ALMA MATER STUDIORUM - UNIVERSITY OF BOLOGNA

SCHOOL OF ENGINEERING AND ARCHITECTURE

DEPARTMENT OF ELECTRICAL, ELECTRONIC AND INFORMATION ENGINEERING "GUGLIELMO MARCONI" - DEI

MASTER DEGREE IN AUTOMATION ENGINEERING

MASTER THESIS
in
Diagnosis and Control

Solutions and algorithms for inertial navigation of railroad vehicles

CANDIDATE
Federico Bacilieri

SUPERVISOR
Prof. Ing. Andrea Tilli

CO-SUPERVISORS
Ing. Alessandro Bosso
Dr. Ing. Christian Conficoni
Dr. Ing. Nicola Mimmo

Academic Year 2016/2017

2nd Session

Abstract

Obiettivo di questa tesi è lo studio e lo sviluppo di soluzioni innovative di navigazione inerziale per applicazioni ferroviarie. Questo strumento risulta infatti utile per il tracciamento del moto durante l'assenza prolungata di sistemi di localizzazione esterni, tipo GPS, come può avvenire in lunghe gallerie. Una volta definite le equazioni fondamentali di meccanizzazione e gli strumenti per la rappresentazione dell'assetto e la costruzione di osservatori di stato, è stata eseguita un'analisi dello stato dell'arte e dei dispositivi disponibili sul mercato, al fine di mettere in evidenza le tecniche attuali per risolvere questo genere di problemi, nonchè le prestazioni dei sistemi già esistenti, in particolare per ciò che concerne le loro prestazioni durante periodi di assenza del GPS. Sono inoltre presentati e caratterizzati i sensori e le misure disponibili su questi veicoli: accelerometri e giroscopi MEMS, GPS e odometria.

Partendo da queste informazioni, sono state proposte varie soluzioni al problema di stima di posizione, velocità e assetto del veicolo con l'obiettivo di valutare quali prestazioni si possono ottenere durante fasi di totale assenza di segnale GPS, con l'ausilio o meno delle mappe del tracciato da percorrere e/o in presenza di non idealità dei sensori come bias o rumore di misura.

Dopo una prima versione basata su un singolo EKF standard, si è scelto di sviluppare una seconda serie di soluzioni separando il problema di stima in ricostruzione di assetto (AHRS) e stima di posizione/velocità, risolti mediante due algoritmi distinti, in modo da sfruttare le informazioni dei sensori in maniera più strutturale e ridurre la complessità. È stato implementato un AHRS basato su filtro di Kalman esteso e uno mediante un osservatore non lineare sul gruppo $SO(3)$, adattato per riceve le misure disponibili a intervalli discreti; inoltre, sono stati sviluppati un EKF di ordine completo e uno ridotto per le dinamiche di traslazione. Successivamente alla loro formulazione, è stata sviluppata una soluzione per l'integrazione dei dati delle mappe negli algoritmi, in modo da fornire correzioni anche in mancanza di GPS e ridurne così la deriva, mantenendo al contempo un ridotto carico computazionale e facilità di integrazione con tutti gli algoritmi sviluppati.

Si è infine proceduto implementando e simulando la soluzione a singolo stadio e le varie combinazioni di INS a due stadi in ambiente Matlab-Simulink, per un confronto e per la verifica delle loro prestazioni in presenza o meno delle correzioni da GPS e/o mappe, nonchè con misure ideali e rumorose e incertezze sulla traiettoria riportata nella mappa, la cui introduzione

ha introdotto sensibili miglioramenti nelle fasi di assenza del GPS. Gli algoritmi a due stadi hanno mostrato prestazioni migliori rispetto alla struttura a EKF singolo la quale presenta, a livello di prestazioni negli scenari di riferimento considerati, un dominio di convergenza troppo limitato per fini pratici.

A conclusione del lavoro, il quale è stato svolto avvalendosi anche della collaborazione di Sadel, sono state gettate delle basi per una successiva analisi rigorosa dell'interconnessione negli INS a due stadi, il cui obiettivo sarà verificare o meno se tale struttura consente la convergenza anche dei bias di accelerometro al fine di ottenere la migliore inizializzazione possibile dei filtri per la fase di *dead reckoning*.

Contents

List of Figures

List of Tables

Introduction

Development of inertial navigation algorithms is nowadays favoured by the technology progress on the manufacturing of *Micro Electro-Mechanical Systems* (MEMS) devices, which allows building miniaturised systems for a wide range of applications, including human motion capture. The integration of data coming from proprioceptive sensors of different nature (accelerometers, magnetometers, gyroscopes and so on) allows for attitude, speed and position reconstruction starting from a known initial condition.

Drift-related errors are inevitable especially in noisy environments and increase with time, therefore periodic updates should be performed from an external source, such as the *Global Positioning System* (GPS), which is possibly not always available, inaccurate and provide data at a limited rate. Inertial navigation systems must then perform the best possible accuracy in order to limit positioning, velocity and orientation error between updates.

The purpose of this work is to apply the techniques of inertial and satellite-aided navigation to the trains sector by adapting them to the unique features of this class of vehicles in order to develop an innovative solution. In particular, it is of interest to evaluate their dead reckoning performance. Another objective is to integrate map knowledge into standard navigation algorithm and to understand how it can help in limiting the intrinsic problem of estimation drift. The thesis has been developed in collaboration with Sadel, an italian company working in the field of train systems for infotainment.

The work is structured as follows. A general overview of the problem of inertial navigation, its basic principles and issues is found in Chapter 1; then, the mathematical fundamentals are given in Chapter 2 and the problem is formulated for the non-inertial framework. Several relevant attitude representations are also presented. Chapter 3 outlines the sensor models considered in this work and their uncertainties, in particular for what concerns MEMS accelerometers and gyroscopes, GPS and odometry.

Chapter 4 presents the Kalman Filter, which is the standard tool to address such problems. Nevertheless, its most popular variations have been outlined, such as *Extended Kalman Filter* (EKF), *Unscented Kalman Filter*

(UKF) and *Invariant EKF* (IEKF). Their main properties and limitations have been described. Chapter 5 points out the state of the art concerning inertial navigation algorithms, as well as the currently-available commercial devices,their generic field of application and overall performance. Moreover, the general approach followed for the design of an innovative algorithm is outlined.

Starting from the solutions found in literature, several models were studied in order to exploit the kinematic constraints of the train and all the available measurements in order to get the best possible estimate by separating the problem of attitude estimation and the navigation algorithm itself and compensate for sensors biases, which are an additional sources of uncertainty.

A single-EKF algorithm is presented in Chapter 6, while several two-stage solutions have been developed in Chapter 7 in order to separate the problem of attitude estimation from position and velocity and to overcome some limitations of the single-stage algorithm.

Moreover, the topic of map-aided navigation was addressed to improve navigation accuracy during GNSS outages. An algorithm was developed in order to exploit the additional information. Chapter 8 describes the proposed solution for map integration.

All the algorithms have been implemented in Matlab-Simulink according to the functional schemes presented in Chapter 9. Simulation results from the different model are depicted in Chapter 10. Ideal and uncertain maps have been used in the simulations as well as ideal and noisy readings to test different working conditions. An overview of the problem of accelerometer bias estimation is depicted in the same chapter and an insight is given about analysis techniques to understand if convergence is possible within a two-stage INS framework. Eventually, conclusions are presented in Chapter 11.

Chapter 1

Inertial Navigation: an overview

The problem of navigation has gained great importance nowadays as increased precision of movements is required in transports and many other engineering fields. This increase is also fed by the small-size MEMS technology and the increased computing performance achieved by embedded devices. *Inertial navigation* can be regarded as a class of sub-problems in which some important constraints are introduced.

First of all, inertial navigation may be defined as "a self-contained navigation technique"[25]. This means that it does not rely on external aiding signal sources to work unlike GPS or instrumental navigation systems for aircraft. Instead, proprioceptive sensors are used to sense motion and angular rates and determine velocity, position and heading. This technique is widely used in aircraft, maritime and land vehicle applications.

Inertial navigation solutions can improve motion estimation accuracy and generally provides information at a higher rate with respect to some external positioning systems. As a matter of fact, these systems may experience some periods of outage during operation, be inaccurate or provide information at an insufficient rate. It is then critical to perform the best possible estimation and track the motion throughout the whole time span in order to continuously provide data. When inertial navigation devices use external signals to periodically correct their estimates they are also said to be performing *aided inertial navigation*[10].

1.1 Working principle

The main principle of inertial navigation is the well-known Newton's Second Law of motion: from the knowledge of the linear accelerations and the rotation rates of the body as well as of its initial kinematic state, it is possible to reconstruct the translational and rotational speed, and therefore the position at a given time. It is essential to determine the orientation of the body in order to integrate the measured accelerations on the right direction, since the sensing equipments are usually solidal to the vehicle in modern applications. Moreover, knowing the attitude is essential in order to discriminate between the true acceleration and the gravity term. As a matter of fact, any accelerometer measures the so-called *specific force*, that is the difference between true and gravity acceleration [10].

Therefore, the main tasks of an *Inertial Navigation System* (INS) are the following [25]:

- Measure angular motion and specific forces and compensate for sensor errors
- Properly integrate the angular rates to get the relative orientation of the body frame with respect to a given reference frame
- Resolve force into the navigation frame and compensate for the gravity term
- Integrate the so-obtained true accelerations to get body speed and position

Almost all inertial navigation systems can be divided among two main categories [10, Chapter 11]. The main difference between the two is the reference frame in which measurements are taken[1]:

Stable Platform Systems (also known as *mechanised INS* or SPS) were the first to be manufactured and present a rotating sensors platform which is kept in a constant orientation coincident to a frame of reference throughout manoeuvring (from which the attribute *stable* comes out). This is made possible by a set of gyroscopes and torque motors that compensate the rotation; high rotational precision is required in order to minimize errors. Hence, accelerometer measurements are always in the reference frame and position can be estimated by simple double integration of data, while attitude is inferred by the rotation angles between vehicle and platform. In order to keep the

[1] Further details about the different reference frames are presented in Chapter 2

right orientation, accurate calibration of the gyroscopes is required in order to compensate their non-idealities, biases and axes non-orthogonality being the most important ones.

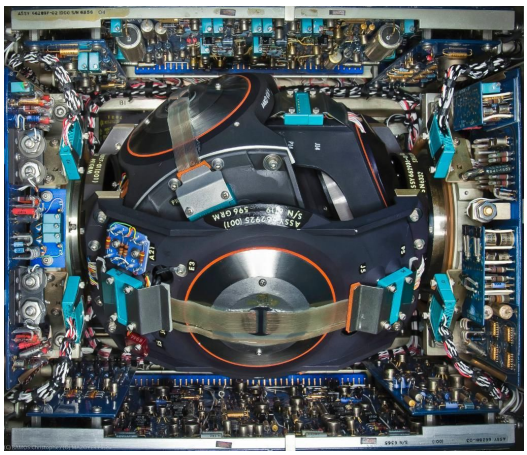

Figure 1.1: Stable-platform IMU of a Concorde aircraft

Strap-down Systems are rigidly attached to the vehicle, in opposition to SPS devices. Therefore, accelerations are measured in the vehicle frame and must be transformed to the proper reference frame prior double integration. Gyroscopes are again used to measure the angular rate and therefore reconstruct the orientation via proper integration. This type of INS is generally cheaper and smaller than stable platform systems and is the most used one nowadays. They have no mechanical complexity at the cost of more computational effort which is on the other hand much more affordable nowadays than at the dawn of navigation era.

AHRS

An *Attitude and Heading Reference System*, or AHRS, is an algorithm estimating the relative orientation of the body with respect to some fixed frame during manoeuvring. In some sense, AHRS algorithm solve a sub-problem of inertial navigation: in fact, once the rotation has been reconstructed, integration of measured accelerations can be performed to determine the vehicle position.

