs	-	-
Bias Turn-on	$\delta b_{turn-on_{gyro}}$	$\delta b_{turn-on_{gyro}} =$ Constant	-	Constant	Average of long term static data
Bias Drift	$\delta b_{drift_{gyro}}$	$\delta b_{drift_{gyro}} = \delta ARW + \delta GM_{gyro} + \delta Q_{gyro} + \delta BI_{gyro} + \delta RRW + \ldots$	-	δARW, δGM_{gyro}, ...	Allan variance and Autocorrelation Function
Wide-band Noise	$\tilde{v}_{gyro}$	$E[\tilde{v}_{gyro}^2] = \sigma_{gyro}^2 f_s$	f_s = Sensor sampling frequency	σ_{gyro}^2	Lag 1 scatter plot and autocorrelation function

9.3.1 Numerical Values for Bias Turn-on

As it has been described at Chapter 8, the bias turn-on is constant during the same turn-on, i.e., once the IMU is powered on, its value remains constant, suggesting that both terms $\delta b_{turn-on_{acc}}$, and $\delta b_{turn-on_{gyro}}$ can be modelled as a random constant. Note that those terms are sometimes referred as the turn-on-to-turn-on bias, or null-shift on the manufacturer data sheets. This parameter is quite often estimated in the navigation system, however due to the fact that no real time application is done in this work, the terms will be estimated as the average of both the sensor outputs. This will be done using a sufficient length of data in order to ensure that the sensor measurement has reached a steady state output.

9.3.2 Numerical Values for Bias Drift

The bias drift parameters to be determined will depend of the components affecting this error source. In fact, according to the equation that defines the bias drift presented at Table 9.1, there at least five different components that can be included in the model. In Chapter 10, the Allan variance method, presented at Chapter 8, will be used as a tool to identify which error sources are the most dominants. Then, with the aid of the autocorrelation function, the numerical values associated to each error can be achieved. Usually, when working with low-performance sensors, the dominant error sources are the angular/velocity random noise, and the exponentially correlated noise. As a consequence, the bias drift is often modelled as a first order Gauss-Markov process, which is completely described by the determination of the two parameters, see Tables 9.2 and 9.3: correlation time, $\boldsymbol{\beta_{acc}}, \boldsymbol{\beta_{gyro}}$, and variance $\boldsymbol{\sigma^2_{\delta GM_{acc}}}, \boldsymbol{\sigma^2_{\delta GM_{gyro}}}$.

Both values are normally easily determined from a plot of the process autocorrelation function, although, as it will be possible to see at the next chapter, in practice, when working with low-performance sensors this is not so easy. In fact, remember here what was stated at Chapter 8, looking at Figure 8.1, that the output measurement data of low-performance sensors is a combination of measurement noise (wide band noise), and correlated noise, where the magnitude of the wide band noise is much larger than the magnitude of the correlated noise. At first glance and as it will be possible to analyse at Chapter 10, an autocorrelation plot constructed using gyro or accelerometer data from such sensors does not permit us to easily identify and separate the correlated process in order to achieve the desired values. Thus, de-noise techniques, presented at Chapter 8, are used as a pre-processing tool to such data prior to the autocorrelation function in order to achieve the desired values.

9.3.3 Numerical Values for Wide Band Noise

According to Equation (9.13), the wide-band noise increases with the sensor sample rate. In order to determine the numerical values for σ^2_{acc}, and for σ^2_{gyro} lets elect to use a discrete time description of the process. In this case, the terms $\tilde{v}_{acc}, \tilde{v}_{gyro}$ are modelled as a band limited white noise process. Thus, the variance of such process σ^2_{acc}, and σ^2_{gyro} can be obtained by reading out the Allan deviation value when the averaging time is equal to $T_s = 1/f_s$. A numerical value for those parameters can also be obtained by just looking at the standard deviation of the sensor

output when it is subjected to a zero input and sampled at a rate much higher than the maximum frequency content of the drift component (Demoz, 2004).

As an illustration, Figure 9.3 represents a schematic of the proposed gyro measurement general error model considering the three error sources discussed above. At first glance, two details are well visible: first, we have a transient at the beginning. Then, the data is biased from the true value of about a constant $\delta b_{Stability} = \delta b_{turn\text{-}on_{gyro}}$.

After some hours, the output tends to stabilizes. It should be noted that no attempt was made to try to characterize the existent initial transient. Although it seems that it is related to the temperature, given the fact that the sensor, after power on, needs to be working for some minutes, this was not done.

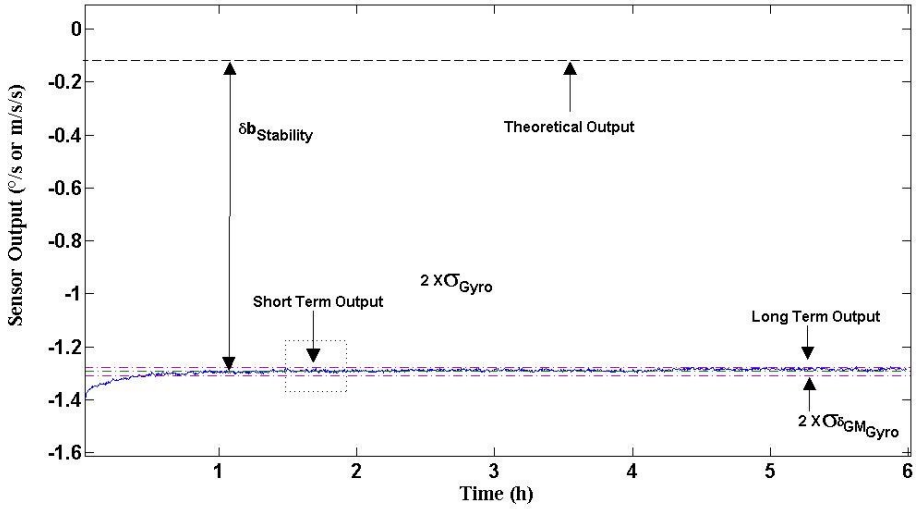

Figure 9.3 Schematic of the gyro general adopted error model.

CHAPTER 10

MMQ50 ERROR MODEL

10 MMQ50 Error Model

The fifth secondary research question targeted in this thesis is a question that has to be answered as a consequence of the explanation of the two previous chapters. This question is how to investigate the possibility of improving the MMQ50 sensors raw data before using it in the strapdown mechanization equations, and also focused in investigating if it is possible to deal with different low performance sensors in the same way.

Chapter 10 intends to answer that question by applying the theory and methodology presented at Chapters 8 and 9. It will give the reader an overview of how to deal with such low-performance IMU sensors, when being a component of our vehicle navigation system. Next, from the adopted compensation form, Equations (9.11) and (9.12), developed at Chapter 9, we will try to achieve numerical values for the time constants, means, and variances by using the mathematical tools presented at Chapter 8. The Allan variance, de-noise techniques, and autocorrelation functions will be used to identify the dominant error sources present in the MMQ50 IMU in order to achieve the best model to describe the MMQ50 accelerometer and gyro measurements.

10.1 MMQ50 IMU Output Raw Data Performance

As a matter of fact, after the theoretical exposition of both chapters 8 and 9, it is time now to plot the inertial sensors raw data in order to see how it looks like. It is time to answer questions such as: I have been collecting inertial raw data for a while. I have three hours of static accelerometer and gyro data; what should I do first? I have static accelerometer data so is it expected to see the gravity acceleration at the z-axis plot? Can I measure the Earth´s rotation rate with my low-performance gyros? What am I expecting to see when plotting inertial raw data?

These, and some others, are the common questions that any newcomers ask to themselves when working with inertial sensors.

Before going into details on determining the numerical values that satisfy the adopted models, let's look at the real raw data obtained with the low-performance IMU MMQ50. Let us just apply some basic concepts to some real data that have been collected with the MMQ50 IMU. A 6-hour data set collected in static conditions with the sampling rate of 450 Hz will be, here, analysed. Figures 10.1 and 10.2 show the output for the MMQ50 Z-axis accelerometer (z accelerometer) and Z-axis rate gyro (z gyro), respectively. Both figures comprise two plots representing the raw data received at the sensors output and the same data detrended (mean removed (Demoz, 2004)). It should be note that the data have been decimated down to 10 Hz for more clarity in the plot.

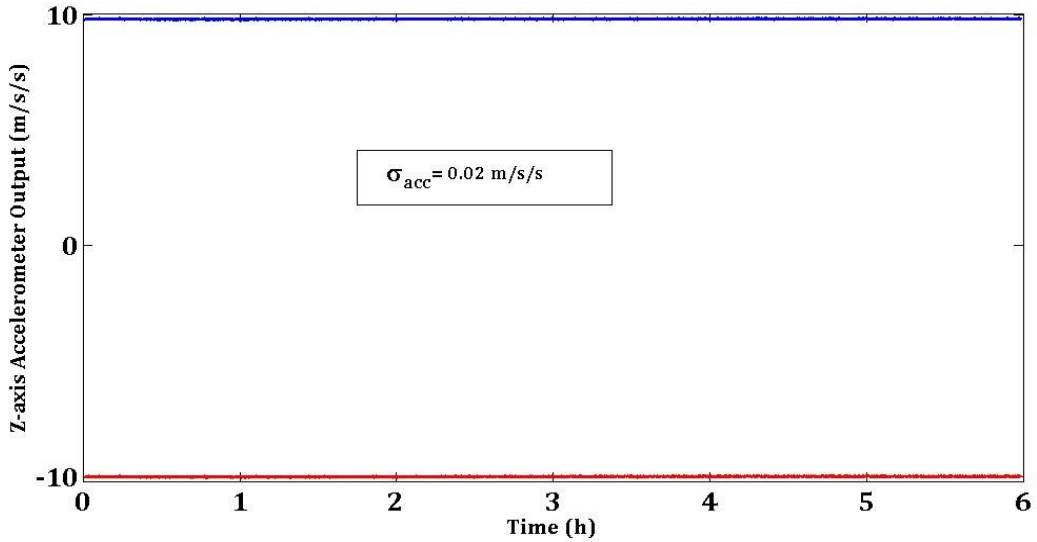

Figure 10.1 MMQ50 Z-axis accelerometer experimental raw data (bottom) and experimental detrended data (top).

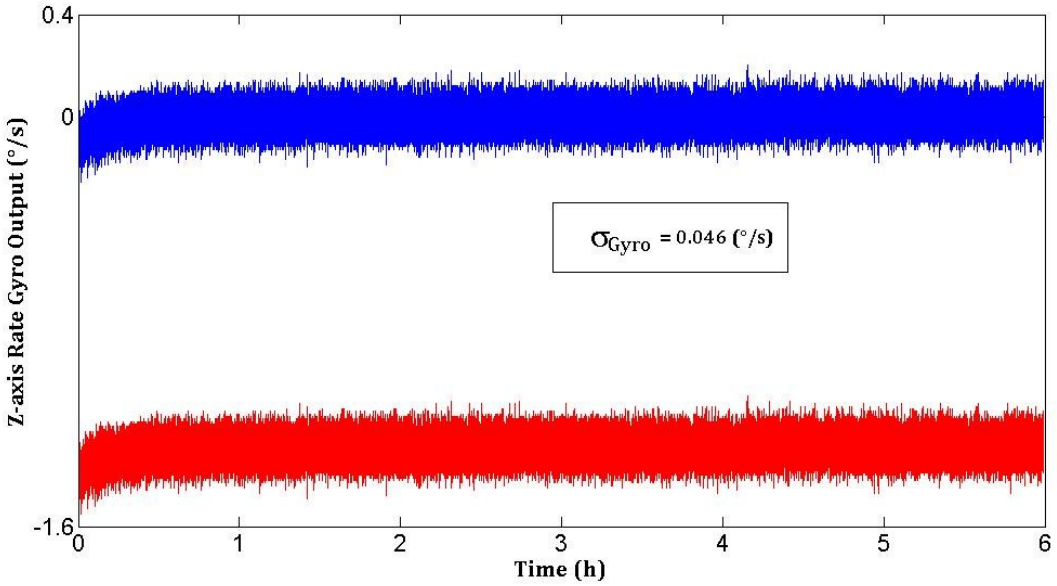

Figure 10.2 MMQ50 Z-axis rate gyro experimental raw data (bottom) and experimental detrended data (top).

Confronting Figure 9.3 with both figures above, it is expected to see a constant bias (bias turn-on), a moving bias (here called bias drift), and wide-band noise. At first glance, if we look at both figures, some details are well visible: first, we have a transient at the beginning. Then the data is biased from the true value of about a constant $\boldsymbol{\delta b_{Turn-on_{acc}}}$ and $\boldsymbol{\delta b_{Turn-on_{gyro}}}$ that has been removed, as it is possible to see at Figure 10.1 top plot, and Figure 10.2 top plot, as well. After some hours, the output tends to stabilize.

Are you somehow disappointed with the plots? For sure, it is not because you have made a mistake when analysing the data but because you are working with low-performance sensors! Welcome to the low-performance sensor world where wide-band noise is highly contaminating your true data.

In order to confirm the general impression that the data are mostly contaminated by uncorrelated noise – to confirm the high presence of wide-band noise (white uncorrelated noise) as the dominant noise source present in the measurement data – a lag1[8] scatter plot followed by the respective autocorrelation function is perform.

Doing a linear fit to the obtained plot, the slope of this linear fit to those points is closely

[8] A lag1 scatter plot in no more than a plot of the data plotted against itself with a lag of one, which shows the degree of correlation in the data.

related to the lag 1 autocorrelation.

For example, white (uncorrelated) noise has a nominal lag 1 autocorrelation of zero, which means that the slope of a linear fit should be around zero. Another example is the case of random walk noise (exponentially correlated Gauss-Markov noise), which has a linear fit slope of about one. Therefore doing a lag 1 scatter plot of the collected data aids us to predict which noise is dominating the data.

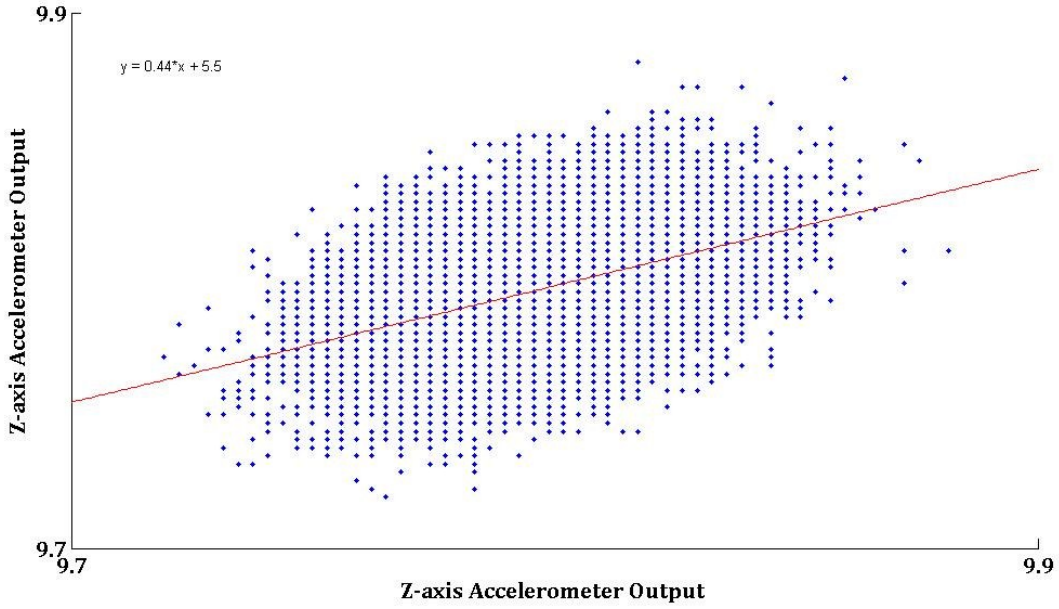

Figure 10.3 Lag 1 scatter plot for MMQ50 rate accelerometer data.

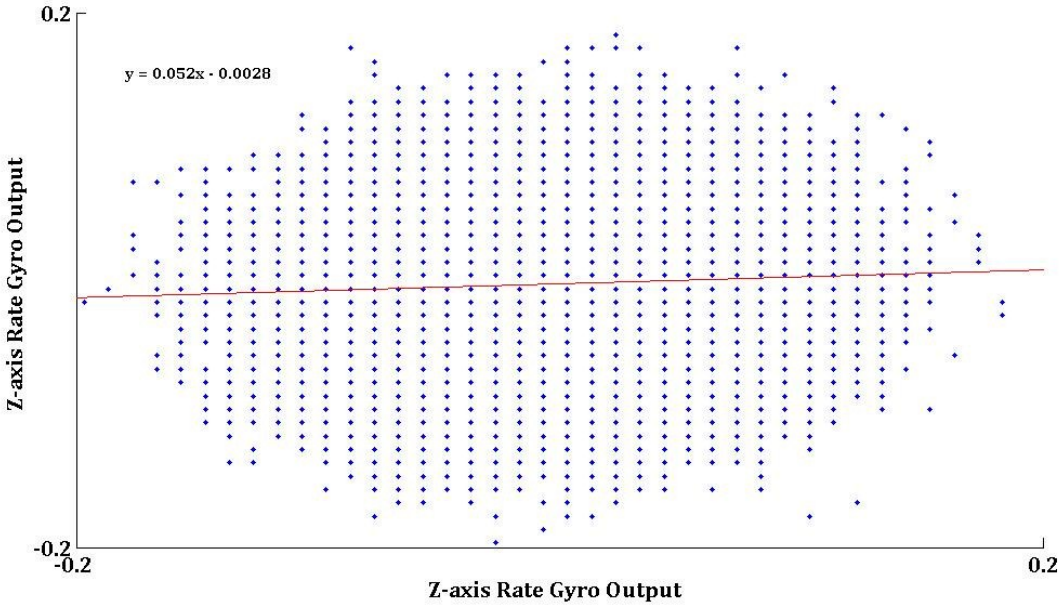

Figure 10.4 Lag 1 scatter plot for MMQ50 rate gyro data.

Based on Figure 10.3, it is possible to reach the following conclusions: the z accelerometer data are from an underlying autoregressive model with moderate positive autocorrelation (bias instability characteristic). Actually, note how the points tend to cluster (albeit noisily) along the diagonal of slope 0.44. Such clustering is the lag plot signature of moderate autocorrelation. This is because the actual output data is a combination of wide band noise and a correlated process, where the magnitude of the wide band noise is much larger than the magnitude of the correlated noise.

From Figure 10.4, it is possible to say that there are no doubts that the sensor is extremely corrupted by white (uncorrelated) noise. In fact, looking at the lag 1 plot and respective autocorrelation function, see Figure 10.6, it can be stated that the wide-band noise dominates the rate gyro output data. In addition and in advance, it can be stated that looking at both Figures, it is possible to suggest that after applying some de-noise techniques to the raw data, it will be possible to reduce part of that wide-band noise and then to model, to some extent, the correlated noise hide behind that high frequency noise.

In order to validate the high presence of wide-band noise in the accelerometer and gyro output data, the respective autocorrelation function is performed. For both plots, just 5 hours of static data decimated down to 10 Hz are used.

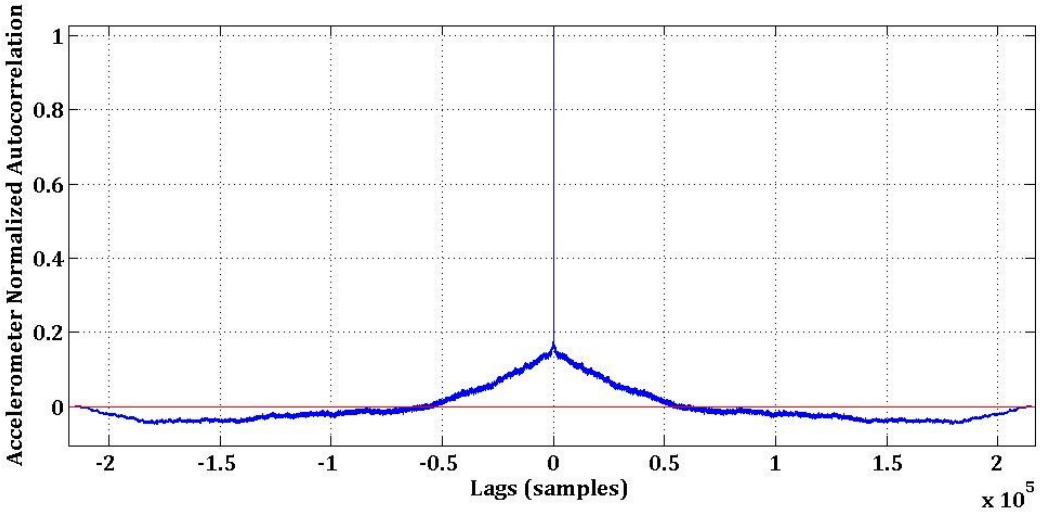

Figure 10.5 MMQ50 accelerometer data autocorrelation.

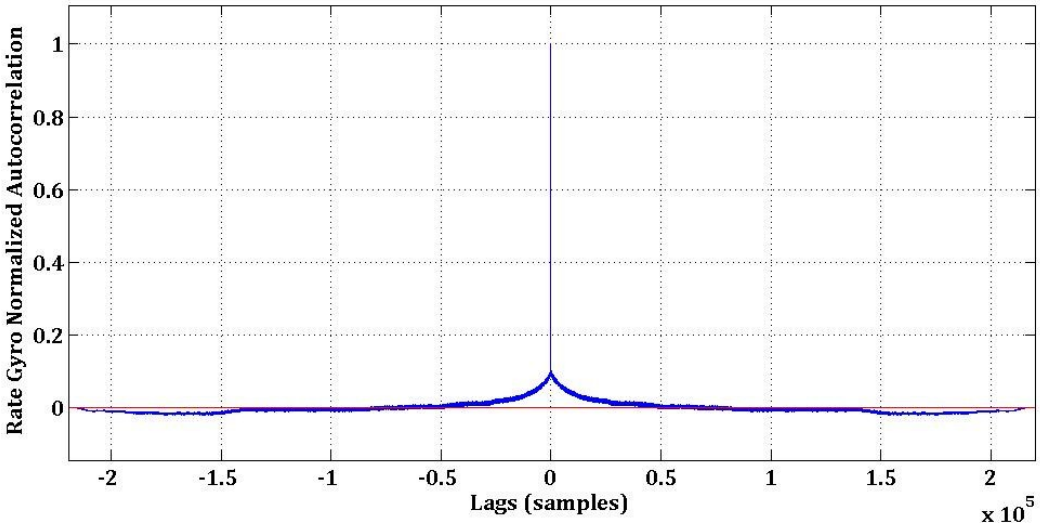

Figure 10.6 MMQ50 rate gyro data autocorrelation.

In fact, based on both the lag 1 scatter plots and autocorrelation functions, it is possible to confirm that both MMQ50 sensors are extremely corrupted by wide-band noise, although with more intensity in the rate gyros than in the accelerometers. Thus, it is possible to state that the standard deviation of this wide band noise will be equal to the standard deviation of the raw output data, over a short period of time, when no input is applied. The numerical values for both the wide noise standard deviation are presented at the last section of this chapter.

10.2 Analysis of Wavelet-Based Signal de-Noising Technique when applied to the MMQ50 Measurements

The use of wavelet signal de-noising as a technique to smooth the raw inertial measurements is wining more and more fans. The idea consists in trying to reduce and/or eliminate the maximum possible wide-band noise present into the inertial measurements, prior to using them into the navigation equations. The methodology followed to achieve such an objective has been discussed at Chapter 8; here just the results of applying such techniques will be presented and analysed.

For instance, Figure 10.7 illustrates an example of the wavelet-based signal decomposition using a db wavelet and an LOD of ten applied to the output of the MMQ50 z gyro in static mode. It should be noted that an exponent of two samples should be used and so, 2^{17} data points are used in the analysis.

According to the theory presented at Chapter 8, we have:

$$S = a_k(t) + \sum_{i=1}^{k} d_i(t) \qquad (10.1)$$

Where:

S	Z-axis rate gyro output (raw/original signal).
k	Level of Decomposition (LOD).
a_{10}	Decomposed component that contains an approximation of the raw signal.
$d_1, d_2, d_3, \ldots d_{10}$	Details of the raw signal. The removed components at different LODs.

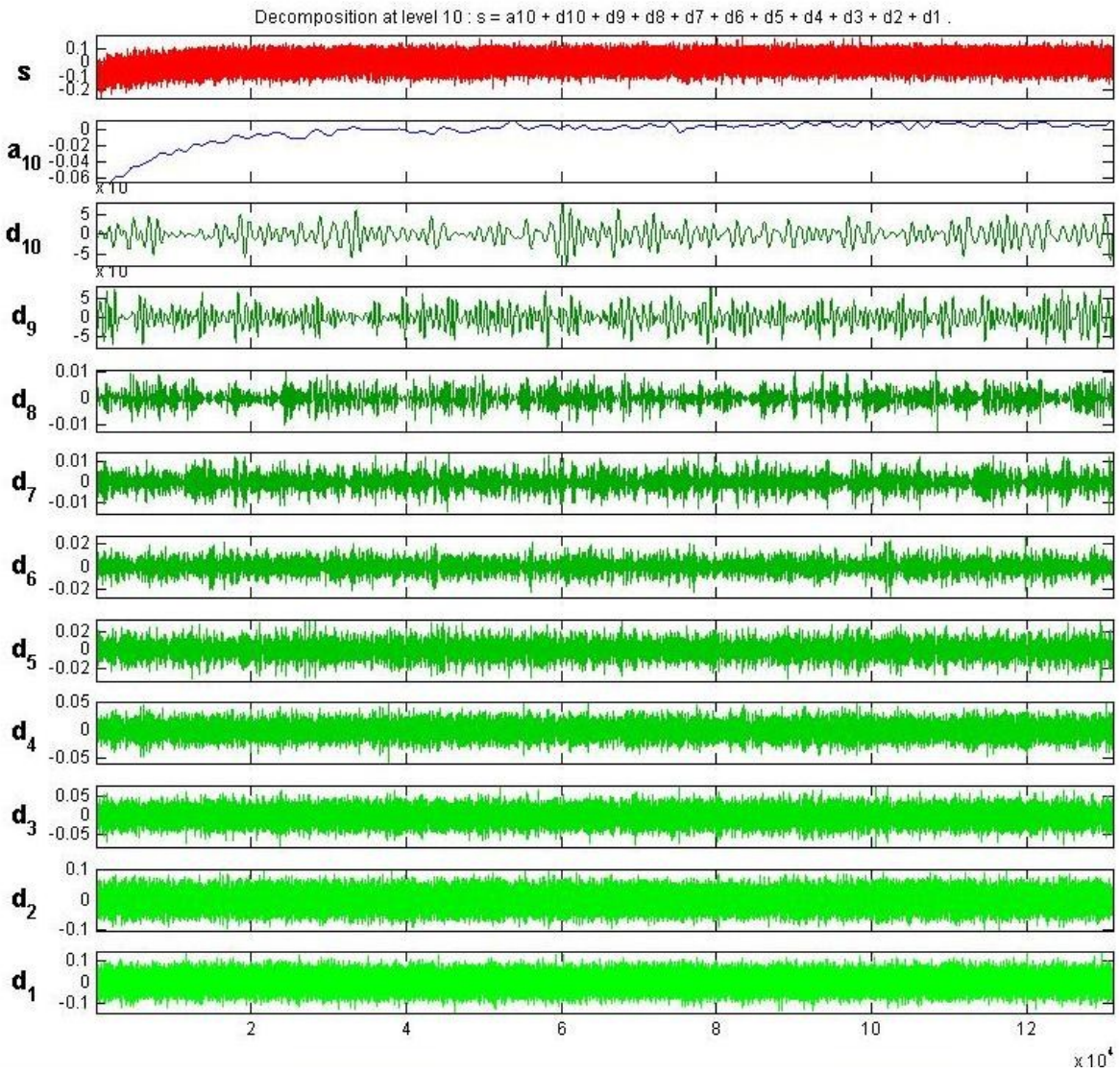

Figure 10.7 Wavelet signal decomposition for the Z-axis rate gyro output in static mode.

Important to realize is that the approximation part represents the majority of the deterministic errors and stochastic correlated noise, whereas de summation of the k-level $d_i(t)$, containing the details for each level of the wavelet decomposition should only contain the high-frequency measurement noise, in addition to some temporally and deterministic correlated error sources. As it has been presented before, the maximum allowable level of de-noising is normally determined by examining the statistic of the removed noise, being the LOD smaller for kinematic dataset to prevent the partial removed of the dynamic signals.

Figure 10.8, illustrates how the allowable LOD for the rate gyro signal has been determined. According to the figure, if we examine the standard deviation of the removed noise

for the original signal, it can be stated that the allowable LOD is 10 for the gyro signals. After 9 levels, the details signal tends to be stable in terms of the standard deviation value (it achieves its minimum after this LOD). It was decided to stop at level ten in order to avoid removing part of the Earth´s rotation rate or its gravity components.

The same analysis has been done to the accelerometer data and the same LOD has been achieved. Thus, hereafter the accelerometer and gyro de-noised static data have been decomposed using the db wavelet with a LOD of 10.

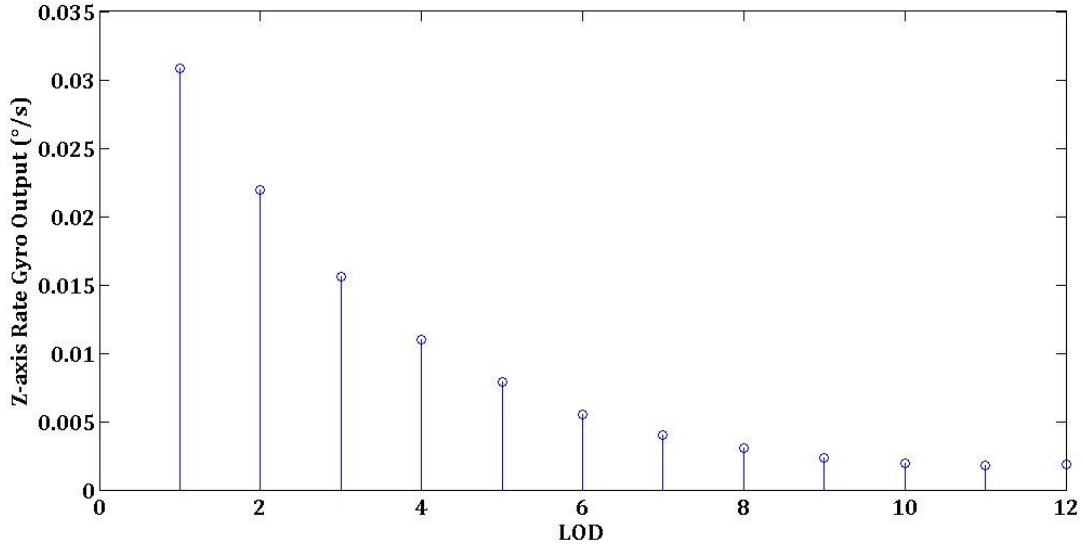

Figure 10.8 Standard deviation of the details reconstructed signal for different levels of de-noising for a static dataset of MMQ50 Z-axis rate gyro.

Therefore, the proposed de-noising method was applied to the real data collected from the MMQ50, in static mode in order to limit the output noise. Figures 10.9 and 10.10 present the smoothed effects of the y accelerometer and z gyro for 3.5 h of data collected in static mode. In both figures, it is evident that most of the noise components have been removed, thus reducing the measurement uncertainty. Table 8 lists the standard deviation of the measurement noise obtained before and after de-noise. According to the values, it is possible to see some reduction in the measurement noise, although not so high, given the bad quality of MMQ50 gyros.

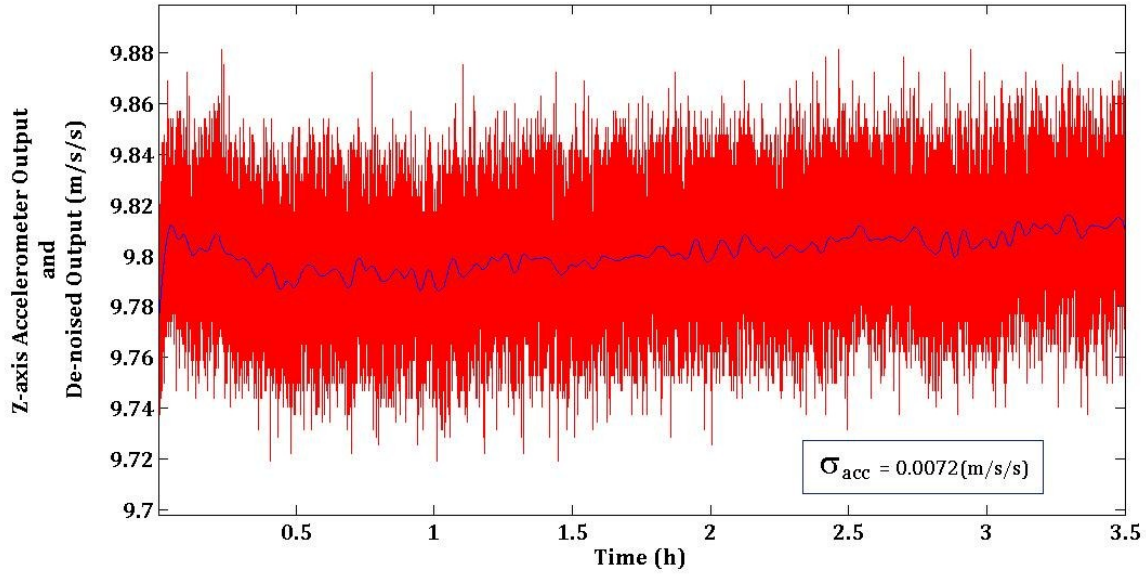

Figure 10.9 Z-axis accelerometer output and de-noised output (lod =10).

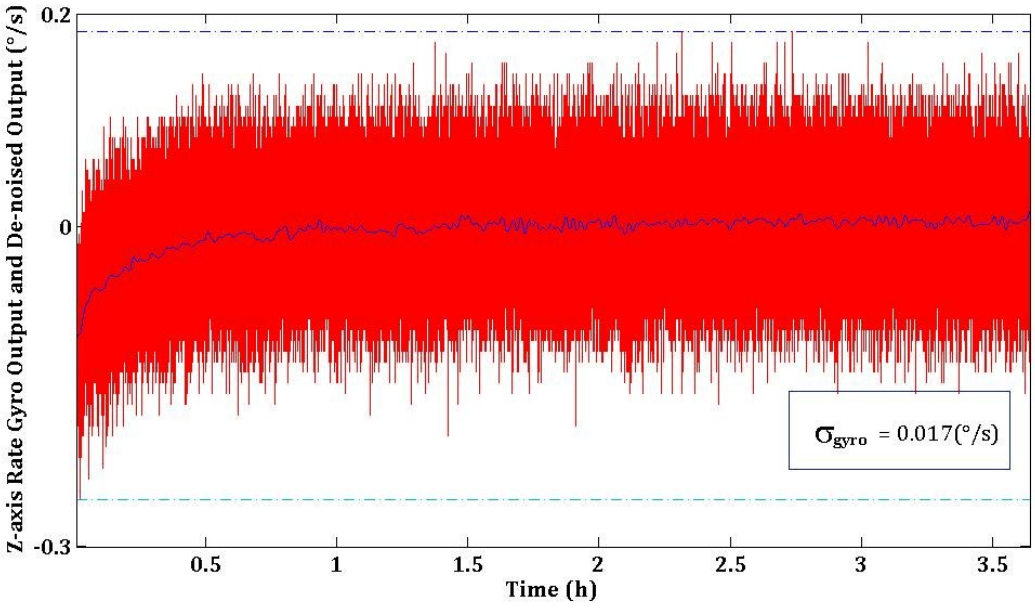

Figure 10.10 Rate gyro output and de-noised output (lod=10).

In both Figures 10.9 and 10.10, the moving bias is clear. After de-noising the inertial measurements, a lag 1 scatter plot with the de-noised signals will be performed. After de-noising, it is clear that we have a correlated error, which according to the lag plot and the autocorrelation function has the behaviour of a correlated noise similar to a Gauss Markov process.

Table 10.1 Standard deviations of the MMQ50 sensor measurements.

Inertial Sensor Measurement	Standard Deviation Before De-noise (σ_{acc}/ σ_{gyro})	Standard Deviation After De-noise (σ_{acc}/ σ_{gyro})
Z-axis Accelerometer	$0.02\ (m/s^2)$	$0.0072\ (m/s^2)$
Z-axis Rate Gyro	$0.046\ (°/s) \approx 166(°/h)$	$0.017\ (°/s) \approx 61(°/h)$

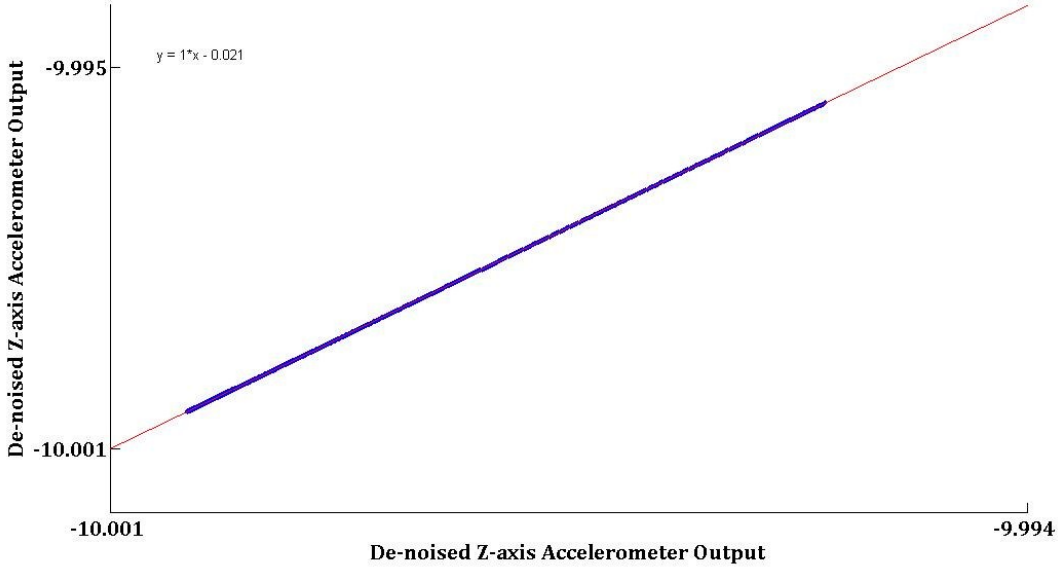

Figure 10.11 Lag 1 scatter plot of the de-noised z-axis accelerometer output.

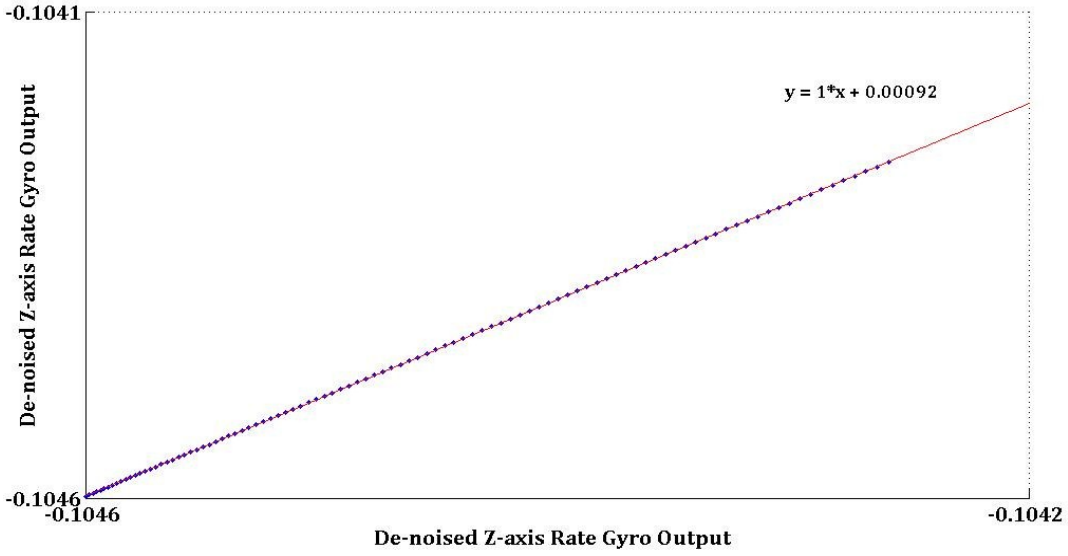

Figure 10.12 Lag 1 scatter plot of the de-noised z-axis rate gyro output.

In fact, if we confront the obtained plots before and after de-noise, it is possible to state that before applying de-noise techniques to the raw data, the measurements were highly contaminated by wide band noise and after de-noise by random walk FM noise, more precisely correlated noise. At its turn, this correlated noise should be analysed with more details using the Allan variance tool.

Following the same procedure as before, the new autocorrelation functions for both de-noised accelerometer and rate gyro measurements is obtained.

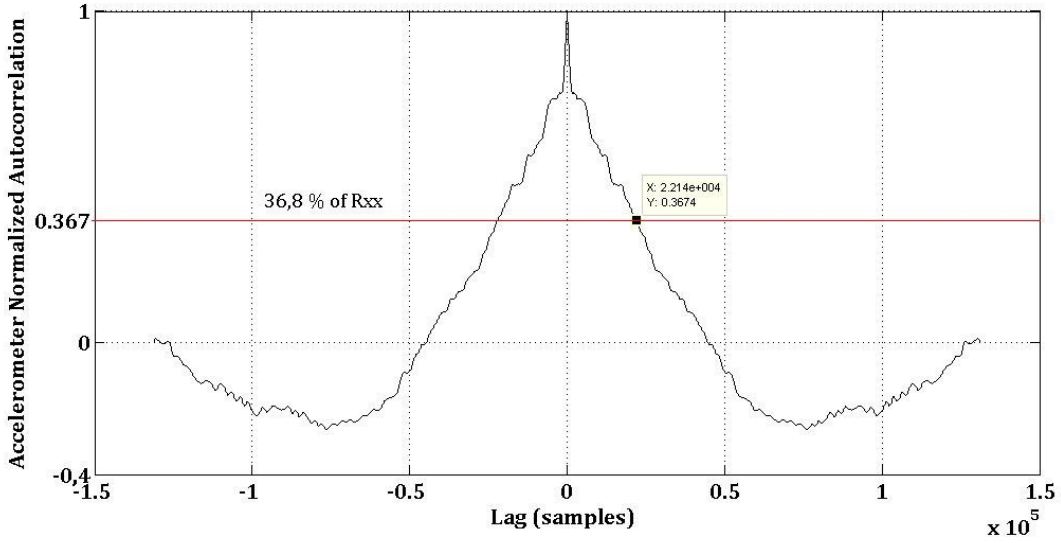

Figure 10.13 Accelerometer measurement autocorrelation after de-noise.

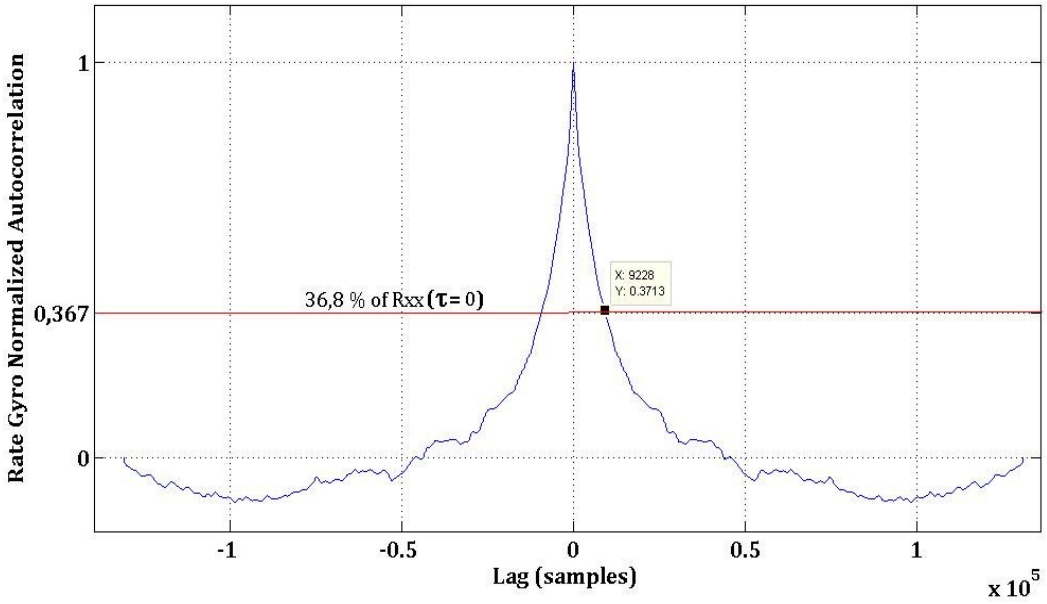

Figure 10.14 Rate gyro measurements autocorrelation after de-noise.

Confronted Figures 10.13 and 10.14 with the respective one before de-noise (Figures 10.5 and 10.6), it can be stated that the present plots looks as an autocorrelation function of a 1st order Gauss-Markov random process instead of an autocorrelation function of a white noise process as the first ones.

10.3 Allan Variance Application to the MMQ50 Sensor Error Characterization

The Allan variance algorithm used to analyse the MMQ50 raw data has been developed by the author according to the methodology described in Chapter 8. The program was developed under the MatLab environment. The input data to this algorithm are in the follow units: °/h for gyros measurements and m/s^2 for accelerometers. Therefore, the Allan deviation obtained will be in the same units.

First a log.log plot of the MMQ50 three axis accelerometers and rate gyros is performed. A long dataset (4 hours) of collected data is used here as an example analysis to investigate the Allan variance analysis for the MMQ50. For this analysis, the data has been sampled at 10 Hz in static mode.

Figures 10.15 and 10.16, presents the Allan standard deviation estimation for the triad

accelerometers and rate gyros respectively, versus cluster time in s.

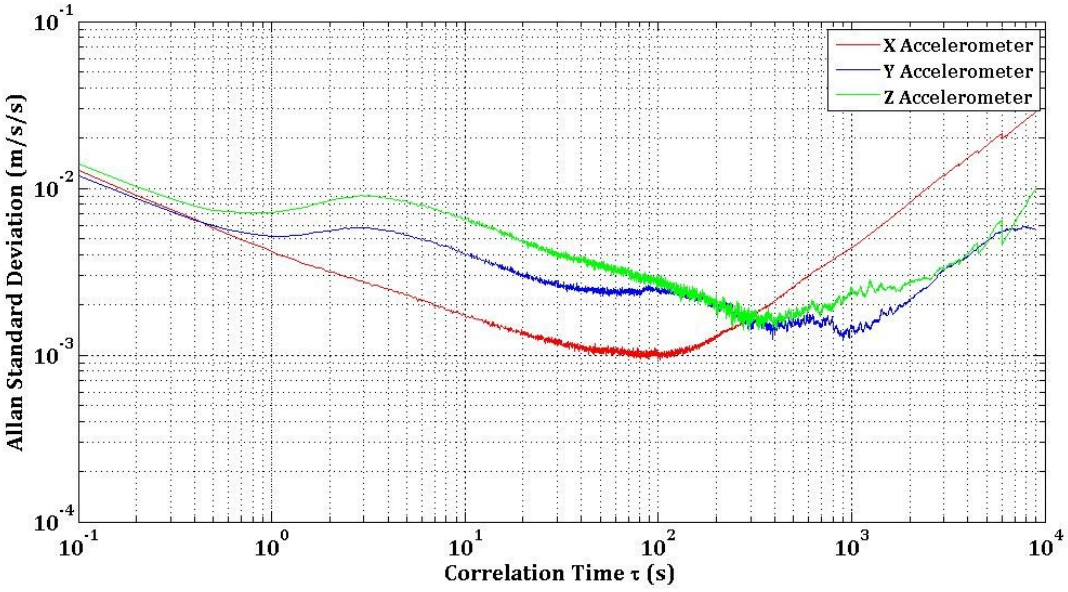

Figure 10.15 MMQ50 accelerometers Allan standard deviation log-log plot.

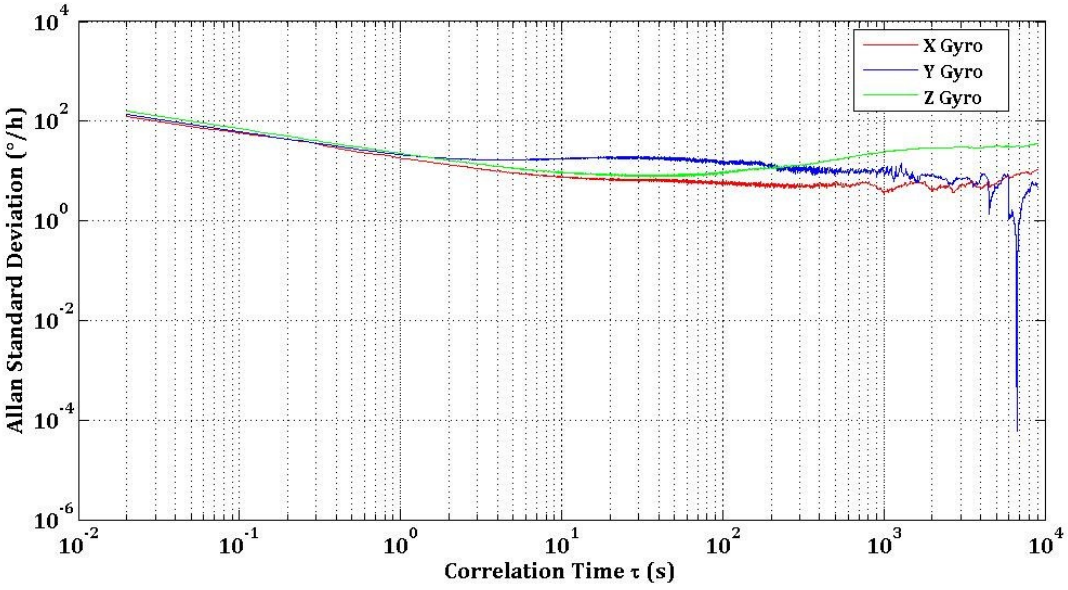

Figure 10.16 MMQ50 rate gyros Allan standard deviation log-log plot.

Generally, observing Figures 10.15 and 10.16, it can be stated that for short averaging times, the Allan deviation is dominated by the noise in the sensor. It is clear the direct correlation between the standard deviation (the noise) of the output vs. time and the slope of the Allan deviation at small τ. This is also referred to as angle random walk (ARW.) Thus, the Allan

standard deviation chart confirms the general impression that the data are mostly uncorrelated noise. As we average over longer and longer times, the variance and, consequently, the deviation decrease up to a minimum value (bias instability). Note that the standard definition of bias instability[9] used by inertial sensor manufacturers is the minimum point on the Allan deviation curve. At some point, the Allan deviation starts to increase again. This is due to the acceleration/rate random walk (A/RRW) in the sensor – inherent instability in the output of the sensor.

According to Figure 10.15, we can say that the log-log plot for both Y-axis and Z-axis accelerometer are similar, the X-axis accelerometer behaves to some extend differently. All three accelerometers start with acceleration white noise, i.e. VRW, up to 1 s for both y and z accelerometer and up to ~ 40 s to the x accelerometer. Then, the y and z accelerometer present an exponentially correlated (Markov) noise in fact, the slope of the Allan standard deviation transits from + ½ to - ½ after 1s up to ~ 40 s. After the Markov noise for the y and z accelerometer, it follows a bias instability noise with some variations ending with random walk. The z accelerometer has a bias instability noise characteristic (40 ~ 100 [s]). The tail time region of the x, y and z accelerometer are quite similar, both have a slope of +1 (rate ramp noise characteristic).

According to Figure 10.16, the three rate gyros behave very similar for short cluster times, deferring for long cluster times. Each rate gyro starts with rate white noise, i.e. ARW, as the dominant noise for short cluster times. In fact, the initial slope of the curve which is approximately - ½ indicates that the predominant error on the rate output is wide-band noise (rate white noise/ARW, see Table 8.1 of Chapter 8) for the first 300 seconds. According to Chapter 8, if we integrate the rate output in order to get attitude, we will integrate wide band noise for the first 300 seconds, which correspond to an attitude error due to angle random walk.

According to Figure 10.16, it can be stated that the y rate gyro behaves worse than the x and z rate gyros, which may indicate malfunctioning of the y rate gyro. In fact, both x and z rate gyros ends with rate random walk noise while the y rate gyro with bias instability with some variations. In fact, we can take a chance and say that the rate white noise (ARW) almost dominates all the time ranges of the y rate gyro measurements.

[9] The best stability you could achieve with a fully modeled sensor and active bias estimation.

Overall, according to both Allan log-log plots, it can be stated that ARW, VRW and bias instability are found in the measurements from the three MMQ50 accelerometers and rate gyros. Thus, it is expected that after applying de-noise the inertial data, part of the noise in the signal will be removed.

In order to investigate the Allan standard deviation curve of static data after applying de-noise analysis, four hours of static data from MMQ50 were collected with a sample rate of 10Hz. First, the Allan variance method was applied to the whole dataset without any de-noise technique applied to the data. Next, the collected data was de-noised, applying the method discussed in Chapter 8; the obtained de-noised the data was then introduced into the Allan variance algorithm in order to obtain the correspondent curves.

Figure 10.17 depicts the root Allan variance of the x accelerometer channel whereas Figure 10.18 depicts the root Allan variance of the z rate gyro channel.

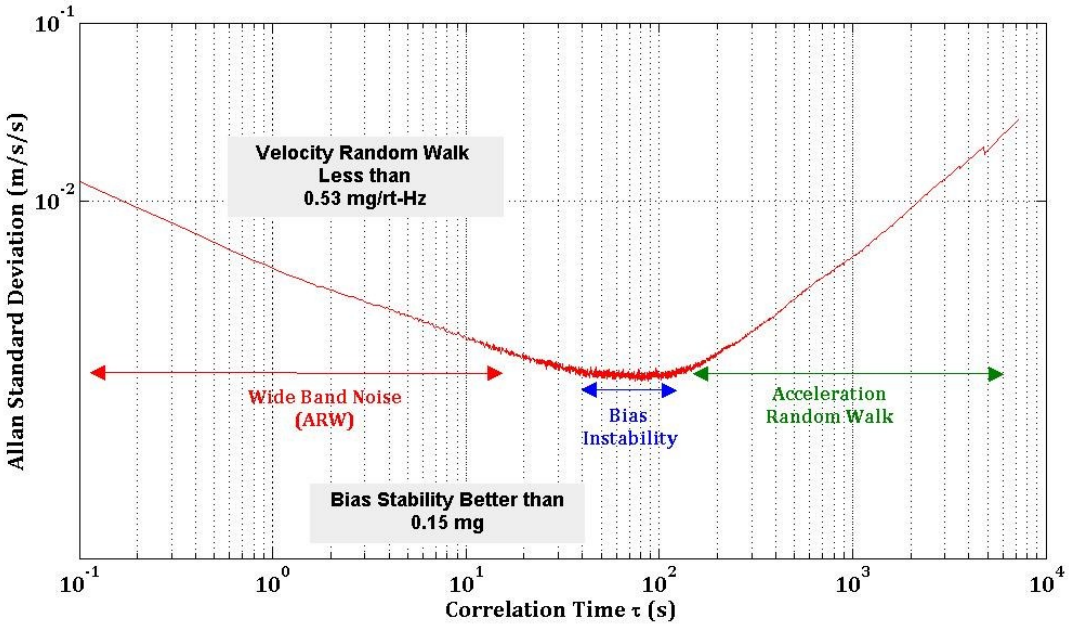

Figure 10.17 Allan Variance plot for MMQ50 X-axis accelerometer.

Figure 10.17 can be broken into three regions. The left most region – the region that corresponds to the short cluster times; high-frequency part; short-term region – of the data depicts a negative slope of about - ½ from where the velocity random walk value 0.53 $mg/\sqrt{Hz}$ is determined (see details about this parameter determination in Tables 8.1 and 8.2). The middle

region shows that the bias stability of the MMQ50 accelerometer is less than 0.15mg. Finally, the right most portion – the region that corresponds to the long cluster times; low-frequency part; long-term region – of the graph depicts the low acceleration random walk of the accelerometer.

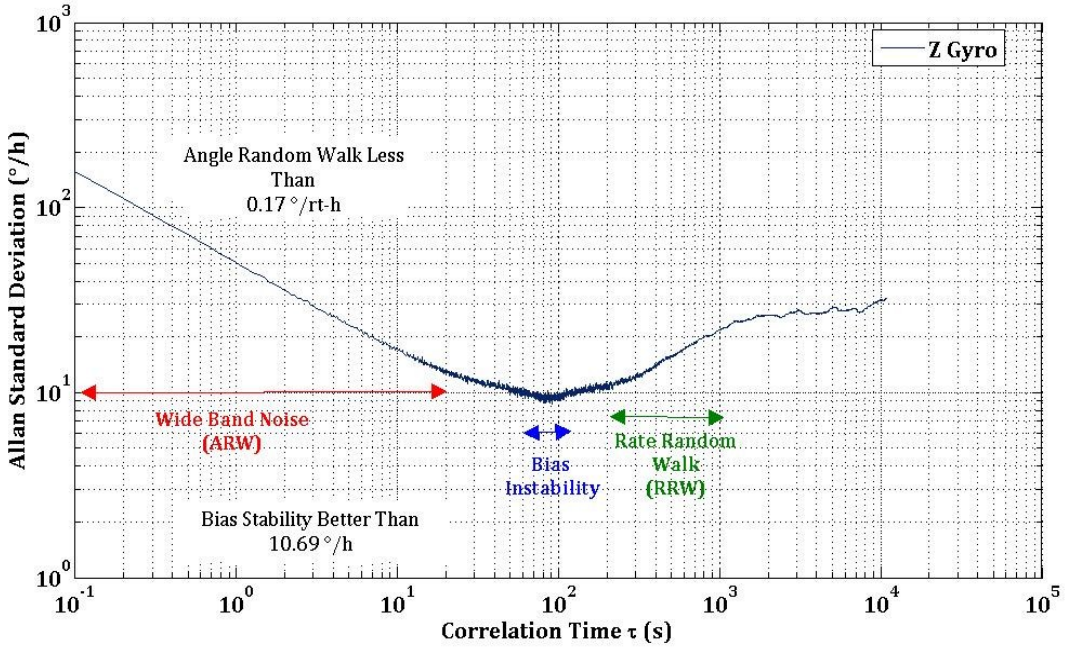

Figure 10.18 Allan Variance plot for MMQ50 z-axis rate gyro.

Figure 10.18 can be broken into three regions as well. The left most region of the data depicts a negative slope of about - ½ from where the angle random walk value 0.1569 °/$\sqrt{h}$ is determined. The middle region shows that the bias stability of the MMQ50 is less than 10.69 °/h for the rate gyro. The right most portion of the graph depicts the low rate random walk of the gyro.

According to both figures, we can say that the output error is a combination of wide band noise, bias instability (null shift/noise floor) and a time varying bias dominant by acceleration and/or rate random walk.

Therefore, the more predominant noise coefficients identified for both the accelerometer and rate gyro are listed in Table 10.2.

Table 10.2 Main noise coefficients for the MMQ50 sensors identified.

Inertial Sensor	Velocity/ Angular Random Walk (VRW/ARW) N	Bias Instability B	Acceleration/ Rate Random Walk R
Accelerometer	$0.53\ \mathrm{mg}/\sqrt{\mathrm{Hz}}$	$< 0.15\ mg$	$0.0257\ \mathrm{mg}.\sqrt{\mathrm{Hz}}$
Rate Gyro	$\approx 0.17\ °/\sqrt{\mathrm{h}}$	$< 10.69\ °.\sqrt{\mathrm{h}}$	$\approx 63\ °/\mathrm{h}/\sqrt{\mathrm{h}}$

From Table 10.2 we can say that being the MMQ50 ARW = $0.17\ °/\sqrt{\mathrm{h}} = 44.4\ °/\mathrm{h}/\sqrt{\mathrm{Hz}}$, then for 10 Hz bandwidth, the attitude error noise uncertainty given by:

$$\sigma = \mathrm{N}/\sqrt{\mathrm{T_s}} \qquad (10.2)$$

will be:

$$\sigma = 44.4\ °/\mathrm{h}/\sqrt{\mathrm{Hz}} \times \sqrt{10\mathrm{Hz}} = \ 140\ °/\mathrm{h}$$

On the other hand, being the VRW = $0.53\ \mathrm{mg}/\sqrt{\mathrm{Hz}}$, then the acceleration noise uncertainty equals:

$$\sigma = 0.53\ \mathrm{mg}/\sqrt{\mathrm{Hz}} \times \sqrt{10\mathrm{Hz}} = 1.67\ \mathrm{mg}$$

Currently, in order to illustrate how different levels of de-noising (LOD) affect the Allan variance characteristic obtained for both inertial sensors lets, apply the Allan variance to the previous dataset after de-noised. Actually, it is supposed to minimize the error due to wide-band noise and so it is expected to remove/diminish the angle/velocity random walk for the short clusters area of the Allan variance.

Thus, next step consists in to de-noise the data applying different LODs. According to the de-noise section, a db 9 with LOD = 5, 8, 10 and 12 is applied to the x accelerometer raw data and z rate gyro raw data. Results are show at Figures 10.19 and 10.20.

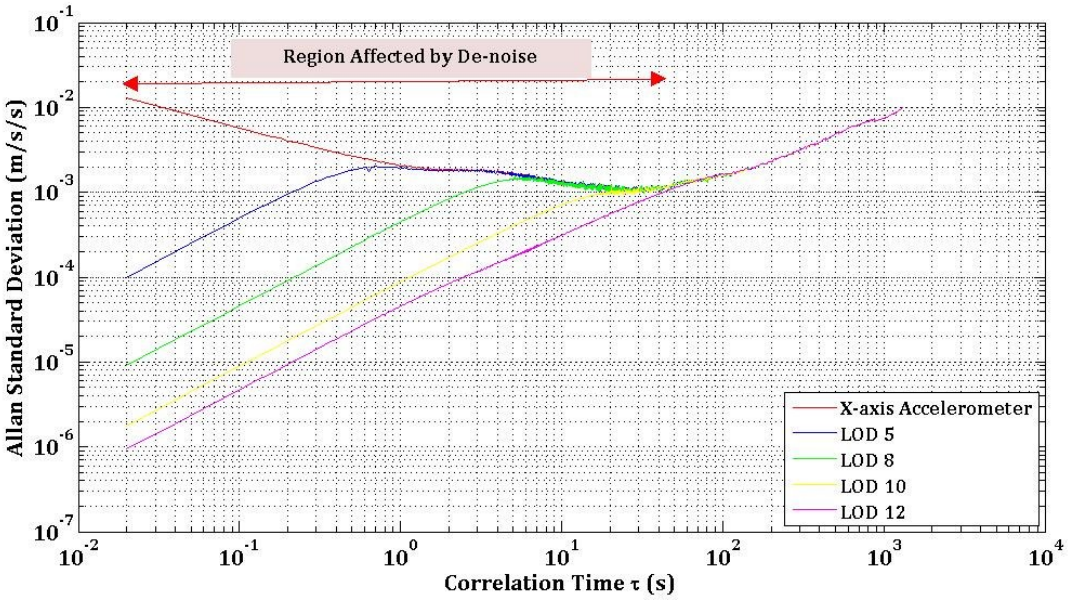

Figure 10.19 Allan Variance plot for MMQ50 X-axis accelerometer in static mode as a function of different lods (after de-noising).

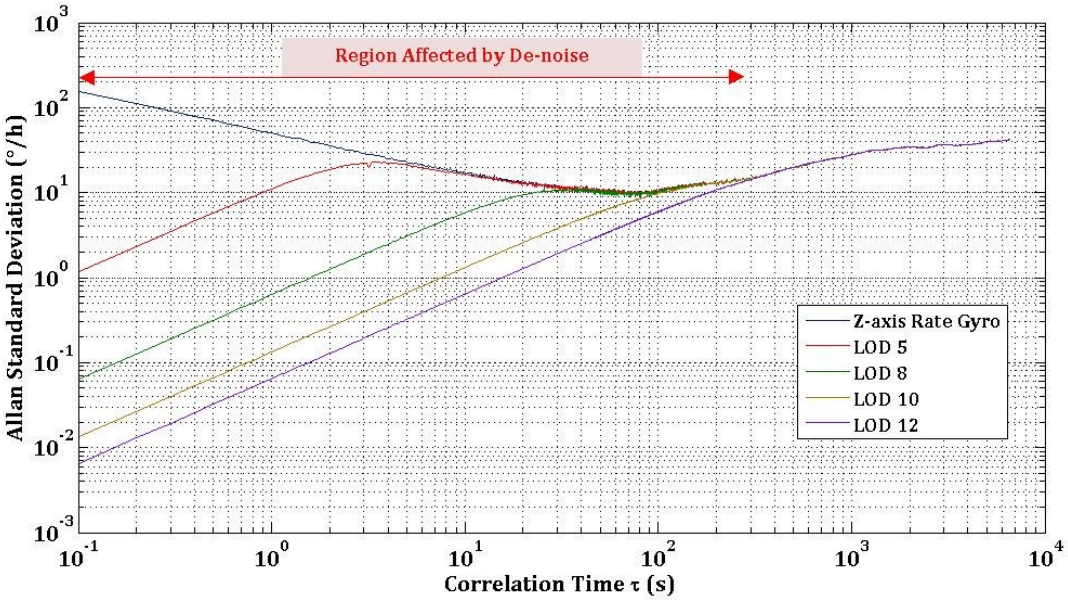

Figure 10.20 Allan Variance plot for MMQ50 Z-axis rate gyro in static mode as a function of different lods (after de-noising).

As it was expected the de-noise method that has been applied to the dataset has minimized the noise present in the short-term clusters. In fact, applying a fifth level of de-noising

to the z rate gyro raw data, the rate random walk, (+ ½ slope) replaces the rate white noise in the time resolution up to 4 s. The rate white noise dominates now in the time scale from 4 s to ~ 40 s. Increasing the LOD, the rate white noise is almost smoothed out being replaced by rate ramp noise characteristic.

As a conclusion, the characteristic of the original signal change with the level of de-noising changing. However, this change is always affecting the short cluster region. In conclusion, it can be stated that de-noise techniques can be used to reduce high frequency noise (short-term noise/wide band noise/measurement noise) having no effect in the low frequency region.

As a matter of fact, for low-performance MEMS-Based inertial sensors, where the predominant error is wide band noise, the steps presented here are considered suitable for any type of sensor belonging to this category. In fact, sensors of this category are corrupted with high frequency noise that makes them, to some extent, very similar when trying to model them. Moreover, it is also possible to conclude that random walks and bias instability are considered as the principal errors, and hence the Allan variance method can used to obtain coefficients of these errors.

10.4 Estimation of MMQ50 Model Parameter Values

In light of the above, the numerical values for the MMQ50 measurement model for the compensation form, Equations (9.11) and (9.12) are determined as follows.

10.4.1 Accelerometer and Gyro Constant Bias for the MMQ50 sensors: $\delta b_{Turn\_on_{acc}}$, $\delta b_{Turn\_on_{gyro}}$

The numerical values for the constant offset bias nominated by the author as $\delta b_{Turn\_on_{acc}}$ and $\delta b_{Turn\_on_{gyro}}$ will be achieve by simply doing the mean of the respective sensor output. Therefore, these values cannot be achieved as a generic value to be used in the model because they will be different for each turn-on. However, it is important to realize that these values are achieved – if in post-processing – after the sensor reaches a steady state output (the data being used must be of a sufficient length).

10.4.2 Accelerometer and Gyro Output Noise for the MMQ50 sensors: $\tilde{v}_{acc}, \tilde{v}_{gyro}$

The numerical values for the wide band noise terms $\tilde{v}_{acc}$, and $\tilde{v}_{gyro}$ are determined by achieving the standard deviation of the detrended accelerometer and gyro output when the sensors are not subjected to a zero-specific force input and zero-rate-input, respectively.

According to Figures 10.13 and 10.14 those values are:

Table 10.3 Wide band noise terms for the MMQ50 accelerometers and gyros.

Wide-band Noise		
$\tilde{v}_{acc}$	$E[\tilde{v}_{acc}^2] = \sigma_{acc}^2 f_s$	$\sigma_{acc} = 0.02\ m/s^2 \approx 2.04\ mg$
$\tilde{v}_{gyro}$	$E[\tilde{v}_{gyro}^2] = \sigma_{gyro}^2 f_s$	$\sigma_{gyro} = 0.046\ °/s \approx 166\ °/h$

10.4.3 Accelerometer and Gyro Bias Drift for the MMQ50 sensors: $\delta b_{drift_{acc}}, \delta b_{dirft_{gyro}}$

According to the analysis that has been done at Section 10.4, it can be stated that the time varying bias is dominant by angular/velocity random walk, bias instability and rate/acceleration random walk.

Moreover, taking into account the values presented at Table 10.3, and the Allan variance methodology presented at Chapter 8, the nature of the MMQ50 rate gyro Allan variance curve, for instance, suggests that the MMQ50 rate gyro can be modelled as:

$$\sigma(\tau) = \frac{B}{0.664} + \frac{60.N}{\sqrt{\tau}} + \frac{R}{60.\sqrt{3}}.\sqrt{\tau} \qquad (10.3)$$

Where:

B Bias instability coefficient.

N Angle random walk coefficient.

R Rate random walk coefficient.

The values for B, N, and R have been achieved taking into account the values presented at Table 10.2.

Figure 10.21 confronts the obtained simulated model applying Equation (10.3) with the experimental rate gyro data plot. According to Equation (10.3), the simulated model just includes angle random walk, bias instability and rate random walk, which is represented in blue in Figure 10.21.

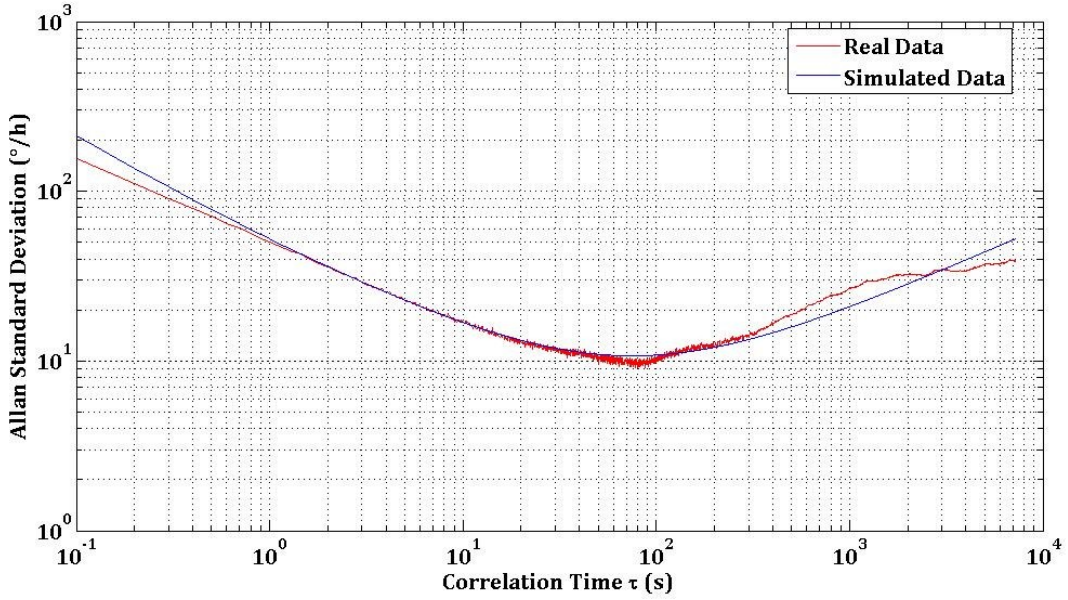

Figure 10- 10.21 MMQ50 Z rate gyro Allan variance and simulated Allan variance (model).

The results of the simulation show a good agreement between the two curves.

CHAPTER 11

PART III CONCLUSION SUMMARY

11 Part III Conclusion Summary

The **specific objectives** associated to Part III were: *i)* to present a clear picture of the inertial sensor signals and errors in order to clearly understand what those sensors really output; *ii)* To present the current state of the art, considering the appropriate methods/techniques, in the field of low performance inertial navigation sensors, associated to the improvement of sensors measurements accuracy; *iii)* To provide the reader with a methodology for developing general inertial measurements models, in the error and compensation form, for the low performance inertial sensors category; *iv)* to investigate the general performance characteristics of the MMQ50 using the Allan variance and wavelet multi-resolution analysis techniques; *v)* to develop and present a simple MMQ50 measurement model, in the error and compensation forms, and finally *vi)* to prove that it is possible to improve low performance inertial raw data before the strapdown mechanization equations.

The second and third secondary research questions associated to the first and second specific objectives of Part III, respectively, were the subject of study at Chapter 8.

In this chapter, we have the opportunity to observe that the task associated to the selection of the adequate IMU, to use in a specific integrated navigation system, is not as easy as it may seem. For instance, the low-performance MEMS-Based inertial sensors, in addition to their miniature size, low power consumption, and low price, have a fabrication process that makes them very sensitive to changes in the surrounding environmental conditions such as temperature, electric and magnetic fields, vibrations and pressure. Consequently, this high sensitivity adds more error types and consequently, higher errors than those of traditional non MEMS-Based inertial sensors. And so, there are no doubts that all the low performance inertial sensors suffer from relatively high measurement noise (wide band noise), which forces scientist to deal with the

problem of recovering the true (desired) signal from noisy data.

Regarding **Q2** of this study, given the fact that the common literature does not provide both easy and comprehensive understandable information about inertial navigation sensors signals and errors, and with the objective of giving an answer to this question, it was crucial to provide the reader with a complete and comprehensive description about inertial sensors signals and errors.

Also and in order to answer to **Q3**, it was also identified, in this chapter, what are the appropriate methods/techniques, in the field of low performance inertial navigation sensors, applied in search of a navigation solution, provided by low performance IMU sensors, with higher accuracy. Wavelet de-noising techniques and Allan variance analysis are the current state of the art of the methods/techniques applied to smooth raw inertial measurements in order to improve the performance of the strapdown algorithm and to identify what type of error source is underlying in the inertial raw data by performing certain operations on the entire length of data, respectively.

In light of the above, we are able to confirm that the specific objectives related to the presentation of a clear picture of inertial sensor signals and errors, in order to understand clearly what those sensors really output and to present the reader with a methodology for analysing error sources inherent to low performance inertial sensors category, have been achieved.

The fourth secondary research question of this study, associated to the third, fourth and fifth specific objectives of Part III, is subject to study at Chapter 9.

Usually a general error model may include several errors that are significant when inertial navigation systems are used for high or very high precision navigation; with low-performance sensors, the scenario is quite different. As a matter of fact, it does not make sense to consider the several error contributions when the output signal is mainly contaminated by wide band noise, even because it is impossible and expensive to account, model and estimate all the errors sources present in both the accelerometer and the gyro measurements. Thus, in conditions where commonly the low performance inertial sensors are used to be integrated with other sensors, a simple error model is used instead of more complex one. Therefore, in Chapter 9, from the general error model developed and presented at Table 9.1, and based on some sensor specifications, see Figures 9.1 and 9.2, a simple error model that includes just the dominant error sources usually present on such sensors category has been developed and presented, see Tables

9.2 and 9.3.

Given the above, and concerning **Q4** which asks how to deal with low performance inertial navigation sensors in the processing of sensor modelling, it can be stated that the strategy that has been presented along Chapter 9 is able to be used as an adequate and simple error model feasible to be applied to any IMU belonging to the low/medium grade category of inertial sensors. In fact, low performance IMU sensors of the same category of the MMQ50 are mainly corrupted with high frequency noise, which makes them, to some extended, similar when trying to model them. Nonetheless, it is considered crucial that some differences related to particular manufacturer specifications of each sensor must be taken into account during the modelling, according to Section 9.2 Figures 9.1 and 9.2.

In light of the above, it is confirmed that **Q4** is satisfactorily addressed, and we are able to confirm that the specific objectives related to provide the reader with a simple methodology for developing general inertial measurements models, in the error and compensation form, for the low performance inertial sensors category, and to investigate a simple MMQ50 measurements error model in the error and compensation form have been successfully addressed.

The fifth research question, associated to the sixth specific objective of Part III, is subject to study at Chapter 10.

The main objective of Chapter 10 is to determine the numerical values that satisfy the adopted MMQ50 sensors measurements models developed and presented at Chapter 9. In order to achieve that, the theory and methodology subjacent to WMRA and Allan variance presented at Chapter 8 are directly applied to the MMQ50 measurements. Therefore, the de-noising of the MMQ50 inertial static sensors data was performed using WMRA with a db wavelet and an LOD of ten applied to the output of the MMQ50 accelerometers and gyros in static mode. The results obtained shown smoothed effects in both sensors measurements. After de-noising the inertial measurements, a lag 1 scatter plot with the de-noised signals was performed in order to analyse the behaviour of the obtained signal after de-noise. The results showed that, instead of wide band noise, we have a correlated error, which according to the lag plot and the autocorrelation function has the behaviour of a correlated noise similar to a Gauss Markov process. In fact, the obtained results proved that the WMRA was able to decrease the wide band noise present in the inertial sensors measurement, highlighting the associated moving bias.

Next, the Allan variance analysis was applied to the inertial data, before and after de-

noise, in order to evaluate the nature of the dominant error sources present affecting the performance of the MMQ50.

The Allan standard deviation plot obtained before de-noising inertial raw data showed that for short averaging times, the Allan deviation is dominated by wide band noise – referred to as ARW for the gyros and VRW for the accelerometers – which confirms that the data are mostly uncorrelated noise. As we average over longer and longer times, the variance, and consequently the deviation decrease up to a minimum value, to the bias instability value. At some point, the Allan deviation starts to increase again due to A/RRW present in the sensor. Overall, we concluded that ARW/VRW, bias instability, and A/RRW are the most common error sources found in the MMQ50 sensor measurements.

Moreover, the nature of the Allan standard deviation curve obtained for the MMQ50 rate gyro, where the time varying bias is dominant by angular/velocity random walk, bias instability and rate/acceleration random walk, suggested that the gyro can be modelled according to Equation (10.3). The associated coefficient (B, N, and R) has been calculated and the simulation was generated, achieving a good agreement between the two curves: the experimental and the simulated.

Finally, the Allan variance technique was applied to the signals after de-noise. The results showed that the characteristic of the original signal change with the level of de-noising changing. However, this change is always affecting the short cluster region, in accordance with the fact that the measurement noise is present in the short term component. As a consequence, de-noise techniques can be used to reduce high frequency noise (short-term noise/wide band noise) having no effect in the low frequency region.

To conclude Chapter 10, the numerical values for the time constants, means, and variances for the various parameters that compose the adopted model developed at Chapter 9 are determined.

In respect of **Q5**, it can be concluded that WMRA applied to inertial data reduces part of the wide band noise underlying the sensors. As a consequence, the output signal can be smoothed highlighting the associated moving bias. The Allan variance analysis can be applied to the obtained signal in order to identify the characteristic in noise to be estimated into the Kalman filter. This leads to an improvement of the MMQ50 raw data prior to using the signal in the SINS developed at Part II.

Therefore, we are able to confirm that the specific objectives prove that it is possible to improve low performance inertial raw data before the strapdown mechanization equations have been achieved.

The **Hypotheses** under investigation in this Part are:

H2: *Low performance inertial navigation sensors signals can be improved, prior to feeding them into the SIMU algorithm, using denoised techniques which are useful to eliminate undesired high frequency components of the inertial signals.*

H3: *The Allan variance analysis is a good method to identify main error sources in the inertial sensors measurements, helping in the determination of relevant numerical values for modelling the sensors.*

H4: *A generic and satisfactory IMU Sensor error model for the MMQ50 IMU can be obtained using specific procedures, which can be applied to different sensor manufacturers of the same category.*

To achieve the specific objectives of Part III, four secondary research questions were defined: **Q2**, **Q3**, **Q4**, and **Q5**, and **H2**, **H3**, and **H4** were formulated.

Each secondary research question was answered successfully and so the hypotheses of Part III are confirmed with the proviso that the manufacturer specifications, related to the IMU in use, are taken into account during the modelling process, i.e., some differences can be found in Section 9.2.

In order to analyse the performance of an integrated navigation system with such model for the MMQ50 sensors, the implementation and analysis of a stochastic model in a loosely coupled IMU/GNSS navigation filter will be explained at Part IV.

PART IV

FILTERING AND INTEGRATION ARCHITECTURES

Absolute, true, and mathematical time, in and of itself and of its own nature, without reference to anything external, flows uniformly and by another name is called duration. Relative, apparent, and common time is any sensible and external measure (precise or imprecise) of duration by means of motion; such as a measure—for example, an hour, a day, a month, a year—is commonly used instead of true time.

— Sir Isaac Newton

CHAPTER 12

ESTIMATION METHODS

12 Estimation Methods

This part appears to answer the sixth secondary question of this research: *What are the more common estimation methods used for integrating inertial and GNSS data? How does one optimally estimate the interest parameters from noise corrupted data?*

This chapter provides an overview of the estimation methods used in this research. It should be stressed that this chapter intends only to present the essential equations that constitute the indirect Kalman filter and EKF. Thus, for more details of derivation of such equations it is recommended the work of (Grewal, et al., 2008).

12.1 Kalman Filtering

In systems that employ INS/SIMU technology, the long-term accuracy deteriorates, mainly due to integration of errors underlying the inertial sensor signals, as well as, computational errors. The problem associated to the trajectory reconstruction using inertial data can be viewed as a state estimation problem. In which, we have the mathematical model, the strapdown mechanization equations, describing the vehicle navigation and we have the raw data provided by the sensors. Given the fact that in the real world we have errors and noise, in order to overcome this problem, the source of errors are modelled as noise (w), added to the input and to the output of the system: being known as process noise and measurement noise. Therefore, associated to this random noise, the system is considered as a stochastic system, where there is the need to determine not only the states but also the uncertainty in the states to be determined.

In INS algorithms, usually the stated of interest consist on the errors of the vehicle's velocity, position, attitude, IMU sensor biases, and errors from other navigation systems. The Kalman filter is the most common technique used to estimate those states. To aid in the

estimation of the states, the KF requires two models: the process model and the observation model. The first one describes the propagation of the states, and the observation model describes the information supplied by a sensor as a function of the states, together with a model of measurement noise. INS error propagation models are the main process models used in most literature. Many common processes cannot be represented adequately using linear models. And so, the EKF is the most widely used approach for a nonlinear filter algorithm.

Up to now, a free inertial navigation algorithm has been developed and validated, as it was shown in Part II. In fact, in such system, the inertial sensors were providing the only input to the navigation system. In many applications, the free inertial performance of the INS is not sufficient to achieve all the navigation accuracy requirements. To compensate for the INS performance deficiencies, other navigation sensors have been utilized to attenuate the undesirable INS error characteristics (e.g., unbounded position error growth and position/velocity/attitude Schuler oscillations). The Kalman filtering has become a standard method use to blend navigation data from INS and other navigation sources in order to achieve a final solution that eliminates undesirable error characteristics of the INS while retaining high bandwidth/low noise output signal signature. The Kalman filtering technique is a general analytical process by which inaccessible system parameters can be estimated based on accessible measurements from the system outputs (Savage, 2007). As applied to the inertial navigation, Kalman filters have been utilized to estimate and correct inaccessible INS errors (e.g., position and attitude errors, inertial sensor errors) based on measurements of accessible INS parameters, compared to equivalent ones provided by alternative navigation data source (e.g., INS computed position compared to calculated position from a GPS receiver aiding device in equivalent formats). The procedure is to estimate and correct INS errors based on the measurements of certain INS parameters, compared to equivalent parameters from another navigation source. This comparison process is no more than a subtraction process that generates an error signal, called the measurement signal containing a composite of errors from both the INS and the navigation aid. Thus, processing the measurement through the Kalman filter allows for the errors that affect the measurement to be independently estimated. Consequently these error estimates are typically corrected through a control process. This control process is incorporated as part of the Kalman filter operations to continuously correct the INS error being estimated.

The considerable INS and aiding device error terms that one wants to account for in the

Kalman filter can be represented analytically through an "error state dynamic equation":

$$\dot{\underline{x}}(t) = A(t)\underline{x}(t) + G_P(t)\underline{w}_p(t) \tag{12.1}$$

Where:

$\underline{x}(t)$	Error states vector. Is a $n \times 1$ state vector.
$\underline{A}(t)$	Error state dynamic matrix. Is a $n \times n$ matrix.
$\underline{w}_p(t)$	Vector of independent white "process" noise sources driving $\underline{x}(t)$. Is a $p \times 1$ noise vector.
$G_P(t)$	Process noise dynamic coupling matrix that couples individual $\underline{w}_p((t)$ components into $\dot{\underline{x}}(t)$. Is a $n \times p$ noise input matrix.