AHRS algorithms can be decoupled from the speed/position estimation problem. This "separation" approach can be useful to reduce complexity of the overall scheme with respect to a single-stage sensor fusion algorithm. Gyroscopes are used to determine body rotation and detect high-frequency

attitude changes, while accelerometers and magnetometers can be used as measures of the local gravity vector and the magnetic North as seen by the vehicle, thus becoming more important during slow changes of the orientation. Since their direction is a priori known in the navigation frame of reference, it is possible to periodically correct the estimated orientation.

1.2 Fields of application

Inertial navigation techniques were first developed for rocket guidance, the first experiments being carried out during the period of World War II. Inertial guidance was further developed also for human space flight and then extended also to land, air and sea navigation as an aid to pilots also during failure of conventional pre-GPS systems (such as VORs/NDBs for air navigation). Nowadays, IMUs are found also on unmanned, autonomous vehicles. Nevertheless, robotics is another important field to apply these techniques. In general, INS can be used to increase the positioning accuracy whenever external systems (such as GPS, infra-red camera devices and so on) output data at a insufficient rate or accuracy. Human and animal motion can be captured thanks to sensors' miniaturization thanks to MEMS technology and smaller-scale computing units. This allows athletes to collect statistics about their performance. Animal motion can be captured with the same techniques, too, in order to track their movements and for other applications.

1.3 Issues of Inertial Navigation

Several issues afflict both types inertial navigation algorithms, the main being estimation drift. Small errors in acceleration bring progressively increasing errors in the estimated speed, and therefore even larger position errors. As a matter of fact, integrators (as well as cascaded integrators) are simply stable systems; moreover, the estimation error is unbounded and grows with time. Similarly, small rates in the angular rate are integrated and eventually lead to large errors in the attitude estimation and thus in the speed and position because the measurements do not undergo the right coordinate transformation [10].

Sensors are the main cause of drifting because of their non idealities. The magnitude of noise varies a lot according to their technology and is related to white noise, thermal noise, measurement bias, external disturbances and so on. Other sources of inaccuracies are model and numerical approximations.

Another disadvantage of stable-platform systems is the mechanical com-

plexity of the moving frame, while strap-down systems generally require more computations to resolve all estimates.

In order to reduce drift and improve accuracy, periodic updates of the algorithm estimated state are performed. A possibility is to use external sources to periodically correct the estimation done by the algorithm. GNSS or other localisation technologies (also based on ground infrastructures or computer vision) can be used. The rate of updates should be sufficiently high to keep the estimation error bounded to acceptable levels.

Chapter 2

Rigid Body kinematics

In this chapter the fundamental kinematic equations to describe rigid body motion are presented in a general fashion and then adapted to the specific framework of Earth-based inertial navigation. Moreover, some relevant reference frames for the current application are described.

2.1 Attitude representation and rotation dynamics

In this section some common and convenient ways to represent rotations are presented, together with the formulation of their derivative, which is useful for implementation on an algorithm.

Rotation matrices

Rotation matrices are the standard tool used to change frame of representation of vector quantities. Given any two reference frames a and b and a generic vector v, it is possible to switch from one to another via the equations

$$v^b = R_{ba} v^a \tag{2.1}$$

$$v^a = R_{ab} v^b = R_{ba}^T v^b \tag{2.2}$$

where the notation R_{ba} is referred to the rotation matrix defining the relative orientation of frame b with respect to frame a. All rotation matrices belong to the special orthogonal group of dimension 3 $SO(3)$, which contains all 3-by-3 orthogonal matrices with positive unitary determinant.

Only 3 out of the 9 parameters of a rotation matrix are independent Such conclusion comes trivially from the constraints on the Cartesian axes of

the new reference system being orthogonal and of unitary norm. Therefore, there are several, less redundant ways to express relative orientation between frames. Each method comes with its advantages and limitations.

Concerning the derivative of R_{ba}, let now ω_{ba}^a the relative rotation of b with respect to a as seen in a frame. Then it can be shown that

$$\dot{R_{ba}} = R_{ba}(\omega_{ba}^a)_\times = (\omega_{ba}^b)_\times R_{ba} \tag{2.3}$$

where $(\cdot)_\times$ denotes the skew operator used for computing the vector product.

Euler Angles

Another common possibility is to use three consecutive rotations in order to have a minimal representation of the relative orientation. Rotations are performed about predefined axes which can belong to one or more different frames. The choice of axes and their order influences the final form of the resulting dynamic equations.

Let the transformation from frame a to b be defined by the following rotations sequence:

1. Rotation of ψ about z^a which defines the reference frame a'
2. Rotation of θ about $y^{a'}$ defining the new reference frame a''
3. Rotation of φ about $x^{a''}$, which eventually defines frame b

Such sequence of rotations is known as the *Roll, Pitch and Yaw* angles [10]. The corresponding rotation matrix is

$$R_{ba} = \begin{pmatrix} C_\theta C_\psi & C_\theta S_\psi & -S_\theta \\ -S_\psi C_\varphi + C_\psi S_\theta S_\varphi & C_\psi C_\varphi + S_\psi C_\theta S_\varphi & C_\theta S_\varphi \\ S_\psi S_\varphi + C_\psi S_\theta C_\varphi & -C_\psi S_\varphi + S_\psi S_\theta C_\varphi & C_\theta C_\varphi \end{pmatrix} \tag{2.4}$$

where $C_\theta = \cos(\theta)$ and $S_\theta = \sin(\theta)$. The derivative of the RPY angles turns out to be

$$\begin{pmatrix} 1 & 0 & -\sin(\theta) \\ 0 & \cos(\varphi) & \cos(\theta)\sin(\varphi) \\ 0 & -\sin(\varphi) & \cos(\theta)\cos(\varphi) \end{pmatrix} \begin{pmatrix} \dot{\varphi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \omega_{ba}^a \tag{2.5}$$

If $\theta \neq \pm\frac{\pi}{2}$, the matrix is invertible and the equation can be rewritten as

$$\begin{pmatrix} \dot{\varphi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \begin{pmatrix} 1 & \sin(\varphi)\tan(\theta) & \cos(\varphi)\tan(\theta) \\ 0 & \cos(\varphi) & -\sin(\varphi) \\ 0 & \sin(\varphi)\sec(\theta) & \cos(\varphi)\sec(\theta) \end{pmatrix} \omega_{ba}^a = E(\varphi,\theta)\omega_{ba}^a \tag{2.6}$$

It has to be noticed that E becomes very large when $\theta \approx \pm\frac{\pi}{2}$, thus leading to numerical errors during integration. This representation should be avoided if the body motion may become very close to this situation. As a matter of fact, that value corresponds to the singularity of the representation, meaning the problem of finding the three angles starting from the rotation matrix is undetermined. There are infinite solutions to the problem because the first and last rotation are performed about the same body axis.

Every type of Euler Angles representation has some singularities that depend on the chosen axes and angles but are always present. This occurs the "shape" of manifold $SO(3)$ does not exactly match that of a sphere and it turns out that it is not possible to define a global diffeomorphism with $\mathbb{R}^3$ because angles whose difference is multiple of 2π define the same rotation. Hence only local diffeomorphisms between the two can be defined.

Another popular Euler Angles representation is the Z-Y-Z transformation, consisting in three rotations about z_0, y_1 and z_2, which therefore happen about different axes of different reference systems. This latter representation is indeed singular when $\theta = 0$ or $\theta = \pi$. In fact, is easy to figure out that the two z axes are parallel in case of singularity, meaning that there are infinite combinations of φ and ψ whose sum is the actual rotation in space.

Quaternions

Quaternions are mathematical objects defined as the union of a scalar and a vector, they can be also regarded as a 4-dimension complex numbers [10, Appendix D]. Sum of quaternions if trivially defined as

$$a + b = \begin{pmatrix} a_1 + b_1 \\ a_2 + b_2 \\ a_3 + b_3 \\ a_4 + b_4 \end{pmatrix} \tag{2.7}$$

while one possible definition of quaternion product is

$$a \circ b = \begin{pmatrix} a_1 & -a_2 & -a_3 & -a_4 \\ a_2 & a_1 & -a_4 & a_3 \\ a_3 & a_4 & a_1 & -a_2 \\ a_4 & -a_2 & a_2 & a_1 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{pmatrix} \tag{2.8}$$

Given two generic frames a and b, it is always possible to describe the transformation from a to b by means of a rotation of some θ degrees about some vector $\boldsymbol{w}$ of unitary norm. Such results is known as the Euler's theorem.

The corresponding quaternion is defined as

$$q_{ba}(\theta, \boldsymbol{w}) = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) \\ \boldsymbol{w} \sin\left(\frac{\theta}{2}\right) \end{pmatrix} = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) \\ \boldsymbol{w}_x \sin\left(\frac{\theta}{2}\right) \\ \boldsymbol{w}_y \sin\left(\frac{\theta}{2}\right) \\ \boldsymbol{w}_z \sin\left(\frac{\theta}{2}\right) \end{pmatrix} \tag{2.9}$$

It can be also easily noticed that $q_{ba}(\theta, \boldsymbol{w})$ and $q_{ba}(-\theta, -\boldsymbol{w})$ represent the same rotation in space. Therefore, the mapping between the set of unitary quaternions and $SO(3)$ is two to one.

Given $q_{ba} = \begin{pmatrix} q_1 & q_2 & q_3 & q_4 \end{pmatrix}^T$, the corresponding rotation matrix is expressed as

$$R_{ba} = \begin{pmatrix} q_1^2 + q_2^2 - q_3^2 - q_4^2 & 2(q_2 q_3 - q_1 q_4) & 2(q_1 q_3 + q_2 q_4) \\ 2(q_2 q_3 + q_1 q_4) & q_1^2 - q_2^2 + q_3^2 - q_4^2 & 2(q_3 q_4 - q_1 q_2) \\ 2(q_2 q_4 - q_1 q_3) & 2(q_1 q_2 + q_3 q_4) & q_1^2 - q_2^2 - q_3^2 + q_4^2 \end{pmatrix} \tag{2.10}$$

Given the angular rate $\omega_{ba}^b = \omega$, the expression of the quaternion derivative for the implementation turns out to be [10, Appendix D]

$$\dot{q}_{ba} = \frac{1}{2} \begin{pmatrix} 0 \\ \omega \end{pmatrix} \circ q_{ba} = \frac{1}{2} \begin{pmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & -\omega_z & \omega_y \\ \omega_y & \omega_z & 0 & -\omega_x \\ \omega_z & -\omega_x & \omega_x & 0 \end{pmatrix} q_{ba} \tag{2.11}$$

$$= \frac{1}{2} \begin{pmatrix} 0 & -\omega^T \\ \omega & \omega_\times \end{pmatrix} q_{ba} \tag{2.12}$$

The main advantage of using quaternions is the absence of singularities during the computations for any possible θ and $\boldsymbol{w}$. Moreover, quaternions do not require any trigonometric function to compute the dynamics. Since it is always possible to switch from quaternions to any other representation and vice versa via proper transformations, they can be used as the actual function to be integrated, while other attitude visualisations can be used in other stages of the algorithm e.g. to perform updates.

The exact discretization of the quaternion derivative can be computed without using the matrix exponential, under the assumption that the angular rate is kept constant between samples, which is reasonable is the sampling time of the device is high enough or there are no sudden rotations.

The continuous-time quaternion dynamics in Equation (2.11) can be rewritten as

$$\dot{q_{ba}} = \boldsymbol{W} q_{ba}$$

whose corresponding discrete-time dynamics are known to be

$$q_{ba}(k+1) = \mathrm{e}^{\boldsymbol{W}T_s} q_{ba}(k)$$

where T_s is the chosen sampling time. Since $\boldsymbol{W}$ is a skew-symmetric matrix, it can be shown that the above equation can be rewritten as

$$q_{ba}(k+1) = \left(\frac{\sin(||\boldsymbol{\omega}||)}{||\boldsymbol{\omega}||} \boldsymbol{W} + \cos(||\boldsymbol{\omega}||) \boldsymbol{I} \right) q_{ba}(k) \tag{2.13}$$

where $\boldsymbol{\omega}$ is defined as

$$\boldsymbol{\omega} = \omega \frac{T_s}{2}$$

2.2 Reference frames

Inertial navigation makes use of several reference frames to represent speed and position. Each one has its own advantages and drawbacks and can be more convenient than another one in some situations.