The equivalent but discrete form of Equation (12.1) is given by:

$$x_k = \Phi_{k,k-1}x_{k-1} + G_{k-1}w_{k-1} \tag{12.2}$$

With:

$$\Phi_{k,k-1} \approx e^{F_{k-1}T} \approx I + F_{k-1}T + \frac{1}{2!}F_{k-1}^2T^2 + \frac{1}{3!}F_{k-1}^3T^3 + \cdots \quad \text{with } T = t_k - t_{k-1} \tag{12.3}$$

The discrete and no-linear form is given by:

$$x_k = f(x_{k-1}, u_{k-1}, k) + G_{k-1}w_{k-1} \tag{12.4}$$

To compensate for the SIMU errors, external measurements, "Z", of better accuracy must be used. The considered measurements, from an aiding device, may be related to some or all the SIMU states considered at Equation (12.1). Moreover, errors associated with the aiding device would also be included in the error state dynamic equation. As a consequence, the navigation

error equation for the aiding device considered must be developed in such a way that they have to fit the format of the above equation as well.

The comparison of the SIMU states (or parameters) with equivalent states provided by the aiding device can be linearly modelled by a "measurement equation" given by:

$$\underline{Z}_n = H_n \underline{x}_n + \underline{v}_n \tag{12.5}$$

Where:

$\underline{Z}$	Measurement vector. See Equation (12.6). Is an m vector.
$()_n$	() at the n^{th} Kalman filter cycle time.
H	Measurement matrix. Is the matrix that gives the relationship between the observations vector "Z" and the state vector $\underline{x}_n$. Is a $m \times n$ matrix.
$\underline{v}_n$	Measurement noise. Is a $m \times 1$ state vector.

The correspondent discrete form, linear and no-linear is given by:

$$\begin{aligned} \underline{Z}_k &= H_k \underline{x}_k + \underline{v}_k \\ \underline{Z}_k &= h(\underline{x}_k, k) + \underline{v}_k \end{aligned} \tag{12.6}$$

The measurement equations presented above result from a linearized version of a general nonlinear "observation equation":

$$\underline{Z}_{Obs_n} = f(\xi_{SIMU_n}, \xi_{Aid_n}) \tag{12.7}$$

Where:

ξ_{SIMU}	SIMU navigation parameters.
ξ_{Aid}	Aiding sensor navigation parameters.
f( )	Functional operator that compares designated equivalent elements of ξ_{INS} and ξ_{Aid} (designed such that it will be zero for an error free system).
$\underline{Z}_{Obs}$	Observation vector resulting from the comparison between SIMU and aiding sensor parameters (would be also zero for an error free system).

The variables $\underline{w}_{\mathrm{k}}$ and $\underline{v}_{\mathrm{k}}$ represent the process and measurement noise, respectively, which are uncorrelated. Thus, the system noise covariance Q and the measurement noise covariance R are given as:

$$\begin{gathered} \mathrm{p}(w) \sim N(0, Q) \\ R(v) \sim N(0, R) \\ \mathrm{E}[w_k w_k^T] = \mathrm{Q_k};\ \mathrm{E}[v_k v_k^T] = \mathrm{R_k} \text{ and } \mathrm{E}\left[w_j v_k^T\right] = \\ 0 \ \text{For all k and j} \end{gathered} \tag{12.8}$$

$$\mathrm{E}[(\hat{\mathrm{x}}_{\mathrm{k}} - \mathrm{x}_{\mathrm{k}})(\hat{\mathrm{x}}_{\mathrm{k}} - \mathrm{x}_{\mathrm{k}})^{\mathrm{T}}] = \mathrm{P_k}$$

Where $\mathrm{P_k}$ corresponds to the error covariance matrix of the estimate of the error state vector to be considered in the application.

The KF equations fall in two groups: measurement update (update phase also called the correction) and the time update (prediction phase).

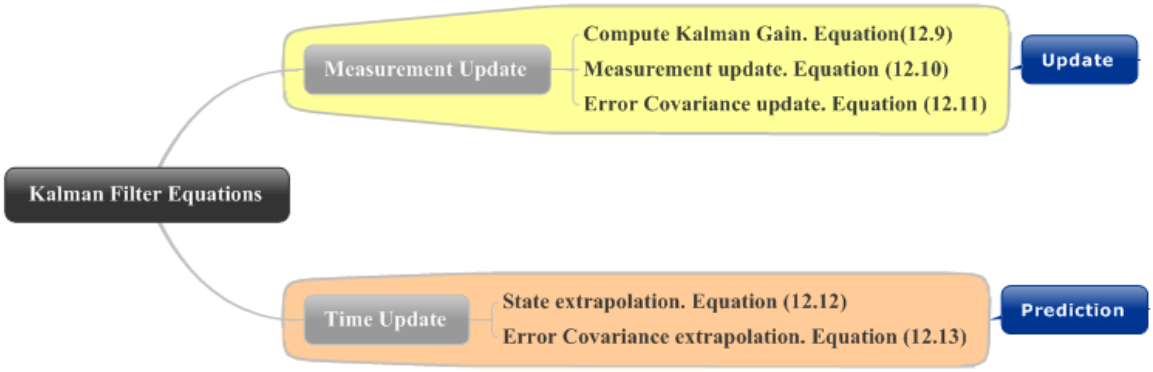

Figure 12.1 Kalman filter equations.

In the measurement update, the available measurements are used to correct the estimated state (measurement update) and the error covariance. In the time update, the two quantities are projected forward with the knowledge of the system model.

According to Figure 12.1, three steps compose the measurement update, whose related equations are summarized in the following set of equations:

Measurement Update

Kalman Gain:

$$K_k = P_k(-)H_k^T\left[H_k P_k(-)H_k^T + R_k\right]^{-1} \qquad (12.9)$$

Measurement Update:

$$\hat{x}_k(+) = \hat{x}_k(-) + K_k[z_k - H_k\hat{x}_k(-)] \qquad (12.10)$$

Error Covariance Update:

$$P_k(+) = [I - K_k H_k]P_k(-) \qquad (12.11)$$

Where:

$(+), (-)$	Notations that stands for a *posterior* and for a *priori*, respectively.
$[z_k - H_k\hat{x}_k(-)]$	Defined as the measurement Innovation.
$\hat{x}_k(+), \hat{x}_k(-)$	*Posteriori* and *priori* states estimate, respectively.
$P_k(+), P_k(-)$	*Posteriori* and *priori* error covariance estimate, respectively.

Again, according to the same figure, the time update is composed by two steps, whose objective is to predict the a *priori* estimate of state and respective error covariance, reflected in the following equations:

Time Update

State Extrapolation:

$$\hat{x}_{k+1}(-) = \Phi_k\hat{x}_k(-) \tag{12.12}$$

Error Covariance Extrapolation:

$$P_{k+1}(-) = \Phi_k P_k \Phi_k^T + Q_k \tag{12.13}$$

Equations (12.9) to (12.12) conclude the DKF algorithm, which is represented at Figure 12.2.

It is important to realize the influences of the system noise matrix, Q, and the measurement noise matrix R.

The Q matrix provides the statistical description of the error model. A large value for Q indicates increased parameter uncertainty and results in noisy estimates, bad estimates. Automatically, with large values for Q, in the prediction, we will have uncertainty in the SIMU data that will grow, which implies that the SIMU will closely follow the GNSS position estimates, which can also be faulty if the estimates are noisy. That is why usually small values are chosen for the Q matrix.

The R matrix tells us how well the measurement noise is modeled. Imperfect modeling of

the measurement observable noise leads to bad estimation quality. Large values for R reflect inaccurate and noisy measurements and might not correct the SIUM sufficiently, while small values imply accurate measurements, which cannot be true, relying more on the measured data than on the model.

In fact, the choice of the two matrixes is a compromise that one should take into account during the process of fusing the data provided by the several sensors.

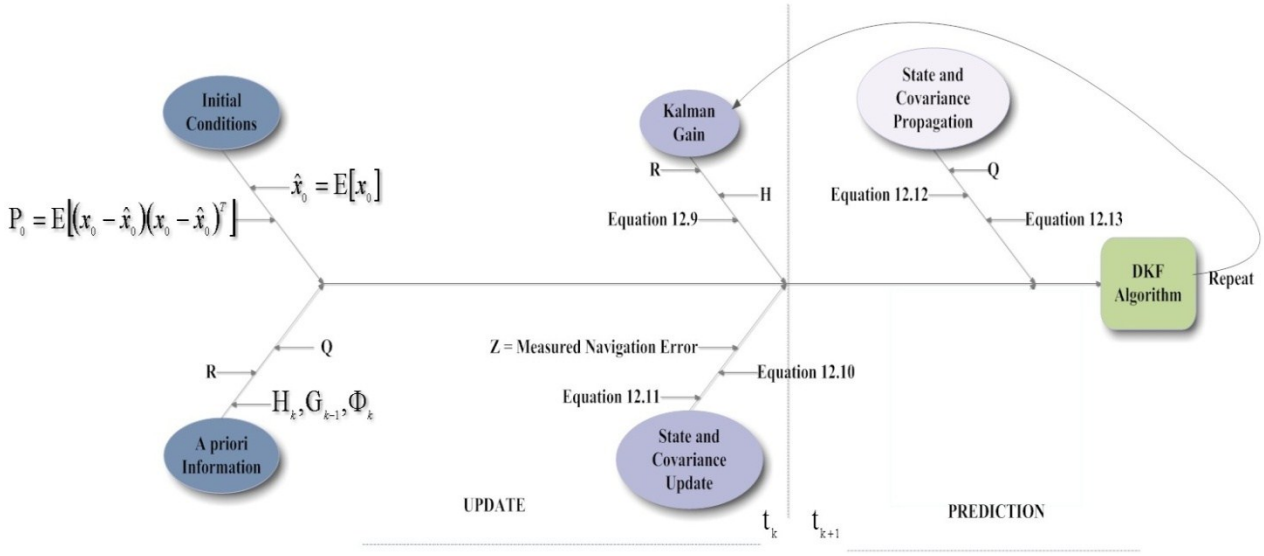

Figure 12.2 DKF algorithm.

12.2 Extended Kalman Filter

Given the fact that the EKF admits nonlinearities in both the system dynamics and measurements equations, while retaining the linear additivity of the driving uncertainties, it is said to be an extension ((Maybeck, 1982).

The nonlinear system model equations, as well as, the nonlinear measurement equations are given by Equations (12.4) and (12.6), respectively.

It should be noted that both the system and the measurement model, although non-linear, are assumed to be approximately linear for small perturbations in the state variables. The main objective of the EKF will be to exploit this linearity around a nominal state trajectory in order to apply the linear estimation concepts. The EKF uses the estimate states $\hat{x}$ as the nominal trajectory, i.e $x = \hat{x}_k$.

The measurement and time update equations for the EKF are given by:

Measurement Update

Kalman Gain:

$$K_k = P_k(-)H_k^{*T}\left[H_k^* P_k(-)H_k^{*T} + R_k\right]^{-1} \tag{12.14}$$

Measurement Update:

$$\hat{x}_k(+) = \hat{x}_k(-) + K_k[z_k - H_k\hat{x}_k(-)] \tag{12.15}$$

Error Covariance Update:

$$P_k(+) = [I - K_k H_k^*]P_k(-) \tag{12.16}$$

Time Update

State Extrapolation:

$$\hat{x}_{k+1}(-) = f_k\hat{x}_k(-) \tag{12.17}$$

Error Covariance Extrapolation:

$$P_{k+1}(-) = \Phi_k^* P_k \Phi_k^{*T} + Q_k \tag{12.18}$$

Where H_k^* and Φ_k^* represent linear approximations terms computed as follows:

$$H_k^* = \frac{\partial h(x,k)}{\partial x}\Big|_{x=\hat{x}_k(-)};\ \Phi_k^* = \frac{\partial f(x,k)}{\partial x}\Big|_{x=\hat{x}_k(-)} \tag{12.19}$$

Equations (12.4) to (12.19) conclude the DEKF algorithm to be used in this research.

CHAPTER 13

AIDING

13 Aiding

As it has been clear, the INS navigation inevitably suffers from unbounded errors in the provided navigation solution. Even supposing that for some applications the associated errors are not a problem, there are other applications where they can be unacceptable. As a consequence, some type of aiding is needed to either bound or reduce these errors.

Different forms of aiding can be used, with diverse results. Some of the most popular aiding devices are the GNSS (more commonly GPS) receivers, magnetometer and barometric altimeter. GPS measurements are widely used in such a type of project. Usually, the measurements are provided by a GPS receiver, while in this thesis it is intend to use GPS and Galileo measurements provided by a $GNSS_{SR}$.

This Chapter intends to give a general description of the use of a GNSS device as an aiding system that aims to bind the position error. Section 13.2 describes some different ways to implement an integration algorithm to fuse the data provided by an SIMU and a GNSS system. An error model of the $GNSS_{SR}$ does not need to be derived, given the fact that a loosely coupled strategy will be implemented.

It should be remarked that only the necessary information about GNSS is here presented. For more details about this topic, it is recommended that one reads (Farrell, et al., 2001), (Groves, 2008) and (Prasad, et al., 2005).

13.1 Aiding with GNSS

Why GNSS measurements? In fact, the idea behind the GNSS as an aiding device was originated in the fact that both, the GNSS receiver and INS/SIMU system, provide the same type of output, position and velocity, but with the particularity that those measurements are provided

in a different manner and suffering from different error characteristics.

According to Part II of this thesis, an SIMU estimates position and velocity by measuring accelerations and angular rates. Also, a GNSS receiver provides position and velocity information directly.

However, given the fact that position and velocity are obtained by double integrating the inertial sensors quantities, this means that even though in short-term the SIMU solution will have an adequate performance, the same will not happen with the long-term error of the SIMU solution, which is time-dependent and unbounded. A GNSS receiver, on the other hand, provides a long-term bounded error, though it experiences short-term error drift mainly due to outages. As a matter of fact, the use of GNSS devices is recognized as the ideal system complement to the INS/SIMU.

Independently of which GNSS receiver we are working with, i.e. GPS, Galileo, or GLONASS, the fact is that all determine the user position in the same manner. The user position is calculated through range measurements based on the Time of Arrival (TOA) of the signal that has travelled from the different satellites to the user receiver. The corresponding TOA will be given by:

$$\text{TOA} = \text{Time instant of Arrival} - \text{Time instant of Transmission} \tag{13.1}$$

As a consequence, and taking into account that the GNSS signals are carried by radio waves in the microwave part of the electromagnetic spectrum, they travel at the speed of light (c), allowing us to calculate the distance between the receiver and the satellite by just multiplying the TOA by c as follows:

$$\text{Distance} = \text{TOA} \times \text{c} \tag{13.2}$$

Looking to Equation (13.2), in order to get the true distance between the satellite and the receiver, it is clear that both the receiver and satellite clocks must be synchronized. In practice, it is impossible to get the synchronization between clocks. This is why the distance assumes the name of pseudo range.

Therefore, the pseudo range will be obtained using Equation (13.3):

$$\rho = \|S_i - u\| + c\Delta t \qquad (13.3)$$

Where:

ρ	Pseudo range from the i´th satellite to the GNSS receiver antenna.
S_i	Position of the i´th satellite. It should be noted that in order to calculate the user position at least four satellites are needed.
u	User GNSS receiver antenna position. The unknown position vector.
Δt	Offset between clocks (satellite and receiver).

The user position will be then calculated as follows:

$$\rho_i = \sqrt{(x_i - x_u)^2 + (y_i - y_u)^2 + (z_i - z_u)^2} + c\Delta t \qquad (13.4)$$

Considering that at least four satellites are needed to compute the position, we will have: i = 1,2,3,4 and four unknowns which are the user position x_u, y_u, z_u and the clock offset Δt.

Given the fact that in this research a loosely coupled integration is to be used, it does not make any sense to develop the GNSS error model. In fact, no error model is needed as it is the position and velocity from the $GNSS_{SR}$ that will be used to get the observation set of the specific $GNSS_{SR}$. The precision of the $GNSS_{SR}$ provided by the data sheet will be used as measurement noise.

The focus of the next section goes to the integration architectures for the SIMU/$GNSS_{SR}$.

13.2 IMU/$GNSS_{SR}$ INTEGRATION ARCHITECTURES

Although there is no universal agreement about formal definitions for the different INS/GNSS or SIMU/GNSS integration architectures, the fact is that it can be accepted all vary in three points (Groves, 2008):

What types of GNSS measurements are used in the integration?

How are the computed corrections applied to the navigator solution?

How the GNSS user equipment is aided or not by the INS and integration algorithm?

Independently of which method we are going to use, they have a common point which is related to the fact that all of them use the IMU data to provide the reference trajectory, while the GPS data is used as the updating system. This is mainly due to the fact that the INS measurement frequency is many times higher than that of the GPS. So, the state vector parameters can be determined with high frequency but only updated, i.e. corrected/compensated, at a lower frequency.

In this thesis, the most widely used definitions are adopted: loosely coupled, tightly coupled, and deep integration. The definitions of each integration architecture are associated to the type of GNSS measurements being used, the first question pointed out before. These are the so called INS/SIMU/GNSS coupling approaches which are individually and briefly defined as follows.

Loosely Coupled

In such architecture, an SIMU/GNSS integration algorithm will use the GNSS position and velocity solution as the measurements inputs to the developed integration algorithm, irrespective of the type of INS/SIMU correction or GNSS aiding used (Groves, 2008). It is a cascaded architecture, where the GNSS user equipment being used incorporates its own navigation filter.

Tightly Coupled

This system uses the GNSS pseudo-range and pseudo-range-rate, delta-range measurements as inputs to the integration algorithm, again irrespective of the type of INS/SIMU corrections or GNSS aiding used.

Deep Integration

In this system, there is a combination of INS/GNSS and GNSS signal tracking into a single estimation algorithm (Groves, 2008). This architecture involves integration and interaction between both the INS and GNSS at the hardware and software levels.

According to the other two points from which the architecture differs, let's talk about the different methods of correcting the navigator solution. Indifferent to what type of GNSS measurements are used and how the GNSS user equipment is aided or not, the correction in a conventional integration architecture, using an error-state Kalman filter, can be either open-loop or closed-loop.

In both configurations, the estimated position, velocity and attitude errors are used to correct the inertial navigation solution, so where lies the difference? In fact, while in the open-loop, the corrections are done within the integration algorithm at each iteration but not fedback to the inertial navigator; in the closed-loop configuration, the errors are fedback to the inertial navigator, either on the Kalman filter iteration or periodically, where they are used to correct the inertial navigation solution itself, see Figure 13.1. As a consequence of the first configuration, only the integrated navigation solution contains the Kalman filter estimates. In the closed-loop configuration, the Kalman filter's position, velocity and attitude estimates are zeroed after each set of corrections is fedback (Groves, 2008). As a consequence and as it happens in the first configuration, here, no independent uncorrected inertial navigation solution exists.

Moreover, in the closed-loop configuration, the accelerometer and gyro errors estimated by the Kalman filter are also fedback to correct the IMU raw data using the compensation form Equations (14.20) and (14.21), as they are input to the inertial navigator developed at Chapter 6. It should be stressed that contrary to the position, velocity, and attitude corrections, the accelerometer and gyro corrections must be applied to every iteration of the navigation equations, with feedback from the Kalman filter periodically updating the accelerometer and gyro errors (Groves, 2008).

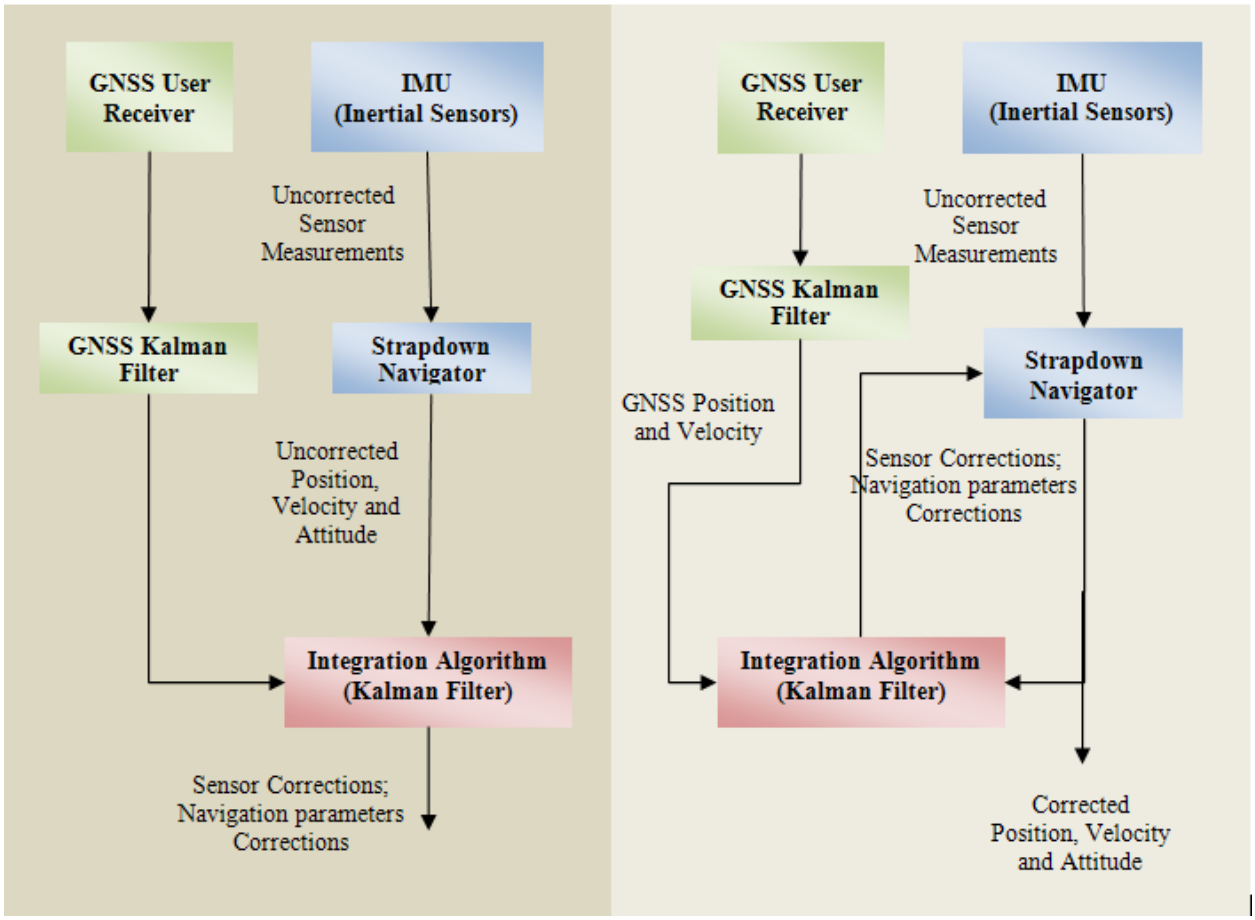

Figure 13.1 Open loop Versus Closed-loop.

The closed loop is called the “feedback” while the open loop the “feed forward”. In fact, in the closed loop, the bias estimated is fedback to the raw sensor output and used in the known error compensation form. In the open case, no error estimates, misalignment errors or sensor bias are fedback to the mechanization equation or raw sensor output. Consequently the errors will grow more rapidly. This is why it is only possible for high end inertial sensors. Instead, the closed loop is used where sensor errors are large, as it is the case of low cost MEMS-Based IMU.

Important to realize is the fact that the choice of open or closed-loop SIMU/GNSS integration is a function of both the IMU quality and the integration algorithm quality (Groves, 2008). In this thesis, a low performance IMU is being used and so only the close-loop configuration is suitable, regardless of the integration algorithm quality.

In the open-loop, the inertial sensor errors remain uncorrected and inertial navigation state errors grow rapidly, introducing large errors in the integrated system. Here is the reason

why the open loop mode can only be used for high end inertial sensors, which are characterized by small errors that will not influence the precision of the solution given the application in study. For low-cost or MEMS based inertial systems, the closed-loop is necessary (Godha, 2006).

Finally, an alternative to an error-state Kalman filter in an INS/SIMU/GNSS integration is a total-state Kalman filter, which instead of the navigation solution error estimates absolute position, velocity, and attitude. In fact, Maybeck (Maybeck, 1982) identified two different methods to fuse the data of INS and other sensor such as GPS, called *indirect* (error-state Kalman filter) filtering and *direct* (total-state Kalman filter) filtering. The open and closed-loop configurations are both indirect filtering due to the fact that both methods model the errors of the INS/SIMU.

13.3 Loosely coupled Close Loop SIMU/GNSS$_{SR}$ Adopted

Given the fact that we are working with low performance IMU, with an autopilot that uses low performance sensors as well, and also given its simplicity of implementation, the loosely-coupled closed loop has been selected as the integration strategy for this study.

In the loose coupled strategy here present, the main aim of the filter is to estimate the errors in the IMU sensors and SIMU solution, using for that the GNSS (GPS and/or Galileo) solution as an external aiding. The GPS and/or Galileo position and velocity process the GPS and/or Galileo raw measurements through a GPS and Galileo Kalman filter.

The IMU/GPS/Galileo raw data is integrated in a post-processing mode and evaluated using the signals provided at the GATE test area. It should be stressed that the GPS/Galileo raw data is provided by the ipexSR software receiver developed by the privately owned company IFEN GmbH, working in the field of satellite navigation. The ipexSR is a high end, multi-frequency and real-time capable software receiver. It is developed in C++, which makes it flexible to be run on a conventional PC and simple to be combined with other applications. Above all, the IMU/GPS/Galileo integration is achieved by employing an EKF in an error state feedback configuration.

Figure 13.2 shows the integration architecture that has been adopted for this thesis. As it is possible to see in the figure, the GPS and Galileo position and velocity are the inputs as measurements to the integration algorithm, the EKF.

The selection of the integration Kalman filter gain and measurement iteration rate are

critical: if measurements are processed too quickly, it leads to a possible unstable filter; in opposition, if measurements are processed too slowly, the observability of the INS errors is reduced (Groves, 2008), and so there is always a compromise that must be account for. Usually the system is tuned so that the integration Kalman filter bandwidth is always less than that of the GNSS Kalman filter. Normally measurement-update intervals of the order of 10 seconds are the typical in loosely coupled systems.

We have, also, the frequency associated to the mechanization equations computation. In fact, the update frequency has to be chosen according to the dynamics expected for the vehicle. Normally, for an UAV, 100 Hz is selected. For a land vehicle, 1 or 10 Hz is considered acceptable. Mainly, it depends on the dynamics expected for the vehicle.

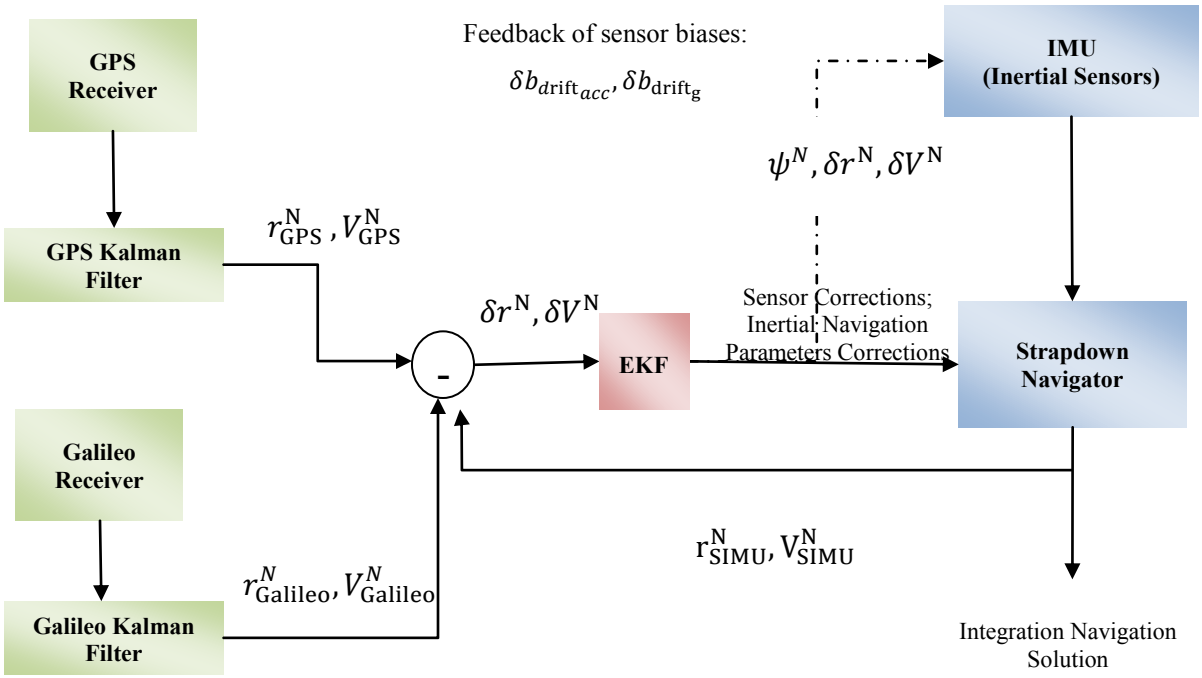

Figure 13.2 Adopted loosely coupled closed-loop SIMU/GNSS integration (adapted from (Groves, 2008)).

CHAPTER 14

SIMU/GNSS$_{SR}$ EKF IMPLEMENTATION AND SPECIFICATIONS

14 SIMU/GNSS$_{SR}$ EKF Specifications

As next, the system (process) and measurements models for the EKF developed are presented.

14.1 State and SIMU/GNSS$_{SR}$ System Model Selected

According to Chapter 12, in order to implement an EKF, we need to develop the system and the measurement model. According to the block diagram presented at Figure 13.2, we want to estimate state errors. No interaction exists between the SIMU and GNSS states in the system model (they only interact through the measurement model). The SIMU solution is used as the reference solution, and the GNSS$_{SR}$ solution is used as the aiding. Each time a GNSS$_{SR}$ update is available, the SIMU errors are "reset" and the SIMU provides a corrected solution at its output.

The EKF will be developed taking into account the EKF equations presented at Chapter 12, as well as, the SIMU error equations introduced in Chapter 6. Therefore, the SIMU system (or process) model is given by:

System Model

The basic system state vector is composed of nine navigation parameter errors: the three attitude errors; the three velocity errors and the three position errors. The error behaviour is expressed by Equation (6.60) which is here reported:

$$\delta\dot{\underline{x}}^{N} = \begin{bmatrix} \dot{\underline{\psi}}^{N} \\ \delta\dot{\underline{v}}^{N} \\ \delta\dot{\underline{R}}^{N} \end{bmatrix} = \begin{bmatrix} F_{\psi\psi} & F_{\psi v} & F_{\psi r} \\ F_{v\psi} & F_{vv} & F_{vr} \\ F_{r\psi} & F_{rv} & F_{rr} \end{bmatrix} . \begin{bmatrix} \underline{\psi}^{N} \\ \delta\underline{v}^{N} \\ \delta\underline{R}^{N} \end{bmatrix} + \begin{bmatrix} C_{B}^{N} & 0_{3\times3} \\ 0_{3\times3} & C_{B}^{N} \\ 0_{3\times3} & 0_{3\times3} \end{bmatrix} \begin{bmatrix} \delta\omega_{IB}^{B} \\ \delta a_{SF}^{B} \\ 0_{3\times1} \end{bmatrix} \quad (14.1)$$

Confronting Equation (14.1) with Equation (12.4), we can write the system model in the same fashion as:

$$\delta\dot{\underline{x}}^{N} = F\,\delta\underline{x}^{N} + G\underline{w} \quad (14.2)$$

Where:

$F = \begin{bmatrix} F_{\psi\psi} & F_{\psi v} & F_{\psi r} \\ F_{v\psi} & F_{vv} & F_{vr} \\ F_{r\psi} & F_{rv} & F_{rr} \end{bmatrix}$ Dynamic matrix.

$\delta\underline{x}^{N} = \left[\underline{\psi}^{N} \quad \delta\underline{v}^{N} \quad \delta\underline{r}^{N}\right]^{T}$ State vector composed by: $\underline{\psi}^{N} = [\psi^{E}\ \psi^{N}\ \psi^{U}]$; $\delta\underline{v}^{N} = [\delta v^{E} \quad \delta v^{N} \quad \delta v^{U}]$ and $\delta\underline{R}^{N} = [\delta l \quad \delta L \quad \delta h]$.

$G = \begin{bmatrix} C_{B}^{N} & 0_{3\times3} \\ 0_{3\times3} & C_{B}^{N} \\ 0_{3\times3} & 0_{3\times3} \end{bmatrix}$ Shaping matrix.

$\underline{w} = [\delta\omega_{IB}^{B} \quad \delta a_{SF}^{B}]$ Forcing vector, consisting of the errors in the measured angular rates and accelerations, respectively.

Given the fact that the inertial measurement errors do not meet the KF assumptions (zero-mean white noise and Gaussian behaviour), Equation (14.1) cannot be used as the process model in the EKF. Therefore, extra states should be added to the SIMU error model given by Equation (14.1). It is expected, with extra states, to account for the effects of the inertial sensor errors, originating what is commonly called the state augmentation process model, which is presented next.

14.2 Augmented System Model in the N-Frame

14.2.1 Gyro and Accelerometers Error Terms

According to the study presented at Part III Chapter 9, we are able to augment the error state provided by (14.1). The uncertainty of the sensors can be expressed as the sum of two terms:

$$\begin{aligned} \delta \underline{a}_{SF}^{B} &= \delta \underline{f}_{IB}^{B} = \delta b_{turn-on_{acc}} + \delta b_{drift_{acc}} + \tilde{v}_{acc} \\ \delta \underline{\omega}_{IB}^{B} &= \delta b_{turn-on_{gyro}} + \delta b_{drift_{gyro}} + \tilde{v}_{gyro} \end{aligned} \tag{14.3}$$

Where the bias turn on $\delta b_{turn-on_{acc}}$ and $\delta b_{turn-on_{gyro}}$ are calculated once at the begining of the post-processing (compensated at the beginning) analysis while, the evolution of the sensor errors, i.e., the accelerometer and gyro bias drift can be modelled as a first order Gauss Markov process, that can be expressed as follows:

$$\begin{aligned} \dot{\delta b}_{drift_{acc}} &= -\frac{1}{T_{ba}} \delta b_{drift_{acc}} + \sqrt{\frac{2}{T_{bacc}} \sigma_{b_{acc}}^2} \\ \dot{\delta b}_{drift_{gyro}} &= -\frac{1}{T_{bg}} \delta b_{drift_{gyro}} + \sqrt{\frac{2}{T_{bgyro}} \sigma_{b_{gyro}}^2} \end{aligned} \tag{14.4}$$

Where T corresponds to the correlation time of the process (accelerometer and gyro bias) and $\sigma_{acc}^2, and\ \sigma_{gyro}^2$ to the associate variance of the process. Note that those values have been determined at Chapter 9 and so for more details, the reader should consult Part III. Moreover, although different noise values could be achieved for the different axes of both types of sensors, it is here assumed that all the gyros and accelerometers have equal noise characteristics. It is assumed that they have equal and uncorrelated variance and correlation time.

Therefore, in the Matrix form, we will have:

$$\delta\dot{b}_{drift_{acc}} = \begin{bmatrix} \delta\dot{b}_{d_{accx}} \\ \delta\dot{b}_{d_{accy}} \\ \delta\dot{b}_{d_{accz}} \end{bmatrix} = \begin{bmatrix} -\frac{1}{T_{bacc}} & 0 & 0 \\ 0 & -\frac{1}{T_{bacc}} & 0 \\ 0 & 0 & -\frac{1}{T_{bacc}} \end{bmatrix} \begin{bmatrix} \delta b_{d_{accx}} \\ \delta b_{d_{accy}} \\ \delta b_{d_{accz}} \end{bmatrix} + \begin{bmatrix} \sqrt{\frac{2}{T_{bacc}}\sigma^2_{b_{acc}}} \\ \sqrt{\frac{2}{T_{bacc}}\sigma^2_{b_{acc}}} \\ \sqrt{\frac{2}{T_{bacc}}\sigma^2_{b_{acc}}} \end{bmatrix} \underline{w}_{acc} \qquad (14.5)$$

$$\delta\dot{b}_{drift_{gyro}} = \begin{bmatrix} \delta\dot{b}_{d_{gyrox}} \\ \delta\dot{b}_{d_{gyroy}} \\ \delta\dot{b}_{d_{gyroz}} \end{bmatrix} = \begin{bmatrix} \frac{-1}{T_{bgyro}} & 0 & 0 \\ 0 & \frac{-1}{T_{bgyro}} & 0 \\ 0 & 0 & \frac{-1}{T_{bgyro}} \end{bmatrix} \begin{bmatrix} \delta b_{d_{gyrox}} \\ \delta b_{d_{gyroy}} \\ \delta b_{d_{gyroz}} \end{bmatrix} + \begin{bmatrix} \sqrt{\frac{2}{T_{T_{bgyro}}}\sigma^2_{b_{gyro}}} \\ \sqrt{\frac{2}{T_{bgyro}}\sigma^2_{b_{gyro}}} \\ \sqrt{\frac{2}{T_{bgyro}}\sigma^2_{b_{gyro}}} \end{bmatrix} \underline{w}_{gyro} \qquad (14.6)$$

Where $\underline{w}_{acc}$ and $\underline{w}_{gyro}$ are the accelerometer and gyro wide band noise terms.

In order to select the correct sensor error model, it is recommended that the behaviour of the errors under the operational scenario of the given application need to be investigated beforehand, as it has been done and presented at Part III. In fact, the choice of what model to use is dependent on the operation time, sensor performance, and working environment. If the operation time is very short, then the errors can be treated practically as constants. If the IMU is to run for a very long time, then the behaviour of the sensor errors must be carefully investigated.

Having characterized the inertial sensors bias, we are now able to present the augmented process model for the EKF by adding Equation(14.3) to Equation (14.1). This Augmented SIMU (ASIMU) model presents a state vector that comprises fifteen states which are presented at Table 14.1.

$$\delta\dot{\underline{x}}^{N} = \begin{bmatrix} \dot{\underline{\psi}}^{N} \\ \delta\dot{\underline{v}}^{N} \\ \delta\dot{\underline{R}}^{N} \\ \delta\dot{b}_{drift_{gyro}} \\ \delta\dot{b}_{drift_{acc}} \end{bmatrix} = \begin{bmatrix} F_{\psi\psi} & F_{\psi v} & F_{\psi r} & 0_{3\times3} & 0_{3\times3} \\ F_{v\psi} & F_{vv} & F_{vr} & C_{B}^{N} & 0_{3\times3} \\ F_{r\psi} & F_{rv} & F_{rr} & 0_{3\times3} & C_{B}^{N} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & diag(\beta_{bg}) & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & diag(\beta_{ba}) \end{bmatrix} . \begin{bmatrix} \underline{\psi}^{N} \\ \delta\underline{v}^{N} \\ \delta\underline{R}^{N} \\ \delta b_{drift_{gyro}} \\ \delta b_{drift_{acc}} \end{bmatrix} + \begin{bmatrix} C_{B}^{N} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & C_{B}^{N} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & I_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & I_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \end{bmatrix} . \begin{bmatrix} \underline{\tilde{v}}_{gyro} \\ \underline{\tilde{v}}_{acc} \\ \underline{w}_{bgyro} \\ \underline{w}_{bacc} \\ 0 \end{bmatrix} \quad (14.7)$$

Where:

$$\beta_{ba} = \begin{bmatrix} -\frac{1}{T_{bacc}} & 0 & 0 \\ 0 & -\frac{1}{T_{bacc}} & 0 \\ 0 & 0 & -\frac{1}{T_{bacc}} \end{bmatrix} \qquad \beta_{bg} = \begin{bmatrix} \frac{-1}{T_{bgyro}} & 0 & 0 \\ 0 & \frac{-1}{T_{bgyro}} & 0 \\ 0 & 0 & \frac{-1}{T_{bgyro}} \end{bmatrix}$$

$\tilde{v}_{acc}, \tilde{v}_{gyro}$	Corresponds to the accelerometer and gyro wide-band noise with variances equal to $\sigma_{acc}^2$and σ_{gyro}^2, respectively. See Chapter 9 for more details.
$\underline{w}_{bgyro}, \underline{w}_{bacc}$	Gauss-Markov process driving noise for the gyros and accelerometers. The corresponding variances are given by $\sigma_{b_{gyro}}^2$and $\sigma_{b_{acc}}^2$, respectively.

The process noise vector is given by:

$$w = \begin{bmatrix} \underline{\tilde{v}}_{\text{gyro}} & \underline{\tilde{v}}_{\text{acc}} & \underline{w}_{\text{bgyro}} & \underline{w}_{\text{bacc}} & 0 \end{bmatrix} \tag{14.8}$$

Then the associated noise covariance matrix Q_{ASIMU} is given by:

$$Q_{ASIMU} = \begin{bmatrix} \text{diag}(\sigma_{gyro}^2) & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & \text{diag}(\sigma_{acc}^2) & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & \text{diag}(\sigma_{b_{gyro}}^2) & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & \text{diag}(\sigma_{b_{acc}}^2) & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \end{bmatrix} \tag{14.9}$$

In the EKF, the Q matrix reflects how the predictions can be trusted, being critical for achieving a practical result. And so, working with low performance data, it is expected that Q must be selected in a pessimist form, i.e. a norm larger than the real one, in order that the filter can converge and exhibit no time lag (Godha, 2006).

It should be noted that the 3×3 diagonal matrices for: $\sigma^2_{gyro}, and\ \sigma^2_{b_{gyro}}$ contain the variances associated with the gyros measurements errors ($\delta\underline{\omega}^{B}_{IB}$) and gyros bias drift errors ($\delta b_{drift_{gyro}}$), respectively. The 3×3 diagonal matrices for: $\sigma^2_{acc}, and\ \sigma^2_{b_{acc}}$ contain the variances associated with the gyros measurements errors ($\delta\underline{aSF}^{B}_{IB}$) and gyros bias drift errors ($\delta b_{drift_{acc}}$). According to Table 9.2 or 9.3 of Chapter 9 related to the Gauss Markov process or taking into account Equation (14.4), the respective spectral desnisities are given by:

$$q_{b_{gyro}} = 2\beta_{bg}\sigma^2_{gyro}; q_{b_{acc}} = 2\beta_{ba}\sigma^2_{acc} \qquad (14.10)$$

Table 14.1 EKF ASIMU Error State Vector.

Type of Error	Description	Symbol	Units
Attitude Errors $\underline{\boldsymbol{\psi}}^{\mathbf{N}}$	East Misalignment	ψ^E	Radians
	North Misalignment	ψ^N	
	Up Misalignment	ψ^U	
Velocity Errors $\delta\underline{\mathbf{v}}^{\mathbf{N}}$	East Velocity Error	$\delta \mathrm{v}^E$	m/s
	North Velocity Error	$\delta \mathrm{v}^N$	
	Up Velocity Error	$\delta \mathrm{v}^U$	
Position Errors $\delta\underline{\mathbf{r}}^{\mathbf{N}}$	Latitude Error	δl	Radians
	Longitude Error	$\delta \mathrm{L}$	
	Altitude Error	δh	m
Accelerometer Errors $\delta\mathbf{b}_{\mathbf{drift_{acc}}}$	X-Accelerometer Error	$\delta b_{\mathrm{drift}_{ax}}$	mg
	Y-Accelerometer Error	$\delta b_{\mathrm{drift}_{ay}}$	
	Z-Accelerometer Error	$\delta b_{\mathrm{drift}_{az}}$	
Gyro Errors $\delta\mathbf{b}_{\mathbf{drift_{gyro}}}$	X-Gyro Error	$\delta b_{\mathrm{drift}_{gx}}$	deg/h
	Y-Accelerometer Error	$\delta b_{\mathrm{drift}_{gy}}$	
	Z-Accelerometer Error	$\delta b_{\mathrm{drift}_{gz}}$	

14.3 Measurement Model

According to our configuration of the SIMU/GNSS integration, the differences between the measurement output by the $GNSS_{SR}$ and the predictions of those measurements from the inertial navigator are used to update the state vector. Given the fact that a loosely-coupled configuration is implemented, the observations to be used will be the position and velocity.

According to Figure (14.2), the GNSS measurements (GPS or Galileo measurements) are subtracted from the SIMU measurements in order to form the observation set given by Equation (14.11).

Thus, the observation equation is:

Observation Equation:

$$Z_{obs} = \begin{bmatrix} v^N_{SIMU} - v^N_{GNSS} \\ R^N_{SIMU} - R^N_{GNSS} \end{bmatrix} \tag{14.11}$$

Measurement Equation:

As a consequence, the measurement equation is:

$$Z = \hat{Z}^N_{obs} - \hat{Z}^N_{obs} = \begin{bmatrix} v^N_{true} + \delta v^N_{SIMU} - v^N_{true} - \delta v^N_{GNSS} \\ R^N_{true} + \delta R^N_{SIMU} - R^N_{true} - \delta R^N_{GNSS} \end{bmatrix} \tag{14.12}$$

$$= \begin{bmatrix} \delta v^N_{SIMU} - \delta v^N_{GNSS} \\ \delta R^N_{SIMU} - \delta R^N_{GNSS} \end{bmatrix}$$

Where:

$\delta R^N_{SIMU}, \delta v^N_{SIMU}$	Error in the SIMU position and velocity solution estimation.
$\delta R^N_{GNSS}, \delta v^N_{GNSS}$	Error in the GNSS (Galileo or GPS) position and velocity solution estimation.
R^N_{true}, v^N_{true}	True position and velocity.

Comparing Equation (14.12) with the one given by Equation (12.6) the corresponding matrices can be achieved as follow:

$$\begin{aligned} x_k &= [\,\psi^N\ \delta V^N\ \delta R^N]|_{SIMU} \\ H_k &= \begin{bmatrix} 0_{3\times3} & I_3 & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & I_3 \end{bmatrix} \\ \underline{v}_{m_n} &= \left[\delta v^N_{GNSS}\ \delta R^N_{GNSS}\right] \\ G_{m_n} &= \begin{bmatrix} I_3 & 0_{3\times3} \\ 0_{3\times3} & I_3 \end{bmatrix} \end{aligned} \tag{14.13}$$

According to Chapter 12 $\underline{v}_k$, it is assumed to have equal and uncorrelated variance, which leads to the measurement covariance matrix given by:

$$R_n = E\left[\underline{v}_{m_n}\underline{v}^T_{m_n}\right] = \begin{bmatrix} I_3\delta GNSS_{velocity} & 0_{3\times3} \\ 0_{3\times3} & I_3\delta GNSS_{position} \end{bmatrix} \tag{14.14}$$

Where $\delta GNSS_{velocity}$ and $\delta GNSS_{position}$ are the $GNSS_{SR}$ velocity and position variance.

Having presented the augmented system model and the measurement model, we can now apply the EKF updates (Chapter 12): measurement update and time update.

Measurement Update

Kalman Gain:

$$K_k = P_k(-)H_k^T\left[H_kP_k(-)H_k^T + R_k\right]^{-1} \tag{14.15}$$

Measurement Update:

$$\hat{x}_k(+) = K_kz_k \tag{14.16}$$

It should be noted that we are using the EKF, where the nominal trajectory is considered to be the previous estimate and with a feedback configuration. As a result $\hat{x}_k(-) = 0$. The errors are feedback into the position, velocity and attitude and so the estimated error state vector is reset.

Error Covariance Update:

$$P_k(+) = [I - K_k H_k]P_k(-)[I - K_k H_k]^T + K_k R_k K_k^T \quad (14.17)$$

Time Update

State Extrapolation:

$$\hat{x}_{k+1}(-) = 0 \quad (14.18)$$

Error Covariance Extrapolation:

$$P_{k+1}(-) = \Phi_k P_k \Phi_k^T + Q_k \quad (14.19)$$

At this point, we have concluded the EKF algorithm which is able to be implemented and certificated in the Matlab environment.

14.4 Error Feedback

Finally, having achieved the error state vector, we have all the needed parameters to correct the SIMU solution, as well as, the inertial sensors bias drift. The feedback of the errors, in order to correct the estimates is divided into two parts: Sensor errors correction and SIMU solution correction.

Sensor errors correction: $\delta b_{drift_{gyro}}$, $\delta b_{drift_{acc}}$

The total accelerometer and gyro error are given by:

$$\begin{aligned} \delta \underline{f}_{IB}^{B} &= \tilde{f}^{B} - f^{B} \\ \delta \underline{\omega}_{IB}^{B} &= \tilde{\omega}_{IB}^{B} - \omega_{IB}^{B} \end{aligned} \quad (14.20)$$

Where $\tilde{f}^{B}, \tilde{\omega}_{IB}^{B}$ are the measured accelerometer and gyro outputs. As a result, the compensation form will be:

$$\begin{aligned} \mathrm{f}^{\mathrm{B}} &= \tilde{\mathrm{f}}^{B} + \delta \underline{\mathrm{f}}_{IB}^{B} \\ \omega_{\mathrm{IB}}^{\mathrm{B}} &= \tilde{\omega}_{\mathrm{IB}}^{\mathrm{B}} + \delta \underline{\omega}_{\mathrm{IB}}^{\mathrm{B}} \end{aligned} \tag{14.21}$$

The measurements of Equation (14.21) will be corrected before the strapdown mechanization, see Chapter 5 for more details, using the error estimates by the EKF, given by:

$$\begin{aligned} \delta \underline{\mathrm{f}}_{IB}^{B} &= \delta \mathrm{b}_{\mathrm{turn-on}_{\mathrm{acc}}} + \delta \mathrm{b}_{\mathrm{drift}_{\mathrm{acc}}} + \tilde{v}_{\mathrm{acc}} \\ \delta \underline{\omega}_{\mathrm{IB}}^{\mathrm{B}} &= \delta \mathrm{b}_{\mathrm{turn-on}_{\mathrm{gyro}}} + \delta \mathrm{b}_{\mathrm{drift}_{\mathrm{gyro}}} + \tilde{v}_{\mathrm{gyro}} \end{aligned} \tag{14.22}$$

SIMU solution errors correction: $\underline{\psi}^{\mathrm{N}}$, $\delta \underline{\mathrm{v}}^{\mathrm{N}}$, $\delta \underline{\mathrm{R}}^{\mathrm{N}}$

According to the error state vector, the only state that can be directly corrected is velocity. This correction will be done according to Equation (14.23).

$$\underline{\mathrm{v}}^{\mathrm{N}} \equiv \hat{\underline{\mathrm{v}}}^{\mathrm{N}} - \delta \underline{\mathrm{v}}^{\mathrm{N}} \tag{14.23}$$

Regarding the attitude correction, we know, from Chapter 6, that the three angle error vectors in the N-Frame are related as:

$$\underline{\psi}^{\mathrm{N}} = \underline{\gamma}^{\mathrm{N}} - \underline{\varepsilon}^{\mathrm{N}} \tag{14.24}$$

In fact, $\underline{\psi}^{\mathrm{N}}$ contains both the error in $\mathrm{C}_{\mathrm{B}}^{\mathrm{N}}$ and $\mathrm{C}_{\mathrm{N}}^{\mathrm{E}}$. In order to obtain $\underline{\varepsilon}^{\mathrm{N}}$,we also need $\delta \underline{\mathrm{R}}^{\mathrm{N}}$ because it describes the errors in $\mathrm{C}_{\mathrm{N}}^{\mathrm{E}}$ as follows:

$$\begin{aligned} \delta \underline{\mathrm{R}}^{\mathrm{N}} &= \mathrm{R}\left(\underline{\varepsilon}^{\mathrm{N}} \times \underline{\mathrm{u}}_{\mathrm{ZN}}^{\mathrm{N}}\right) + \delta \mathrm{R}\, \underline{\mathrm{u}}_{\mathrm{ZN}}^{\mathrm{N}} \\ &\Updownarrow \end{aligned} \tag{14.25}$$

$$\begin{bmatrix} \delta R_x/R \\ \delta R_y/R \\ \delta R_z/R \end{bmatrix} = \begin{bmatrix} \varepsilon_y \\ -\varepsilon_x \\ 0 \end{bmatrix} + \delta R \, \underline{u}_{ZN}^{N}$$

From Equation (14.25), it is possible to extract $\underline{\varepsilon}^{N}$ and, as a consequence, the value of $\underline{\gamma}^{N}$can be achieved using Equation (14.24).

Once we have the all necessary parameters, the SIMU states are corrected using:

$$\begin{aligned} C_B^N &\equiv \left[I + \left(\underline{\gamma}^N \times\right)\right] \hat{C}_B^N \\ C_E^N &\equiv \left[I + \left(\underline{\varepsilon}^N \times\right)\right] \hat{C}_E^N \end{aligned} \tag{14.26}$$

14.5 Practical considerations

14.5.1 Level Arm

The level arm corresponds to the offset existing between the position of the IMU and the $GNSS_{SR}$ antenna, due to the fact that the sensors were not installed in, exactly, the same position. Therefore, if we consider that l^B is the correspondent offset vector between the $GNSS_{SR}$ and the MMQ50 IMU – important to note that the GNSS antenna for the Piccolo was the same as the one used for the $GNSS_{SR}$. The autopilot unit was connected to the Roke Manor triple-GNSS antenna via quad signal splitter.

Considering that that $l^m$is the level arm vector in the mapping frame, then, we have:

$$l^m = \begin{bmatrix} x_{IMU}^m - x_{GNSS}^m \\ y_{IMU}^m - y_{GNSS}^m \\ z_{IMU}^m - z_{GNSS}^m \end{bmatrix} \tag{14.27}$$

In order to correct this value, the IFEN provided the vector from the centre of the IMU to the GPS antenna. The m corresponds to the vehicle frame. The vector is given by:

$$l^{m} = \begin{bmatrix} 1.1561469 \\ 1.4439087 \\ -0.2983429 \end{bmatrix}^{T} \quad (14.28)$$

In order to get the level arm vector in the B-Frame, l^{B}, a transformation between the two frames must be performed beforehand. The IFEN also provided the vector correspondent to the vehicle to body rotation, $v^{B}_{m\,=\,vehicle}$. The vector is given in degrees by:

$$v^{B}_{m} = \begin{bmatrix} 0.49236 \\ -1.26309 \\ -152.123 \end{bmatrix}^{T} \quad (14.29)$$

Finally in order to get the level arm vector in the B-Frame, we have to perform the C^{B}_{m} using the transformation matrix given by Table 3.1 and then we get:

$$l^{B} = C^{B}_{m} l^{m} \quad (14.30)$$

As a consequence, the lever arm correction for the measurement model is given by:

$$Z_{obs} = \begin{bmatrix} v^{N}_{SIMU} - v^{N}_{GNSS} - C^{N}_{B}\omega^{B}_{IB}C^{B}_{m}l^{m} + \omega^{N}_{IN}C^{N}_{B}C^{B}_{m}l^{m} \\ R^{N}_{SIMU} - R^{N}_{GNSS} - R^{-1}C^{N}_{B}C^{B}_{m}l^{m} \end{bmatrix} \quad (14.31)$$

14.5.2 ASIMU EKF Parameters

Table 14.2 EKF model practical considerations.

State number	Description	Parameters

10-12	Gyro Bias drift $\delta\dot{b}_{drift_{gyro}}$	1st order Gauss Markov Model with: $\sigma^2_{b_{gyro}} = [0.046 \quad 0.046 \quad 0.046]$ and $T_{bgx} = 200s$
13-15	Accelerometer Bias drift $\delta\dot{b}_{drift_{acc}}$	1st order Gauss Markov Model with: $\sigma^2_{b_{acc}} = [0.02 \quad 0.02 \quad 0.02]$ and $T_{bax} = 200$ s

CHAPTER 15

PART IV CONCLUSION SUMMARY

15 Part IV Conclusion Summary

The **specific objective** associated to Part IV was: To develop and present the architecture strategy to be used to fuse SIMU/$GNSS_{SR}$ data.

The sixth secondary research question associated to these specific objectives is the subject of study of Chapters 12, 13 and 14.

Chapter 12 starts with the general presentation of the equations related to the KF and EKF.

Chapter 13 does not go into details about the several GNSS but instead gives a general description of the use of a GNSS device as an aiding system. Different ways to implement an integration algorithm to fuse data provided by an SIMU and a GNSS system are described in this chapter as well.

Regarding the first part of **Q6** of this study, Chapter 12 and Chapter 13 presented the common estimation methods and architectures for integrating the INS/SIMU and GNSS data. In Chapter 13, it is concluded that the closed-loop integration, loosely coupled scheme, is considered as a more suitable approach for low-cost SIMU/$GNSS_{SR}$ integration in land vehicle applications, due to the large error associated to the sensors. It was also stated that, for the loosely coupled integration, there is no need for a GNSS error model due to the fact that we are directly using the position and velocity from the $GNSS_{SR}$ to form the observation set. Additionally, the precision of the $GNSS_{SR}$ provided by the data sheet will be used as measurement noise.

The particularity of the closed-loop configuration in relation to the open loop is related to the fact that the EKF estimated SIMU errors are send back to the mechanization equations in the SIMU block and to the IMU raw data before mechanization, see Figure 13.2.

Regarding the second part of **Q6**, "How does one optimally estimate the interest parameters from noise corrupted data? " Chapter 14 presented with detail the system (process) and measurements models, using the EKF developed at Chapter 12.

Equation (14.1) represents the basic system model, with a state vector of nine navigation parameter errors to be estimated. However, considering that we are working with corrupted data that according to Chapter 9 and 10 do not meet the KF assumptions (zero-mean white noise and Gaussian behaviour), an ASIMU is proposed to. The system models extra states, to account for the effects of the inertial sensor errors, to be added to the basic one. This new system model is composed of fifteen navigation parameter errors.

In order to answer to the second part of **Q6**, Equation (14.7) represents the system model for the ASIMU/$GNSS_{SR}$ algorithm while Equation (14.11) represents the measurement model using data provided by a $GNSS_{SR}$. In chapter 14, the numerical values to be used in the noise covariance matrix Q_{ASIMU} are presented.

In light of the above, we are able to confirm that the specific objective related to **Q6** has been achieved. Part IV finalizes with the presentation of the error model for the SIMU/ $GNSS_{SR}$ system ready to be filtered through its implementation in Matlab.

PART V

PLATFORM DESIGN

Exact science and its practical movements are no checks on the greatest poet, but always his encouragement and support ... The sailor and traveller, the anatomist, chemist, astronomer, geologist, phrenologist, spiritualist, mathematician, historian and lexicographer are not poets, but they are the lawgivers of poets and their construction underlies the structure of every perfect poem.

— Walt Whitman

CHAPTER 16

UAV HARDWARE IMPLEMENTATION

16 UAV Hardware Implementation

The ANTEX-M is an UAV collaboration project with some institutes[10] that aims to develop UAV as systems and technology demonstrators in areas concerning UAV development, defence, aeronautics and civil protection. One of the platforms developed for the project ANTEX-M is the ANTEX-M X02 (see Figure 16.2).

For the UAV to be fully operational and perform its mission, a good design and implementation of the hardware platform must be done. It should be stressed that even though it was not possible to perform the desired flight tests, the hardware and software complete development and implementation is presented for future works in this field.

Part V contains two chapters concerning the hardware and software implementation necessary to achieve the goals of this project. The first chapter related to the hardware, describes the aircraft, onboard avionics, as well as, all system architecture for this thesis. The second chapter describes all the software developed to perform the flight tests.

Chapter 16 begins with an overview of the UAV avionics, in order to give the reader an image of the characteristics and current state-of-the-art of the existing UAV avionics hardware. This section ends summarizing the existing payloads such as sensors, computers, power suppliers, specific payloads, and data links.

Afterwards, a general idea of this project's whole system, the airborne and the plus

[10] Portuguese Air Force Academy; the Institute of Geodesy and Navigation of University FAF Munich; University of California at Berkeley, Center for Collaborative Control of Unmanned Vehicles (C3UV); Swedish Defense Research Agency (FOI); Embraer, Empresas Brasileiras de Aeronáutica S.A (EMB), and Honeywell.

ground system is shown and described. It should be stressed that the airborne system is divided in two parts which are related to the hardware that has been developed to achieve this project's goal; and the rest of the avionics which consists of the hardware existent at the PAFA project. An overall description and detailed explanation of each part of the airborne and ground systems will be addressed. Last of all, the selected hardware, as well as, its description is presented.

16.1 UAV Avionics Payload Outline

An UAV payload capacity varies in weight from a few grams - Micro and Mini UAV - to over 1000 kg – High Altitude Long Endurance (HALE) UAV (Bento, 2008). Different UAV platforms have different mission applications and, therefore, different payload capacities are required. Although the UAV payload adds to the overall weight, the advancing technology and its miniaturization has led to smaller, less expensive and more functional payloads. Therefore, the capabilities of a wide range of UAV have been enhanced.

The UAV payload can be classified in three general types: avionics support (navigation and control support); sensing (information collection) and communications support (data links). So, the purpose of this section is to provide an overview of the characteristics and current state-of-the-art of the avionics support type. It should be stressed that almost all the information presented in this section has been compiled from the UAV roadmap released by the U.S. Department of Defence (DoD) presented in (US DoD, 2002). At the end of this section, a brief summary of the existing sensors, computers, and, power supply is presented.

Basically the avionics comprises the navigation and control support package, which consists of: adequate sensors, especially for navigation and control, flight computer, onboard processing and data storage, and power supply.

16.1.1 Navigation and Control Sensors

The flight control algorithm performance of an autonomous or semi-autonomous UAV depends mostly on the information provided about its state. This information can be obtained from several sensors, depending on both the vehicle platform and its mission application. For instance, a non-autonomous UAV does not need information about its attitude and therefore does not need to carry attitude sensors. In any case, the most important sensor types have been identified and divided into three general categories, which are related to the type of

measurements associated with the sensors. And so, there are sensors for navigation, sensors for flight control, and sensors for automatic take-off and landing measurements.

16.1.1.1 Navigation Measurements Sensors

To determine the vehicle state, some essential measurements must be considered. Particularly, the following measurements are needed: 3D position; ground velocity; accelerations; attitude; heading; angular rates and time. Such measurements can then be obtained from several navigational units such as the INS; Global Navigation Satellite System (GNSS), such as the Global Positioning System (GPS) or Differential GPS (DGPS).

16.1.1.2 Flight Control Measurements Sensors

For an autonomous or even semi-autonomous unmanned vehicle, attitude, linear accelerations and some air data must be provided to its flight control algorithm. Attitude information, pitch, roll and heading angles, can be obtained from several ways. Pitch and roll angles can be obtained from the IMU, from the digital measurement unit (DMU) or from the vertical gyro unit (VGU). Whereas, heading angle may be obtained from the IMU, magnetometer or compass. The angular rates, pitch, roll and yaw rate, can be obtained from the IMU, DMU, or the rate gyro unit (RGU). Also important are the linear accelerations (longitudinal, lateral and vertical), obtained from the IMU, DMU or accelerometers. Air data information like airspeed and barometric altitude, can be extracted from the air data unit (ADU), optical/laser sensors or Pitot tubes. An alternative to the individual gyros or IMU is a self-contained attitude heading and reference system (AHRS).

16.1.1.3 Automatic Take-off and Landing Measurements Sensors

Some specific operations, such as automatic take-off and landing usually require absolute altitude above ground level (AGL) measurements and/or a very accurate database. To obtain such information several sensors are available. The sensors widely used for automatic take-off and landing are the following ones: sonar (AGL); radar altimeter (AGL); Laser/Lidar (AGL); GPS (MSL), and barometric (MSL).

16.1.2 Onboard processing

One of the most significant and critical factor concerning, especially, the autonomous UAV development is the increased computational processing required to support all the flight operations. The most common model used in UAV systems is to relay most of the acquired flight data back to the ground control station (GCS) for post-processing. Nevertheless, this model is not ideal for real time missions or even when the communication bandwidth is limited. So, depending on the vehicle as well as on its mission, two different types of processing must considered: onboard processing and on-the-ground processing. Some research groups have opted for doing ground processing instead of onboard processing, but usually onboard real time embedded data processing is preferred. Thus, before choosing which platform to work with, it is important to define which type of processing is going to be used. Should all the computing power be located on-board or located on-the-ground? In fact, onboard processing means more UAV payload and consequently a greater size, weight and power (SWAP) but it also leads to more autonomy. On-the-ground processing always relies on working data links and can possibly result in significant weight savings. On the other hand, having onboard computers to process the algorithms makes the system less susceptible to signal latency problems and the algorithms can, also, be processed at higher rates. So, opting for the onboard processing concept implies flight computers and/or processors onboard the UAV to process all the algorithms necessary to control it.

Several computers, adequate for the different types of UAV platforms and mission applications, have been successfully developed in the last few years. Thus, alternatives can be found in the market. So far, for onboard processing, the following typical processors can be considered: Programmable Logic Controller (PLC); Micro-Processors (μP); Embedded Computer; Digital Signal Processors (DSP); Application Specific Integrated Circuit (ASIC) and Field Programmable Gate Array (FPGA).

A PLC is no more than a microprocessor specially used in the automation of industrial processes. Unlike general purpose computers, the PLC is designed for extended temperature ranges, dirty or dusty environments, immunity to electrical noise, and resistance to vibration and impact. It consists of a central processing unit (CPU), memory areas and appropriate circuits to receive input/output data. The PLC works by looking at the inputs and depending upon their state, turning on/off the outputs.

A μP is a programmable digital electronic component, incorporating the functions of a CPU on a single integrated circuit (IC), with as much as possible integrated on a chip, the microprocessor derived from the reduction of the word size of the CPU from 32 bits to 4 bits. One or more microprocessors typically serve as the CPU in a computer system, embedded system or handled device. Given the low cost and high flexibility of μPs they are widely used in consumer products (particularly mobile phones) where low level programming is typical.

An embedded computer is a special-purpose system in which the computer is completely encapsulated by the device it controls. Unlike a general-purpose computer, such as a personal computer, an embedded system performs one or a few pre-defined tasks, usually with very specific requirements. Since the system is dedicated to specific tasks, design engineers can optimize it, reducing the size and cost of the product. Embedded computers are the majority of the systems used in UAV applications. For instance, the PC104 is the most used, mainly due to being small and usually having low power requirements, which make them great for applications where available space is a constraint as is the case of Mini UAV. PC104 systems are widely used in factories, laboratories, vehicles, and almost anywhere where devices need to be controlled by a programmable computer. Almost anything is possible for a PC104 module. It has many peripheral boards such as: power supply; serial I/O ports; video controllers; control relays; GPS receivers; wireless communications and DSP slave. Thus a stack of PC104 modules can be attached as components of a larger circuit board. The main advantage of this PC is the fact that most of the programming development tools used for PCs can be used for a PC104 system. This reduces both the cost of purchasing new tools and also the learning curve of the programmers and hardware designers.

The DSP is a specialized μP designed specifically for digital signal processing, generally in real time. They are specially used for processing speech, image and video data. Usually a DSP is used when large repetitive real-time computations must be performed. DSP functionality can also be achieved using FPGA chips.

For example, obstacle and avoidance algorithms for UAV applications usually require systems that use specific sensors to detect and interpret the environment around the vehicle. These sensors, normally, need real time signal processing capabilities which are generally available on the DSP. Thus, while the DSP provides processing capabilities for navigation, control and sensor processing algorithms, the FPGA provides, for instance, custom interfaces for

aircraft hardware.

An ASIC is an IC customized for a particular use, rather than for general-purpose use. For instance, a chip designed solely to run a cell phone is an ASIC. It often combines the functionality of a number of components in a single package. ASIC applications can be found in the areas of entertainment, multimedia and avionics.

Small UAV guidance, navigation and control systems have limited processing power mainly due to stringent SWAP constraints. The current state of the art in Mini UAV is to give them the capacity of seeing and avoiding obstacles, tracking moving targets, cooperating with other manned and unmanned aircraft or tolerating unpredicted flight conditions. To go further, more powerful processing technologies are needed. Consequently, not only the SWAP constraints but also the necessity of intelligent UAV have led to alternative processing technologies such as FPGA, in order to meet the requirements of high performance embedded systems operating in real time. A FPGA is a semiconductor device containing programmable logic components and programmable interconnections. The programmable logic components can be programmed to duplicate the functionality of basic logic gates or more complex combinational functions such as decoders or simple math functions.

However, ASIC solutions were used extensively in many high performance embedded computing systems defence applications in the past. To maintain and upgrade ASIC based systems is currently very expensive. Thus, the preference of vendors such as Texas Instruments, IBM, Motorola and Analogue devices was for DSP processors. Nevertheless, the DSP do not always comply with the performance requirements of high performance embedded computing applications. Therefore, all the attention has turned towards the development of new types of processing architectures such as the FPGA. In fact, the price and performance of FPGA have improved enough over the past several years, making them very attractive when compared with using only DSP technology. In many cases, a single, high-performance FPGA can function as a coprocessor, performing tasks that would otherwise require the use of multiple DSP. It is true that most implementations for on-board processing rely on embedded computing systems or DSP. However, a considerable effort has been made in order to demonstrate the performance advantages of FPGA technology especially for video applications and image processing. It is not expected that the FPGA will replace the DSP and general-purpose processors in all signal processing applications. For instance, FPGA are usually considered to have high power

consumption. Many embedded processors require less power than a FPGA, but low power chips rarely offer comparable performance. Thus, in the choice of the computing system, typical design criteria including performance, power consumption, cost, memory usage, and programming requirements must be considered. So, depending on the application and platform required, alternative approaches have to be considered.

16.1.3 Data Storage

If, on one hand, sophisticated airborne sensors are getting versatile and generate more and more data, on the other hand, data links between the air vehicle and the GCS have limited bandwidth. Obviously, a data loss during a mission can be very costly. Given that, nowadays, data storage on board is a very important issue which must be taken into account. Fundamentally, the data storage technologies more used on board air vehicles are: the magnetic tape recorders; Flight data record (FDR); direct access storage devices (DASD); solid state flash disk and the solid state drive (SSD, also called solid state disk). The magnetic tape records are widely used in airborne platforms for analogue or digital data storage but their volume and weight are not always compatible with small payloads.

In Mini UAV, where the SWAP is an important factor, the choice of the data storage device will be a constraint. A disk or direct access storage device is, then, too big and too heavy for being used in such air vehicles. Thus, normally SSD and flash disks are used as storage devices. Nevertheless, the SSD have some limitations associated with its capacity. But anyway, the SSD are a good choice because they have, in some cases, access times twice or faster than those of the fastest hard drives. They have lower power consumption and make no noise but they use costly technology. The flash disk is also a good choice for using in Mini UAV. They have very low power consumption and failure rate and are typically used as replacements for hard drives, especially in installations exposed to extreme conditions.

16.1.4 Power Supply

It is well-known that power has to be supplied to the avionics. Batteries and/or generators are the power source more commonly used for such operation. The battery choice plays an important role on the hardware selection and integration, not only due to its weight but also because it will greatly affect the UAV endurance. Rechargeable batteries are preferred for this

type of application because they are more cost effective and do not need to be removed from the vehicle. Most of the remote controlled (R/C) aerial vehicles use rechargeable nickel cadmium (Ni-CAD), nickel zinc (Ni-Zn), or nickel metal hydride (Ni-MH) batteries mainly due to their high current draw, long life and very small recharge memory effect. However, as this type of battery is designed for a long life, they are consequently very heavy. In Mini UAV, a battery which can provide the necessary amount of electric charge to enable the vehicle to fly around 20 minutes is considered to be enough for research applications. Many electronic devices such as CD and MP3 players, digital cameras and so on, use lithium batteries, which are sold in most electronics stores in many sizes, shapes and voltages. In particular, the Lithium-Polymer (LiPo) batteries provide an enormous amount of electric current enabling the vehicle to fly for around 20 minutes.

16.2 Sensing

UAV payloads mainly depend on the type of mission that they have been designed for. They often include one or more of the following payloads (sensors): imaging payload; electronic warfare (EW) payload and nuclear, biological and chemical (NBC) payload. Therefore, brief overviews of the existing and emerging sensors are presented.

16.2.1 Imaging Payload

Gathering information to support intelligence, surveillance and reconnaissance (ISR) missions depends on both the UAV platform and sensors. Currently, the technologies available to support the ISR missions are: Video; electro-optic/infra-red/laser designator (EO//IR/LD) serving as a multi-mode payload for UAV and Synthetic Aperture Radar (SAR). The ability of the EO sensors to provide imaging with a good resolution depends mostly on the light conditions of some sources such as stars or solar light. Thus, traditional EO sensors are unable to penetrate clouds and its performance in rain is very poor, providing also a low resolution for long ranges. IR technology depends on thermal energy. Thus, thermal imaging uses tiny differences in temperature within a scene to build a detailed picture. However, if in a scene no thermal contrast characteristics are exhibited, the IR sensor will not be so effective. Nevertheless, recent advances in EO and IR systems have overcome and minimized the effects of many of these limitations, using for instance, a combination of the EO, IR and SAR sensors. SAR is an active system which

operates in the microwave band. Distinct from other systems such as infrared, purely optical and spectral systems, the longer wavelength microwave energy of SAR is able to penetrate smoke, rain, cloud, fog, mist, providing, consequently, all weather, day and night imaging with the capability of spatial resolution up to one foot. However, the minimum weight of common SAR systems is at least 30Kg, making it unacceptable for installation in Micro, Mini and even some tactical UAV. Up to now, this kind of platforms have not benefited from this precise real time imagery. Nevertheless, technological developments and the miniaturization of some components have led to miniature SARs.

The interferometry and moving target indicator (MTI) are two technologies which have recently been used in order to enhance the utility of SAR. Thus with a combined payload of EO/IR and SAR sensors, it is possible to achieve an all-weather system that can detect and focus objects over a very wide area with a high resolution. It is believed that this type of combination will become dominant in most UAV and, consequently, this will displace earlier technologies such as infrared line scanners. However, this combination does not solve the problem of foliage penetration and camouflage. Therefore, a new technology is emerging, the multispectral/hyperspectral imaging (MSI/HI) payload for UAV.

For Micro and Mini UAV, a big challenge is the development of CMOS cameras. A small UAV with a CMOS digital video sensor can provide real time forward reconnaissance. This technology enhances functionality through low cost chip integration while minimizing imager size, power consumption and weight. CMOS cameras make lots of noise. However, their small size and low cost provide significant advantages for both civilian and military applications.

16.2.2 EW Payloads

Mini UAV are the leading platform being introduced in the EW environment. Signals intelligence (SIGINT) is one of the major branches of the EW, followed by jammer equipment. Traditionally, the EW equipment has been utilized just in manned aircraft, due to their high power consumption and large size but, currently and mainly due to the miniaturization of RF components, there is some EW equipment capable of being installed in UAV.

SIGINT is usually subdivided into: electronic intelligence (ELINT) and communications intelligence (COMINT). While ELINT basically deals with radar emissions, the COMINT deals with voice and data communication. The basic idea of SIGINT is to gather and process

intelligence by monitoring, during a conflict, the use of the electromagnetic spectrum by the opponent. In 2003, a Global Hawk flew from California to Germany with an EADS ELINT reconnaissance sensor, which allowed it to detect electromagnetic signals from several sources, including air defence radars. As the Global Hawk flew over the North Sea, the data acquired by the sensor was being transmitted to the GCS where the detected emitters were identified.

16.2.3 NBC Payloads

NBC are detectors, sniffers capable of picking up the tiniest traces of nuclear, biological and chemical weapons (UVS payloads). A good example is the Flight Inserted Detector Expendable for Reconnaissance (FINDER), a tiny UAV which is launched from under the wing of a Predator, in order to sniff out areas considered by the crew to be suspicious.

16.3 Communications Support

Essentially, the UAV data link is responsible for performing two different functions: an up-link function, which allows the GCS to control the UAV and its payload; a down-link function, which normally provides two channels for transmitting UAV status and sensor data to the GCS. For the UAV control, it is usual to use a radio frequency (RF) transmitter to control the aerial vehicle within visual range. The UAV platform must carry the entire payload that will enable it to transfer information from on-board to the ground and vice-versa. Data is generally transmitted from the UAV directly to the GCS through line of sigh (LOS) or through a relay which in some of the more sophisticated UAV systems (Medium Altitude Long Endurance (MALE) and HALE UAV), can be by SATCOM.

The UAV can also be used as relay station to extend communication links. This is currently a big challenge in multi-vehicle applications. Thus, the types of channels available for the UAV to operate comprise radio frequency (RF) or microwave frequencies and include LOS, airborne relay and SATCOM. Given the fact that wideband microwave SATCOM links require a large antenna installed on the vehicle, this type of data link is not practical for Micro and Mini UAV. Otherwise, wireless data links are widely used by Mini UAV to send commands and receive telemetry or payload data. Wireless data links can be divided into digital and analogue links. As a good example of an analogue link, we have a UHF video signal transmission whereas digital links are well used for communicating between the GCS and the vehicle on-board

computer. It is important that the LOS problem should be taken into account. Occasionally, when the data link is blocked, a remote pilot (radio controller R/C) data link is often used. Therefore, this should be considered in the project. Wireless LANs sometimes are not desirable, mainly due to the inherent technology limitations. For instance, the typical range of a common 802.11 network with standard equipment is around tens of meters, which is considerable insufficient for some UAV applications. If an additional range is needed, repeaters or additional access points must be purchased.

Other emerging wireless technologies, such as WiMAX, offer ranges up to a hundred or more kilometres. Despite the rapid progress that wireless networking has made in the recent years, certain limitations of this technology have been inescapable. WiMAX (802.16) technology will become a significant addition to the field, offering high-bandwidth, low-cost wireless connectivity to homes, businesses, and eventually mobile users at distances up to 100 km. WiMAX uses different types of wireless technologies which can be used in UAV. It should be stressed that each of these technologies has advantages, as well as, disadvantages. When opting for wireless technology, some factors must be taken into account such as the best performance/power consumption ratio and the capability to handle interference problems. Currently the most widespread wireless local area network (WLAN) available today is based on the IEEE 802.11 standard. However, due to characteristics like the design and complexity of the protocol stack of 802.11, it translates into relatively expensive, bulky and power hungry modules. Although these networks can be suitable to interconnect devices like computers; there is an enormous potential market to provide wireless communication capabilities to smaller and cheaper devices running on batteries without the need for frequent recharging. Such devices include computer peripherals, biomedical monitoring appliances, surveillance units and many other sensing and actuation devices, such as our platform. One of these network types is the new ZigBee standard. This standard objective is to provide an extreme low power and a low bit rate PAN within the main purpose of enabling sensor networks. Another low power and low cost wireless network technology is Bluetooth, which operates in the 2.4 GHz ISM band using frequency hopping spread spectrum (FSSS). Due to its frequency agility techniques, Bluetooth presents mechanisms that deal smoothly with noise and narrow band interference.

The choice of the communication technology for UAV strongly depends on the intended application. There are many applications with different requirements on data link performance

between the GCS and the UAV or among UAV themselves if we consider multi-vehicle applications. For instance in Mini UAV, and due to their SWAP limitations, the low cost and low-power wireless networks are preferable to the RF modems.

Table 16.1 presents all the different sensors, power supplies, computers/data loggers, specific payload, and data links discussed along these sections.

Table 16.1 Avionics Support (navigation and control systems).

Measurement	**Sensors**	**Power Supply**	**Computers/Data Logger**	**Specific Payload**	**Data Link**
Vehicle State (3D Position)	INS, GPS, GNSS, DGPS, Vision (cameras), INS/GPS, INS/GPS/Vision	Ni-CAD Ni-Zn Ni-MH LiPo	**Computers:** PLC Embedded Systems (PC104) ASIC DSP FPGA	**Imaging:** EO/IR/LD SAR MTI MSI/HIS CMOS cameras	**Up-Link:** SATCOM Wireless LAN (Wi-Fi) Wireless PAN (Bluetooth) Wireless MAN (Wimax) Radio modem link Cellular networks **Down-Link** SATCOM Wireless LAN Wireless LAN (Wi-Fi___33) Wireless PAN (Bluetooth) Wireless MAN (Wimax) Cellular networks
Linear accelerations	INS,DMU, Accelerometers				
Angular Rates (Pitch, Roll and Yaw rate)	INS, DMU, RGU				
Attitude (Pitch/Roll)	INS, DMU, VGU, GPS		**Data Logger:** Magnetic tape recorder FDR DASD SSD	**EW:** ELINT sensors COMINT sensors	
Azimuth/Heading	INS, Compass, FVU				
Air Data (airspeed and static pressure)	ADU, Pitot tubes				
Altitude ASL/AGL	INS,GPS,Radar altimeter, Sonar, Laser				

16.4 Project Airborne and Ground System Hardware Configuration

Figure 16.1 presents an overall description of the different project hardware parts and their interaction. The hardware setup involves the airborne system hardware and the ground system hardware. The hardware developed for this project is presented in the violet box in the airborne system. The rest of the hardware is related to the one that exists in the PAFA, the PITVAN ANTEX UAV project. The new hardware includes implementation of an on-board computer, adequate sensors for navigation and control, interface sensor hardware, data storage, and power supply. The hardware parts of the airborne system, as well as, the parts of the ground system are described in more detail next.

16.4.1 Airborne System

The airborne system is composed of the following components: the UAV platform (Figure 16.2); the autopilot Piccolo II avionics (Figure 16.3); UHF and GPS antenna, and communications hardware. Details of each component are presented next.

16.4.1.1 UAV platform

The UAV platform to be used in the flight tests is the Portuguese Air Force UAV Antex-X02, which has the following specifications:

- Maximum Take Of Weight (MTOW): 10 Kg
- Wing Span: 2.4 m
- Payload: 4 Kg
- Endurance/fuel capacity: 0.3 h/0.2 L

16.4.1.2 Piccolo II avionics

The UAV Antex X02 is equipped with the Piccolo II autopilot provided by Cloud Cap Technology (CCT). The Piccolo II is a complete integrated avionics system for flying, especially small UAV. The all system includes avionics hardware and software, ground-station hardware and software, and also a development and simulation

environment (Vaglienti, et al., 2008).The Piccolo Autopilot II physical characteristics are the following:

- Length: 10 cm
- Width: 4 cm
- Height: 1.5 cm
- Weight: 233 g
- More details about the autopilot are given in Section 16.5.

16.4.1.3 UHF and GPS antenna;

The Piccolo has two SMA coaxial connectors which are used for the UAV UHF and GPS antenna connections. The output voltage for active GPS antenna is 3V. Note that it is recommended that low loss RG174 or RG316 cable be used for all external cabling. Piccolo Plus unit can be alternately configured for 5V antenna power.

16.4.1.4 Communications hardware.

The communication between the autopilot and the ground station is provided by a 900 MHz/2.4 GHz radio modem. The operator interface allows control of the UAV to be switched to autonomous or manual mode by the flick of a switch located on the Futaba transmitter. As it can be seen in Figure 16.1, there is also a switch before the flight servos, in order to allow the remote operation mode when necessary. Thus, the remote pilot on the ground can send control signals to the actuators, via the Futaba transmitter, which will be received on board by the RC receiver. When the autonomous mode is activated, the onboard flight mode switch redirects the computed control outputs coming from the Piccolo autopilot to the servo board, to control the actuators. The pilot manual control is mainly used for take-off and landing, as well as in the event of an emergency. In fact, when manual control is activated, the UAV can be flown like a typical RC aircraft.

16.4.2 Ground System Hardware

The ground control station has the following components: The Piccolo Ground Control

Station (GCS) from Cloud Cap; operator interface computer with software; Futaba pilot console for manual pilot control and the RF radio modem.

16.4.2.1 Piccolo Ground Control Station

The main role of the piccolo ground control station unit is to manage the communications to the avionics, the operator interface and the pilot manual control. This unit has the responsibility of streaming telemetry command and control data to and from the operator interface computer.

The operator interface computer runs a Windows 2000 ® operating system and custom developed software, which permits the end user to configure and operate the Piccolo System. For instance, the telemetry page, Figure 16.4, of this software provides the end user with an interface to: perform a pre-flight checklist; configure the desired gains of the control system in the Autopilot; program a desired route for the UAV via waypoints; display the reading of all the sensors aboard the UAV, and display the current position of the UAV in a geographical chart.

Thus, the operator interface environment software allows for the total monitoring of the UAV health status, including attitudes, rates, accelerations, airspeed, ground speed, GPS location, etc. The pilot manual control is mainly used for take-off, landing and emergency situations.

16.4.2.2 Futaba Pilot Console

The pilot manual control, which is mainly used for take-off, landing and emergency situations, is a commercial Futaba radio remote control, as it has been discussed before.