2.2.1 Inertial reference frame

A reference frame is termed *inertial* if Newton laws apply. In particular, it must hold that a body is steady or in linear uniform motion with respect to such reference if and only if the sum of all forces acting on it is zero. The inertial frame origin may be arbitrary as well as its orientation and can be still or moving with constant linear motion, but may not accelerate or rotate. In addition, a second frame is inertial if and only if it is in a fixed position or in linear uniform motion with respect to an inertial one.

Assuming the navigation frame to be inertial can be reasonable in our context as long as the movements happen in a limited space, e.g. in case of indoor navigation or localization.

2.2.2 ECEF frame

The *Earth-Centred, Earth-Fixed* frame (ECEF) is solidal to the Earth and centred at its centre of mass. Axes are defined as follows [10, 25]:

- the x axis is directed towards the intersection between the Prime Meridian and the Equator
- the z axis goes through the true North Pole, that is the Earth rotation axis

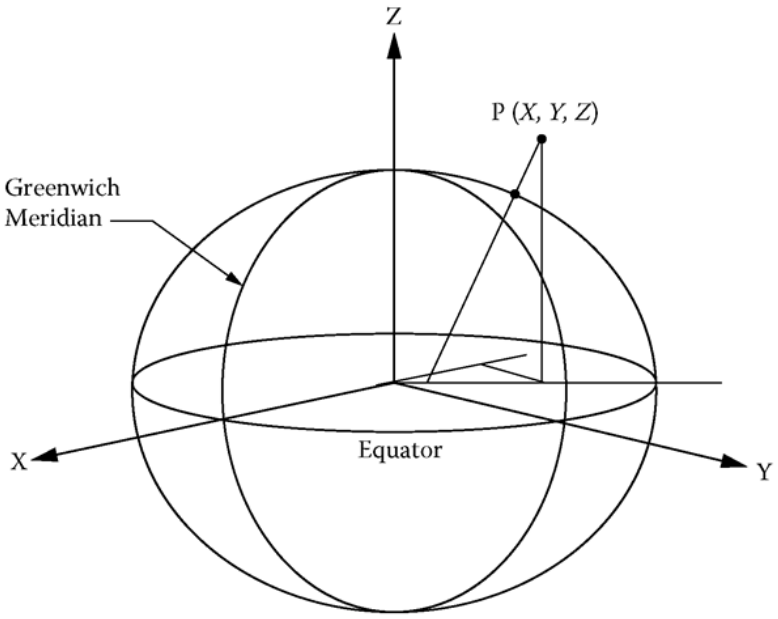

Figure 2.1: ECEF reference frame

- the y axis completes the right-handed coordinate system

The Earth rotation rate with respect to an inertial frame can be computed considering the daily rotation and the revolution about the Sun and turns out to be

$$\omega_{ie} = \frac{2\pi\,(365.25+1)\,rad}{365\cdot 24\cdot 3600s} \approxeq 7.292\cdot 10^{-5} rad/s \tag{2.14}$$

Since the ECEF frame is in rotation, it is *not* an inertial one. However, this may be neglected in some applications and therefore the *Earth-Centred, inertial* (ECI) frame is obtained, whose axes are defined for some epoch.

2.2.3 NED frame

The *North-East-Down* frame (NED), or *Local Tangent Plane* (LTP) is defined locally with its center at the vehicle's position [10, Chapter 2]. As it can be seen in Figure 2.2, x and y axes point northwards and eastwards respectively, while the z axis is directed inwards along the normal to the sphere (or ellipsoid). In this way the $x-y$ plane is always tangent to the Earth's surface. The NED can be particularly convenient to easily represent the speed of a vehicle. This frame is not inertial because of its rotation.

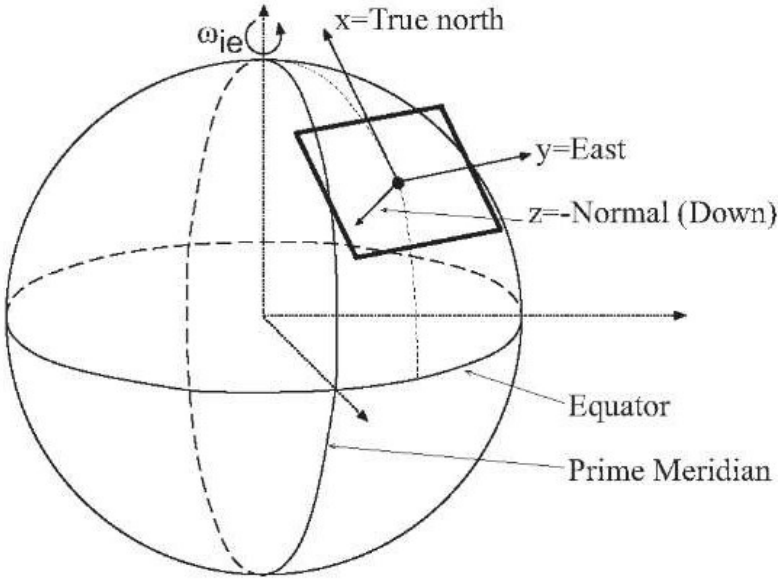

Figure 2.2: ECEF and NED reference frames

Moreover, it is possible to switch from a vector representation in ECEF frame to NED by means of the following rotation matrix [10, 25]:

$$R_{ne} = \begin{pmatrix} -\sin(\phi)\cos(\lambda) & -\sin(\phi)\sin(\lambda) & \cos(\phi) \\ -\sin(\lambda) & \cos(\lambda) & 0 \\ -\cos(\phi)\cos(\lambda) & -\cos(\phi)\sin(\lambda) & -\sin(\phi) \end{pmatrix} \tag{2.15}$$

where ϕ and λ are latitude and longitude of the NED frame origin. As it can be seen from Figure 2.3, the ECEF frame is first rotated by λ radians about z axis, and then of $\phi + \frac{\pi}{2}$ radians about axis y', whose direction is defined by the Cartesian reference frame obtained after the first transformation. Further details on rotation matrices and attitude representations are contained in Section 2.1

Similarly, the NEU (North-East-Up) frame can be defined by simply reverting the direction of the z axis.

2.2.4 Body frame

A body-fixed reference frame is the one that arises naturally from the characteristics of the body shape and motion and is rigidly attached to it. The x axis usually points towards the direction of the vehicle, the z one to the bottom of the object while the y is oriented so to complete the right-handed coordinate system.

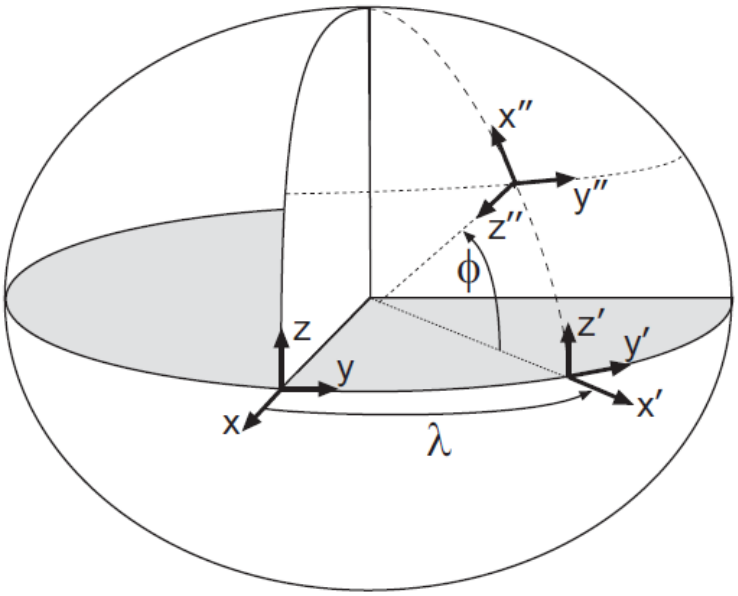

Figure 2.3: From ECEF to NED

The body frame is not inertial in general because of vehicle rotation. Sensors often have their axes aligned with the body frame, otherwise they are usually mounted in a fixed orientation with respect to the vehicle, so this additional transformation is precisely known or can be compensated during the design phase.

2.3 Translation dynamics

In this section the fundamental equations to describe the translational dynamics of a rigid body are obtained and specialised for inertial navigation.

2.3.1 Fundamentals

Given a vector p and two reference frames a and b whose origins are related by vector $\rho = O_a - O_b$, then it is possible to switch the frame of representation of p by means of

$$p^b = \rho^b + R_{ba} p^a \tag{2.16}$$

and its derivative is given as

$$\dot{p^b} = v^b = \dot{\rho^b} + \dot{R_{ba}} p^a \tag{2.17}$$
$$= \dot{\rho^b} + \dot{R_{ba}} p^a + R_{ba} \dot{p^a} \tag{2.18}$$
$$= \dot{\rho^b} + R_{ba} \left((\omega^a_{ba})_\times p^a + v^a \right) \tag{2.19}$$

By taking another derivative on v^b it is possible to obtain a very important equation of all inertial navigation systems. It is an exact derivation and can be specialised for different cases and any two reference frames. We have that

$$\ddot{p^b} = \dot{v^b} = \ddot{\rho^b} + \dot{R_{ba}} \left((\omega^a_{ba})_\times p^a + v^a \right) + R_{ba} \left((\omega^a_{ba})_\times v^a + (\dot{\omega^a_{ba}})_\times p^a + \ddot{p^a} \right) \tag{2.20}$$
$$= \ddot{\rho^b} + R_{ba} \left(2(\omega^a_{ba})_\times v^a + (\omega^a_{ba})_\times (\omega^a_{ba})_\times p_a + \dot{\omega^a_{ba}})_\times p^a + \ddot{p^a} \right) \tag{2.21}$$

2.3.2 Translational navigation equations

As outlined in Chapter 1, the fundamental equation of inertial navigation is Newton's Second Law of motion, whose most generic formulation is

$$m\ddot{p} = \sum F_i = F_I$$

where m is the (constant) body mass and the F_I term corresponds to the sum of all forces that are physically applied to the body, including gravity. The relation may be rewritten in a more convenient form in terms of force per units of mass in order to take out m and by introducing the *specific force* $f = \frac{1}{m} F_I$, which is the quantity actually read by accelerometers, thus obtaining the fundamental equation of inertial navigation[1]

$$\ddot{p^i} = f^i + G^i \tag{2.22}$$

where G^i is the gravity acceleration vector. By combining it with 2.20 and assuming b as the inertial reference frame and sharing the origin with a, we have that [10]

$$f^i + G^i = R_{ia} \left(2(\omega^a_{ia})_\times v^a + \left((\omega^a_{ia})_\times (\omega^a_{ia})_\times + (\dot{\omega^a_{ia}})_\times \right) p^a + \ddot{p^a} \right)$$

that is

$$\ddot{p^a} = R_{ai} \left(f^i + G^i \right) - 2(\omega^a_{ia})_\times v^a - \left((\omega^a_{ia})_\times (\omega^a_{ia})_\times + (\dot{\omega^a_{ia}})_\times \right) p^a \tag{2.23}$$

[1] In the *inertial* reference frame

If a is chosen to be the ECEF frame (subscript e) then ω_{ie} is constant and the following equations are obtained:

$$\dot{p^e} = v^e \tag{2.24}$$
$$\dot{v^e} = f^e - G^e - 2(\omega^e_{ie})_\times v^e - ((\omega^e_{ie})_\times (\omega^e_{ie})_\times)\, p^e \tag{2.25}$$
$$= R_{eb} f^b - G^e - 2(\omega^e_{ie})_\times v^e - ((\omega^e_{ie})_\times (\omega^e_{ie})_\times)\, p^e \tag{2.26}$$

where v^e is the body speed in the ECEF frame as seen from the ECEF frame. Moreover, it can be noticed the presence of the centrifugal effect term $((\omega^e_{ie})_\times (\omega^e_{ie})_\times)\, p^e$. However, its contribution can be neglected in most practical cases, in particular when MEMS sensors are used, due to their relatively high noise levels [12].

The dynamic equation of v^e will be specialised for the representation in NED and body frames in Equations (6.2) and (7.6), when the related INS mechanisation equations are obtained.