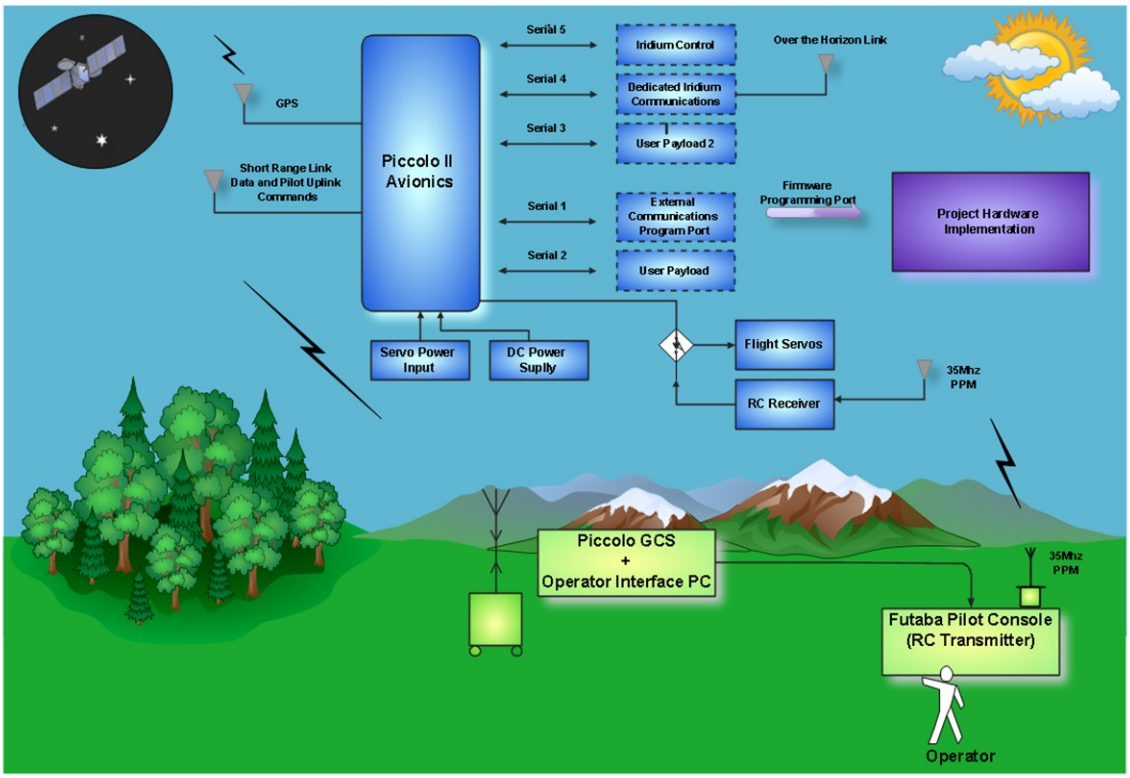

Figure 16.1 Project Hardware configuration.

Figure 16.2 Antex X02 platform.

Figure 16.3 Piccolo II autopilot.

16.5 Autopilot Details

16.5.1 Autopilot Overview

The Piccolo includes three Tokin CG-16D rate gyros and three ADXL202 accelerometers. It uses those three rate gyros and three accelerometers to determine the UAV's attitude (Vaglienti, et al., 2008). It includes a dual ported mpxv50045 4KPa dynamic pressure sensor, an absolute ported mpx4115a barometric pressure sensor, and an air temperature sensor (Vaglienti, et al., 2006). Together this data is used to calculate true airspeed (TAS), absolute altitude, and ambient air temperature. The piccolo uses its Motorola G12 GPS to provide the UAV's basic groundspeed and geodetic position. Piccolo includes a sophisticated data link. The communication between the autopilot and the ground station is provided by a 900 MHz/2.4 GHz radio modem. The operator interface allows for the control of the UAV to be switched to autonomous or manual mode (remote operation) by the flick of a switch located on the Futaba transmitter see Figure 16.1. Thus, and also important to realize, that when necessary, the remote pilot on the ground can send control signals to the actuators, via the Futaba transmitter, which will be received on board by the RC receiver. When the autonomous mode is activated than the onboard flight mode switch redirects the computed control outputs coming from the Piccolo autopilot to the servo board to control the actuators. The manual control is mainly used in the take-off and landing phases, as well as in the event of an emergency. In fact, when the manual control is activated, the UAV can be flown like a typical RC aircraft.

As an unmanned aircraft is only as useful as its payload, the Piccolo provides several

means of connecting to payloads, including CAN, asynchronous serial and discrete general purpose I/O. Data received from the payload (s) can be downlinked over the main data link and demuxed by the operator interface. For instance, in the laboratory, the Piccolo autopilot can be linked to a simulation computer via the CAN bus to enable hardware in-the-loop testing. In the Piccolo front panel (see Figure 16.3) it is possible to see the following external interface: UHF and GPS antenna; pitot and static pressure ports and, a 44-pin external interface connector. The main interface connector provides servo power control interface, payload and programming serial ports and general purpose I/O.

All versions of Piccolo support two two-wire RS232 universal asynchronous receiver transmitter (UART) serial ports, but, as it can be seen in Figure 16.3, the Piccolo II has an additional external interface connector, providing three two-wire RS232 interfaces. COM2 which can support baud rates up to 115200 is mainly used for external communications/firmware programming. The COM1 default function is for payload.

16.5.2 Piccolo Program Port

The Piccolo II has tree serial ports: two RS-232 serial ports at autopilot's 44-pin, and an extra payload serial port. One of the two serial ports is called the payload port while the other is the program port.

The serial port can be used to retrieve state information such as GPS latitude, longitude and altitude, as well as, IMU/Baro information. Additionally, waypoints can be sent through this port to the Piccolo to update, for instance, its flight plan. And so, the embedded onboard computer is linked to this serial port.

The program port is mainly used for external communications transmissions, and also for firmware programming.

In particular, all the needed information, provided by the autopilot, can be accessed through the serial port using for that, for instance, the Cloud Cap ComSDK library or developing a specific code based on the communication protocol, as described in Chapter 17. In fact, it is possible to develop a module capable of communicating directly with the UAV autopilot avionics using such libraries or otherwise developing and implementing the Cloud Cap Piccolo communication protocol based on (Vaglienti, 2007). This will be topic of discussion in the next chapter.

16.5.3 Piccolo Hardware in-the-Loop (HIL)

The Piccolo hardware in-the-loop simulator is a well-designed simulator that allows for the aircraft control laws and mission functionality to be tested without risking hardware in flight tests. The Piccolo developer's kit includes a HIL simulator that can be used to test the performance of a Piccolo implementation (Vaglienti, et al., 2008). The HIL simulator is based on the external CAN interface. So, when in simulation mode, the Piccolo sends control surface signals and receives sensor inputs from the CAN interface. In the laboratory, a PC with a CAN interface card running the HIL simulation software completes the necessary hardware to perform such a laboratory test.

16.5.4 Piccolo Software in-the-Loop (SIL) Simulation

The SIL simulation provides the same functionality as the HIL simulation, with the difference that the Piccolo and the ground station firmware run on a PC instead of the actual avionics and ground station hardware.

When compared to the HIL method, which requires an autopilot, ground station, power supplies, etc., the SIL method has the advantage of just needing a PC.

16.6 UAV Hardware Selection and Integration

In this section, the avionics support selected to achieve the project goals are presented.

Figure 16.4 illustrates the hardware selected, as well as the integration between systems, which are described next.

16.6.1 Sensors

The following sensors have been selected: two IMU[11]; Piccolo autopilot with GPS, speed, and air data information.

[11] Note that although two IMU are presented in Figure 16.4, one of the sensors was not used in the project. The Memsense nIMU included has not been used for achieve the project purposes. In fact, it will be used for future projects. Therefore, the characteristics of the nIMU will not be presented along the thesis.

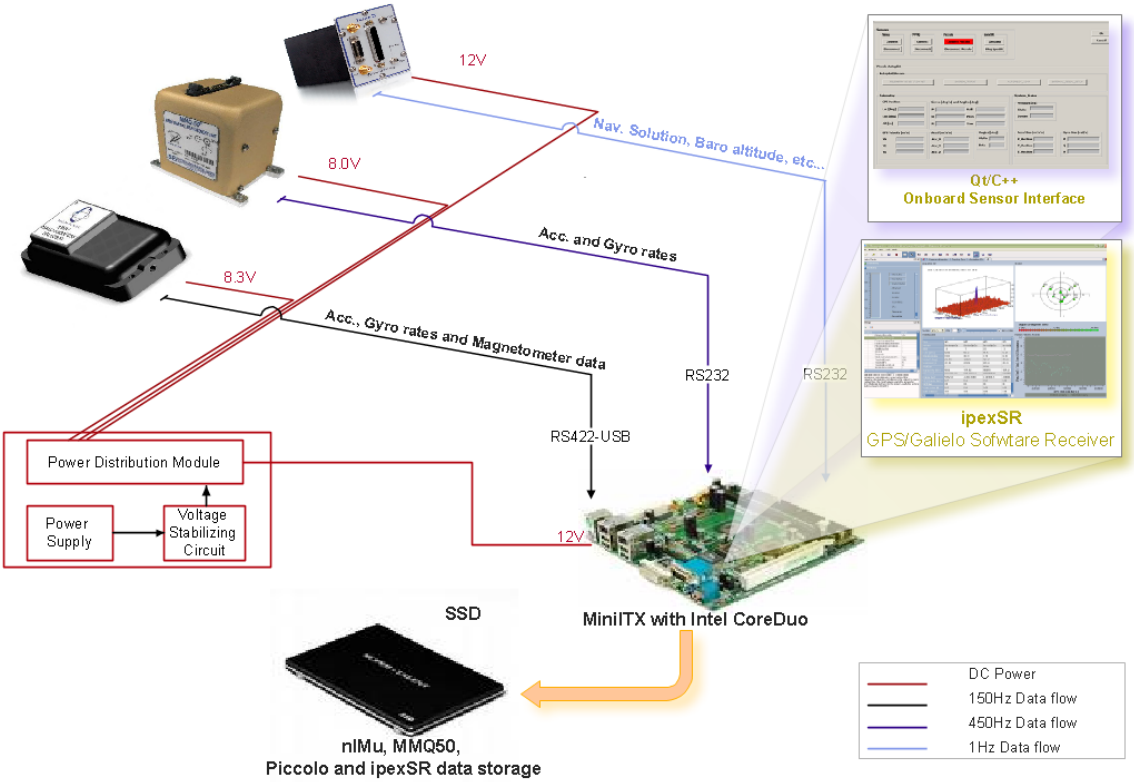

Figure 16.4 Project hardware selected.

16.6.1.1 Inertial Measurement Units

Although the Piccolo autopilot also provides accelerometer, and gyro measurements information, topic of discussion of subsection 16.3.1.2, at this point just the Systron Donner MMQ50 is presented.

Therefore, the MMQ50 provides 3D acceleration, and 3D rate of turn, with a sample frequency of 450 Hz.

Size: 48 mm x 48 mm x 65 mm

Weight: < 0.227 Kg

Power: 12 VDc at <5 Watts total

I/O: RS-232

The MMQ50 uses asynchronous serial communication across the RS-232 standard method. In particular, the miniature quartz MMQ50 is made by a compromise between two technologies: quartz and silicon. This sensor incorporates solid-state quartz micro machined inertial rate sensors and uses high performance, and, low cost silicon MEMS accelerometers. It is well known that silicon sensors have very good performance but, on the other hand, they have a very high temperature sensitivity. In spite of this, accelerometers and gyros should be temperature compensated; otherwise the drift in the measurements will grow along the time.

The MMQ50 IMU is an ideal sensor for embedded applications where extremely small size, low cost, and low power consumption are a requirement, as it is the case of this project.

Moreover, it offers substantial performance for a very attractive price. Technical data for this sensor are presented at Table 16.1.

Table 16.2MMQ50 Sensor performance. Source (MMQ50 Technical data)

Parameter		Gyros	Accelerometers
Bias Turn-on Stability	deg/h or mg	≤ 100 (1σ)	≤ 2.5 (1σ)
Bias In-run Stability	deg/h or mg	50 – 200 (1σ)	3 (1σ)
Scale Factor In-Run Stability	ppm	≤ 5000 (1σ)	≤ 5000 (1σ)
Bandwidth	Hz	50	50
Angle Random Walk/velocity random walk	$deg/\sqrt{h}$ or $mg/\sqrt{Hz}$	0.35	0.5

16.6.2 Onboard Computer

According on the application and platform, several alternative approaches have been under consideration. And so, an embedded computer is considered.

Taking into account the payload requirements and also the PC104 advantages presented before, the PC104 new model has been selected for the onboard processing tools.

CHAPTER 17

UAV SOFTWARE IMPLEMENTATION

17 UAV Software Implementation

It is well known that the flight control algorithm performance of an autonomous or semi-autonomous UAV depends mostly on the information provided about its state. Consequently, this information is provided by the multisensory data fusion algorithm, developed in the navigation module. Most important, it is the way how the multisensory data can be concomitantly obtained. Thus, the big challenge in this chapter is to provide an onboard sensor Graphical User Interface (GUI) module capable of receiving data from several sensors in a concurrent mode. The onboard sensor GUI developed, and shown in Figure 17.2, and described in this chapter, allows for a user to select which sensors to use during flight tests. An overview of the used and developed software is presented. The software environment that has been chosen for the GUI application is first presented. Following this, the software design architecture is depicted with more specific details related to the onboard sensor fusion graphical user interface (GUI) developed. Particular attention is also paid to the data flow of the software, in order to provide the reader with an idea of how the software interacts with the sensors and other systems. Afterwards, a description of the MEM IMU's integration in the software developed is presented, and lastly, a lay out of the developed C^{++}class is presented.

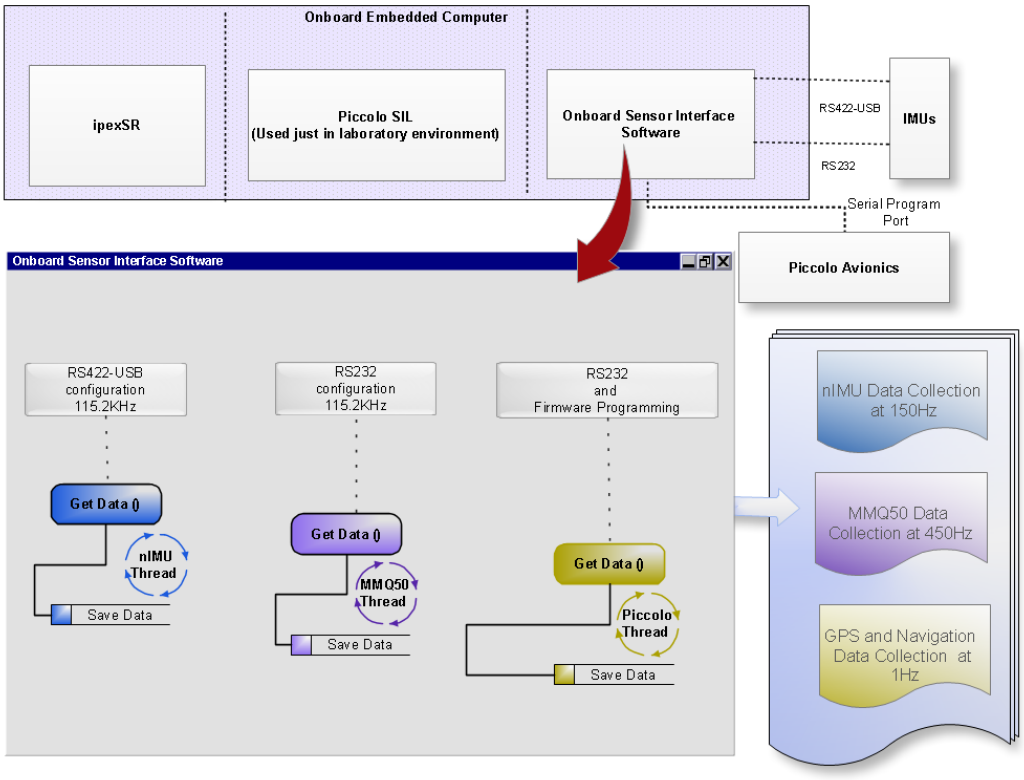

Figure 17.1 Software Architecture.

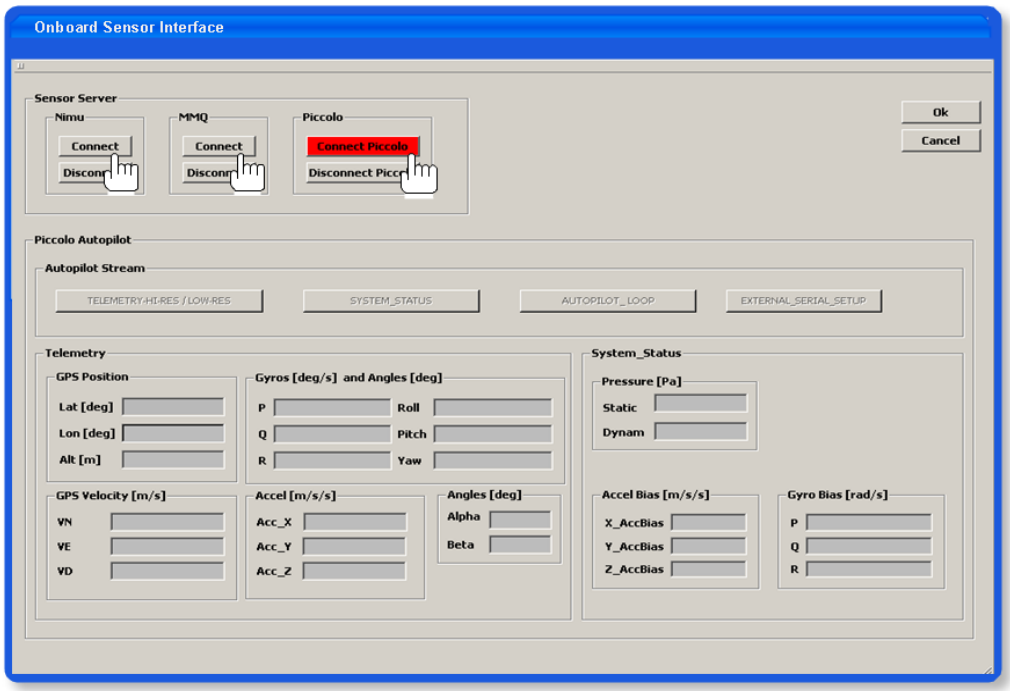

Figure 17.2UAV Onboard sensor interface GUI.

17.1 Software Development

The onboard sensor GUI is a multithread module that has been developed using Qt and Microsoft Visual 2005 with C^{++} and MFC. Qt is a cross-platform application development framework, widely used for the development graphical user interface programs. Therefore, in order to have a cross-platform as well as a multithread application, Qt integrated with standard C^{++} programming language has been chosen for this project. Moreover, to allow for communication with the Piccolo autopilot, a Windows binary release of Cloud Cap Technology's communications library has been used. Due to some requirement restrictions related to the use of the ipexSR, the platform for this project must be Windows XP.

17.2 Software Architecture

Qt integrated with C^{++} has been chosen as the program language for the GUI implementation. In this section, the software architecture of the GUI, systems, subsystems and related tasks will be described.

Figure 17.1 shows the UAV onboard software architecture necessary to enable the flight tests.

On the whole, the software can be divided in the following parts: ipexSR (Galileo Software Receiver), Piccolo Autopilot Software in the Loop (SIL), and, the onboard sensor GUI. Additionally, the GUI module is divided in two subsystems: the sensor server and the sensor data logger subsystems.

According to the figure, tree programs can run concurrently on the onboard embedded computer: the Galileo Software Receiver, ipexSR, developed by the Institute of Geodesy and Navigation, the onboard sensor GUI module specifically developed for this project and, the autopilot Piccolo.

17.2.1 Piccolo Autopilot Integration

In order to easily integrate the Piccolo autopilot in the onboard sensor module developed, it was vital to understand the Piccolo´s communication protocol. To better understand this

protocol, Cloud Cap technology provides a communications Standard Developer Kit (SDK) which has been used to communicate with the autopilot. This communication specifies the protocol between the Piccolo avionics and Piccolo ground station, which is the same that is used between the avionics and other applications connected via its program port. The Cloud Cap Communications library is a hybrid mix of both C and C^{++} modules, which provides a valuable tool for developing specific code to interact with the Piccolo avionics.

17.2.2 Piccolo SIL

The Piccolo SIL is used just for a laboratory environment. The SIL configuration provides the same functionality as a HIL setup, but without the avionics autopilot and GS hardware connected. Using the SIL simulator environment, PC applications take the place of the autopilot and GS. To setup this environment, it is only necessary to have a Windows-based PC. This simulation environment, as well as the HIL environment, allows the user to test the developed GUI application in the laboratory, without risking the UAV in a flight test. Although the SIL cannot replace flight testing, it is considered a good way to prevent some failures by detecting software bugs and other deficiencies preventing one from putting UAV and related hardware in risk.

17.2.3 Piccolo Autopilot Stream

The data sent to or from the avionics are made-up of multiple bi-directional streams which are multiplexed onto the wireless channel. Each stream represents an endpoint in the avionics being characterized by a physical port or a software endpoint. Table 1 of Communications for the Piccolo avionics (Vaglienti, 2007) defines eight numbers of stream types. The endpoint Autopilot is one of those eight types, supported by the Piccolo protocol with the identifier AUTOPILOT-STREAM whose knowledge was important for the development of the Piccolo integration module. This stream defines the primary command and control interface for the Piccolo avionics. Furthermore, sensor and autopilot state information are sent from this endpoint. So, using the Communications SDK, the autopilot stream will be used to carry autopilot state information from the avionics to the sensor interface and sensor telemetry information from avionics to the sensor interface.

The autopilot stream data travels between both the Piccolo avionics autopilot and the

operator interface, and the Piccolo avionics autopilot and the onboard embedded computer.

As depicted in Figure 17.1, the data will be received by the embedded computer, via the Piccolo avionics program serial port. Running both links simultaneously is referred to as multi-homed operation. Then, the data on the autopilot stream will be identical to that received by the Operator Interface on the ground. Every 10-20 ms (recommended by Cloud Cap) the Piccolo algorithm developed in C^{++} checks for data on the serial port.

17.2.4 Piccolo Communication Implementation

The SDK library provides a communication manager class that abstracts the low level of talking to one or more Piccolo system. It can work over serial lines or over an IP network, as either a client or a server. This class provides a number of functions for sending and receiving packets from avionics, serving as the basis to interact with the Piccolo avionics. To use the SDK library, it is first necessary to instantiate a CCommManager object, passing it the necessary constructor arguments to determine how the communications manager will work. For example: if SIL is used then the CCommManager (0,_localhost: 2001, 0) must be used; if real time application with the avionics connected to a serial port is used instead, then CComManager (COMi,__ , 2001) is used. Once instantiated and set up, the communications manager uses the RunNetwork () function, with a typical call period of 10ms, to service the communications endpoints.

Data coming from the avionics are contained in streams, which are contained in queues. To get such data, it is first necessary to get a pointer to the queue with the identifier for the stream endpoint desired. The Autopilot stream defines many different packet types, each of which follows the format given in Table 21 of reference (Vaglienti, 2007). The Autopilot stream packet size that it will be used in the project will be the TELEMETRY_HIGH_RESOLUTION and SYSTEM-STATUS_HIGH_RESOLUTION packets. To obtain this data, using the SDK library, we need to use the functions defined in the Autopilot Packet module. This module provides functions for encoding and decoding packets from the autopilot stream, as documented in (Vaglienti, 2007).

17.2.5 Onboard Sensor GUI

The main objective of the onboard sensor interface software is to read data from several

sensors concomitantly, and store them for post processing. This interface is a multithread module consisting of four major threads which are, according to Figure 16.1: the user interface thread, the nIMU sensor thread (not use for this project), the MMQ50 sensor thread, and, the Piccolo thread autopilot. Note that as the software receiver, the ipexSR runs in a separate module; it is not taken into consideration in this description. Each thread is represented in a different colour and, even though working concomitantly, it should be stressed that they are not synchronized. In the same figure, it is also possible to see that the start of each thread is delayed when compared to the first one. Another point is that those threads cannot be initialized at the same time.

The sensor interface GUI developed with Qt and C^{++} and shown in Figure 2.2 allows a user to graphically select which sensors to use during the flight tests, as well as, which one to connect at first. It should be stressed that no order of selection is previously required. As a consequence, the nIMU sensor, MMQ50 sensor, and Piccolo autopilot can be connected in order to run concurrently in separate threads.

As it has been stated before, the GUI is divided in two subsystems: the sensor server and the sensor data logger. The sensor Server subsystem is running into the user interface thread while the others tree threads are running in the sensor data logger subsystem. These two subsystems are briefly described as follow.

17.2.6 Sensor Server

The sensor server has the task to connect the nIMU, MMQ50 and Piccolo autopilot to the onboard embedded computer using a C^{++} serial communication class that has been developed for each type of communication. The serial communication class between sensors and computer has been developed taking into account specific requirements concerning the different types of connection. For instance, and according to Figure 16.1, the nIMU connects to the computer via the USB port; the MMQ50, as well as, the Piccolo avionics connect to the computer via a serial port (RS-232). And so, if the communication between sensors and the computer succeeds, it is possible to receive data without errors and provide, consequently, the sensor data logger with the sensor measurements.

Most important is that, since sensors do not operate with the same frequencies, we will have an asynchronous sensor data flow in the system, and, consequently, in the navigation loop. Thus reference time will be need for later synchronization purposes.

17.2.7 Sensor Data Logger

The sensor data logger is software implemented. The nIMU, MMQ50, and Piccolo autopilot threads are implemented in this subsystem with the responsibility of logging the several sensors data during the flight tests, to be used by the navigation algorithm in a post-processing mode. All data will be stored at different rates in separates files which will be decoded a posteriori using a C^{++} class. Furthermore, as data writing to a file is considered a non-real time task, although data collection will be done in real time mode, this application is considered a non-real time application. If a real time application is required then other considerations, related especially with synchronization purposes, must be taking into account.

17.3 Software Data Flow

Taking into account Figures 17.1 and 17.2 the generic flow of the software is now presented.

For instance, if the user presses the MMQ connect button on the GUI, it allows the user to configure the serial port and configuration parameters desired for communication. If this configuration succeeds, then the GUI calls the Run ThreadMMQ ( ) which creates the MMQ thread responsible for receiving measurements provided by this sensor. The main function of this thread is to read and save the data into a file in a loop until the user stops the respective communication, which is done by pressing the MMQ disconnect button on the GUI.

As a result, the MMQ data file contains accelerometer data, and Gyro rates information collected at 450Hz. Whenever the Piccolo connect button is pressed, the sensor interface GUI allows the user to configure the Piccolo avionics communication. It should be stressed that if working with the Piccolo SIL the user must select the communication via TCP/IP at address local host: 2001. Otherwise, if connected through the program port, the user must select serial communication at address COMi. Then, if this connection succeeds, the GUI calls the Run ThreadPiccolo ( ), which creates the thread to run the communication with Piccolo concurrently with the other two threads, until the user stops the connection. The function of this thread is to get telemetry data from the avionics and save it in two different files: the Piccolo telemetry data file and the Piccolo system status file. The first file contains data coming from the GPS/INS navigation filter, such as, position, NED groundspeed, Euler angles, barometric altitude, wind

velocity, static pressure, indicated airspeed, and, accelerometers and gyro measurements. Navigation health, rate bias, acceleration bias are provided by the second file.

17.4 IMU Integration Implementation

Although two IMU have been integrated in the developed software, only the MMQ50 is considered as part of this thesis.

In order to integrate the MMQ50 in the software developed, a good knowledge about the MMQ50 message data format had to be considered. The IMU sensors data format specifications, as well as, communication parameters necessary to allow the user to receive data from sensors are presented now.

17.4.1 MMQ50 Message Format

MMQ data messages are transferred across RS-232, at baud rate of 115200 bits per second, 8 data bits per byte, no parity, 1 start and 1 stop bit per byte. All serial outputs are a tag word followed immediately by a checksum and then one or more data words.

The MMQ50 message transmission is sent Low-Byte, High-Byte, that is, for each 16 bit data word, the least significant byte is transmitted first, followed by the most significant byte. Each word is separately byte-reversed. Each message contains a checksum as word 2. The arithmetic sum of the entire message should be zero in the low 16 bits received correctly.

The MMQ50 data message structure is depicted at Figure 17.3. And so, according to that figure, the data message header word is 0 × 7FFF. The second word is a 2s-complement of the 16 bit sum of the rest of the entire message. The data word count is nine where six belongs to the payload data, and which meanings are present at Table 17.1. A message word length is 16 bits and so, due to the fact that the protocol uses s standard Universal Asynchronous Receiver Transmitter (UART), two bytes are required in order to make up one message word.

The normal data message is the message transmitted in the nominal operational state, at a nominal rate of 450 Hz. Accelerations and rates are in units of $\mathrm{m/s^2}$ and $\circ/\mathrm{s}$. For delta-velocity and delta-theta, the acceleration and rate must be divided by the output rate. The scale factor that must be used to convert accelerometer data is 326:3 $\mathrm{counts/m/s^2}$ (nominally 3200 $\mathrm{counts/m}$), and 100 $\mathrm{counts}/\circ/\mathrm{s}$ for the rate sensors.

As an illustration, Figure 17.4 shows an example of how the sample message is received

and then parsed. MMQ50 data is received in a character format and then converted in hexadecimal format. Remember that coded data are sent byte-reversed, so byte 1 is the first byte received, byte 2 is the second byte received, and so on. Accelerometer and gyro are temperature compensated, but no temperature information is provided. After converting the payload elements of the data packet (accelerometer, and, gyro samples) to hexadecimal values, they must then be converted to values that represent usable data (e.g., rotational rate, G- force, gauss). In order to be able to convert the sample the following equation must be used:

$$\text{Result} = \text{Raw\_Payload\_Value} \times \text{Digital\_Sensitivity} \qquad (17.1)$$

Where, Raw_payload_Value are values taken from the sample payload corresponding to each element, for instance, to the Z component of the accelerometer, and, gyro, respectively, as it will be shown in the following example. Digital sensitivity for this sensor is presented at the table below. Thus, the correspondent value in engineering units for the Z accelerometer, and Z rate gyro would be given by Table 17.2.

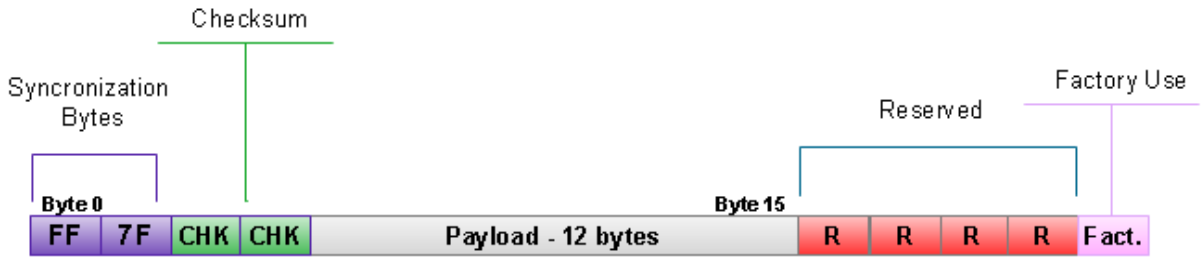

Figure 17.3 MMQ50 sample structure.

Table 17.1 MMQ50 normal data message format.

Word #	Meaning	Length (bits)	Minimum (Decimal/Hexadecimal)	Maximum (Decimal/Hexadecimal)
1	Compensated X Accelerometer	16	- 32750/8012	32750/7FEE
2	Compensated X Rate	16	- 32750/8012	32750/7FEE
3	Compensated Y Accelerometer	16	- 32750/8012	32750/7FEE
4	Compensated Y Rate	16	- 32750/8012	32750/7FEE
5	Compensated Z Accelerometer	16	- 32750/8012	32750/7FEE
6	Compensated Z Rate	16	- 32750/8012	32750/7FEE

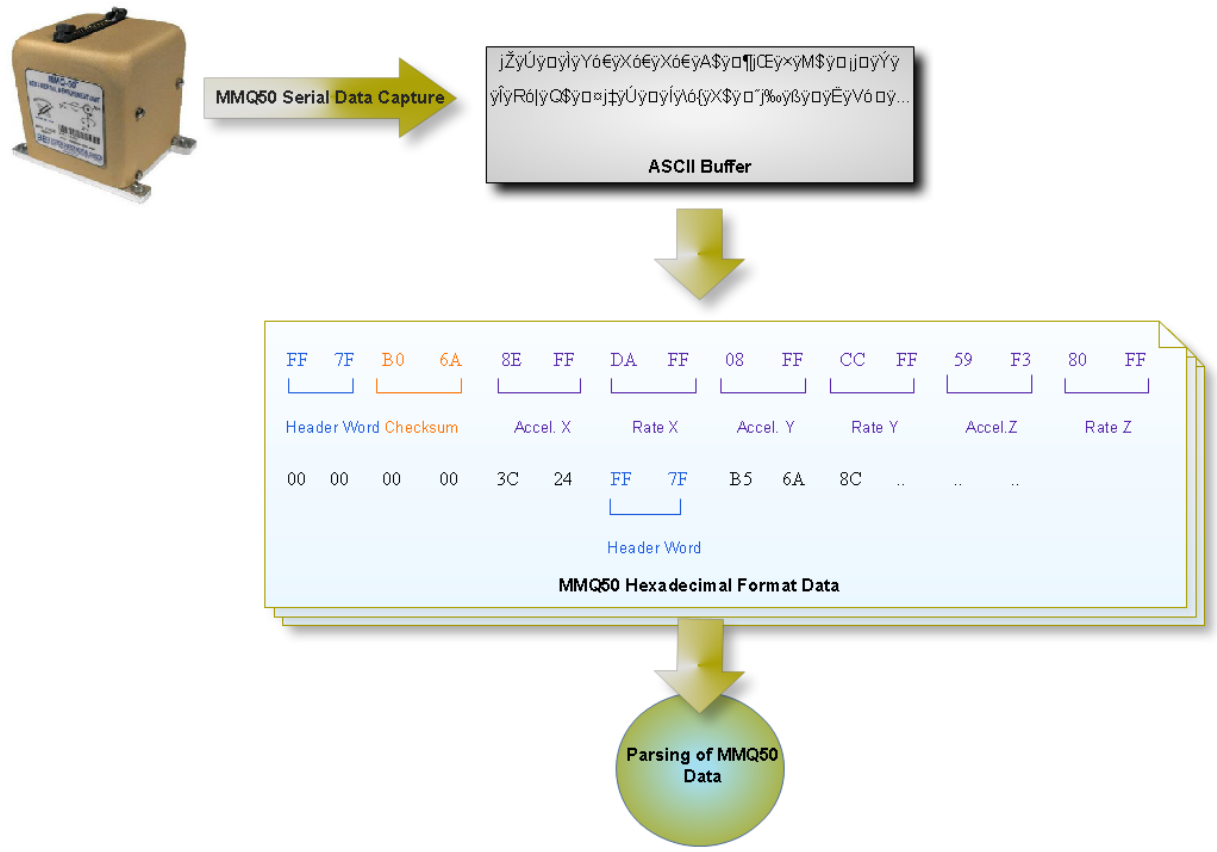

Figure 17.4 MMQ50 sample message.

Table 17.2 MMQ50 data parsing example.

Component	Digital Sensitivity	Raw_Payload_Value for Z measurements (16 bits)	Result (Equation 17.1)
Z Accelerometer	$3.06466 \times 10^{-3}\ \mathrm{m/s^2/counts}$	$F359_{16} = (-3239)_{10}$	$-9.9264\ \mathrm{m/s^2}$
Z Rate Gyro	$10^{-2}\ °/\mathrm{s/counts}$	$FF80_{16} = (-128)_{10}$	$1.01230\ °/\mathrm{s}$

CHAPTER 18

VAN TESTS

18 Van Tests

This chapter is devoted to the presentation of the results obtained for the closed loop loosely coupled SINS/$GNSS_{SR}$ system. In order to perform the van-test, all the equipment presented at Chapter 16, Figure 16.1, purple box, has been embedded into the van, according to Figure 18.1.

The main goal of this chapter is to analyse the results obtained for the IMU/$GNSS_{SR}$ integration algorithm with the van-tests. The van tests consisted in the collection of real data, in the GATE Test area, for post processing analysis.

The specific objectives of this chapter are the investigation of the performance of the developed SIMU/ $GNSS_{SR}$ integrated navigation algorithm in Part IV; the investigation of the performance when the GATE Galileo measurements are included, and, finally, the confrontation of the performance of the integrated SIMU/ $GNSS_{SR}$ solution with the solution provided by the commercial autopilot.

Once those specific objectives are achieved, we will be able to answer the last secondary research question of this thesis, here remembered: *Can we guarantee a navigation solution with at least the same accuracy as the one provided by a commercial autopilot navigation solution? Does the Galileo data improve the navigation solution?*

Again, it is important to remember that the evaluation of the navigation algorithm developed, at Part IV Chapter 15, was performed in post processing using real data, obtained with the datalloger developed by the author and presented at Chapters 16 and 17.

18.1 Campaign Description

A reference trajectory obtained by an external GPS- RTK receiver will be compared with both, the autopilot solution (Piccolo) and the IMU/$GNSS_{SR}$ integration algorithm solution, in order to validate the solidity of the developed algorithm. The external GPS receiver is a Novatel SPAN-SE with a Novatel L1/L2 Model 502 antenna.

The SIMU/$GNSS_{SR}$ raw data is integrated in a post-processing mode and evaluated using the signals provided at the GATE test area. The GPS/Galileo raw data is provided by the ipexSR software receiver.

Dynamic tests for logging MMQ50 IMU data, Piccolo autopilot data, Galileo/DATE position with precise reference position data was performed in GATE Berchtesgaden, on 18 June 2010. GATE was on Virtual Satellite Mode (VSM). The duration of the test was conducted for about 53 minutes.

Before driving the bus, a 15-minute static calibration was performed, in order to get initialization values for the strapdown mechanization equations. This point is called the GATE Central Point (GCP); corresponding to Zone 1, see Figure 18.2.

Table 18.1 summarizes the alignment values obtained at GCP.

Table 18.1 Alignment values.

Initial conditions	Values
$\underline{\mathbf{R}}^{\mathbf{GEO=ENU}}(\mathbf{0})$	$\begin{bmatrix} l(0)(\text{deg}) \\ \text{L}(0)(\text{deg}) \\ \text{h}(0)\ (\text{m}) \end{bmatrix} = \begin{bmatrix} 47.610 \\ 12.97 \\ 668.15 \end{bmatrix}$
$\underline{\mathbf{v}}^{\mathbf{GEO=ENU}}(\mathbf{0})$	$[0 \quad 0 \quad 0]^{\text{T}}$
$\mathbf{C}_{\mathbf{B}}^{\mathbf{GEO=ENU}}(\mathbf{0})$	$\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} -0.77 \\ -1.03 \\ 199.55 \end{bmatrix}$ (deg) introduced into Equation (A.33)in order to obtain $\text{C}_{\text{B}}^{\text{ENU}}$

After the 15 minutes, the dynamic tests started. We drove to Turning Point West, then to Turning Point East at GATE Office, and then stopped at Bus Stop GCP, see Figure 18.2. As next, we started the large test tour West, Zone 2. Stopped again at GCP. After, we drove to the parking area Königssee, where several cyclic turns were performed, Zone 3. Stopped and then we started a high speed test on B20 with a drive to roundabout Berchtesgaden and back to Bus stop GCP via Kohliasl. During the entire described trajectory, the data was collected for post-processing tests in order to evaluate the proposed SIMU/$GNSS_{SR}$ algorithm.

Figure 18.1 Van test onboard equipment.

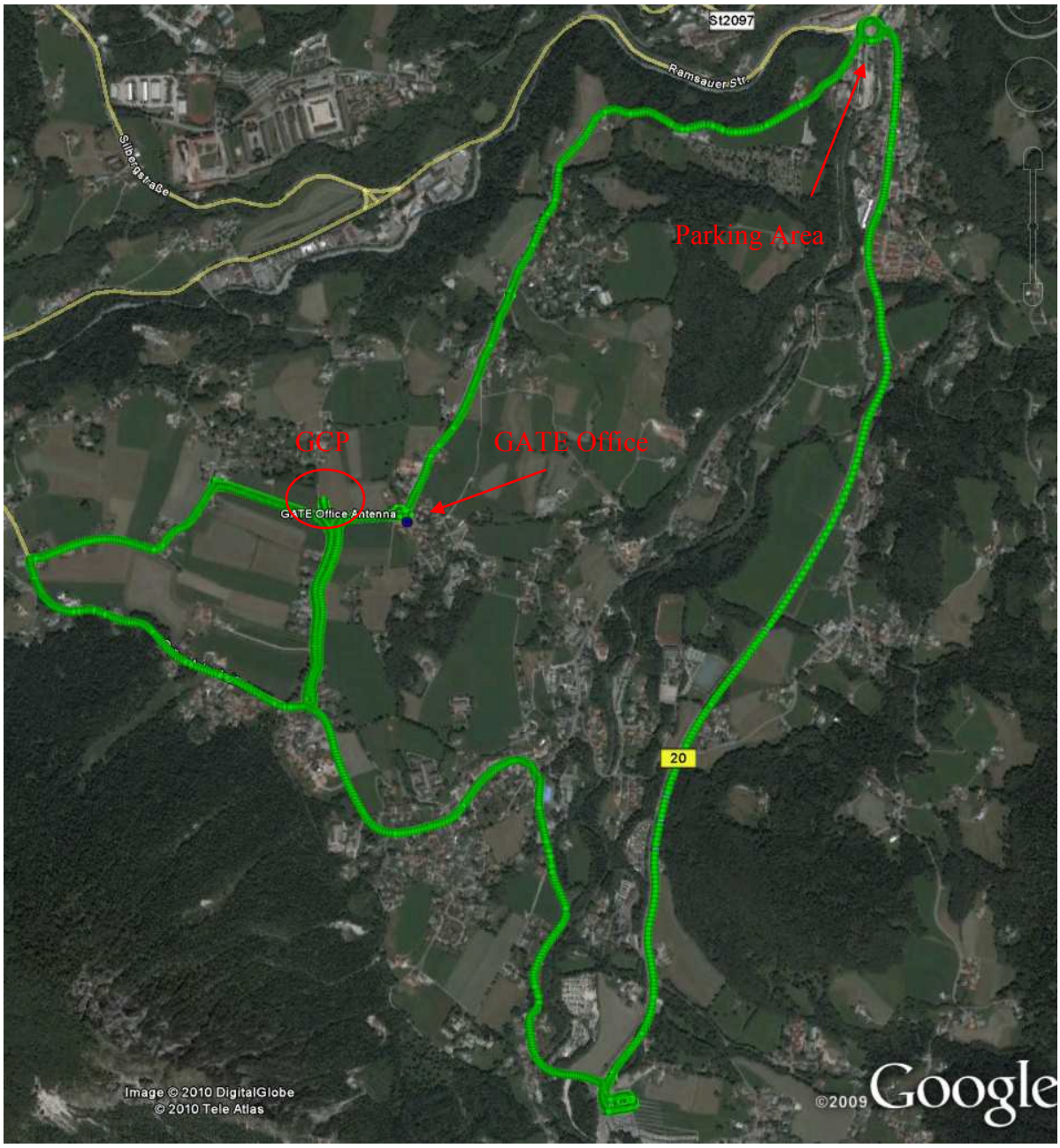

Figure 18.2 Van test trajectory performed.

18.2 Results and Discussion

In order to be able to apply the developed EKF, there are some parameters that must be input to the system, such as the system and measurement noise covariance matrices, Q and R, and the initial error covariance matrix, P_0. The parameter values are presented at Table 14.2. Level arm values are also needed and presented at Chapter 14 by Equation (14.30).

18.2.1 Inertial Sensor Signals

Figures 18.3 and 18.4 illustrate the accelerometer raw data from both the MMQ50 IMU and the Piccolo autopilot IMU, for about 53 minutes of collected data, corresponding to the trajectory described before. Although the MMQ50 frequency is 450 Hz and the Piccolo IMU solution is provided at 1 Hz, the MMQ50 raw data has been sampled to 1 Hz too.

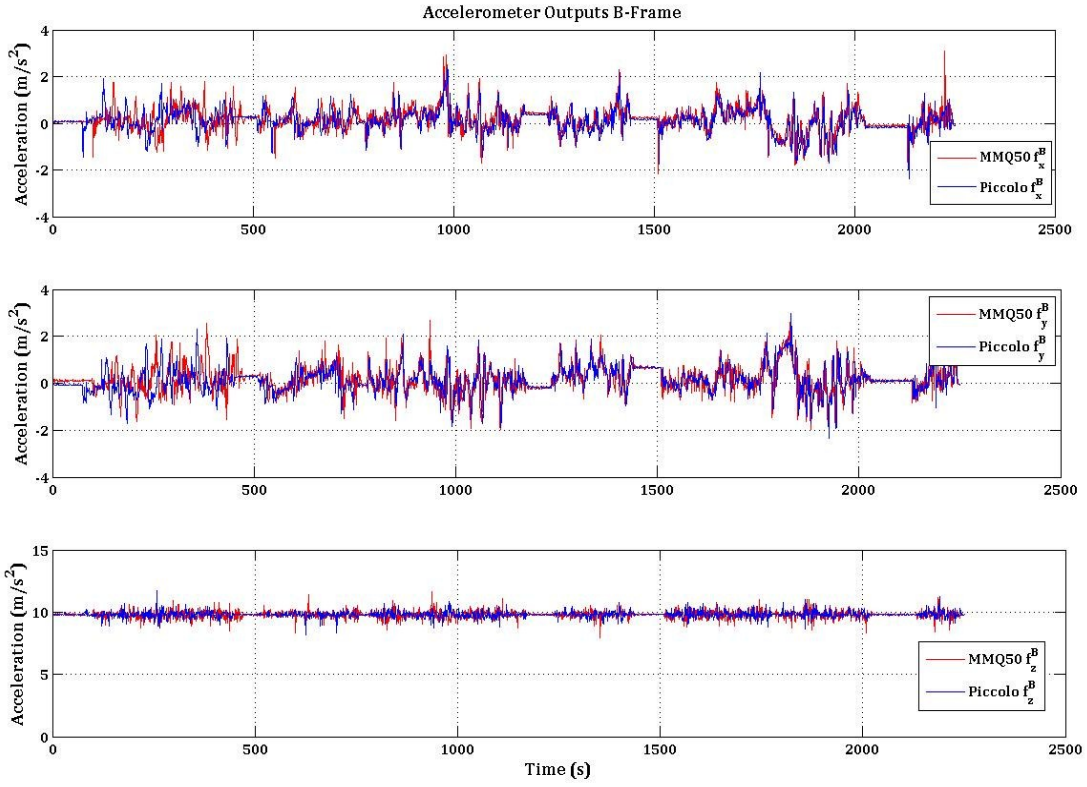

Figure 18.3 MMQ50 and Piccolo IMU accelerometer raw data in the B-Frame.

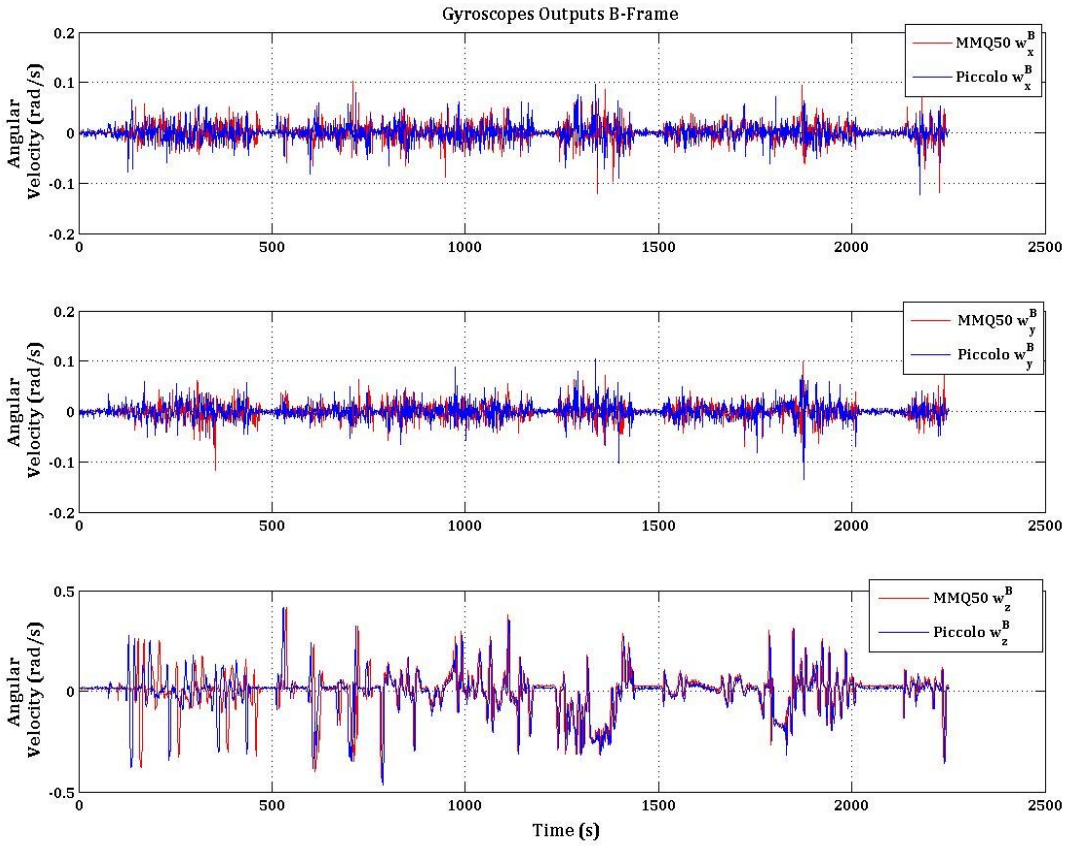

Figure 18.4 MMQ50 and Piccolo IMU accelerometer raw data in the B-Frame.

It should be stressed that, during the tests, we encountered some problems with the data provided by the MMQ50. The collected data for the first 15 minutes was not good, maybe due to temperature stabilization. Consequently, the results presented at Figures 18.3 and 18.4 are for the rest of the static data, approximately 1 minute, and for the dynamic part of the rest of the run.

Looking at Figure 18.3, at the beginning, when stopped at GCP point, we can see zero output in both x and y accelerometers while the z accelerometer is reading the local gravity component. These results are obtained for both IMU sensors, being compatible with a static leveled platform.

It can be stated that there is a good match between the data provided by both the IMU sensors. Looking at the gyros outputs, it is possible to see that the y and, principally, the z gyro present, for both sensors, some spurious signals. In fact, for gyros that cannot sense the Earth's rotation, it is clearly acceptable and understandable that the technology behind both types of

gyros is sensitive to undesired linear acceleration and spurious vibrations acting on the sensor. It is clear that both sensors, due to location where they were placed, were highly sensitive to the car vibrations and to tight corners, which can cause changes in angular velocities up to 30º/s.

Comparing the inertial signals provided by both sensors, the MMQ50 and Piccolo IMU, the following statistics have been achieved:

Table 18.2 Statistic of inertial raw data.

	Accelerometer Measurements $\mathbf{m/s^2}$			**Gyros Measurements** $\mathbf{rad/s}$		
	$\mathbf{f_x^B}$	$\mathbf{f_y^B}$	$\mathbf{f_z^B}$	$\mathbf{w_x^B}$	$\mathbf{w_y^B}$	$\mathbf{w_z^B}$
MMQ50						
Mean	0.1573	0.1764	9.8	0.000486	0.0000053	-3.739×10^{-6}
STD	0.5142	0.5879	0.3021	0.01702	0.0146	0.0976
STD (after denoised)	0.285	0.3543	0.0592	0.0042	0.00339	0.085
Piccolo IMU						
Mean	0.1099	0.138	9.807	0.000152	0.0002327	-0.0067
STD	0.4861	0.5557	0.2452	0.01725	0.01621	0.09787

The respective figure of errors between sensor axes is given by Figures 18.5 and 18.6. From the statistics obtained from both sensors, it is not clear if the MMQ50 sensors are of better quality than the Piccolo one. It can be concluded that both sensors belong to the same category.

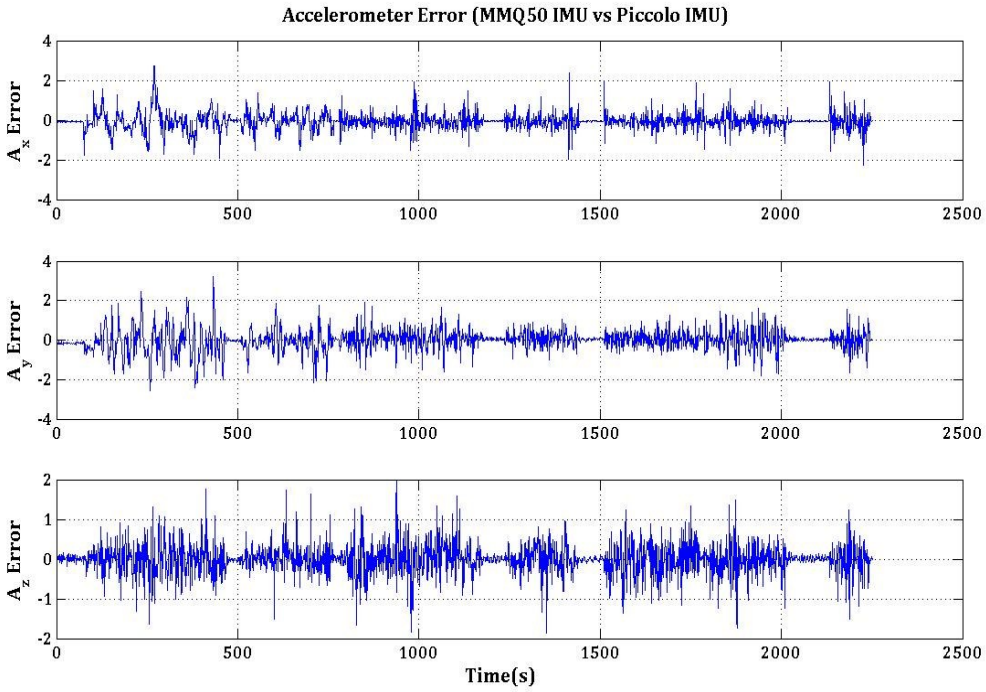

Figure 18.5 Differences between MMQ50 accelerometer raw data and the Piccolo accelerometer raw data.

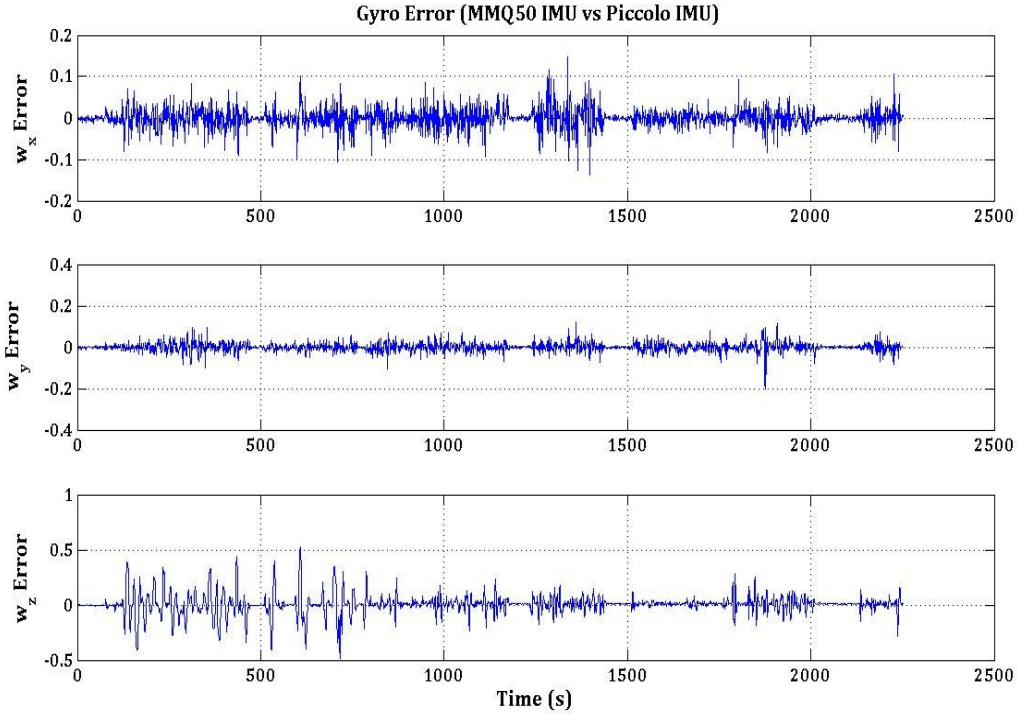

Figure 18.6 Differences between MMQ50 gyros raw data and the Piccolo gyros raw data.

The statistic values obtained for the MMQ50 sensors data, after denoised, are also presented at Table 18.2. MMQ50 raw data have been denoised with a wavelet db6 (Daubechies LOD of 6) and the results are presented at Figures 18.7 and 18.8.

According to both figures, it is clear that the white noise, present in the high frequency component, has been removed. Therefore, the EKF intends to estimate the low frequency noise component.

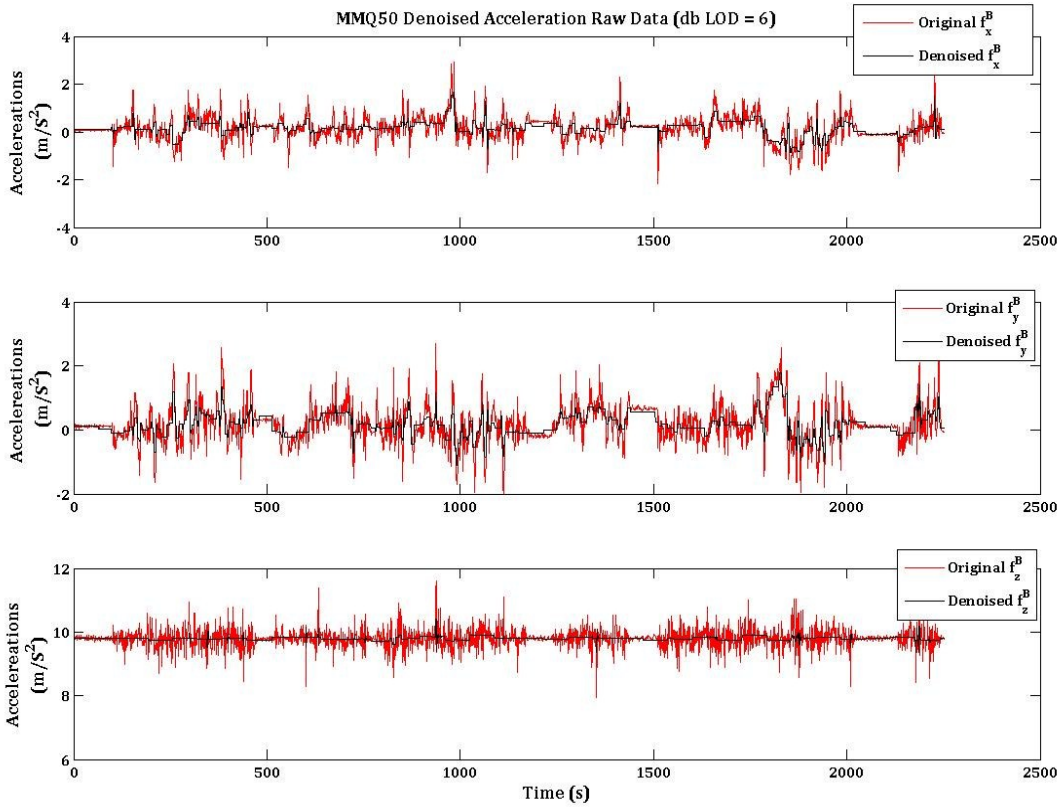

Figure 18.7Denoised MMQ50 accelerometer raw data.

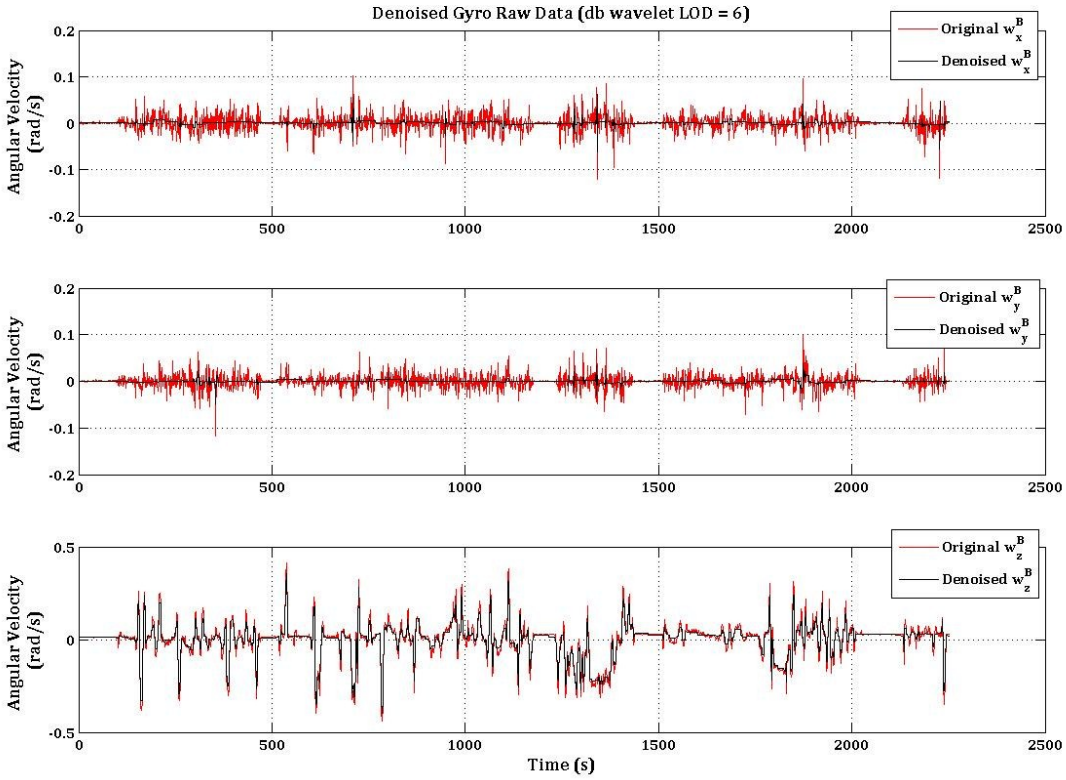

Figure 18.8 Denoised MMQ50 gyro raw data.

It is important to realize that the denoised data, obtained from Figures 18.7 and 18.8, will be the data to be fed into the mechanization equations, in order to get the SIMU standalone solution.

18.2.2 Estimated Position

18.2.2.1 Latitude/Longitude/Height

Figure 18.9 corresponds to the horizontal reference trajectory, divided into the three zones. Static data collection was done at GCP, which belongs to Zone 1. The 3D trajectory for the GPS-RTK (reference), Piccolo, SIMU stand alone and SIMU/GPS is depicted at Figure 18.10. In this figure, it is clear the difference in altitude for the Piccolo and SIMU solution. The SIMU/GPS follows the GPS-RTK reference trajectory in altitude as well.

Figure 18.11 depicts the vertical trajectory for the same solution. Looking at the Piccolo curve, it can be stated that it suffered some outage, which is why the curve seems to be delayed.

In fact, we have to admit that this work has a gap related to the synchronization of the several data. This is a problem that must be overcome in future work. Nonetheless, the problem related to the Piccolo delay has been overcome in the analysis done in this thesis.

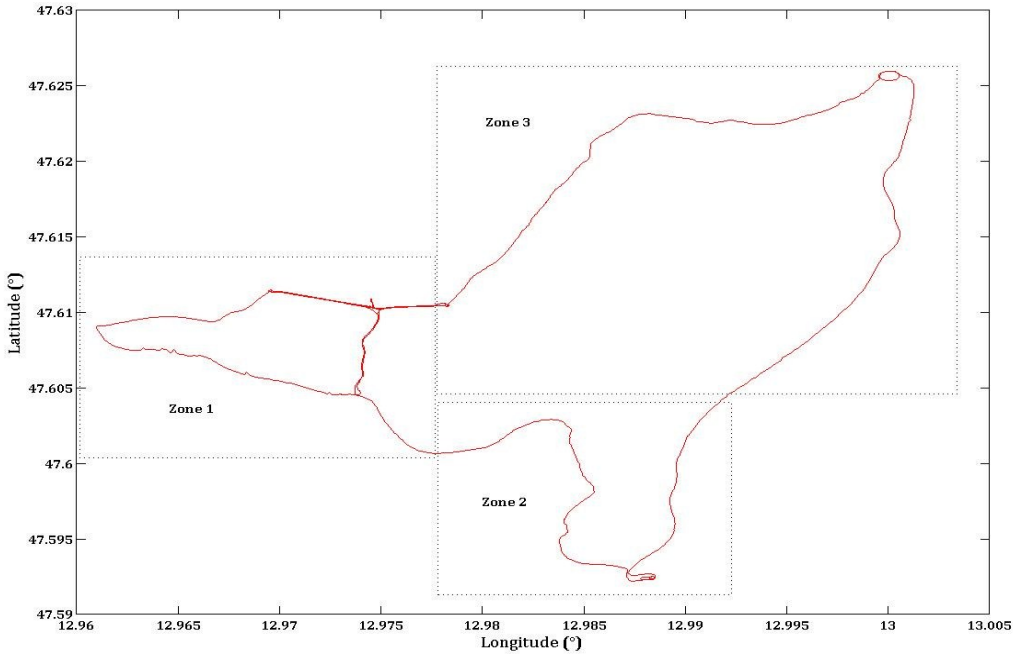

Figure 18.9 Horizontal reference trajectory

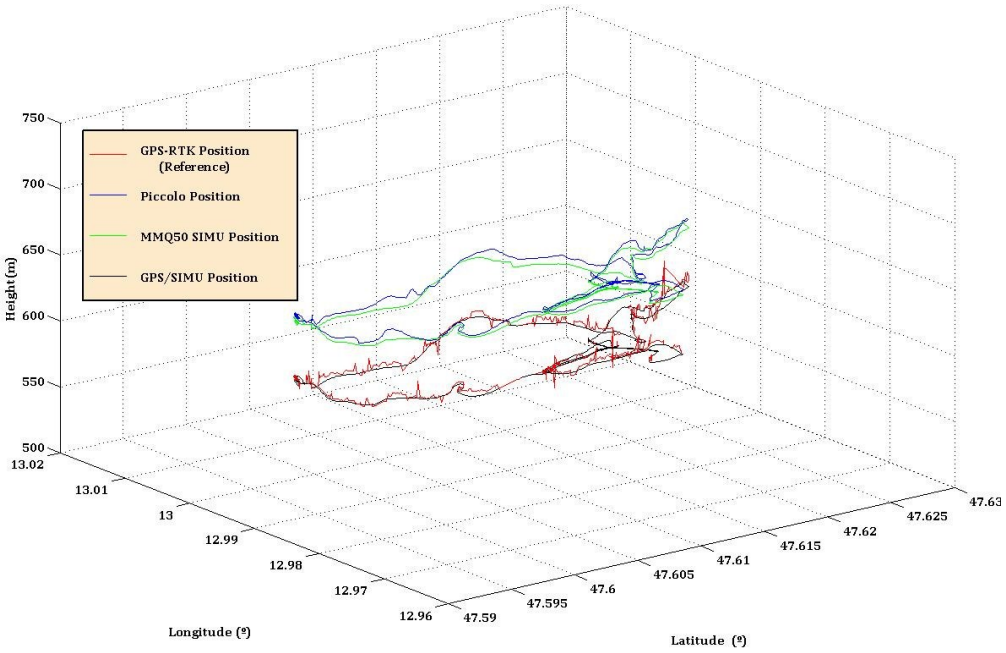

Figure 18.10 3D trajectory: GPS-RTK reference trajectory (red); piccolo position (blue); SIMU solution (green) and SIMU/GPS (black).

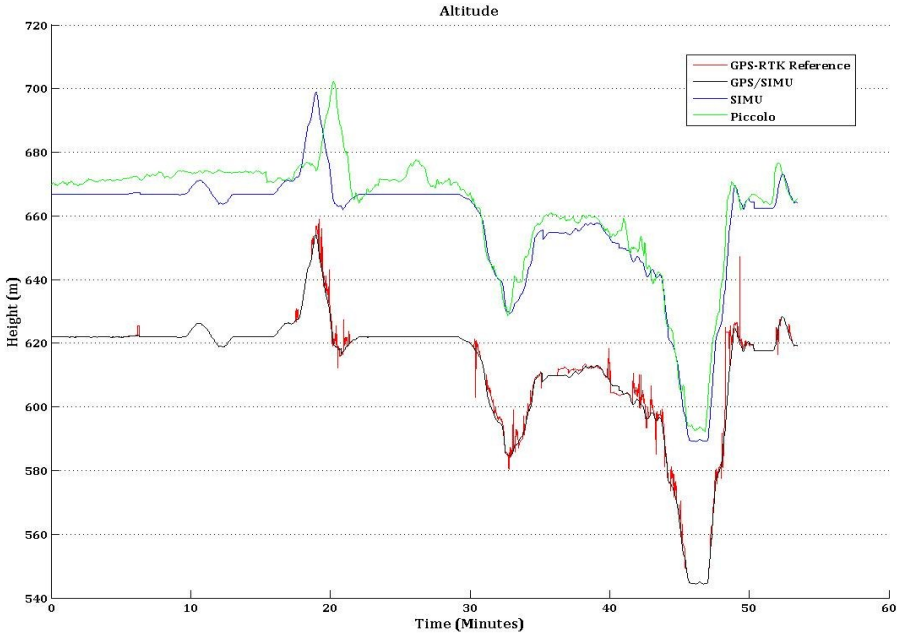

Figure 18.11Altitude: GPS-RTK reference height (red); SIMU/GPS (black); SIMU alone (blue) and Piccolo height (green).

According to Figures 18.10 and 18.11, it is clear that the developed SIMU/GNSS$_{SR}$

position solution has the same behavior of the reference position. The estimated position follows the reference curve, not only in latitude and longitude, but also in respect to altitude.

Regarding the Piccolo position solution, it can be stated that although it behaves as the reference in latitude and longitude, the same is not achieved for altitude, where we can find bigger differences between the two solutions. Moreover, comparing the solution provided by the Piccolo with the one provided by SIMU stand alone, it is possible to say that they behave in a very similar way regarding the latitude, longitude and altitude. Figure 18.12, for instance, depicts the horizontal solution obtained by both in Zone 2, in order to make it clear how close the two solutions are. Figure 18.13 depicts the horizontal position, in the same zone, for the Piccolo and the reference. Confronting the two figures, it can be stated that the Piccolo position solution behaves more as the SIMU standalone solution than as the reference. As a consequence it is possible to conclude that the SIMU solution accuracy is similar to the one provided by the Piccolo solution.

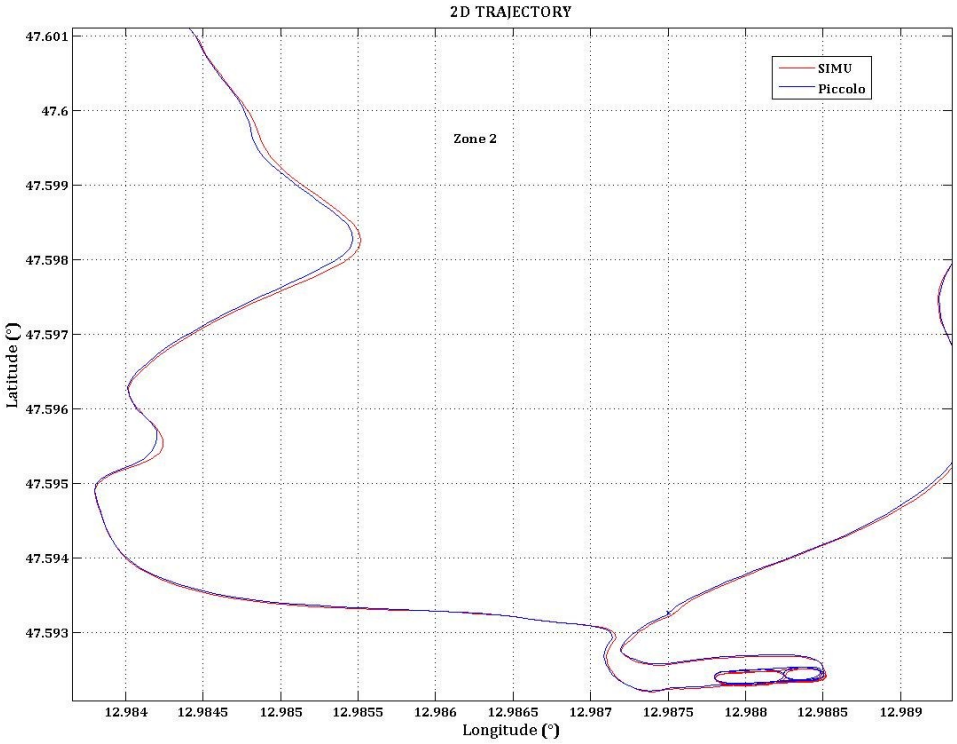

Figure 18.12 SIMU (red) versus Piccolo (blue) geodetic position in Zone 2.

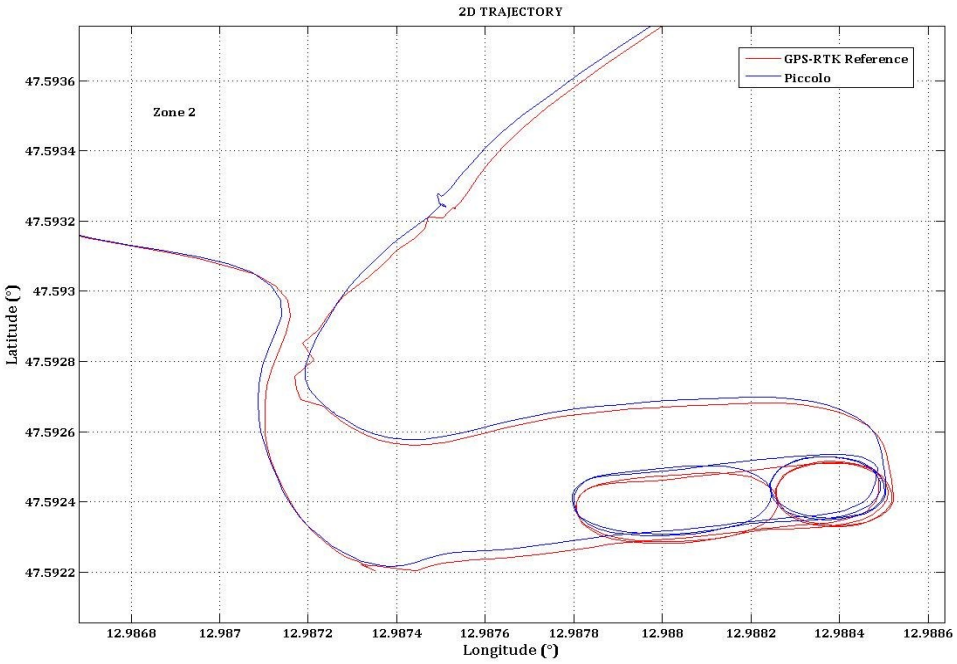

Figure 18.13 Reference (red) versus Piccolo (blue) geodetic position in Zone 2.

Galileo Measurements

Figures 18.5 to 18.7 presents the 2D trajectory obtained with Galileo receiver 1 frequency E1.

It should be remembered that Zone 1 is the zone where the static data have been collected, which is why we have more Galileo measurements in the GCP area. However, if we take into account what happened at Zone 2 and 3, it can be concluded that we have some lack of Galileo data during the dynamic trajectory.

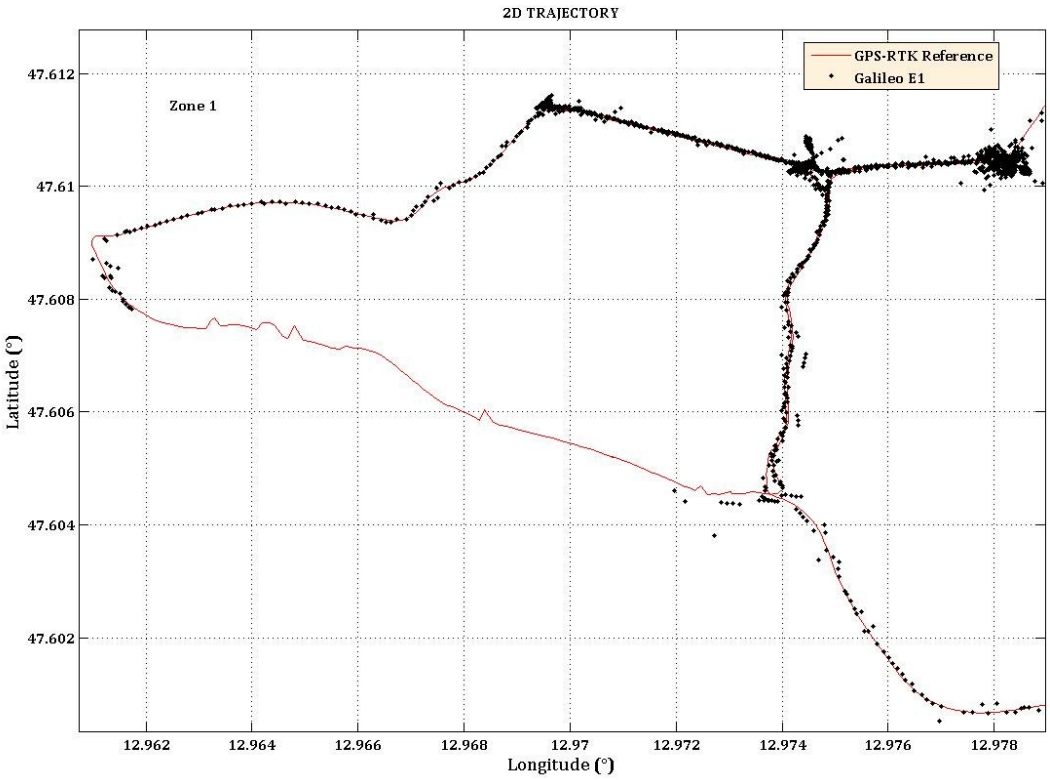

Figure 18.14 Zone 1 with inclusion of Galileo E1 measurements.

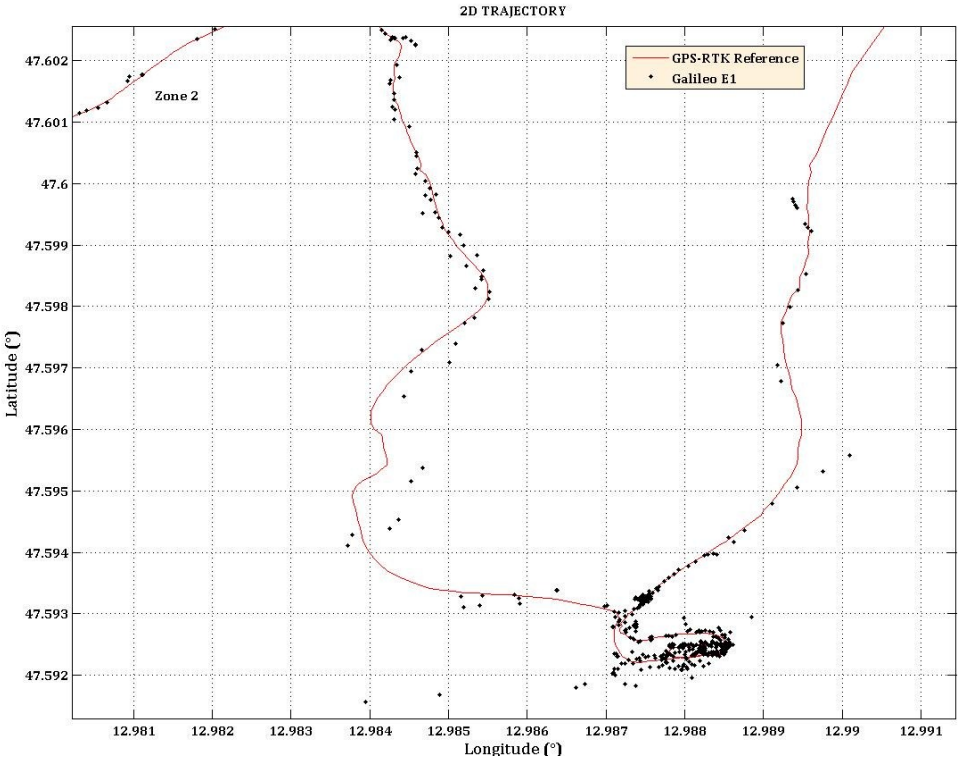

Figure 18.15 Zone 2 with inclusion of Galileo E1 measurements.

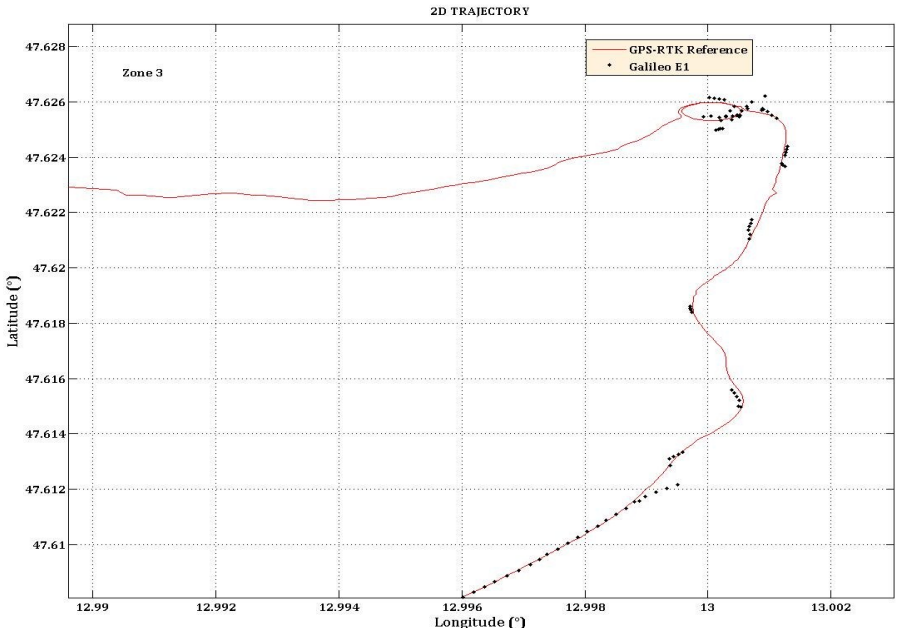

Figure 18.16 Zone 3 with inclusion of Galileo measurements.

Unfortunately, and according to the exposed figures, it can be concluded that not enough Galileo data were provided to evaluate the EKF with the Galileo data, as it is possible to see at Figure 18.20. The error in altitude was also too big to use such data in the filter.

As a consequence, it is possible to answer the second part of question seven, which asks, "Does the Galileo data improve the navigation solution?" As a matter of fact, the evaluation was not performed, given the fact that Galileo presents several outages and, also, mainly due to the bad quality of the vertical solution provided by the GATE Galileo receiver 1.

Consequently, the results for the SIMU/$GNSS_{SR}$ only include the integration with the GPS and, so, it is called SIMU/GPS along this analysis.

18.2.2.2 Errors in Latitude/Longitude/Height

Figures 18.18 to 18.19 are related to the position parameter (latitude, longitude and height) propagation errors. The exposed errors were computed as:

- The difference between the estimated position solution provided by the SIMU/GPS integration EKF and the reference solution provided by the GPS-RTK.
- The difference between the estimated position solution provided by the Piccolo solution and the reference solution provided by the GPS-RTK.

In order to better understand the difference between the two solutions, the SIMU/GPS and Piccolo, and to give an answer to the first part of **Q7**, the statistic values associated to the errors are presented at Table.

Figure 18.18 also presents the error obtained using the SIMU stand alone, just as a curiosity, and Figure 18.20 depicts the Galileo position parameters propagation error.