Moreover, if frame a is chosen to be inertial, the above equations simply boil down to

$$\dot{p^a} = v^a \tag{2.27}$$
$$\dot{v^a} = R_{ab} f^b - G^a \tag{2.28}$$

2.4 Geographical coordinates

A convenient way to represent an object's position on Earth is by geographic coordinates: latitude φ, longitude λ and height h. Moreover, it is of interest to use the ECEF body speed represented with respect to the Local Tangent Plane v^n_e, since it is a more "natural" choice from everyday experience. Hence, a transformation between ECEF and geographical coordinates must be performed. If Earth is assumed to be perfectly spherical[2] with radius $R_m \approxeq 6.3781 \cdot 10^6$m, from the definition of the ECEF frame in Subsection 2.2.2 we have that

$$x^e = (R_m + h)\cos(\varphi)\cos(\lambda) \tag{2.29}$$
$$y^e = (R_m + h)\cos(\varphi)\sin(\lambda) \tag{2.30}$$
$$z^e = (R_m + h)\sin(\varphi) \tag{2.31}$$

[2] Additional details about Earth's models (including the WGS84 ellipsoid) can be found in [10] as well

Moreover, it is also true that

$$v^e = \frac{\partial p^e}{\partial t} = \begin{pmatrix} \frac{\partial p^e}{\partial \phi} & \frac{\partial p^e}{\partial \lambda} & \frac{\partial p^e}{\partial h} \end{pmatrix} \begin{pmatrix} \dot{\phi} \\ \dot{\lambda} \\ \dot{h} \end{pmatrix}$$

Therefore, after some computations [10, Section 2.5.3] one obtains

$$v^e = \begin{pmatrix} -\sin\phi\cos\lambda & -\cos\phi\sin\lambda & \cos\phi\cos\lambda \\ -\sin\phi\sin\lambda & \cos\phi\cos\lambda & \cos\phi\sin\lambda \\ \cos\phi & 0 & \sin\phi \end{pmatrix} \begin{pmatrix} \dot{\phi} \\ \dot{\lambda} \\ \dot{h} \end{pmatrix}$$

By remembering $v^e = R_{ne}^T v_e^n$ and from Equation (2.15), the final expression of the geographical coordinates' derivative eventually turns out to be

$$\begin{pmatrix} \dot{\phi} \\ \dot{\lambda} \\ \dot{h} \end{pmatrix} = \begin{pmatrix} \frac{1}{R_m+h} & 0 & 0 \\ 0 & \frac{1}{(R_m+h)\cos(\phi)} & 0 \\ 0 & 0 & -1 \end{pmatrix} v_e^n \tag{2.32}$$

It is also possible to use more accurate Earth models by considering the WGS84 ellipsoid, for example. The resulting equations present the same structure enriched with parameters taking into account eccentricity, flatness and so on.[10, Section 2.3.2]

Chapter 3

Sensors

As will be better outlined in Section 5.1, sensor sets are very similar across different inertial navigation units. Their components have to be properly characterised in order to design a suitable and enough accurate model to develop an algorithm on. The sensors typologies that are considered in this work are the following:

- Accelerometers
- Gyroscopes
- GNSS
- Odometry

All of them are affected by several types of errors depending on their working principle and manufacturing technology. Magnetometers are not considered because electric power groups of trains may heavily interfere with the measurements, thus making impossible to detect the Earth's weak magnetic field, whose strength is about 0.45 Gauss on average. Barometers are not considered in this work as they are not suitable for train applications, too, because of the pressure difference that can arise inside tunnels.

MEMS technology is very popular today for sensors manufacturing. It consists of using miniaturised electro-mechanical elements. Their main advantages are reduced size, power consumption and production cost, yet on the other hand they usually present less accuracy with respect to other constructive techniques. In addition, they require very low maintenance and are suitable forr use in harsh environments [31].

MEMS technology is used to manufacture a wide range of detectors such as accelerometers, gyroscopes, compasses, humidity sensors, microphones and so on. They also find applications in the biomedical and optics field.

Moreover, also miniaturised actuators and structures can be built for some precision applications.

3.1 Accelerometers and gyroscopes

As mentioned in Chapter 1, accelerometers measure not only body acceleration but the so-called *specific force* [10], that is the true acceleration minus the gravity term. On the other hand, gyroscopes measure the angular rate of the vehicle. They both take measurements with respect to an inertial reference frame. They can be based on different technologies; for example, the first devices were based on mechanical solutions and some gyroscopes use optical phenomena to detect rotation [21]. MEMS devices will be considered in this work as they are very popular in integrated INS solutions.

MEMS accelerometers typically exploit either some piezoelectric or capacitive effect, depending on the manufacturer choice. The latter may be used also to build micro-actuators [5] on the same principle and requires less processing in general. They are typically built as movable proof masses held into place via springs and mechanical suspensions[1]. A number of micro-plates on the moving part and the frame act as capacitors.

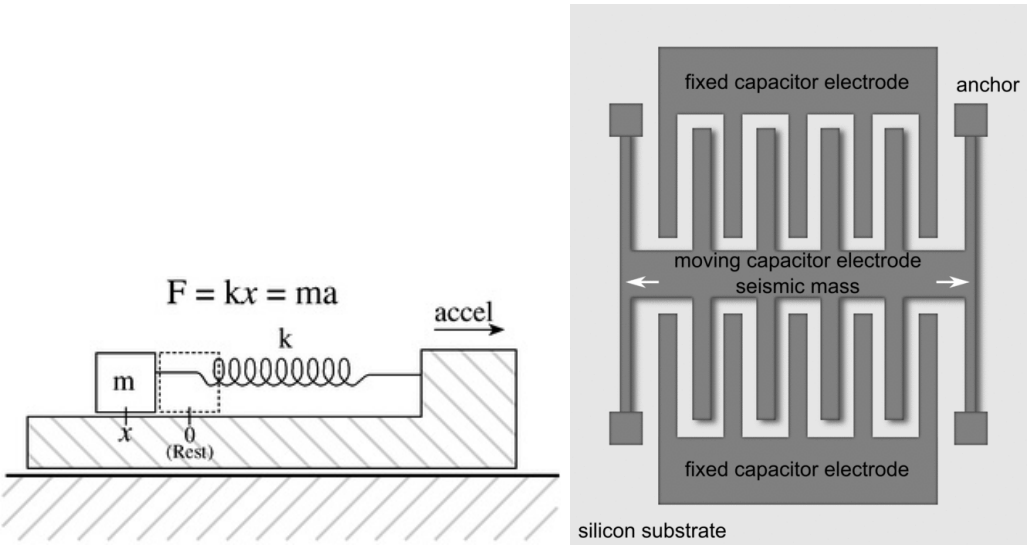

Figure 3.1: Working principle of accelerometers (left) and CAD representation of a capacitive effect accelerometer (right)

[1]This helps to motivate the *specific force* measurement: the detected acceleration is the reaction term holding the mass in place

On the other hand, gyroscopes usually consist of micro-masses put into vibration by capacitive actuation and proper circuitry to keep them in movement along a defined axes. If the overall system is put into rotation about an axis orthogonal to motion's, then a second vibration mode is triggered along the third direction induced by the Coriolis acceleration. This motion can be sensed by additional, properly placed capacitive micro-devices. The amplitude of the phenomenon is function of the angular rate and allows to detect rotation thanks to a properly-designed sensing system.

Some MEMS gyroscopes can also detect the angle of rotation instead of just the speed [21]; such devices are called *whole-angle mode gyroscopes* or *hemispherical resonator gyroscope* and the resonating element can have the shape of a wineglass.

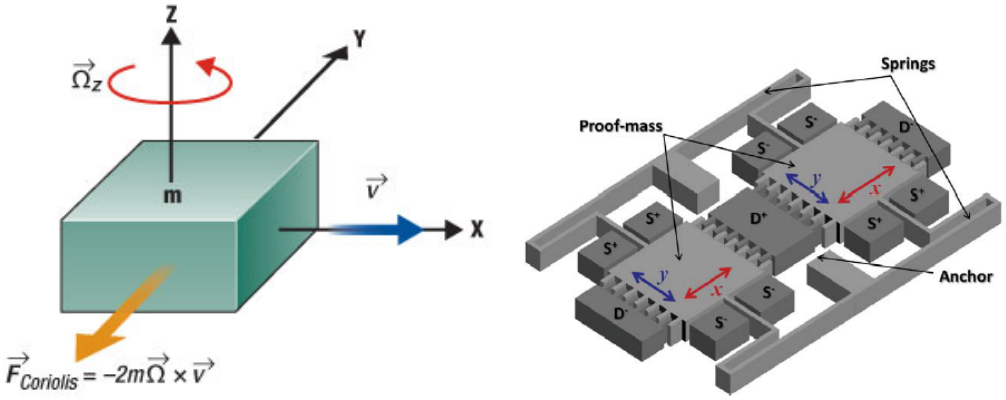

Figure 3.2: Working principle of gyroscopes (left) and CAD representation of a Tuning Fork Gyroscope, or TFG (right)

3.1.1 Model

Ideal sensor readings are always affected by several sources of errors. Some of the most common non-idealities that can be present[25, Chapter 8] include

- Scale factors and cross-coupling between the axes. Gyroscopes may also be affected by some components depending on body acceleration, too.

- Measurement offset, also known as *bias*, which is a slowly time-varying additive disturbance. Its ratio of change is referred as *bias stability* and commonly measured in terms of the Allan Variance computed over long-term measurements in rest conditions [23]

- Measurement noise. This disturbance can be reasonably modelled as a zero-mean white noise, its power density depending on the quality of the sensors. This effect is particularly relevant when MEMS technology is used. On the other hand, the advantages of such sensors are reduced cost and very small size, allowing miniaturized solutions

The amplitude and relevance of such effects depends a lot also on their technology. In this work sensors white noise and bias will be considered. Scale factors and other effect are not covered since they are often corrected via factory calibration and their coefficients do not vary a lot in time with respect to sensor biases, which change a lot during operation and also between device switch-ons[4]. In a stochastic framework, bias behave as random walks, that is the integral of some white noise, but can be also regarded as a constant when building a functional model. Therefore, accelerometer and gyroscope readings are modelled as

$$u = a^b - g^b + b_u + \nu_u \tag{3.1}$$

$$\omega = \omega_{ib}^b + b_\omega + \nu_\omega \tag{3.2}$$

where u and ω are the actual accelerometer and gyroscope readings, a^b the true acceleration, g^b the gravity vector, ω_{ib}^b the true angular rate with respect to an inertial frame of reference, b_u and b_ω the sensor biases and ν_u,ν_ω the white noise.

Because of their relatively high noise level, accelerometer and gyroscope usually require some pre-elaboration on their readings prior being used inside any navigation algorithm. Proper low-pass filters can already be enough to achieve acceptable levels; alternatively, more complicated techniques may be used.

3.2 Global Navigation Satellite System (GNSS)

The expression *Global Navigation Satellite System* (GNSS) denotes all the navigation systems based on satellite constellations to provide accurate positioning of a vehicle or object. Each satellite transmits a specific radio signal which is decoded by the receiver to determine its position on the Earth. Currently active GNSS systems include GPS (USA) and GLONASS (Russia), while the Galileo positioning system (EU) is under development and its constellation is expected to be completed and full functionality to be achieved by 2020[9]. Many commercial receivers can even access several of these GNSS systems.

The basic idea behind GNSS and the other systems is triangulation. If the satellites' position is known as well as their distance from the receiver it is possible to reconstruct the position. Each satellites continuously transmits information about its own position, clock accuracy and less accurate information about the other satellites' positions and the time the message was transmitted[2]. To this purpose, very high-precision atomic clocks are equipped on each satellite.

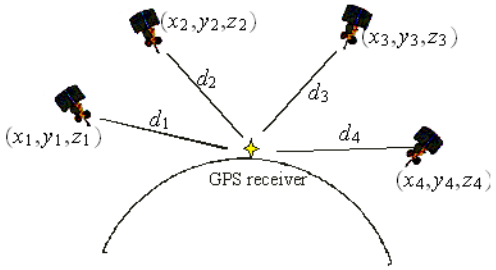

Figure 3.3: Working principle of GNSS systems

The vehicle's GNSS receiver determines its distance from a specific satellite by comparing the time of arrival of the signal with the transmission time which is contained inside the message and multiplying by the speed of light. However, this information is not accurate because of the amplitude of the light speed being influenced by the means of propagation and disturbances (e.g. clouds) as well as the relatively high clock inaccuracy mounted on the GNSS device with respect to the atomic-grade ones mounted on the satellites. The distances from each source computed by the receiver are also called *pseudo-ranges*.

[2]Thus providing also a time information to the user

At least three[3] different radio signals are needed to reconstruct the vehicle coordinates and height under ideal conditions. The unknown clock drift can be recovered by measurements from a fourth satellite so that local time estimation reaches the required precision;. Several possibilities exist to resolve the position, from least squares fitting to Newton-Gauss methods [25, Appendix D]. The more satellites are visible to the receiver, the higher the accuracy will be. Finally, control stations on Earth continuously monitor the satellite segment of each GNSS system and correct for inaccuracies or drifts, which have strong effects on systems precision due to the extremely high speed of light.

GNSS receivers can also provide direct information on the vehicle speed and direction on the local tangent plane to the Earth, without need to post-elaborate position data. GNSS position accuracy is generally within few meters, but depends on several factors; some of them are radio interferences, satellite signal being blocked by buildings or trees, multipath (i.e. signal reflection), atmospheric conditions and so on. Typical values for position accuracy range from 2 to 5 meters and from 0.2 to 0.4 m/s for speed. Yet, these values may vary according to the previously-mentioned operating conditions and there is no true known value for the standard deviation, thus modelling GNSS noise may be a complex task[1, Chapter 3].

Augmentation techniques can be used to achieve better precision, such as Differential GPS or SBAS[10, Chapter 8]. Some receivers may mount two antennas for enhanced heading and positioning resolution.

Moreover, Differential GPS (D-GPS) uses several ground antennas at known fixed positions to provide more information to the receiver which can then resolve its kinematic state with a much better position. This technique employs ground infrastructures and requires a communication channel between them and the vehicle, so it can be used only in specific, limited-area applications and with proper receivers, which is clearly not the case for trains on normal passenger or freight lines.

3.2.1 Model

In this work, GNSS position and speed are modelled as the true values plus some white noise to model the system's inaccuracy according to the aforementioned statistics, that is

$$p_{GNSS} = p_{true} + \nu_{GNSS}$$

[3] Actually, a fourth satellite should be used to discriminate between the two points that come from the intersection of three spheres in 3-D space

with p_{GNSS} being the actual measurement p_{true} the true value and ν_{GNSS} the white noise. Velocity information is assumed to be available in the three dimensions and also in the form of ground speed plus heading. Such conversion is usually carried out automatically by the receiver.

3.3 Odometry model

It is often possible to access the vehicle's total travelled distance and/or speed by means of on-board sensors, as in the case of trains and land vehicles in general. Some details on its accuracy requirements are specified in [26], in particular for what concerns admissible errors during wheel-slip or skidding, that is when relative motion occurs at the point of contact between wheels and rail during deceleration or acceleration, respectively, as well as how to detect such anomalous conditions. Speed odometry error has to be contained within 2% of the speed when travelling slower than 50 km/h and 1% otherwise.

In this work, odometry speed measurement is assumed to be available at all times and is modelled as the true value plus some proper white noise in this work, while a fixed bias is not considered. Hence, speed from odometry is modelled as

$$v_o = v_{o,true} + \nu_o \tag{3.3}$$

where $v_{o,true}$ is the true body speed value, ν_o the white disturbance and v_o the actual reading.

Chapter 4

Kalman Filtering

The *Kalman Filter* (KF) is a type of linear observer which is widely used in noisy estimation applications such as control and navigation, in order to get the best possible estimate of the internal state of a system in a stochastic framework [15].

Just like other observes, the Kalman Filter works in a two-step fashion: prediction and correction. In the prediction phase the current state estimate and the predicted error covariance are propagated forward in time. Once measurements are available, correction can take place and the system's states are updated according to the error between the actual measurements and their estimate (also known as *innovation*). The greatest advantage of the Kalman Filter with respect to other structures is that the stochastic characteristics of the estimate error (that is, covariance and cross-covariance) are also propagated.

4.1 Continuous-time Kalman Filter

Suppose to have a linear, time-invariant (LTI) continuous-time system perturbed by noise which can written in the form

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) + Dw(t) \\ y(t) &= Cx(t) + \nu(t) \end{aligned}$$

where $w(k)$ and $\nu(k)$ are zero-mean, white Gaussian stochastic processes with known covariances Q and R, respectively. System matrices A, B and C must be known in advance.

Q and R are commonly referred to as *process noise* and *measurement noise*. Their characterisation is essential in order to get the best possible be-

haviour of the filter in presence of disturbances. In some sense, they describe the level of accuracy of the filter prediction and how much the filter can "trust" the external measurements, respectively. The estimates' stochastic characterisation is defined by the *covariance matrix* P.

Given this framework, the KF in its continuous-time formulation has the form

$$\begin{aligned}
\dot{\hat{x}}(t) &= A\hat{x}(t) + Bu(t) + K(t)(y(t) - \hat{y}(t)) \\
\dot{P}(t) &= AP(t) + P(t)A^T + DQD^T - K(t)RK(t)^T \\
K(t) &= P(t)C^T R^{-1} \\
\hat{y}(t) &= C\hat{x}(t)
\end{aligned}$$

The estimate provided by the Kalman Filter can be shown to be unbiased; if noise is Gaussian it is also the maximum likelihood one, meaning that it minimises the error covariance (P) as well as the mean-squared error from the true value. Assume now that pair (A, C) to be detectable and (A, D) to be stabilisable. Then, it can be also shown that the KF error covariance P is bounded and - most importantly - the presented filter formulation is globally asymptotically stable, hence state estimation error will eventually reach zero [10, Chapter 5]. This version of the KF, in which update occurs continuously in time is also called the *Kalman-Bucy filter* [6].

If the system and filter representation are continuous-time but measurements come at given time instants only (e.g. as from a digital processor) then the *Hybrid Kalman filter* is obtained [30]. Given the usual LTI system in state-space form, its prediction equations are as follows:

$$\dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t) \tag{4.1}$$

$$\dot{P}(t) = AP(t) + P(t)A^T + DQD^T \tag{4.2}$$

When measurements are available and update is performed, the following relations are used to correct the estimate:

$$\begin{aligned}
K &= P(t)C^T \left(CP(t)C^T + R\right)^{-1} \\
x_c(t) &= \hat{x}(t) + K\left(y(t) - C\hat{x}(t)\right) \\
P_c(t) &= \left(I - KC\right)P(t)
\end{aligned}$$

Corrected values x_c and P_c are then used to re-initialise Equations (4.1) and (4.2).

4.2 Discrete-time Kalman Filter

Suppose to have an LTI, discrete-time system perturbed by noise which can written in the form

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + Dw(k) \\ y(k) &= Cx(k) + \nu(k) \end{aligned}$$

The Kalman Filter equations for the prediction step determine the so-called *a priori* estimates and are the following:

$$\begin{aligned} \hat{x}(k|k-1) &= A\hat{x}(k-1|k-1) + Bu(k-1) \\ P(k|k-1) &= AP(k-1|k-1)A^T + DQD^T \end{aligned}$$

while the state and covariance update equations are

$$\begin{aligned} K(k) &= P(k|k-1)C^T \left(CP(k|k-1)C^T + R\right)^{-1} \\ \hat{x}(k|k) &= \hat{x}(k|k-1) + K(k)\left(y(k) - C\hat{x}(k|k-1)\right) \\ P(k|k) &= \left(I - K(k)C\right)P(k|k-1) \end{aligned}$$

where $\hat{x}(k|k)$ and $P(k|k)$ are also called *a posteriori* estimates of state and covariance, respectively.

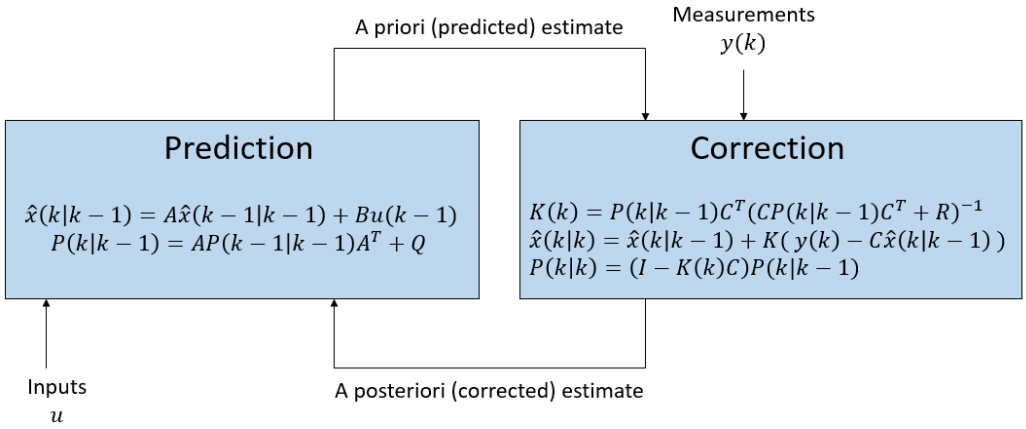

Figure 4.1: Overview of the Kalman Filter algorithm

It has to be noticed that the filter equations are written in a recursive fashion which makes implementation easy. Moreover, correction is not strictly required to take place at *every* algorithm step nor at some regular rate. This happens if some measurements are bad or missing[10] or are simply provided with a higher period than the algorithm cycle time. If this holds there is a loss of the properties presented in Section 4.1, in particular for what concerns optimality of P.

4.3 Extended Kalman Filter

The *Extended Kalman Filter* (EKF) is a sub-optimal formulation of the Kalman Filter and is widely used in inertial navigation applications. It is commonly applied to non-LTI systems.

A crucial aspect to keep in mind about the EKF is that its optimality properties outlined in [13] and Section 4.1 cannot be guaranteed because of the non linearity of the system. As a matter of fact, the KF matrices are functions of the state error covariance which is computed exactly for an LTI system but just approximately to the first order in the other cases. This can be a potential source of errors; as a consequence, too large inaccuracies may result in bad behaviour or even divergence of the filter [29]. However, the EKF shows acceptable performances if applied when higher-order terms are not too relevant.

The EKF has the same structure of the standard filter, with the difference that matrices A,C,D are now built by means of linearisation about the current state estimate (and control input), meaning that they are possibly complex, non-linear functions of state, inputs and noises. Given a generic non linear system in the form

$$\begin{aligned} \dot{x} &= f(x, u, w) \\ z &= h(x, \nu) \end{aligned}$$

The matrices to be used for the EKF are obtained as

$$\begin{aligned} A(k) &= \left.\frac{\partial f(x,u,w)}{\partial x}\right|_{x=\hat{x}(k|k-1),u=u(k-1),w=0} \\ D(k) &= \left.\frac{\partial f(x,u,w)}{\partial w}\right|_{x=\hat{x}(k|k-1),u=u(k-1),w=0} \\ C(k) &= \left.\frac{\partial h(x,\nu)}{\partial x}\right|_{x=\hat{x}(k|k-1),\nu=0} \end{aligned}$$

Linearising about $w = 0$ and $\nu = 0$ comes directly from the assumption of zero-mean noise in Section 4.1. Should this not hold, the zero vector has to be replaced by their respective mean value. As a final remark, state prediction should be carried out by using the actual non-LTI model for greater accuracy with respect to a linearised version computed from matrices A, D and C. The EKF can be applied to both continuous- and discrete-time systems.

4.4 Unscented Kalman Filter

The *Unscented Kalman Filter* (UKF) is another variant of the standard Kalman Filter to be used in non linear systems [14] with the main purpose is to overcome the limitations and flaws of the EKF [29].