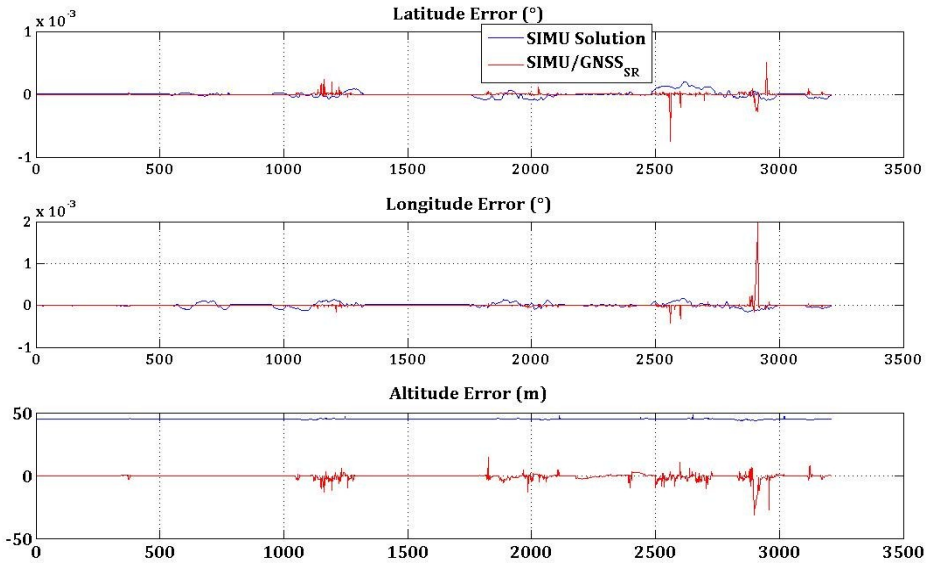

Figure 18.17 Errors for the SIMU standalone solution (blue) and for the SIMU/GPS solution (red).

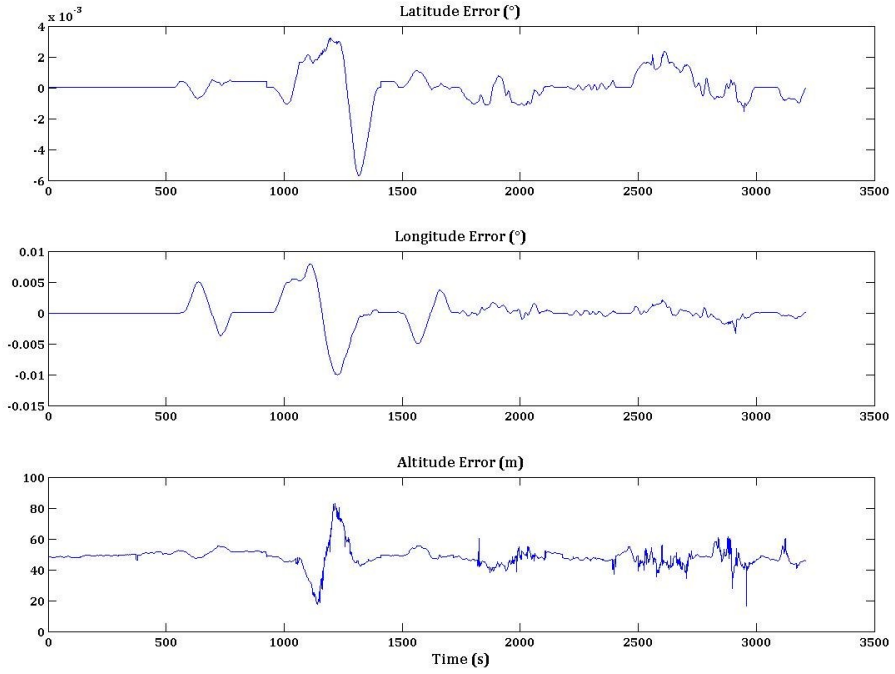

Figure 18.18 Errors in position for the Piccolo provided solution.

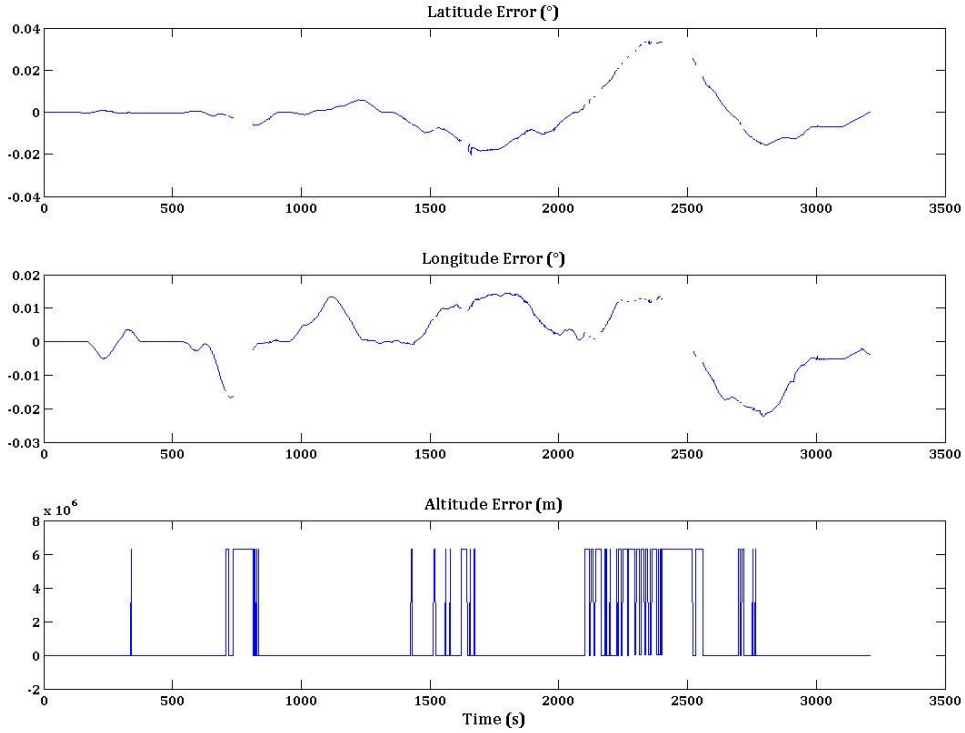

Figure 18.19 Errors in position for the Galileo E1 provided solution.

Confronting Figure 18.18 with Figure 18.19, it can be stated that the position difference errors are bigger for the Piccolo parameters and for the SIMU altitude parameter. The Piccolo figures present some spurious signals for some points. Those differences arises due to the outages problems that the Piccolo suffered during all the run. It is important to remember that the solution provided by the Piccolo is the filtered solution, obtained with low performance embedded inertial sensors, and so, with large sensor errors and poor observability, it is normal that the solution diverges for some seconds.

Table 18.3 presents a statical summary for the position parameters error propagation, obtained using the SIMU/GPS and the Piccolo solution. In order to easily understand those errors, the angular position values have been converted to meters.

Table 18.3 Position parameters propagation errors statistics.

	Position Error (m)		
	X	**Y**	**Z**
SIMU/GPS			
Mean	0.4213	0.727	0.7765
STD	3.64	2.637	3.722
Piccolo Solution			
Mean	7.74	5.863	9.12
STD	4.459	9.275	4.526

According to the exposed values, there are no doubts that the SIMU/GPS solution is of a better quality than the Piccolo solution.

18.2.3 Estimated Velocity

Figures 18.21 to 18.23 depict the estimated NED velocity, for the reference and for the Piccolo provided solution. The velocities are given in meters per second.

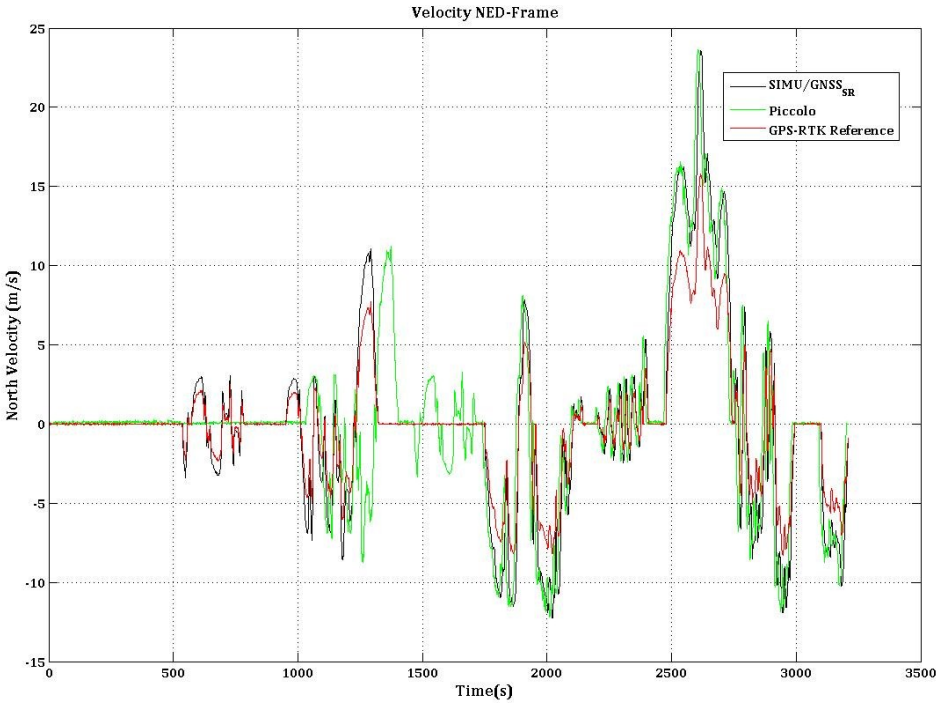

Figure 18.20 SIMU/GNSS$_{SR}$ estimated Piccolo and Reference North velocity.

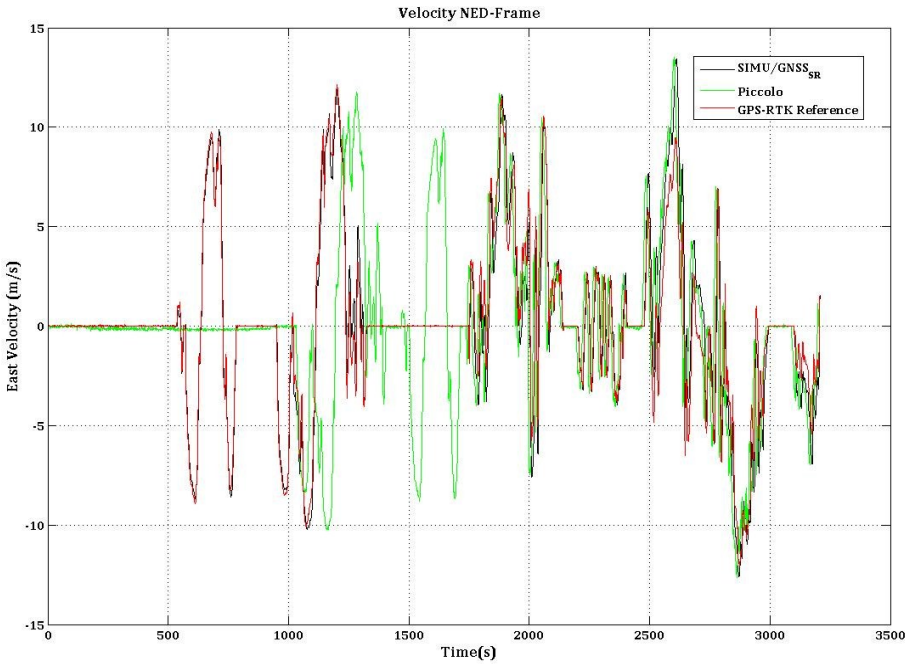

Figure 18.21 SIMU/GNSS$_{SR}$ estimated Piccolo and Reference East velocity.

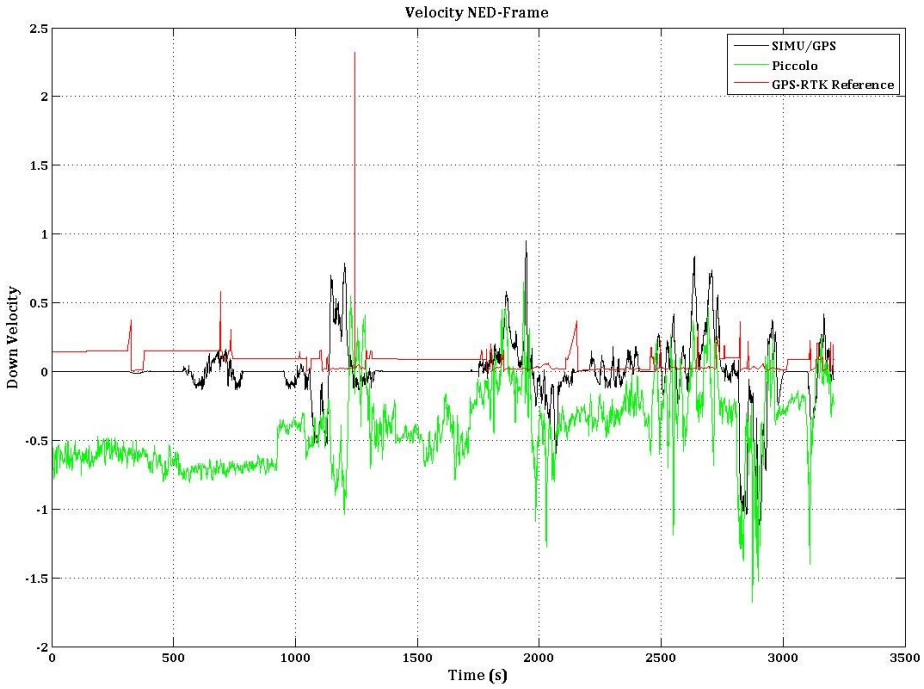

Figure 18.22 Estimated, Piccolo and Reference Down velocity.

18.2.3.1 Errors in velocity

As it has been done for the position errors, Table 18.4 presents the values for the velocity parameter error propagation and attitude errors obtained using the SIMU/GPS and the Piccolo solution. Those values are also in meters per second.

According to Figure 18.21 and 18.22, the estimated North and East velocities have the same behavior of the reference velocity. However, some differences appear at the down velocity. In fact, the vertical velocity is of a lower accuracy for both the SIMU/GPS solution and for the Piccolo solution.

Table 18.4 Velocity parameters propagation errors statistics.

	Velocity Error $(\mathbf{m/s})$		
	$\mathbf{V_N}$	$\mathbf{V_E}$	$\mathbf{V_D}$
SIMU/GPS			
Mean	-0.023	0.0026	-0.00232
STD	1.894	1.416	4.017
Piccolo Solution			
Mean	0.037	-0.05409	0.3982
STD	3.135	3.575	4.946

Table 18.4 presents the mean errors and standard deviation associated to the estimated velocities and to the Piccolo velocity solution. Once again, the results are of better quality for the estimated solution. The std has its maximum for the Down component in both solutions.

18.2.4 Estimated Attitude

The comparison results for the estimated attitude, Piccolo attitude and reference attitude are presented at Figure 18.24.

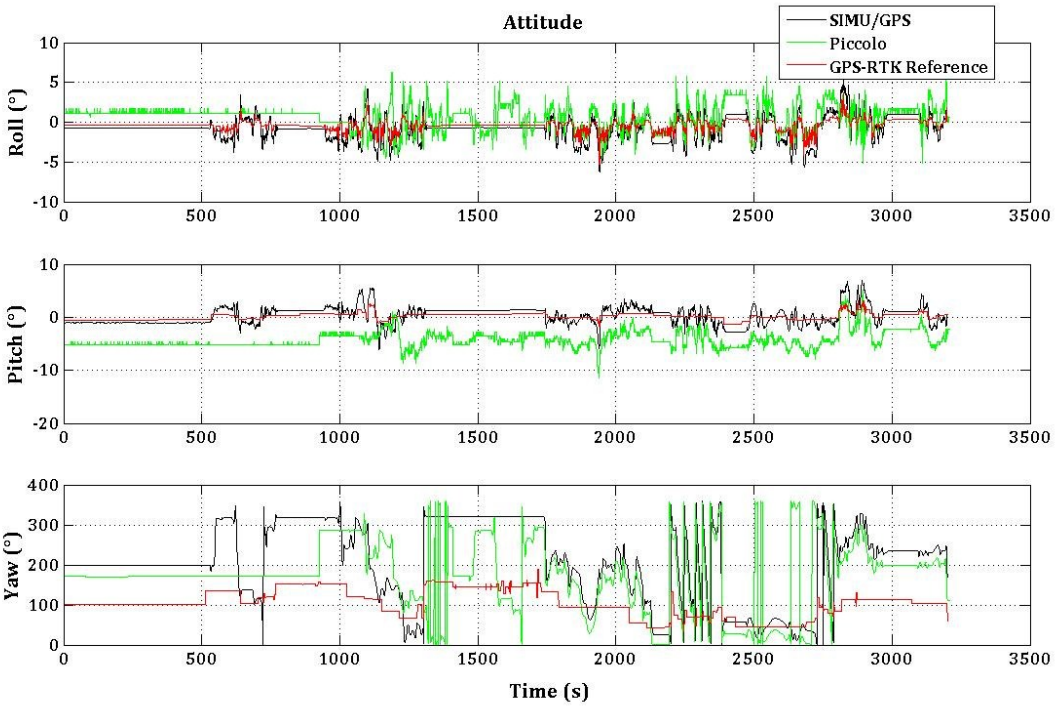

Figure 18.23 Attitude: SIMU/GPS (black), Piccolo (green) and Reference (red).

In order to better understand the results, we cannot forget that the test was done with a car. Thus, big changes in roll and pitch were not expected. In fact, they do not change significantly. However, the same does not happen with the heading, whose determination is very demanding. According to Figure 18.23, for both the roll angle and the pitch angle, the solution provided by the SIMU/GPS is satisfactory. In fact, the values vary around zero. We have some small changes due to tight corners.

Regarding the Piccolo solution, the roll and pitch angles have a behavior similar to the SIMU/GPS solution, but with large differences from the reference. It should be noted that the Piccolo solution is the filter solution, but in an open-loop configuration. In fact, the Piccolo inertial raw data is not corrected, as it is the case of the MMQ50 raw data.

Relating to the heading, the results are not good enough. In fact, we have to realize that there was no other independent sensor to measure the heading. For instance, we could have used magnetometer measurements in the developed EKF measurement model, in order to directly observe the heading. The SIMU/GPS does not behave as the reference. They seem to be in a disagreement. In the Piccolo solution, the heading jumps from positive to negatives values. It should be remembered that the car had been in a roundabout, for one minute approximately, and

its impact is also reflected in the data. One interesting analysis could be to analyze what would happen when removing all this effects. As a matter of fact, with UAV, these extreme conditions are not so usual.

The results shown at Figure 18.23 are reflected at Table 18.3, which presents a statical summary of the attitude errors in degree.

Table 18.5 Attitude parameters propagation errors statistics.

	Attitude Error (full run) (deg)		
	Roll	**Pitch**	**Heading**
SIMU/GPS			
Mean	- 1.8	0.1063	17.1
STD	4.56	2.452	35.2
Piccolo IMU			
Mean	3.45	5.23	18.9
STD	6.9	4.7	37.07

Analyzing the table, the values that have been obtained for the roll and pitch indicate a better performance in the estimation provided by the SIMU/GPS solution than by the Piccolo. The heading values do not satisfy, even though, they are of better quality than the ones of the Piccolo.

Overall, it can be stated that the SIMU/GPS solution offers a better quality than the Piccolo. It does not offer decimeter or centimeter level position accuracy but is better than the Piccolo. As a matter of fact, the values are not excellent, although, they are better than the Piccolo solution. In the low performance world, where sensors are far from perfect, it is impossible to expect more.

18.3 Part V Conclusion Summary

The **specific objectives** associated to Part V were: *i)* to present procedure guidelines, to develop all of the hardware and software required for this type of projects; *ii)* to investigate the performance of an SIMU/ $GNSS_{SR}$ integrated navigation algorithm, in a loosely coupled closed-loop architecture; *iii)* to investigate the performance when the GATE Galileo measurements are included; *iv)* to confront the performance of the integrated SIMU/ $GNSS_{SR}$ solution obtained in the Van-Test scenario with the solution provided by the commercial autopilot.

The seventh secondary research question, **Q7**, of this study, associated to the first specific objective of Part V, is the subject of study of Chapters 16 and 17.

Chapter 16 is completely devoted to the hardware implementation necessary to achieve the goals of this project. First, an UAV payload outline is provided, in order to contextualize the selection of the hardware. Special attention is paid to the UAV navigation and control sensors; UAV onboard processing; data storage and power supply. A brief overview of the UAV sensing is also introduced, as well as, the question related to the communication support. Table 16.1 summarizes the more common UAV avionics support.

Afterwards, Figure 16.1 depicts the general idea subjacent to this project. It is important to realize that the whole system – PAFA, the PITVAN ANTEX UAV project – is shown. According to that figure, the airborne system is divided in two parts, which are related to the hardware that has been developed to achieve this project's goal (violet box); the rest of the avionics are the hardware existent at the PAFA project. Just the violet box is described. Detailed explanation of each part of the airborne and ground system is addressed. This chapter finalizes with the presentation and description of the project hardware selected.

Chapter 17 complements Chapter 16, in order to give a complete answer to **Q7.** Equally relevant, Chapter 17 describes all the software developed to perform the flight tests. Chapter 17 presents information that is generally forgotten by the research student community. Usually the data logger is not developed by the authors but instead by outsourcing. Given the fact that this was not the case in this research and given the difficulty of such an implementation, it has been considered essential to devote one chapter to this topic.

In this chapter, first, the onboard sensor GUI module, developed for receiving data from several sensors in a concurrent mode, is described (see Figure 17.1).

Following this, the software design architecture is depicted with more specific details related to the developed onboard sensor fusion GUI (see Figure 17.2). Particular attention is, also, paid to the data flow of the software, in order to provide the reader with an idea of how the software interacts with the sensors and other systems.

The developed software can be divided in the following parts: $GNSS_{SR}$; the Piccolo Autopilot SIL, and, the onboard sensor GUI. Additionally, the GUI module is divided in two subsystems: the sensor server, and the sensor data logger subsystems. The sensor server is responsible for saving the collected data while the data logger is responsible for receiving and decoding the data.

The way the Piccolo autopilot SIL was integrated in the onboard sensor module is described, as well as, the IMU data logger. It is important to realize that without this data logger the mission would never have been accomplished.

Regarding **Q7** of this study, Chapters 16 and 17 clearly identify the prominent considerations, related to the platform design, to take into account when working in the field of UAV integrated navigation systems. Given the fact that a lack of information related to the hardware and software implementation has been identified, it was crucial to include, in both chapters of this research, procedure guidelines, to develop all of the hardware and software required for this type of project.

In light of the above, we are able to confirm that the specific objective related to **Q7** has been achieved and, as a consequence **Q7** is satisfactorily addressed.

The eighth research question and last of this research, **Q8**, associated to the last three specific objectives of this thesis, is subject to study at Chapter 18.

The main objective of Chapter 18 is to present the results obtained for the closed loop loosely coupled SINS/$GNSS_{SR}$ system developed at Part IV. To achieve this objective, all the hardware selected and presented at Chapter 16 has been embedded into the van, according to Figure 18.1. It is important to stress that given the constraints associated to flying the UAV in the GATE area and also given the payload restrictions, we had to perform a van test instead of flight tests.

The developed algorithm was tested under a specific scenario and in post processing mode. From the analysis of the performed evaluation, some important conclusions can be drawn. First, the solution provided by the SIMU stand alone is of the same quality as the solution

provided by the Piccolo autopilot. Important to remember is that the MMQ50 raw data was previously denoised, in order to reduce the high frequency noise component (measurement noise component) which cannot be estimated into the EKF.

Second, from the estimated position and velocity figures, and also from the statistical analysis, it is possible to conclude that the developed SIMU/$GNSS_{SR}$ position/velocity solution has the same behavior of the reference position/velocity. Regarding the Piccolo position solution, it can be stated that although it behaves as the reference in latitude and longitude, the same is not true for altitude, where the differences are higher.

Regarding the attitude estimation and according to Figure 18.23, for both roll angle and pitch angle, the solution provided by the SIMU/GPS is satisfactory. However, the same is not true for the Piccolo, which presents big differences from the reference.

Relating to the heading, the results are not good enough. The fact of having a SIMU/$GNSS_{SR}$ integrated algorithm that only depends on the SIMU for heading calculation, using for that low performance gyros, is reflected in the bad results obtained for the heading angle, whose values are exposed at Table 18.3.

Finally and taking into account the several statistics that have been done and presented confronting both solutions (the SIMU/GPS solution and the Piccolo solution), we can conclude that the SIMU/GPS solution is of a better quality than the one provided by the Piccolo. As a consequence, with the proposed algorithm, it is clear that we can guarantee a navigation solution with at least the same accuracy as the one provided by a commercial autopilot navigation solution.

In order to answer the second part of **Q8**: *Does the Galileo data improve the navigation solution?* Figure 18.19 presenting the errors in position for the Galileo E1 provided solution, specially, the altitude errors, shows that the Galileo data quality was not sufficient to be included into the EKF navigation solution.

The **Hypotheses** under investigation in this Part is:

H5: *The navigation solution provided by the developed and implemented SIMU/$GNSS_{SR}$ when confronted with the one provided by the commercial Autopilot produces better results*.

To achieve the proposed specific objectives of Part V, two secondary research questions

were defined: **Q7**, and **Q8**, and **H5** was formulated.

Q7 has been successfully answered. First part of **Q8** has been also successfully addressed. However part of question **Q8** has not been successfully addressed, given the bad quality of the Galileo E1 data in some areas, making it impossible to test the EKF with data from the Galileo measurements.

Bearing this in mind, it is possible to validate H5. As a matter of fact, the implemented solution is of a better quality then the one provided by the Piccolo autopilot.

PART VI

THESIS SUMMARY

And from true lordship it follows that the true God is living, intelligent, and powerful; from the other perfections, that he is supreme, or supremely perfect. He is eternal and infinite, omnipotent and omniscient; that is, he endures from eternity to eternity; and he is present from infinity to infinity; he rules all things, and he knows all things that happen or can happen.

— Sir Isaac Newton

CHAPTER 19

CONCLUSIONS AND RECOMMENDATIONS

19 Thesis Conclusions

Even knowing that the current legislation and standards only allow the UAV to fly in segregated airspace, the UAV range of applications is on the rise. However, the emergence of new UAV applications, not only for the military world but also for the civil one, implies that the UAV must be able to fly, not only in a segregated airspace, but also in an un-segregated airspace. Several and distinct UAV applications are emerging, demanding steps to be taken in order to allow those platforms to operate in an un-segregated airspace. Autonomy is the key for this integration. Subjacent to this autonomy, we have the autopilot, which plays a fundamental role in the UAV state determination and navigation. High level of autonomy in the UAS platforms is the way to guarantee a safe and secure integration in an un-segregated airspace.

However, given the 21st century global economic paradigm, it can be stated that commercial autopilots start to become a non-economic solution for academic research in the field of UAV. In fact, the problem is of balance. Usually, the academic projects are based on inexpensive platforms but on expensive commercial autopilots. Most of the commercial autopilots available on the market are not open source, so users are forced to accept the navigation solution performance provided by the navigation system inside the autopilot. There is a pressing need to overcome this problem, being the solution a navigation system with a compact, light, low power consumption and cheap characteristics.

Nowadays, UAV technology – specially the high availability of low cost, low power consumption, and small size navigation sensors – has matured to the extent that it is widely used, not only by the military, civil and commercial market but also by the Academics world.

After the exposed, we have tried, along this work, to evaluate if it is feasible to develop a low-cost SIMU/GNSS$_{SR}$ integration algorithm for UAV state estimation – using low performance inertial sensors and a GPS/Galileo software receiver. Bearing this in mind, this

research has tried to answer the leading question: *How should an integrated SIMU/*$GNSS_{SR}$ *algorithm, working with low performance inertial navigation sensors (specifically, the Systron Donner MMQ50) and aiming to replace a commercial autopilot navigation solution, be developed and implemented?* This question was broken into eight secondary questions that have support all the thesis.

Regarding question **Q1:** "*What is the purpose of developing a SIMU and is it possible to provide a generic SIMU algorithm, suitable for different categories of IMU sensors, as a complete and comprehensive "recipe" able to be implemented in any computer*?" we concluded that the purpose of developing a SIMU is associated to the fact that the IMU does not provide, at its output, a navigation solution but instead, inertial measurements. In order to convert those data into navigation parameters, position, velocity and attitude, a set of mechanization equations must be developed and implemented. On the other hand, an easy and complete SIMU "recipe" has been identified, developed, implemented and tested, proving to be easily implemented in a computer, for different categories of IMU sensors, with the proviso that the differences that may exist will be on the correction of the IMU raw data (Part II, Chapters 5 and 6).

Regarding question **Q2:** "*Does the available literature provide easy, comprehensive and understandable information about inertial navigation sensor signals and errors? What are the most important properties of the inertial navigation sensor signals*?" we have concluded that the common literature does not provide easy and comprehensibly understandable information about inertial navigation sensors signals and errors. As a consequence, we have provide the reader with a complete and comprehensive description of inertial sensors signals and errors (Part II, Chapter 8).

In order to answer **Q3: "***What is the current state of the art in regard to appropriate methods/techniques, in the field of low performance inertial navigation sensors signals in search for a higher accuracy navigation solution*?" we have identified the more common and appropriate methods/techniques, in the field of low performance inertial navigation sensors, applied in search of a navigation solution, provided by low performance IMU sensors, with higher accuracy. Wavelet de-noising techniques and Allan variance analysis are the current state of the art of methods/techniques applied to smooth raw inertial measurements in order to improve the performance of the strapdown algorithm, and to identify what type of error source is underlying the inertial raw data, by performing certain operations on the entire length of data,

respectively (Part II, Chapter 8).

Concerning **Q4: "***How to deal with low performance inertial navigation sensors in the sensor modelling process?*" Chapter 9 has presented a strategy to be used as an adequate and simple error model feasible to be applied to any IMU belonging to the low/medium grade category of inertial sensors. In fact, low performance IMU sensors of the same category of the MMQ50 are mainly corrupted with high frequency noise, which makes them, to some extent, very similar when trying to model them, with the proviso of some differences related to particular manufacturer specifications of each sensor, which must be taken into account during the modelling process (Part II, Chapter 9).

In respect to **Q5: "***Is it possible to improve the MMQ50 sensor raw data before feeding it to the SIMU mechanization equations?*" we have concluded that the WMRA, applied to the inertial data, reduces part of the wide band noise underlying the sensors. As a consequence, the output signal can be smoothed, highlighting the associated moving bias. Allan variance analysis can be applied to the obtained signal, in order to identify the characteristic in noise in order to be estimated into the Kalman filter. As a consequence, we have an improvement of the MMQ50 raw data prior to using the signal in the SINS developed at Part II.

Regarding the first part of **Q6: "***What are the more common estimation methods used for integrating inertial and GNSS data?***"** in Chapter 12 and Chapter 13, we have presented the common estimation methods and architectures for integrating INS/SIMU and GNSS data. Being for this research considered that the closed-loop integration, loosely coupled scheme, is a more suitable approach for low-cost SIMU/$GNSS_{SR}$ integration in land vehicle applications, due to large error associated to the sensors.

Regarding the second part of **Q6**: "*How does one optimally estimate the interest parameters from noise corrupted data?*" Chapter 14 has presented, with detail, the system (process) and measurement models, using the EKF developed at Chapter 12. Equation (14.7) represents the system model for the ASIMU/$GNSS_{SR}$ algorithm while Equation (14.11) represents the measurement model using the data provided by a $GNSS_{SR}$.

Regarding **Q7: "***What are the most prominent considerations for a platform design to be used in a van test or a flight test environment?*" Chapters 16 and 17 have clearly identified the prominent considerations, related to the platform design, to take into account when working in the field of UAV integrated navigation systems. Given the fact that a lack of information related

to the hardware and software implementation has been identified, it was crucial to include, in both chapters of this research, procedure guidelines to develop all of the hardware and software required for this type of project.

Finally, regarding to the last question of this thesis, **Q8: "***Can we guarantee a navigation solution with at least the same accuracy as the one provided by a commercial autopilot navigation solution? Does the Galileo data improve the navigation solution?*" and taking into account the several statistics that have been obtained and presented, while confronting both solutions, the SIMU/GPS solution and the Piccolo solution, we can conclude that the SIMU/GPS solution is of a better quality than the one provided by the Piccolo. As a consequence, with the proposed algorithm, it is clear that we can guarantee a navigation solution with at least the same accuracy as the one provided by a commercial autopilot navigation solution. Even if the second part of **Q8** was not answered, due to the bad quality of Galileo E1 data, in a definite way.

The main contributions of the Thesis are:

Chapter 2

Presentation of a clear overview on the inertial navigation basic concepts and the differences between an ISA, IMU, and INS as well as other relevant information related to inertial navigation sensors.

Chapter 3

Presentation of the several coordinate frames presenting a summary of all the needed DCM between the several coordinate's frames using along this thesis and the Earth model, used in Part II, as well as some other concepts such as transport rate, radii of curvature.

Chapter 5

The presentation of a single speed strapdown algorithm using DCM. The developed SIMU algorithm has been proposed as a complete recipe able to be directly implemented in computer, and easy to be understandable and implemented even from the point of view of a non-system analyst. Table 5.1 summarizes the inertial navigation equations in several coordinate frames.

Chapter 6

The main contribution of Chapter 6 is the development and presentation of the time rate differential equations describing the propagation of position, velocity and attitude errors in the strapdown navigator implemented. The objective of those equations is to illustrate how errors

propagate in time.

With the SIMU developed and implement, with the error model developed and implemented, we are now able to develop the MMQ50 sensor measurement error model.

Chapter 8

The presentation of a complete and comprehensive description of inertial sensors signals and errors. Figure 8.1, left side, is an illustration of how inertial measurements are composed. As it is possible to observe that they are composed by the true signal plus errors, grouped into deterministic and stochastic components. Different grades of inertial navigation units contain different levels, and types of errors, but in fact, no sensor is immune to any of them. Indeed, all types of inertial sensors exhibit errors such as bias, scale factor, and noise, among others. In order to really understand what in practice those terms are, Figures 8.2 and 8.3 illustrate the classification adopted and used by the author for inertial sensor errors.

Regarding the current state of the art concerning the theory and methodology associated to the improvement of low performance IMU sensors measurements, another main contribution of Chapter 8 is the presentation of the WMRA technique and the Allan variance tool. The first one to be used to minimize the measurement noise presents in the low performance sensors outputs and the second to be used as a tool able to identify the several stochastic errors underlying the inertial raw data.

Chapter 9

The development and presentation of general and simple inertial sensors measurement models in the error a compensation form for low performance IMU sensors, in particular for the MMQ50. Table 9.1 summarizes the accelerometer general error model adaptable to gyros, while Tables 9.2 and 9.3 summarize the adopted measurement model for the accelerometers and gyros expressed in the error form. Based on the adopted model, the corresponding compensation forms for the accelerometer and gyro measurements that will be used to compensate inertial data before going through the strapdown navigator are provided as well.

Chapter 10

The main contribution of Chapter 10 is the development and presentation of a methodology to be applied to low performance inertial navigation sensors in order to improve their measurements before being used in the SIMU mechanization equations. In this chapter, a step by step methodology has been applied to the MMQ50 sensor measurements. Given the fact

that there is no standard mode to deal with the low performance class of inertial sensors, the step by step methodology presented is considered a good way to deal with this category of IMU. It should be noted that the big difference lies in the form how the simple model is achieved from Figure 9.2.

Chapter 12

The presentation of the general equations related to KF and EKF in order to understand how to related the SIMU equations with the one provided the filters.

Chapter 13

Presentation of a general description of the use of a GNSS device as an aiding system that intends to bind the position error. The different ways to implement an integration algorithm to fuse data provided by an SIMU and a GNSS system are presented.

Associated to the fact that a low performance IMU is used, as well as, an autopilot that also uses low performance sensors, and also given the associated simplicity of implementation, the loosely-coupled closed loop selected for this research is presented at Figure 13.2. This figure describes the approach to be implemented in Matlab for the SIMU/ $GNSS_{SR}$ algorithm to achieve the main goal of the thesis to answer the main question that drives this research.

Chapter 14

The development of a loosely coupled close loop SIMU/$GNSS_{SR.}$ In this model, a 15 state EKF is proposed in order to compensate for the gyro and accelerometer bias. During the EKF implementation, it is also assumed that we have small errors over some nominal trajectory, which in the low performance sensors is not always guaranteed.

Chapter 14 presents also the error feedback equations that should be used to compensate both the inertial raw data prior mechanization equations and the SIMU solution.

Considerations about level arm and practical values for the developed EKF are also provided in this Chapter, see Table 14.2.

Chapters 16 and 17

Presentation of the procedure guidelines to develop all of the hardware and software required for the research project, with details of the developed data logger.

19.1 Recommendations and Future Work

Time synchronization is an important requirement that must be taken into account when dealing with multisensory fusion applications, as it is the case of this project. In fact, precise time synchronization has not been considered due to time limitations. This must be overcome because it is, directly, related to achieving a better data fusion performance.

Identification and modelling of the noise terms, underlying the inertial sensor signals, were topics under discussion in Part III of this thesis. An extensive study related to this topic has been done; however. There is, still, a long road ahead with respect to the improvement of the low performance sensors. For instance, as the Allan variance applied to MMQ50 data identified that the time varying bias is also dominant in angular/velocity random and rate/acceleration random walks, in future works, these noise terms should be considered in the system.

Since the collected data in the first 15 minutes were not satisfactory, maybe due to the temperature stabilization, it would be interesting to evaluate the integration of thermal compensation, using an integrated solution.

Taking into account the identified noise underlying any IMU, different types of KF should be developed, including the maximum terms possible, in order to evaluate which ones improve the navigation solution the best.

According to Chapter 17, another IMU sensor was included in the multithread interface module developed: the MEMSense nano IMU (nIMU). Since this sensor provides serial digital outputs of 3D acceleration, 3D rate of turn, and 3D magnetic field data (it has a magnetometer inside), it would be interesting to explore and analyse the improvements that an extra IMU, with magnetometer information, could bring to the solution's accuracy.

Different trajectory generators, capable of being used as a test platform to test and analyse different types of SIMU mechanization algorithms under specific and several scenarios, should be developed. For instance, to evaluate what the effects for SIMU sensors in an UAV Zig-Zag trajectory profile.

The Kalman filter that was implemented to evaluate the performance of the MMQ50 SIMU/GPS integration algorithm was a standard EKF. Therefore, it is considered crucial to investigate other filters, such as the unscented Kalman filter. A comparison of several filters under specific scenarios could be a good way to achieve a more robust navigation filter for each

specific application.

Finally, presentation, as soon as possible, of a step by step instrumentation project, related to the flight test to be realized in the GATE test area with the PAFA UAV. Steps must be taken in order to overcome the restrictions associated to the feasibility of performing an UAV flight test in the GATE area, in order to test the developed algorithm under realistic flight conditions. If these flight tests are done fast, the PAFA UAV will be the first UAV to fly in the GATE.

It is expected that the proposed work will, in the near future, bring higher performance results, even if working with low performance sensors.

BIBLIOGRAPHY

Allan, W D. 1966. Statistics of Atomic Frequency Standards. 1966. Vol. 54, pp. 221-230. 2.

Austin, Reg. 2010. *Unmanned Aircraft Systems: UAVS design, development and deployment.* [ed.] Lan Moir, Allan Seabridge e Roy Langton. United Kingdom : Aerospace Series:John Wiley, 2010.

Barbour, Neil M. 2009. *Inertial Navigation Sensors.* Munich : RTO-EN-SET-116, 2009.

Bekir, Esmat. 2007. *Introduction to Modern Navigation Systems.* Singapore : World Scientific Publishing Co. Pte. Ltd., 2007.

Bento, Maria de. 2008. Unmanned Aerial Vehicles: An Overview. *Inside GNSS.* January/February 2008, pp. 54-61.

Bortz, E. J. 1971. *A New Mathematical Formulation for Strapdown Inertial Navigation.* 1971. IEEE Transactions on Aerospace and Electronics Systems. Vols. AES-7, Nº1, págs. 61-66.

Britting, K. R. 1971. *Inertial Navigation System Analysis.* New York : John Wiley and Sons, 1971.

Bruton, A. M., Schwarz, K. P. and Skaloud, J. 1999. The Use of Wavelets for the Analysis and De-noising of Kinematic Geodetic Measurements. *K.P Schwarz (Ed.) Geodesy Beyond 2000: The Challenges of the First Decade Birmingham.* Berlin Heidelberg : Springer-Verlag, July 19-30, 1999. pp. 227-232.

Demoz, Gebre-Egziabher. 2004. *Design and Performance Analysis of a Low-Cost Aided Dead Reckoning Navigator.* Stanford : A Dissertation submitted to the department of Aeronautics and Astronautics and the comittee on graduate studies of Stanford University, 2004.

DoD. 4 July 1997. *"Department Of Defense World Geodetic System 1984".* s.l. : NIMA TR8350.2, 4 July 1997.

Doitsidis, L., Valavanis, K.P., K.P. and Tsourveloudis, N. 2004. A Framework for Fuzzy Logic Based UAV Navigation and Control. New Orleans, LA April : s.n., 2004.

Dorobantu, R. and Gerlach, C. 2004. *Investigation of a Navigation-Grade RLG SIMU Type iNAV-RQH.* Munich : s.n., 2004. IAPG/FESG (ISSN 1437-8280).

El-Sheimy, Naser y Nassar, Sameh. 2004. Wavelet De-Noising for IMU Alignment. *IEEE A&E SYSTEMS MAGAZINE.* October de 2004.

Farrell, A. J. y Barth, M. 2001. *The Global Positioning System and Inertial Navigation.* New York : McGraw-Hill, 2001.

Gavrilets, V. 2003. *Autonomous Aeorbatic Maneuvring of Miniature.* s.l. : Massachussetts Institute of Technology, 2003. PhD Thesis.

Gelb, A. 1974. *Applied Optimal Estimation.* Fifth Edition. s.l. : MIT Press, 1974.

Godha, S. 2006. *Performance Evaluation of Low Cost MEMS-Based IMU Integrated With GPS for Land Vehicle Navigation Application.* Canada : MSc Thesis, Department of Geomatics Engineering, University of Calgary, 2006. UCGE Report Nº 20239.

Godsill, S.J., Doucet, A. and West, M. 2000. Monte Carlo Smoothing for Nonlinear Time Series. Tokyo, Japan : In Symposium on Frontiers of Time Series Modeling. Institute of Statistical Mathematics, 2000.

Graps, Amara. 1995. An Introduction to Wavelets. *Computing in Science and Engineering.* June de 1995, Vol. 2, págs. 50-61.

Grewal, M, Weill, L y Andrews, A. 2001. *Global Positioning Systems, Inertial Navigation and Integration.* s.l. : Wiley, 2001.

Grewal, Mohinder S. and Andrews, Angus P. 2008. *Kalman Filtering. Theory and Practice Using Matlab.* Third. New Jersey : Wiley, 2008. 978-0-470-17366-4.

Groves, Paul D. 2008. [aut. libro] GNSS Technology and Apllications Series. *Principles of GNSS,Inertial, and Multisensor Integrated Navigation Systems.* 2008, págs. 97-109.