As a matter of fact, matrix P plays a crucial role as it contains the covariance of the state estimation error. The information about how noise covariance is "reshaped" by the system dynamics should be propagated in the best possible way since it strongly influences the Kalman gain K and thus the correction performed by the observer, as can be seen from the previous formulations of the filter.

The EKF might be not applicable for all non-LTI systems. First, the Jacobian matrix A may be complex to derive or not precisely known for some systems or in some states; secondly, the expression APA^T is just a first-order approximation of the predicted value of P and is exact only for linear, time-invariant systems. This may be not sufficient to perform the right state correction on some non-linear systems in which higher-order terms are more relevant. This is way the EKF is also referred to as "First-Order Filter". It is noteworthy to point out that higher-order formulations of the EKF exist, but are hard to derive, computationally expensive and applicable for low noise levels [8].

The main concept behind the UKF is that "it is easier to approximate a probability distribution rather than a generic non-linear function" [14]. Even if also the EKF propagates a probability description of the estimate, the Unscented Kalman Filter uses a more complete characterisation built by using higher-order moments rathe than just mean and covariance.

The core part of the UKF is to propagate a number of carefully-chosen states and expected outputs (called *sigma points* or just σ*-points*) via the state transition function. Then, such points are transformed by the non-linear function (the state dynamics, in this case). The new distribution moments are computed and the state estimate is eventually corrected accordingly. This technique has the advantage to propagate higher-order moments of the involved probability distributions with higher precision than the EKF, even though not exactly. The level of achieved accuracy depends on the number and the choice of sigma points.

4.5 Invariant Extended Kalman Filter

Another advanced structure for state estimation in a stochastic framework is the so-called *Invariant Extended Kalman Filter*[3]. This can be especially

useful if the state-space is non linear but can be described by Lie groups or mathematical structures featuring symmetries, in general.

In this filter, an exponential mapping can be used to perform a transformation into a subset of $\mathbb{R}^N$, whose size is the same of the corresponding Lie algebra; this is not performed by a standard EKF, which has the "additional" task to converge into it and does not feature this invariance property of the IEKF.

Because of how the use of such exponential mapping instead of a simple linearisation, the IEKF is structurally more robust with respect to the standard Extended Kalman Filter in terms of state estimation, even if neither has optimality guarantees.

Chapter 5

Products for terrestrial inertial navigation and the proposed approach

Preliminary research is necessary to understand the level of performance of currently available inertial navigation devices as well as the standard methodologies for algorithms to approach the problem of inertial navigation. These starting information will be used in order to develop a valuable solution to be applied to trains.

Nowadays, Kalman filtering is definitely one of the most common tools to develop inertial navigation algorithms [10, 25]. In particular, the Extended KF is used because of the system being non linear. INS can be developed in different ways according to the available signals: there may be a single EKF[22], two cascaded filters [18], both structures having advantages and disadvantages in terms of complexity, modularity and domain of convergence in practical situations. Other, more complex techniques, such as Invariant EKFs can be used as well [3]. Additional details and relevant results about Kalman filters are presented in Chapter 4.

Moreover, several non-linear observers have been developed in literature for inertial navigation. They are used for both position/velocity estimation and are also quite popular for attitude reconstruction by exploiting the properties of the rotation group $SO(3)$ [17, 18]. Finally, techniques from adaptive control are commonly employed to estimate and compensate for sensors non-idealities, such as biases [18]; their approach is generic and can be exploited under many engineering fields.

5.1 Market Analysis

A market research was carried out to figure out the state of the art concerning commercial inertial navigation systems. Data were collected about the performance limits of each device as well as their operative fields. In particular, dead reckoning performance were considered to understand the entity of the drift after prolonged period of GPS outages. The purpose of this research is also to understand the sensors available on board across several solutions according to their intended application field.

VectorNav

VectorNav is a company specialized in high-end embedded navigation systems [28]. Two of its most remarkable products are the *VN-200* and *VN-300* GPS/INS modules. They both have 3-axis MEMS accelerometer, MEMS gyroscope, magnetometer and a barometer[1]. They also have a small size which makes them suitable for integration on several types of hardware. In particular, VN-200's dimensions are 36 x 33 x 9 mm.

Figure 5.1: VectorNav VN-200

GPS data are updated at a rate of 5 Hz and the navigation algorithm runs at a 400 Hz rate. Thermal calibration can be requested; in such a way sensors noise is better compensated over the whole range of operating temperatures

[1]Such sensor sets is often called *10-Axis IMU* in a commercial context, even if such nomenclature is misleading

(-40 °C to +85 °C), otherwise just a 25 °C-calibration is performed. Some accuracy information about the VN-200 module are the following:

- Horizontal position accuracy: 2.5 m
- Vertical position accuracy: 5 m
- Velocity accuracy 0.05: m/s

However, no information about its operative capabilities without GPS could be gathered.

SBG Systems

SBG Systems is specialized in ING/GPS modules [19]. Some of its most remarkable module series are *Ellipse*, *Ekinox* and *Apogee*. They are all characterised by reduced dimensions and a number of communication interfaces, as well.

Concerning *Ellipse* modules, they are equipped with MEMS accelerometers, gyroscopes. Some also have magnetometers and pressure sensors and can exploit the vehicle odometry (where available) to improve their performance. The declared accuracy during GPS availability is of 2 meters for positioning and 0.1 m/s for speed estimation. Data are gathered from inertial sensors at a 200 Hz rate and processed at 1 kHz rate inside the unit, together with GPS position and speed data.

It is noticeable to point out that their performance in absence of GPS signal is documented on their website. The company declares an error of 12 meters over a 6-minute, 2.5 km journey in urban environment without GPS aiding. Such discrepancy boils down to about 3 meters if odometry aiding is enabled [20].

Advanced Navigation

Advanced Navigation in an Australian company specialised in high-quality robotics and navigation technologies[2]. *Spatial* is a miniaturized navigation module with thermally-calibrated sensors able to perform properly under a wide range of operating conditions. The device mounts MEMS accelerometers, gyroscopes, a triaxial magnetometer and a pressure sensor. Moreover, external odometry can be used as an additional aid.

Concerning its navigation precision, the *Spatial* features a position accuracy of 2 m and 3 m on the horizontal and vertical directions, respectively, a speed precision of 0.05 m/s and a sub-degree attitude precision.

Figure 5.2: Advanced Navigation Spatial

Dead-reckoning capabilities of the device are briefly described according to a test carried out along a 1.6 km urban tunnel. The test was carried out in 2.5 minutes with the following results:

- Position error turns out to be 5 % (about 90 meters) without odometry aiding
- If odometry is used inside the algorithm, then the position error becomes about 1 % of the overall path (17 meters)

According to the company website, the *Spatial* module can also work without much performance degradation in a urban environment where GNSS coverage and accuracy is poor. The estimate of the vehicle's kinematic state is computed at 1 kHz rate, as the other devices typically do.

Trimble

The *Trimble Aardvark 88788-50* embedded module is designed to provide stable position estimation during GNSS outages or bad coverage such as in urban canyons where buildings may obstruct the line of sight with the satellites [27]. It is equipped with gyroscopes and GPS receiver only and lacks accelerometers. Unlike the previous modules, it cannot resolve the the vehicle's speed autonomously so it must receive informations from odometry while its gyroscope determine the orientation in order to properly integrate and obtain the position.

Despite its "reduced" sensors set, the module is declared to have 2 meters of positioning accuracy and 3 meters of elevation precision.

5.2 Summary of market analysis

The above-presented solutions present good and similar performances in terms of position, speed and orientation accuracy. The positioning error is of 2-3 meters for the horizontal components and of 2-5 meters along the vertical one among all the presented devices during GNSS availability. In addition, all the solutions in this chapter are strapdown IMUs, which are definitely the most popular nowadays.

Most of the solutions are designed for land and air vehicles. Anyway, application on trains is possible even if there is no explicit reference to it. Every module comes with different communication interfaces and protocols and has limited size and low power consumption, which makes them suitable for designing embedded systems. Their sensors set is very similar and MEMS technology is almost always present. Moreover, the use of odometry can further improve the algorithm performance even if it is strictly required for the *Trimble* only.

On the other hand, few details could be collected about the performance during GNSS outages, even if the relative positioning error was not very different between the *Ellipse* and *Spatial* modules. In addition, it is not even clear whether other devices available on the market can actually operate without a regular update from the GNSS system. It is also to be pointed out that use of the magnetometer, which is present on many platforms, might be impossible under some operating conditions and this can worsen the overall performance since one of the assumed measurement is practically useless.

A more comprehensive list of GPS/INS devices from the same and other manufacturers can be found at [7].

5.3 The proposed approach

According to the information gathered in the previous sections, the approach in this work is to develop first a standard EKF-based algorithm because of its popularity. Its detailed formulation is presented in Chapter 6.

Several two-block solutions have been developed, too, and are presented in Chapter 7. The main purpose of this second approach is to separate the problem of attitude estimation from position and velocity and to overcome some limitations of the single-stage algorithm that have been detected during simulations, the limited use of explicit information on attitude being the first.

As a matter of fact, accelerometer readings can be also used to provide partial information about the inclination of the vehicle without relying just on low-rate GNSS position and speed data; moreover, using two separate

stages helps reducing overall complexity and opens to modularity. Further details on the interconnections are found in Chapter 9.

In particular, an AHRS based on a standard EKF and one on a formulation of the explicit complementary filter on $SO(3)$ have been developed and adapted to the available information sources. The choice of implementing a non linear observer is related to their popularity[17] in AHRS algorithms.

Concerning translation dynamics, a full-order observer and a reduced one have been implemented in Simulink, the latter exploiting train lateral velocity being structurally zero in order to quantify the performance improvement, especially during GNSS outages.

Chapter 8 eventually describes the proposed solution for integrating maps during navigation. It has been developed in order to easily integrate on the inertial navigation solutions designed in the previous chapters with as few modifications as possible, require limited computational effort and provide as much explicit information as possible.

Chapter 6

Single-Stage INS Algorithm

The first INS algorithm was developed on the basis of a standard EKF structure as reported in Section 4.3.

6.1 Mechanization equations

6.1.1 Speed

The vehicle speed is represented in the North-East-Down frame, which is not assumed to be inertial. Therefore, all relative rotations have been taken into account. We have that[10]

$$\dot{v_e^n} = \begin{pmatrix} v_N \\ v_E \\ v_D \end{pmatrix} = R_{ne}\dot{v_e} + R_{ne}(\omega_{en}^e)_\times v_e \tag{6.1}$$

By replacing $\dot{v_e}$ from Equation (2.26) we have that

$$\dot{v_e^n} = R_{ne}(R_{eb}(u - b_a) + g^e - 2(\omega_{ie}^e)_\times v_e) + R_{ne}(\omega_{en}^e)_\times v_e$$

Where f^b has already been replaced by the unbiased accelerometer readings. This brings to the final dynamics equation for the speed

$$\dot{v_e^n} = R_{nb}(u - b_a) + g^n - (\omega_{en}^n + 2\omega_{ie}^n)_\times v_e^n \tag{6.2}$$

where all terms are functions of measurable quantities, system states or known constants. u is the vector of accelerometer readings, b_a is its biases, $g^n = \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix}$ is the gravity acceleration, ω_{en} is the rotation between the NED

and ECEF frame due to the vehicle motion on the Earth surface defined as

$$\omega_{en}^{n} = \begin{pmatrix} \dot{\lambda}\cos(\phi) \\ -\dot{\phi} \\ -\dot{\lambda}\sin(\phi) \end{pmatrix} \tag{6.3}$$

and the Earth rotation rate vector in the NED frame is

$$\omega_{ie}^{n} = \begin{pmatrix} \omega_{ie}\cos(\phi) \\ 0 \\ -\omega_{ie}\sin(\phi) \end{pmatrix} \tag{6.4}$$

where ω_{ie} is defined as in equation (2.14)