IEEE Std 528. 2001. Standard for Inertial Sensor Terminology. New York : Institute of Electrical and Electronics Engineers, Inc., 2001.

IEEE STD 647. 2006. IEEE standard specification format guide and test procedure for single-axis laser gyros. 2006, págs. 68-80.

IEEE Std. 952. 1997. *IEEE Standard Specification Format Guide and Test Procedure for Single-Axis Interferometric Fiber Optic Gyros.* 1997.

Jaffe, R., y otros. 2004. *Maximizing Bandwith in the Low Cost Miniature MEMS Quartz IMU.* 2004. Joint Navigation Conference. Vol. Session 2A.

Jun, M. and Roumeliotis, S.G. 1999. State estimation via sensor modeling for helicopter control using an indirect Kalman Filter. s.l. : In: IEEE/RSJ International Conference on Intelligent Robots and Systems, 1999.

Langelaan, J. and Rock, S. 2004. Navigation of Small UAVs Operating in Forests. Rhode

Island, Providence : Guidance, Navigation and Control Conference, August 16-19 , 2004.

Lawrence C. Ng. 1993. *On The Application of Allan Variance Method for Ring Laser Gyro Performance Characterization.* IEEE Standards Publications . 1993. 641-1981.

Liu, Fuqiang, et al. 2007. MEMS Gyro's Output Signal De-noising Based on Wavelet Analysis. Harbin, China : Proceedings of the IEEE, 2007.

Mallat, Stephane G. 1989. A theory for Multiresolution Signal Decomposition:The wavelet Representation. 1989. Vol. 1, pp. 674-693.

Maybeck, Peter S. 1982. *Stochastic Models, Estimation, and Control.* New York : Academic Press, Inc., 1982. Vol. 2.

Memense. 2007. Memsense. *Memsense Web Site.* [Online] 2007. [Cited: November 10, 2007.] www.memsense.com.

Memsense. 2008. Nano Inertial Measurement Unit Series Documentation Version 2.11. *Memsense web Site.* [Online] August 2008 http://www.memsense.com/docs/nIMU_Datasheet_v2.11.pdf. DN00010.

Merwe, Van der and Wan. 2004. Sigma-point Kalman filters for integrated navigation. 2004.

Moritz, H. 1980. Geodetic Reference System 1980. [Online] 1980. http://www.gfy.ku.dk/~iag/handbook/geodeti.htm.

Nassar, Sameh. 2005. Accurate INS/DGPS Positioning Using INS Data De-Noising and Autoregressive (AR) Modeling of Inertial Sensor Errors. 2005. Vol. 59, pp. 283-294.

Nikolos, Y., Doitsidis, L. and Christopoulos, V. 2003. Roll Control of Unmanned Aerial Vehicles using Fuzzy Logic, ,. 2003. Vols. vol. 4, no. 2, pp. 1039-1047.

Papoulis, A. 1991. *Probability, Random Variables, and Stochastic Process.* s.l. : McGraw-Hill, 1991. Vol. III.

Prasad, Ramjee and Ruggieri, Marina. 2005. *Applied Satellite Navigation Using GPS,Galileo and Augmentation Systems.* s.l. : Artech House, 2005. 1-58053-814-2.

Russell and Jeffrey, B. 2007. The Myth of the Flat Earth. *American Scientific Affiliation.* 03 14, 2007.

Samrat, Sabat L., et al. 2009. Characterization of Fiber Optics Gyro and Noise Compensation using Discrete Wavelet Transform. s.l. : IEEE, 2009. pp. 909-913.

Savage, Paul G. 2007 *Strapdown Analytics.* Second. s.l. : Strapdown Associates, Inc., Vol. I.

—. 2007. *Strapdown Analytics.* Second. s.l. : Strapdown Associates, Inc., Vol. II.

Schmidt, George T. 2009. *INS/GPS Technology Trends.* s.l. : RTO-EN-SET-116, 2009. NATO RTO Lecture Series,Low-Cost Navigation Sensors and Integration Technology.

Schumacher, C., Sahjendra, N. 2000. Nonlinear control of multiple UAVS in close-coupled formation flight. *AIAA Guidance, Navigation, and Control Conference and Exhibit.* Denver, CO : s.n., August 14-17, 2000.

Schwarz, P. K. and Wei, M. 2006. *INS/GPS Integration for Geodetic Applications.* s.l. : Department of Geomatics Engineering, 2006.

—. 1997. INS/GPS Integration for Geodetic Applications. *Lecture Notes of ENGO 623.* s.l., The University of Calgary : Department of Geomatics Engineering, 1997.

—. 2006. INS/GPS Integration for Geodetic Applications. *Lecture Notes of ENGO 623.* s.l., The University of Calgary : Department of Geomatics Engineering, 2006.

Stockwell, Walter. 2010. Angle Random Walk. *Crossbow.* [En línea] 2010. [Citado el: 8 de January de 2010.] http://www.xbow.com/support.

St-Pierre, M. y Gingras, D. 2004. *Comparison between the unscented Kalman filter and the extended Kalman filter for the position estimation module of an integrated navigation information system.* University of Parma : IEEE Proceedings, Intelligent Vehicles Symposium, 2004.

Taha, R., Noureldin, M. and El-Sheimy, N. 2003. Improving INS/GPS Positioning Accuracy During GPS Outages Using Fuzzy Logic. Portland, Oregon : Proceedings of the 16th annual Technical Meeting of Satellite Devision, Institute of Navigation (ION) GPS-GNSS, 2003. pp. 499-508.

Tehrani, M M. 1983. *Ring Laser Gyro.* 1983. SPIE. Vol. 412, pp. 207-222.

Titterton, David y Weston, John. 2004. *Strapdown Inertial Navigation.* Second. s.l. : IEE Radar, Sonar, Navigation and Avionics Series 17, 2004.

US DoD. 2002. Unmanned Aerial Vehicles Roadmap 2002-2030. *Department of Defense Office of the Secretary of Defense.* [Online] 2002. http://www.acq.osd.mil/usd/uav_roadmap.pdf.

Vaglienti, B. 2007. *Communications for the Piccolo Avionics Version 2.0.4.* s.l. : Cloud Cap Technology, 2007.

Vaglienti, B., et al. 2008. *Piccolo System User's Guide Version 2.1.0.* s.l. : Cloud Cap Technology, 2008.

—. 2006. *Piccolo Avionics External Interface Specifications.* s.l. : Cloud Cap Technology, 2006.

Walker, James S. 1999. *A Primer on Wavelets and their Scientific Applications.* s.l. : Chapman & Hall CRC Press, 1999.

Weinberg, M. and Kourepenis, A. 2006. *Error Sources in In-Plane Silicon Tuning Fork Gyroscopes.* 2006. Vol. 20. 3.

Woodman, Oliver J. 2007. *An Introduction to Inertial Navigation.* Cambridge : University of Cambridge, 2007.

Zhong, S. and Cherkassky, V. 2000. Image De-noising using Wavelet Thresholding and Model Selection. Vancouver, BC, Canada : s.n., 2000.

A. Appendix: Mathematical Preliminaries

This appendix has the purpose of defining the notation used along this thesis, as well as, some of the fundamental mathematical tools which are considered essential in the derivations of some of the equations.

a. Notation

C_B^A	DCM that transforms a vector from its coordinate frame B projection form to its coordinate frame A projection form.
$\underline{\omega}_{AB}$	Angular rate of coordinate frame B relative to coordinate frame A. It should be stressed that if A-Frame is non-rotating frame, corresponds to the angular rate that would be measured by angular rates sensors mounted on frame B.
$(\underline{\omega}_{AB}^B \times)$	Skew symmetric form of $\underline{\omega}_{AB}$ represented by the square matrix given by Equation (A.7).
$\underline{\omega}_{IE}^E$	$= \begin{bmatrix} 0 \\ \omega_{IE} \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 7.292115 \times 10^{-5} \\ 0 \end{bmatrix}$ rad/s Rotation of the E-Frame with respect to the I-Frame.

b. Attitude Representations

Various mathematical representations can be used to define the attitude between two coordinate frames. In this thesis, special emphasis is given to the direction cosine matrix (DCM), the rotation vector associated, and Euler angles.

In this thesis, the attitude propagation is done using DCM and rotation rates; thus this two rate equations are here discussed. The Euler angles are used as a navigation output and so, a brief overview on how to get Euler angles from DCM is presented as well.

DCM

In order to transform an arbitrary vector $\mathrm{r}^{\mathrm{B}} = \begin{bmatrix} \mathrm{X} \\ \mathrm{Y} \\ \mathrm{Z} \end{bmatrix}_{\mathrm{B}}$ from one coordinate frame (B-Frame) to another frame (A-Frame), a transformation matrix capable of relating the two different coordinate frames can be used as follows:

$$r^{\mathrm{A}} = \begin{bmatrix} \mathrm{X} \\ \mathrm{Y} \\ \mathrm{Z} \end{bmatrix}_{\mathrm{A}} = \mathrm{C}_{\mathrm{B}}^{\mathrm{A}} r^{B} = \mathrm{C}_{\mathrm{B}}^{\mathrm{A}} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}_{\mathrm{B}} \tag{A.1}$$

Where $\mathrm{C}_{\mathrm{B}}^{\mathrm{A}}$ represents the transformation matrix between the A-Frame and the B-Frame, which is called a DCM. The columns of this matrix represent the unit vectors in B axes projected along the A axes.

DCM Equalities:

$$\begin{gathered} \mathrm{C}_{\mathrm{A}}^{\mathrm{B}} = (\mathrm{C}_{\mathrm{B}}^{\mathrm{A}})^{\mathrm{T}} \\ \mathrm{C}_{\mathrm{A}}^{\mathrm{B}} (\mathrm{C}_{\mathrm{A}}^{\mathrm{B}})^{\mathrm{T}} = \mathrm{I} \\ \left(\mathrm{C}_{\mathrm{B}}^{\mathrm{A}}\right)^{-1} = (\mathrm{C}_{\mathrm{B}}^{\mathrm{A}})^{\mathrm{T}} = \mathrm{C}_{\mathrm{A}}^{\mathrm{B}} \end{gathered} \tag{A.2}$$

Chain Rule for DCM products:

$$\mathrm{C}_{\mathrm{A}}^{\mathrm{D}} = \mathrm{C}_{\mathrm{B}}^{\mathrm{D}} \mathrm{C}_{\mathrm{C}}^{\mathrm{B}} \mathrm{C}_{\mathrm{A}}^{\mathrm{C}} \tag{A.3}$$

Vector Transformation Characteristics:

$$\begin{gathered} \left(\underline{\mathrm{v}}^{\mathrm{A}} \times\right) = \mathrm{C}_{\mathrm{B}}^{\mathrm{A}} \left(\underline{\mathrm{v}}^{\mathrm{B}} \times\right) (\mathrm{C}_{\mathrm{B}}^{\mathrm{A}})^{\mathrm{T}} \\ \underline{\mathrm{v}}^{\mathrm{A}} \times \underline{\mathrm{w}}^{\mathrm{A}} = \mathrm{C}_{\mathrm{B}}^{\mathrm{A}} \left(\underline{\mathrm{v}}^{\mathrm{B}} \times \underline{\mathrm{w}}^{\mathrm{B}}\right) \end{gathered} \tag{A.4}$$

DCM Rate Equation:

In order to be able to characterize the navigation of a body, its attitude angular orientation with respect to the navigation frame has to be propagated in time. Thus, equations describing the rate of change of the direction cosine matrix have to be derived.

The rate of change of $\mathrm{C}_{\mathrm{B}}^{\mathrm{A}}$ can be expressed mathematically as (Savage, 2007):

$$\dot{\mathrm{C}}_{\mathrm{B}}^{\mathrm{A}} = \mathrm{C}_{\mathrm{B}}^{\mathrm{A}}(\underline{\omega}_{\mathrm{AB}}^{\mathrm{B}} \times) \tag{A.5}$$

Considering the A-Frame being the "Wander Azimuth" N-Frame, and the B-Frame being the Body Frame, Equation (A.5) becomes:

$$\dot{\mathrm{C}}_{\mathrm{B}}^{\mathrm{N}} = \mathrm{C}_{\mathrm{B}}^{\mathrm{N}}(\underline{\omega}_{\mathrm{NB}}^{\mathrm{B}} \times) \tag{A.6}$$

Where $(\underline{\omega}_{\mathrm{NB}}^{\mathrm{B}} \times)$ represents the skew-symmetric matrix for the angular rates of the B-Frame with respect to the N-Frame expressed as follows:

$$\underline{\omega}_{\mathrm{NB}}^{\mathrm{B}} \times = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{A.7}$$

This matrix is derived from the angular rates in the B-Frame with respect to the I-Frame ω_{IB}^{B}, and from the earth angular ratesω_{IN}^{N}, according to Figure A.1.

Therefore, substituting $\omega_{\mathrm{NB}}^{\mathrm{B}}$, form Figure A.1, into Equation (A.6), in terms of the individual inertial angular rotation rates of Frames N and B, we obtain:

$$\dot{\mathrm{C}}_{\mathrm{B}}^{\mathrm{N}} = \mathrm{C}_{\mathrm{B}}^{\mathrm{N}}\left(\underline{\omega}_{\mathrm{IB}}^{\mathrm{B}} \times\right) - (\underline{\omega}_{\mathrm{IN}}^{\mathrm{N}} \times)\mathrm{C}_{\mathrm{B}}^{\mathrm{N}} \tag{A.8}$$

Where:

$(\underline{\omega}_{IB}^{B} \times)$	Skew-symmetric matrix form of $\underline{\omega}_{IB}^{B}$, angular rate of B-Frame relative to I-Frame (as projected on B-Frame axes). Angular rates directly obtained from the IMU gyros.
$(\underline{\omega}_{IN}^{N} \times)$	Skew-symmetric matrix form of $\underline{\omega}_{IN}^{N}$, angular rate of N-Frame relative to I-Frame (as projected on N-Frame axes). Where $\underline{\omega}_{IN}^{N} = \omega_{IE}^{N} + \omega_{EN}^{N}$

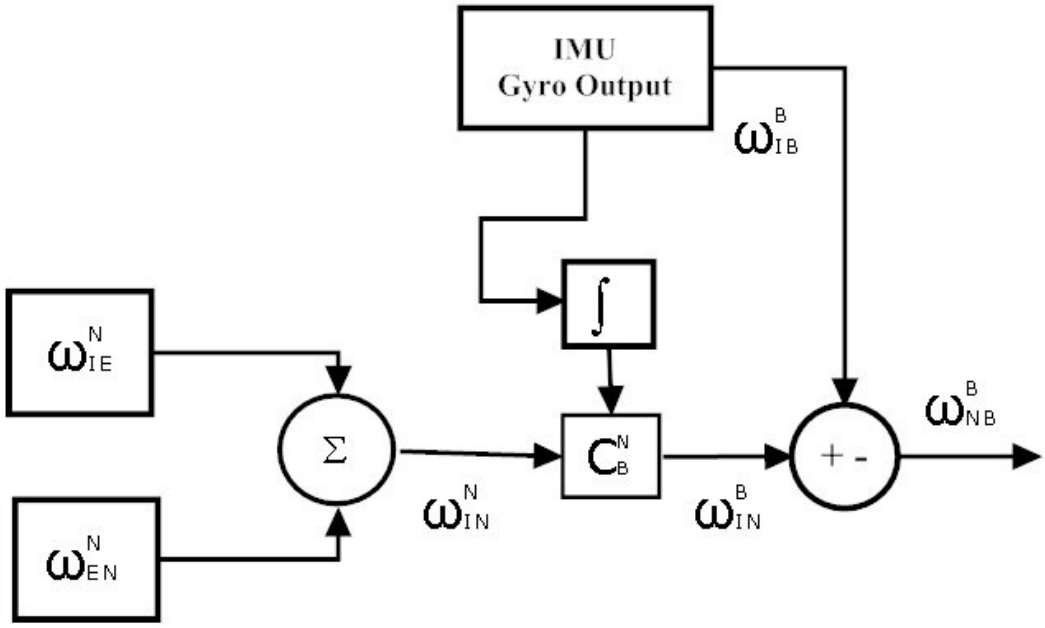

Figure A. 1Angular rates compensation algorithm.

The special particularity of the Equation (A.8) is related to the fact that the coefficient matrix multiplying the angular rates is a DCM; hence, its components can never exceed one magnitude. Therefore, assuming that B and N Frames are chosen to have finite angular rates, this equation has no singularities for any attitude of the B-Frame relative to the A-Frame.

It should be noted that the normality and/or orthogonality of the DCM must be preserved along the developed algorithm. The normalization of the DCM can be performed by normalizing each column vector periodically according to:

$$C_{B}^{N} = \hat{C}_{B}^{N} - \frac{1}{2}\left(\hat{C}_{B}^{N}\hat{C}_{B}^{N^{T}}\right)\hat{C}_{B}^{N} \qquad (A.9)$$

For more information the reader should consult (Savage, 2007).

Rotation Vector

The rotation vector defines an axis of rotation, and magnitude for a rotation about the rotation vector (Savage, 2007). According to Figure 3, the rotation vectorΦ, is directed along the axis of rotation and has a magnitude equal to the rotation angle in radians. It is considered that the B-Frame is initially aligned with the B-Frame, and then rotated to a new attitude around the rotation vector. The B-Fame is now the new attitude of A-Frame. In fact, the application of the rotation vector to B-Frame is to rotate each of the B-Frame coordinate axis unit vectors ($\underline{u}_{iB}$) from A-Frame about $\underline{u}_{\Phi}$ through the angle Φ into the "final" B-Frame. As a consequence, the respective equation for the rotation vector can be defined as:

$$\underline{\Phi} = \begin{bmatrix} \Phi_x \\ \Phi_y \\ \Phi_z \end{bmatrix} = |\Phi| \begin{bmatrix} \cos\alpha \\ \cos\beta \\ \cos\gamma \end{bmatrix} \tag{A.10}$$

Where $|\Phi|$ corresponds to the magnitude of the rotation vector, and the angles $\alpha, \beta, and\ \gamma$ are the angles between the axis of rotation and a specific coordinate frame.

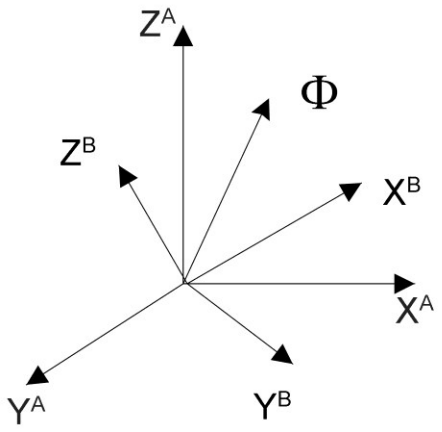

Figure A.2 Rotation vector.

Rotation Vector Rate Equation

According to Bortz[12] (Farrell, et al., 2001), it can be defined that the rotation vector satisfies the following differential equation:

$$\underline{\dot{\Phi}} = \underline{\omega}_{AB}^{B} + \frac{1}{2}\underline{\Phi} \times \underline{\omega}_{AB}^{B} + \frac{1}{|\Phi|^2}\left(1 - \frac{|\Phi|\sin|\Phi|}{2(1-\cos|\Phi|)}\right)\underline{\Phi} \times (\underline{\Phi} \times \underline{\omega}_{AB}^{B}) \quad \text{(A.11)}$$

Where:

$\underline{\dot{\Phi}}$	Rotation vector rate.
$\underline{\omega}_{AB}^{B}$	Angular velocity between the two coordinate frames. In this case rate of rotation of the B-Frame with respect to the A-Frame.

For a small Φ angle, Equation (A.11) it can be rewritten in a simpler form as:

$$\underline{\dot{\Phi}} = \underline{\omega}_{AB}^{B} + \frac{1}{2}\underline{\Phi} \times \underline{\omega}_{AB}^{B} + \frac{1}{12}\left(\underline{\Phi} \times \left(\underline{\Phi} \times \underline{\omega}_{AB}^{B}\right)\right) \quad \text{(A.12)}$$

DCM in Terms of Rotation Vector

According to Savage (2007), the DCM from B-Frame to A-Frame can be written in terms of the rotation vector as:

$$C_B^A = I + \frac{sin\Phi}{\Phi}(\underline{\Phi}\times) + \frac{(1-cos\Phi)}{\Phi^2}(\underline{\Phi}\times)(\underline{\Phi}\times) \quad \text{(A.13)}$$

[12] Given the fact that the above equation was first developed by Bortz (A New Mathematical Formulation for Strapdown Inertial Navigation, 1971), it is quite usual to be referred to as the Bortz equation.

Where:

$$\underline{\Phi}\times = \begin{bmatrix} 0 & -\Phi_z & \Phi_y \\ \Phi_z & 0 & -\Phi_x \\ -\Phi_y & \Phi_x & 0 \end{bmatrix} \tag{A.14}$$

According to Bortz (1971), if Φ is very small, then $\frac{sin\Phi}{\Phi} \approx 1$ and the second-order term can be neglected. Consequently, Equation (A.13) above can be approximated by:

$$C_B^A \approx I + \left(\underline{\Phi}\times\right) \tag{A.15}$$

Euler Angles

In order to define the orientation of a body with respect to the local-level frame, the Euler angles, namely, pitch roll and heading (θ, ø,ψ) are introduced. As a matter of fact, the Euler angle rotation sequence is a classical method for describing the attitude between two coordinate frames.

An Euler angle sequence is a set of sequential rotations of a given coordinate frame around the frame's coordinate axes that positions it at a new attitude after the rotation sequence is completed. The final attitude of the displaced coordinate frame depends on the magnitude and axis of each sequential rotation in the selected Euler sequence. A common Euler angle sequence used to describe the attitude of body axes relative to the L-Frame consists of a heading rotation about the Z^L, followed by a pitch rotation about the displaced Y^L, followed by a roll rotation about the displaced X^L, (Savage, 2007). Thus, the Euler angle sequence: heading, pitch, and roll uniquely define the attitude of the body axes relative to the L-Frame, which results in the following:

Rotation 1 [ψ, about Z]:

$$C_1 = \begin{bmatrix} c\psi & -s\psi & 0 \\ s\psi & c\psi & 0 \\ 1 & 0 & 1 \end{bmatrix} \tag{A.16}$$

Rotation 2 [θ, about displaced Y, but with a change in sign to change the sense]:

$$\mathrm{C}_2 = \begin{bmatrix} c\theta & 0 & \mathrm{s}\theta \\ 0 & 1 & 0 \\ -\mathrm{s}\theta & 0 & c\theta \end{bmatrix} \tag{A.17}$$

Rotation 3 [ϕ, about displaced X]:

$$\mathrm{C}_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\phi & -\mathrm{s}\phi \\ 0 & \mathrm{s}\phi & c\phi \end{bmatrix} \tag{A.18}$$

The resulting DCM from B-Frame to L-Frame transformation matrix is given by:

$$\begin{aligned} \mathrm{C}_{\mathrm{B}}^{\mathrm{L}} = \mathrm{C}_1\mathrm{C}_2\mathrm{C}_3 &= \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix} \\ &= \begin{bmatrix} \mathrm{c}\theta\mathrm{c}\psi & -c\phi\mathrm{s}\psi + \mathrm{s}\phi\mathrm{s}\theta\mathrm{c}\psi & \mathrm{s}\phi\mathrm{s}\psi + \mathrm{c}\phi\mathrm{s}\theta\mathrm{c}\psi \\ \mathrm{c}\theta\mathrm{s}\psi & c\phi\mathrm{c}\psi + \mathrm{s}\phi\mathrm{s}\theta\mathrm{s}\psi & -\mathrm{s}\phi\mathrm{c}\psi + \mathrm{c}\phi\mathrm{s}\theta\mathrm{s}\psi \\ -\mathrm{s}\theta & \mathrm{s}\phi\mathrm{c}\theta & c\phi\mathrm{c}\theta \end{bmatrix} \end{aligned} \tag{A.19}$$

If the DCM from B-Frame to N-Frame is desired, the above matrix has to be multiplied by the $\mathrm{C}_{\mathrm{L}}^{\mathrm{N}}$ as follows:

$$\mathrm{C}_{\mathrm{B}}^{\mathrm{N}} = \mathrm{C}_{\mathrm{L}}^{\mathrm{N}}\mathrm{C}_{\mathrm{B}}^{\mathrm{L}} \tag{A.20}$$

Euler angles in Terms of Direction Cosines.

The DCM Equation (A.19) can be used to obtain expressions for the Euler angles parameters in terms of the elements of the $\mathrm{C}_{\mathrm{B}}^{\mathrm{L}}$ matrix as follows (Savage, 2007):

Pitch:

$$\theta = tan^{-1}\left(\frac{s\theta}{c\theta}\right) = tan^{-1}\left(\frac{-C_{31}}{\sqrt{C_{32}^2 + C_{33}^2}}\right) \tag{A.21}$$

Roll and Platform Heading (Yaw):

When $|C_{31}| \neq 1 \ \wedge \ |C_{31}| < 0.999$:

$$\begin{aligned} \phi &= tan^{-1}\left(\frac{sin\phi}{cos\phi}\right) = tan^{-1}\left(\frac{C_{32}}{C_{33}}\right) \\ \psi &= tan^{-1}\left(\frac{sin\psi}{cos\psi}\right) = tan^{-1}\left(\frac{C_{21}}{C_{11}}\right) \end{aligned} \tag{A.22}$$

When $|C_{31}| \leq -0.999$:

$$\psi - \phi = tan^{-1}\left(\frac{C_{23} - C_{12}}{C_{13} + C_{22}}\right) \tag{A.23}$$

$C_{31} \geq -0.999$:

$$\psi + \phi = \pi + tan^{-1}\left(\frac{C_{23} + C_{12}}{C_{13} - C_{22}}\right) \tag{A.24}$$

c. DCM Between the Several Coordinate Frames

DCM from E-Frame to GEO-Frame (North Pointing: ENU)

Referring to Figure A.3, the transformation matrix from the E-Frame to the GEO-Frame proceeds with the following sequence of two rotations:

Rotation 1 [positive rotation (right-hand sense) of angle L about Y^E]:

$$C_1 = \begin{bmatrix} cL & 0 & -sL \\ 0 & 1 & 0 \\ sL & 0 & cL \end{bmatrix} \tag{A.25}$$

Rotation 2 [negative rotation of angle l about X', but with a change in sign to change the sense:

$$C_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cl & -sl \\ 0 & sl & cl \end{bmatrix} \tag{A.26}$$

Thus, as a result, the DCM from E-Frame to GEO-Frame is given by:

$$C_E^{GEO} = C_2 C_1 = \begin{bmatrix} cL & 0 & -sL \\ -slsL & cl & -slcL \\ clsL & sl & clcL \end{bmatrix} \tag{A.27}$$

Note that s(.) and c(.) stands for sin(.) and cos(.), respectively, and that $(l, L)$are the geodetic latitude and longitude respectively.

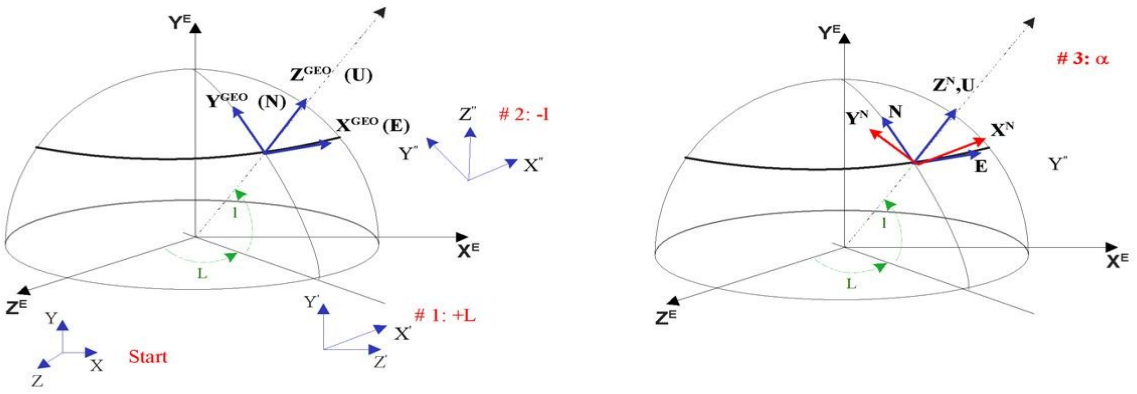

Figure A. 3 E-frame with y axis along earth´s rotation axis and equivalent "Wander Azimuth" N-Frame.

DCM from E-Frame to N-frame (Wander Azimuth)

In order to get the DCM matrix from E-Frame to the "Wander Azimuth" N-Frame, an additional transformation from the GEO-Frame to the N-Frame, through the wander azimuth angle, as shown in Figure A.3 (right side), is needed.

The C_{GEO}^{N} matrix is given by:

Rotation 3 [positive rotation α angle about the $Z^{''}$]:

$$C_{GEO}^{N} = \begin{bmatrix} c\alpha & s\alpha & 0 \\ -s\alpha & c\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \text{(A.28)}$$

Thus using Equation (A.28), we get the correspondent C_E^N matrix:

$$C_E^N = C_{GEO}^N C_E^{GEO} = \begin{bmatrix} D_{11} & D_{12} & D_{13} \\ D_{21} & D_{22} & D_{23} \\ D_{31} & D_{32} & D_{33} \end{bmatrix} = \begin{bmatrix} c\alpha cL - \mathrm{s}L s l s\alpha & s\alpha \mathrm{c}l & -\mathrm{s}L c\alpha - \mathrm{c}L s l s\alpha \\ -s\alpha \mathrm{c}L - s l s L c\alpha & c l c\alpha & -\mathrm{s}L s\alpha - s l c L c\alpha \\ c l s\mathrm{L} & s l & c l c\mathrm{L} \end{bmatrix} \quad \text{(A.29)}$$

In addition, latitude (l), longitude (L), and wander angle α can be extracted from the C_E^N direction cosine matrix as follows:

$$l = \tan^{-1}\left(\frac{D_{32}}{\sqrt{D_{12}^2 + D_{22}^2}}\right); \quad L = \tan^{-1}\left(\frac{D_{31}}{D_{33}}\right)$$
$$\alpha = \tan^{-1}\left(\frac{D_{12}}{D_{22}}\right); \ \psi_{True} = \psi_{Platform} - \alpha \quad \text{(A.30)}$$

Note that the true heading is obtained by subtracting the wander angle from the platform heading (Euler angle obtained).

DCM from E-Frame to L-Frame (North Pointing: NED)

If we want the DCM from E-Frame to the L-Frame, it is just a question of aligning the axes from GEO (ENU) to NED, using the corresponding DCM:

$$C_{(GEO)ENU}^{L\,(NED)} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \quad \text{(A.31)}$$

Thus using Equations (A.27) and (A.31) we get the following C_E^L matrix:

$$C_E^L = C_E^{NED} = C_{GEO(ENU)}^{NED} C_E^{GEO(ENU)} = \begin{bmatrix} -\mathrm{s}L s l & c l & -\mathrm{c}L s l \\ \mathrm{c}L & 0 & -\mathrm{s}L \\ -c l s\mathrm{L} & -s l & -c l c\mathrm{L} \end{bmatrix} \quad \text{(A.32)}$$

DCM from B-Frame to L-Frame (North Pointing: NED)

The transformation matrix $\mathrm{C}_{\mathrm{B}}^{\mathrm{L}}$ has been derived before and is given by Equation (A.19).

DCM from B-Frame to GEO-Frame (North Pointing: ENU)

If the DCM from B to GEO frame is desired, we have to proceed as follow:

$$\mathrm{C}_{\mathrm{B}}^{\mathrm{GEO}} = \mathrm{C}_{\mathrm{L}}^{\mathrm{GEO}}\mathrm{C}_{\mathrm{B}}^{\mathrm{L}} = \begin{bmatrix} \mathrm{c}\theta\mathrm{s}\psi & \mathrm{c}\phi\mathrm{c}\psi + \mathrm{s}\phi\mathrm{s}\theta\mathrm{s}\psi & -\mathrm{s}\phi\mathrm{c}\psi + \mathrm{c}\phi\mathrm{s}\theta\mathrm{s}\psi \\ \mathrm{c}\theta\mathrm{c}\psi & -\mathrm{c}\phi\mathrm{s}\psi + \mathrm{s}\phi\mathrm{s}\theta\mathrm{c}\psi & \mathrm{s}\phi\mathrm{s}\psi + \mathrm{c}\phi\mathrm{s}\theta\mathrm{c}\psi \\ \mathrm{s}\theta & -\mathrm{s}\phi\mathrm{c}\theta & -\mathrm{c}\phi\mathrm{c}\theta \end{bmatrix} \tag{A.33}$$

DCM from B-Frame to N-Frame

For the DCM from B to N frame, we just have to multiply Equation (A.28) with $\mathrm{C}_{\mathrm{B}}^{\mathrm{GEO}}$ in the following way:

$$\begin{aligned} \mathrm{C}_{\mathrm{B}}^{\mathrm{N}} &= \mathrm{C}_{\mathrm{GEO}}^{\mathrm{N}}\mathrm{C}_{\mathrm{B}}^{\mathrm{GEO}} \\ &= \begin{bmatrix} \mathrm{c}\theta\mathrm{s}\psi c\alpha + \mathrm{c}\theta\mathrm{c}\psi s\alpha & \mathrm{c}\phi\mathrm{c}\psi c\alpha + \mathrm{s}\phi\mathrm{s}\theta\mathrm{s}\psi c\alpha - \mathrm{c}\phi\mathrm{s}\psi s\alpha + \mathrm{s}\phi\mathrm{s}\theta\mathrm{c}\psi s\alpha & -\mathrm{s}\phi\mathrm{c}\psi c\alpha + \mathrm{c}\phi\mathrm{s}\theta\mathrm{s}\psi c\alpha + \ \mathrm{s}\phi\mathrm{s}\psi s\alpha + \mathrm{c}\phi\mathrm{s}\theta\mathrm{c}\psi s\alpha \\ -s\alpha\mathrm{c}\theta\mathrm{s}\psi + \mathrm{c}\theta\mathrm{c}\psi c\alpha & -\mathrm{c}\phi\mathrm{c}\psi s\alpha - \mathrm{s}\phi\mathrm{s}\theta\mathrm{s}\psi s\alpha - \mathrm{c}\phi\mathrm{s}\psi c\alpha + \mathrm{s}\phi\mathrm{s}\theta\mathrm{c}\psi c\alpha & \mathrm{s}\phi\mathrm{c}\psi s\alpha - \mathrm{c}\phi\mathrm{s}\theta\mathrm{s}\psi s\alpha + \ \mathrm{s}\phi\mathrm{s}\psi c\alpha + \mathrm{c}\phi\mathrm{s}\theta\mathrm{c}\psi c\alpha \\ \mathrm{s}\theta & -\mathrm{s}\phi\mathrm{c}\theta & -\mathrm{c}\phi\mathrm{c}\theta \end{bmatrix} \end{aligned} \tag{A.34}$$

Note that as expected, if the wander angle is set to zero, Equation (A.34) is the same as (A.33).

d. Angular Rates Between I-frame and Computational Frames

From Equation (A.5) we get the general equation for the angular rate of frame B with respect to frame A, in a skew-symmetric form:

$$\left(\underline{\omega}_{AB}^{B}\times\right)=C_{A}^{B}\dot{C}_{B}^{A} \tag{A.35}$$

$\underline{\boldsymbol{\omega}}_{IL}^{L}/\underline{\boldsymbol{\omega}}_{INED}^{NED}$

The angular rate $\underline{\omega}_{IL}^{L}$ can be obtained applying the rule of vector addition to the angular velocity vector. Thus the angular rate associated to the rotation of the L-Frame about the I-Frame with its components in the L-Frame is given by:

$$\underline{\omega}_{IL}^{L}=\omega_{IE}^{L}+\omega_{EL}^{L} \tag{A.36}$$

Where $\underline{\omega}_{IE}^{L}$ is obtained in the following manner:

$$\underline{\omega}_{IE}^{L}=C_{E}^{L}\underline{\omega}_{IE}^{E} \tag{A.37}$$

Using $\underline{\omega}_{IE}^{E}$ (Notation), and the matrix C_{E}^{L}, we obtain:

$$\underline{\omega}_{IE}^{L}=\begin{bmatrix} cl \\ 0 \\ -sl \end{bmatrix}\omega_{IE} \tag{A.38}$$

Using (A.38) and (A.51) into (A.36), we obtain:

$$\underline{\omega}_{IL}^{L}=\begin{bmatrix} cl(\omega_{IE}+\dot{L}) \\ -\dot{l} \\ -sl(\omega_{IE}+\dot{L}) \end{bmatrix} \tag{A.39}$$

Proceeding in a similar way for the other reference frame, we will obtain the several angular rates that relate the I-Frame with GEO and N frames.

$\boldsymbol{\underline{\omega}_{IGEO}^{GEO}/\underline{\omega}_{IENU}^{ENU}}$

$$\underline{\omega}_{IGEO}^{GEO} = \underline{\omega}_{IE}^{GEO} + \underline{\omega}_{EGEO}^{GEO} \tag{A.40}$$

Where $\underline{\omega}_{EGEO}^{GEO}$ is obtained from (A.48), and $\underline{\omega}_{IE}^{GEO}$ is obtaining in the following manner:

$$\underline{\omega}_{IE}^{GEO} = C_{E}^{GEO}\underline{\omega}_{IE}^{E} \tag{A.41}$$

Using $\underline{\omega}_{IE}^{E}$ (Notation), and the matrix C_{E}^{GEO} given by (A.27), we obtain:

$$\underline{\omega}_{IE}^{GEO} = \begin{bmatrix} 0 \\ cl \\ sl \end{bmatrix} \omega_{IE} \tag{A.42}$$

Using (A.42) and (A.48) into (A.40), we obtain:

$$\underline{\omega}_{IGEO}^{GEO} = \begin{bmatrix} -\dot{l} \\ cl(\omega_e + \dot{L}) \\ sl(\omega_e + \dot{L}) \end{bmatrix} \tag{A.43}$$

$\boldsymbol{\underline{\omega}_{IN}^{N}}$

$$\underline{\omega}_{IN}^{N} = \underline{\omega}_{IE}^{N} + \underline{\omega}_{EN}^{N} \tag{A.44}$$

Where $\underline{\omega}_{EN}^{N}$ is obtained from (A.54), and $\underline{\omega}_{IE}^{N}$ is obtaining in the following manner:

$$\underline{\omega}_{IE}^{GEO} = C_{E}^{N}\underline{\omega}_{IE}^{E} \tag{A.45}$$

Using $\underline{\omega}_{IE}^{E}$ (Notation), and the matrix C_{E}^{N} given by (A.29), we obtain:

$$\underline{\omega}_{IE}^{N} = \begin{bmatrix} s\alpha cl \\ clc\alpha \\ sl \end{bmatrix} \omega_{IE} \tag{A.46}$$

Using (A.46) and (A.54) into (A.44), we obtain:

$$\underline{\omega}_{\mathrm{IN}}^{\mathrm{N}} = \begin{bmatrix} -\dot{l} + cls\alpha\omega_{\mathrm{IE}} \\ \dot{\mathrm{L}}cl + clc\alpha\omega_{\mathrm{IE}} \\ sl \end{bmatrix} \tag{A.47}$$

e. Angular Rates Between E-frame and Computational Frames

$\underline{\boldsymbol{\omega}}_{\mathbf{EGEO}}^{\mathbf{GEO}}/\underline{\boldsymbol{\omega}}_{\mathbf{EENU}}^{\mathbf{ENU}}$

Thus, the angular rate $\omega_{\mathrm{EGEO}}^{\mathrm{GEO}}$ in the skew-symmetric form can be derived as follow:

$$(\underline{\omega}_{\mathrm{EGEO}}^{\mathrm{GEO}} \times) = \mathrm{C}_{\mathrm{E}}^{\mathrm{GEO}}\dot{\mathrm{C}}_{\mathrm{GEO}}^{\mathrm{E}} = \begin{bmatrix} 0 & -\dot{\mathrm{L}}sl & \dot{\mathrm{L}}cl \\ \dot{\mathrm{L}}sl & 0 & \dot{l} \\ -\dot{\mathrm{L}}cl & -\dot{l} & 0 \end{bmatrix} \tag{A.48}$$

Therefore:

$$\underline{\omega}_{\mathrm{EGEO}}^{\mathrm{GEO}} = \begin{bmatrix} -\dot{l} \\ \dot{\mathrm{L}}cl \\ \dot{\mathrm{L}}sl \end{bmatrix} \tag{A.49}$$

Thus the angular rate vector using relation (A.7) is given by:

$$\underline{\omega}_{\mathrm{EGEO}}^{\mathrm{E}} = \mathrm{C}_{\mathrm{GEO}}^{\mathrm{E}}\underline{\omega}_{\mathrm{EGEO}}^{\mathrm{GEO}} = \begin{bmatrix} -\dot{\mathrm{I}}\mathrm{cL} \\ \dot{\mathrm{L}} \\ \dot{\mathrm{I}}\mathrm{sL} \end{bmatrix} \tag{A.50}$$

As a result, the respective angular rate ω_{EGEO}^{E} can be calculated as follows:

$\boldsymbol{\underline{\omega}_{EL}^{L}}$

The angular rate vector in the skew-symmetric form associated to both reference frames w_{EL}^{L} is therefore, using (A.33):

$$(\omega_{EL}^{L}\times) = C_{E}^{L}\dot{C}_{L}^{E} = \begin{bmatrix} 0 & \dot{L}sl & -\dot{l} \\ -\dot{L}sl & 0 & -\dot{L}cl \\ \dot{l} & \dot{L}cl & 0 \end{bmatrix} \tag{A.51}$$

This corresponds to the follow angular rate vector:

$$\underline{\omega}_{EL}^{L} = \begin{bmatrix} \dot{L}cl \\ -\dot{l} \\ -\dot{L}sl \end{bmatrix} \tag{A.52}$$

$\boldsymbol{\underline{\omega}_{EN}^{N}}$

The angular rate vector in the skew-symmetric form associated to both reference frames $\underline{\omega}_{EN}^{N}$ is therefore, using (A.29):

$$(\omega_{EN}^{N}\times) = C_{E}^{N}\dot{C}_{N}^{E} = \begin{bmatrix} 0 & 0 & \dot{L}cl \\ 0 & 0 & \dot{l} \\ -\dot{L}cl & -\dot{l} & 0 \end{bmatrix} \tag{A.53}$$

$$\underline{\omega}_{EN}^{N} = \begin{bmatrix} -\dot{l} \\ \dot{L}cl \\ 0 \end{bmatrix} \tag{A.54}$$