6.1.2 Position

The position expressed in geographic coordinates (latitude, longitude and height) has the following nonlinear dynamics from Equation (2.32):

$$\begin{aligned} \dot{\phi} &= \frac{v_N}{R_m + h} \\ \dot{\lambda} &= \frac{v_E}{(R_m + h)\cos(\phi)} \\ \dot{h} &= -v_D \end{aligned}$$

where R_m m is the Earth mean radius. They can be rewritten in matrix form as

$$\dot{p} = \begin{pmatrix} \dot{\phi} \\ \dot{\lambda} \\ \dot{h} \end{pmatrix} = \begin{pmatrix} \frac{1}{R_m+h} & 0 & 0 \\ 0 & \frac{1}{(R_m+h)\cos(\phi)} & 0 \\ 0 & 0 & -1 \end{pmatrix} v_e^n = T(\phi, h) v_e^n \tag{6.5}$$

6.1.3 Attitude

The dynamic of the attitude R_{nb} comes from equation (2.3) and is given by[10]

$$\dot{R}_{nb} = R_{nb}\left(\omega_{nb}^{b}\right)_{\times} = R_{nb}\left(\omega_{ib}^{b} - \omega_{in}^{b}\right)_{\times} = R_{nb}\left((\omega - b_g) - \omega_{ie}^{b} - \omega_{en}^{b}\right)_{\times} \tag{6.6}$$

where ω denotes gyroscope readings, b_g the sensor bias and ω_{ie}^{b} and ω_{en}^{b} are the rotations in body frame of the aforementioned angular rates. Quaternion representation has been chosen for dynamics implementation. Hence, from Equations (2.11) and (6.6) the final expression can be obtained as

$$\dot{q}_{nb} = \frac{1}{2}\begin{pmatrix} 0 \\ \omega - b_g - \omega_{ie}^{b} - \omega_{en}^{b} \end{pmatrix} \circ q_{nb}$$

6.1.4 Sensor biases

As mentioned in Section 3.1, sensor biases are slowly time-varying additive disturbances to the readings, so they can be modelled as constants:

$$\dot{b_a} = \mathbf{0} \tag{6.7}$$

$$\dot{b_g} = \mathbf{0} \tag{6.8}$$

6.1.5 Measurements

The available measurements for the single-stage formulation of the INS algorithm are GNSS position and speed, the former being already expressed in geographical coordinates. Thus, the measurement model is simply

$$z = \begin{pmatrix} p \\ v_e^n \end{pmatrix} \tag{6.9}$$

For the sake of performance analysis, the complete speed vector is assumed to be fully known.

6.2 EKF design

Linearisation of the full-order, single stage EKF is performed based on Euler Angles. The state transition matrix for the EKF is in the form [22]

$$A = \begin{pmatrix} J_v \\ J_p \\ J_\rho \\ \mathbf{0}_{3\times 15} \\ \mathbf{0}_{3\times 15} \end{pmatrix} \tag{6.10}$$

where the sub-matrices are defined as

$$J_v = \begin{pmatrix} J_{vv} & J_{vp} & -R_{nb}(u - b_a) & -R_{nb} & \mathbf{0}_3 \end{pmatrix}$$

$$J_p = \begin{pmatrix} T(\phi, h) & J_{pp} & \mathbf{0}_{3\times 9} \end{pmatrix}$$

$$J_{pp} = \begin{pmatrix} 0 & 0 & -\frac{v_N}{(R_m+h)^2} \\ v_E \frac{\sin(\varphi)}{((R_m+h)\cos(\varphi))^2} & 0 & -\frac{v_E}{(R_m+h)^2\cos(\varphi)} \\ 0 & 0 & 0 \end{pmatrix}$$

$$J_\rho = \begin{pmatrix} (\mathbf{0}_{3\times 6} & EA_2) + EA_3 & \mathbf{0}_3 & -E \end{pmatrix}$$

where E is the matrix for computing the RPY angles derivative, as defined in Equation (2.6), and

$$A_2 = \begin{pmatrix} 0 & -\dot{\psi}\cos\theta & 0 \\ -\dot{\theta}\sin\varphi + \dot{\psi}\cos\varphi\cos\theta & -\dot{\psi}\sin\varphi\sin\theta & 0 \\ -\dot{\theta}\cos\varphi - \dot{\psi}\sin\varphi\cos\theta & -\dot{\psi}\cos\varphi\sin\theta & 0 \end{pmatrix}$$

$$A_3 = -R_{bn}\begin{pmatrix} 0 & \frac{1}{R_m+h} & 0 & 0 & 0 & \frac{-v_E}{(R_m+h)^2} & \mathbf{0}_{1\times3} \\ \frac{-1}{R_m+h} & 0 & 0 & 0 & 0 & \frac{v_N}{(R_m+h)^2} & \mathbf{0}_{1\times3} \\ 0 & \frac{-\tan\phi}{R_m+h} & 0 & \frac{-(v_E)^2}{(R_m+h)(\cos\phi)^2} & 0 & \frac{v_E\tan\phi}{(R_m+h)^2} & \mathbf{0}_{1\times3} \end{pmatrix} + \\ + R_{bn}\begin{pmatrix} \mathbf{0}_{1\times3} & w_{ie}\sin\phi & \mathbf{0}_{1\times5} \\ \mathbf{0}_{1\times3} & 0 & \mathbf{0}_{1\times5} \\ \mathbf{0}_{1\times3} & w_{ie}\cos\phi & \mathbf{0}_{1\times5} \end{pmatrix}$$

$$J_{vv} = \begin{pmatrix} \frac{v_D}{R_m+h} & -2w_{ie}\sin\phi - \frac{2v_E\tan\phi}{R_m+h} & \frac{v_N}{R_m+h} \\ 2\omega_{ie}\sin\phi + \frac{v_E\tan\phi}{R_m+h} & \frac{v_D}{R_m+h} + \frac{v_N\tan\phi}{R_m+h} & 2\omega_{ie}\cos\phi + \frac{v_E}{R_m+h} \\ -2\frac{v_N}{R_m+h} & -2\left(w_{ie}\cos\phi + \frac{v_E}{R_m+h}\right) & 0 \end{pmatrix}$$

$$J_{vp} = \begin{pmatrix} 2v_E\omega_{ie}\cos\phi - \frac{v_E^2}{(R_m+h)(\cos\phi)^2} - \omega_{ie}^2(R_m+h)\cos2\phi & 0 & \frac{v_E^2\tan\phi}{(R_m+h)^2} - \frac{v_Nv_D}{(R_m+h)^2} - \frac{\omega_{ie}^2}{2}\sin2\phi \\ 2v_N(\omega_{ie}\cos\phi - v_D\sin\phi) + \frac{v_Ev_N}{(R_m+h)(\cos\phi)^2} & 0 & -\frac{v_E}{(R_m+h)^2}(v_N\tan\phi + v_D) \\ 2v_E\omega_{ie}\sin\phi + \omega_{ie}^2(R_m+h)\sin2\phi & 0 & \frac{v_E^2+v_N^2}{(R_m+h)^2} - \frac{\omega_{ie}^2}{2}(1+\cos2\phi) \end{pmatrix}$$

where the RPY angles derivative can be obtained from the prediction model. The linearisation of the observation dynamics is simply given as

$$C = \begin{pmatrix} \boldsymbol{I}_6 & \mathbf{0}_{6\times9} \end{pmatrix} \tag{6.11}$$

while the process noise distribution matrix comes out from the linearisation as

$$D = \begin{pmatrix} R_{nb} & \mathbf{0}_3 & -R_{nb} & \mathbf{0}_3 \\ & & \mathbf{0}_{3\times12} & \\ \mathbf{0}_3 & E(\varphi,\theta) & \mathbf{0}_3 & -E(\varphi,\theta) \\ \mathbf{0}_3 & \mathbf{0}_3 & \boldsymbol{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \boldsymbol{I}_3 \end{pmatrix} \tag{6.12}$$

No other update possibilities are considered. State is corrected only when GNSS measurements come in the form of position plus speed

Chapter 7

Two-Stage INS Algorithms

The single-stage, EKF-based algorithm proposed in Chapter 6 suffers from several limitations:

- The accelerometer is used only for speed estimation. Since gravity is inherently present in the readings, the sensor may be used also as an inclinometer to get additional measurements about attitude. In this way, it would be possible to perform additional state updated at a much higher rate than GNSS can provide so to improve performance
- Velocity expressed in NED frame, meaning that it is more difficult to exploit the kinematic constraints of the train
- Complex, tightly-coupled structure

In order to overcome such limitations a double-structure algorithm has been proposed.

7.1 Overview

The full-order, two-stage model estimates the whole geodetic position, the full 3-D speed vector as seen in the body frame, the relative orientation between body and NED frames, as well as accelerometer and gyroscope biases.

The model is made up by two interconnected modules. The first one is the AHRS, while the latter is an Extended Kalman Filter for the translational navigation dynamics. The first advantage of using a double structure is simplicity because less states are propagated and the complexity is kept limited. At first, no assumptions are made about the vehicle, so any type of motion can be followed by the algorithm. Then, a reduced-order observer for the translational dynamics will be proposed in order to exploit the kinematic

constraints on the train speed, in order to improve performance. Moreover, in this work two possible AHRS are developed and tested into simulations to determine which one gives better performances and several combinations of AHRS and translational dynamics observer will be put into test.

Finally integration with maps will be developed and integrated. The full procedure will be explained on Chapter 8

7.2 EKF-based AHRS

The first proposed AHRS is based on the standard EKF structure presented in Section 4.3 and on the "Dual Structure EKF" presented in [18]. Both accelerometer and GPS will be used for state correction at different rates.

7.2.1 Mechanization equations

Roll, pitch and yaw angles are chosen for the internal attitude representation. Moreover, gyroscope bias is taken into account and estimated.

It is known that RPY angles present a singularity for $\theta = \pm\frac{\pi}{2}$, that is a 90-degree positive or negative pitch. However, since this is clearly not an operating condition for trains, the problem can be neglected.

According to equation (2.6), the AHRS mechanization equations turn out to be as follows:

$$\dot{\Theta} = \begin{pmatrix} \dot{\varphi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \begin{pmatrix} 1 & \sin(\varphi)\tan(\theta) & \cos(\varphi)\tan(\theta) \\ 0 & \cos(\varphi) & -\sin(\varphi) \\ 0 & \sin(\varphi)\sec(\theta) & \cos(\varphi)\sec(\theta) \end{pmatrix} \left(\omega_{nb}^{b} - b_g\right) \tag{7.1}$$

$$= E\left(\varphi, \theta\right)\left(\omega - \omega_{ie}^{b} - \omega_{en}^{b} - b_g\right) \tag{7.2}$$

$$\dot{b_g} = 0 \tag{7.3}$$

where $\omega_{ie}^{b} = R_{bn}\omega_{ie}^{n}$ and $\omega_{en}^{b} = R_{bn}\omega_{en}^{n}$

The available measurements are the gravity vector (as provided by the accelerometers) and the yaw angle from the GPS receiver. As already mentioned in Section 3.1, accelerometers measure the difference between true and gravity acceleration, so what can be actually read at rest or in uniform linear motion is

$$a = -g^{b} = R_{bn}\begin{pmatrix} 0 \\ 0 \\ -g \end{pmatrix}$$

that is the opposite of the gravity acceleration as seen in body axes, corresponding to a vector pointing upwards in NED frame. From the definition

of R_{bn} in equation (2.4) we have that

$$a = \begin{pmatrix} g\sin(\theta) \\ -g\sin(\varphi)\cos(\theta) \\ -g\cos(\varphi)\sin(\theta) \end{pmatrix}$$

Therefore, the final measurement vector turns out to be

$$z = \begin{pmatrix} u_x \\ u_y \\ u_z \\ \psi \end{pmatrix} = \begin{pmatrix} g\sin(\theta) \\ -g\sin(\varphi)\cos(\theta) \\ -g\cos(\varphi)\cos(\theta) \\ \psi_{GNSS} \end{pmatrix}$$

During manoeuvring, the accelerometer readings should be compensated by taking out the true acceleration and the sensor bias. Otherwise, this results in a different direction of the assumed gravity vector, thus implying errors in the estimate. However, the attitude error will remain bounded.

An estimate of the acceleration terms can be done by using the gyroscope readings, the speed estimate computed from the navigation algorithm and some guess of the vehicle's acceleration $\hat{\dot{v}}_e^b$ which may be computed by a standard state-variable filter.

Its basic structure in continuous-time is depicted in Figure 7.1. If it response is fast enough, its output $\hat{v}$ will basically follow the input v and hence an estimate of its derivative can be accessed as the value entering the integrator.

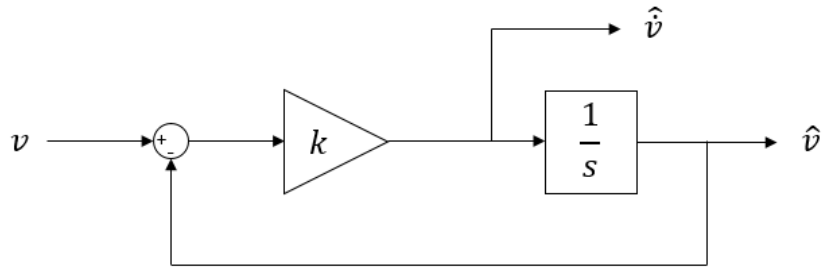

Figure 7.1: First-order state-variable filter scheme

Filter tuning should be a trade-off between fast response and limited high-frequency noise amplification. Therefore, the compensated accelerometer readings are computed as

$$u = a - b_a - \hat{\dot{v}}_e^b - (\omega - b_g + \omega_{ie}^b)_\times v_e^b \tag{7.4}$$

where a are the actual readings from the triaxial sensor.

7.2.2 EKF design for the AHRS

An Extended Kalman Filter structure is used to manage the state correction of the AHRS. The linearised model is used to propagate the covariance matrix P at each step and compute the Kalman gain K, according to Section 4.3.

Moreover, the update stage is carried out in two different ways:

- if no external heading information is available, the state is updated via the three accelerometer readings only. This update is partial in some sense, because it cannot eliminate completely the attitude error but keeps it bounded on the roll and pitch components. It is carried out after every prediction step
- when a new heading information is available, both sources are used. This update is typically performed at the rate in which GPS data arrive

Linearisation of AHRS equations

From equation (7.1) the linearisation of the AHRS process about some generic state $\begin{pmatrix}\Theta & b_g\end{pmatrix}^T$ turns out to be

$$A = \begin{pmatrix} \frac{\partial \varphi}{\partial \Theta} & \\ \frac{\partial \theta}{\partial \Theta} & E(\varphi, \theta) \\ \frac{\partial \psi}{\partial \Theta} & \\ \mathbf{0}_3 & \mathbf{0}_3 \end{pmatrix}$$

where

$$\begin{aligned} \frac{\partial \varphi}{\partial \Theta} &= \begin{pmatrix} C_\varphi T_\theta \hat{w}_y - S_\varphi T_\theta \hat{w}_z & (1+T_\theta^2)(S_\varphi \hat{w}_y - C_\varphi \hat{w}_z) & 0 \end{pmatrix} \\ \frac{\partial \theta}{\partial \Theta} &= \begin{pmatrix} -(S_\varphi \hat{w}_y + C_\varphi \hat{w}_z) & 0 & 0 \end{pmatrix} \\ \frac{\partial \psi}{\partial \Theta} &= \begin{pmatrix} SC_\theta (C_\varphi \hat{w}_y - S_\varphi \hat{w}_z) & \frac{S_\theta (S_\varphi \hat{w}_y + C_\varphi \hat{w}_z)}{C_\theta^2} & 0 \end{pmatrix} \\ \frac{\partial \Theta}{\partial b_g} &= E(\varphi, \theta) \end{aligned}$$

where $\hat{w} = w - b_g$ and $SC_\theta = \sec\theta$. It has to be pointed out that ω_{nb} is a function of the attitude as well since the Earth and NED angular rates have to be resolved in the body frame via the rotation matrix; however, since the additional terms are small in amplitude, they have been neglected in the linearisation

Concerning the "full" observation function (that is, when GNSS is available), its linearisation about the current state is

$$C = \begin{pmatrix} \frac{\partial u}{\partial \Theta} & \frac{\partial u}{\partial b_g} \\ \frac{\partial \psi_{GNSS}}{\partial \Theta} & \frac{\partial \psi}{\partial b_g} \end{pmatrix} = \begin{pmatrix} \frac{\partial u}{\partial \Theta} & \mathbf{0}_3 \\ 0 \quad 0 \quad 1 & \mathbf{0}_{1x3} \end{pmatrix}$$

that becomes in the end

$$C = \begin{pmatrix} 0 & gC_\theta & 0 & \\ -gC_\varphi C_\theta & gS_\varphi S_\theta & 0 & \mathbf{0}_3 \\ gS_\varphi S_\theta & -gC_\varphi C_\theta & 0 & \\ 0 & 0 & 1 & \mathbf{0}_{1x3} \end{pmatrix}$$

When GNSS heading is not available, matrix C reduces to the first three rows with no further modification. Finally, matrix D describes how the process noise distributes across the states and turns out to be the following function of the state:

$$D = \begin{pmatrix} E(\varphi, \theta) & \mathbf{0}_3 \\ \mathbf{0}_3 & \boldsymbol{I}_3 \end{pmatrix}$$

7.3 Non linear AHRS

The second AHRS algorithm developed has its core based on a complementary filter on the group $SO(3)$ and adapted to the available measurements of the current framework. The internal attitude representation is in quaternions and gyro biases are estimated as well.

7.3.1 Mechanization equations

As in the EKF-based AHRS proposed in Section 7.2, the relative orientation between NED and body frame is computed. From Equation (2.11) and Equation (7.1) the quaternion and gyro bias derivative simply write down as

$$\dot{q}_{nb} = \frac{1}{2} \begin{pmatrix} 0 \\ \omega - b_g - \omega_{ie}^b - \omega_{en}^b \end{pmatrix} \circ q_{nb} = \frac{1}{2} \begin{pmatrix} 0 \\ \omega_{nb}^b \end{pmatrix} \circ q_{nb}$$
$$\dot{b_g} = 0$$

7.3.2 Non linear observer design

The presented non linear observer (NLO) is the so-called *Explicit Complementary Filter* on $SO(3)$ [17]. It requires at least two distinct reference directions, in order to guarantee complete state convergence. Should there be only one measurement available, convergence of at least one of the rotation angles is still achieved.

The vectors' directions must be known in some frame of reference[1] (which is going to be the NED in this case) and their value in body frame has to be accessible from some measurements. Then, their readings have to be compared with the estimates coming from the current attitude estimation to perform correction.

Let v_{0i} be the known vectors as seen in the NED frame and $v_i = R_{bn}v_{0i}$ their value in the body frame. Let $\hat{v_i} = \hat{R}_{bn}v_{0i}$ the estimates of v_i. An output injection term can be defined as

$$\omega_m = \sum_{i=1}^{n} k_i(v_i \times \hat{v_i}) \tag{7.5}$$

where $\times$ denotes the usual vector product. ω_m can be seen as an angular rate trying to bring the estimated directions onto the corresponding measurements by adjusting the attitude. Then, the observer equations are defined as[17, Appendix B]

$$\dot{\hat{q}}_{nb} = \frac{1}{2}\begin{pmatrix} 0 \\ \omega - b_g - \omega_{ie}^b - \omega_{en}^b + k_P\omega_m \end{pmatrix} \circ \hat{q_{nb}}$$
$$\dot{\hat{b}}_g = -k_I\omega_m$$

where k_i, k_P and k_I are positive tuning gains. The choice of k_P and k_I influences the dynamic behaviour of the filter in terms of response, overshoot and so forth. The k_is should be chosen according to the relative degree of confidence towards each measurement, hence considerations about sensors noise may come into play. This filter structure is also modular as it can be expanded if additional direction-providing sensors are used (e.g. magnetometers) and information can be properly merged by tuning the k_is.

It can be proven the explicit complementary filter is locally exponentially stable and converges for almost all initial conditions to the true attitude and gyro bias trajectories. This results are stronger than those from the EKF, which are indeed just local and hold inside some domain around the linearisation point. This observer structure is similar to a sort of non linear

[1] They do not necessarily have to be constant vectors

PI controller, where the integral term (corresponding to the gyroscope bias observer) is used to nullify the steady-state error.

In this framework, the reference vectors are given as

$$v_{01} = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}^T$$
$$v_{02} = \begin{pmatrix} 0 & 0 & -g \end{pmatrix}^T$$

that is the geographical North and the opposite of the gravity vector.

A direct measure of v_1 is not available because of the lack of a magnetometer; therefore, it is reconstructed via the GNSS heading information by means of a "fictitious" rotation matrix

$$v_1 = R_{bn}(\hat{\varphi}, \hat{\theta}, \psi_{GNSS}) \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

In order to adapt the NLO to the non-continuous information from GNSS, a fictitious heading value is instead used and updated every time ψ_{GNSS} is available. An additional single-state observer is used to recreate the fictitious measurement by using the last row of Equation (7.1)

v_2 is obtained by the raw accelerometer readings when the body is at rest, or by the compensated ones otherwise, computed again according to Equation (7.4).

7.4 Full-order translational navigation EKF

The proposed INS algorithm for the navigation dynamics is realised by means of an Extended Kalman Filter. It has been chosen to express the speed state in the body-fixed coordinate system. This has the advantage of the accelerometers entering linearly in the model since they measure accelerations in the same reference.

7.4.1 Mechanization of translation equations

Starting from $v_e^b = R_{bn} v_e^n$, one can write the speed as seen in the body frame as

$$\begin{aligned} \dot{v_e^b} &= R_{bn} \dot{v_e^n} + R_{bn} (w_{bn}^n)_\times v_e^n \\ &= R_{bn} \left(R_{nb}(u - b_a) + g^n - (w_{en}^n + 2w_{ie}^n)_\times v_e^n \right) + R_{bn} (w_{bn}^n)_\times v_e^n \\ &= u - b_a + R_{bn} g^n - R_{bn} (w_{en}^n + 2w_{ie}^n - w_{bn}^n)_\times v_e^n \\ &= u - b_a + R_{bn} g^n - (w_{en}^b + 2w_{ie}^b - w_{bn}^b)_\times v_e^b \end{aligned}$$

and by remembering $w_{bn} = w_{in} - w_{ib} = w_{ie} + w_{en} - w_{ib}$[2], one eventually obtains

$$\dot{v_e^b} = a - b_a + R_{bn} g^n - (w_{ib}^b + w_{ie}^b)_\times v_e^b \tag{7.6}$$

Hence, with reference also to Equation (2.32), the overall navigation dynamics write down are as follows:

$$\dot{p} = \begin{pmatrix} \dot{\phi} \\ \dot{\lambda} \\ \dot{h} \end{pmatrix} = \begin{pmatrix} \frac{1}{R_m + h} & 0 & 0 \\ 0 & \frac{1}{(R_m+h)\cos(\phi)} & 0 \\ 0 & 0 & -1 \end{pmatrix} R_{nb} v_e^b = T(\phi, h) R_{nb} v_e^b$$

$$\dot{v_e^b} = a - b_a + R_{bn} g^n - (\omega + \omega_{ie}^b)_\times v_e^b$$

$$\dot{b_a} = 0$$

where $\omega_{ie}^b = R_{bn} \omega_{ie}^n$ according to equation (6.4) and by remembering that the gyroscope measures the angular rate with respect to an inertial frame. The current attitude estimate must be retrieved from some AHRS algorithm, for example one of the two previously presented in this work. Refer to Figure 9.3 for the complete interconnection scheme of the two stages.

The position is expressed directly in geodetic coordinates for easier use. Alternatively, additional conversions would be required in order to manage GNSS data. Once again, a spherical Earth model is used for sake of simplicity and local gravity is assumed constant everywhere, while the Earth rotation rate is not neglected.

The available measurements are GPS position and speed as seen in the body frame resulting in the trivial observation model

$$z = \begin{pmatrix} p \\ v_e^b \end{pmatrix}$$

Since GPS speed natively comes in NED frame, it has to be rotated via the estimate attitude provided by the EKF-based AHRS, which might be another source of estimation errors if the attitude is not precisely estimated.

The common assumption for GPS velocity being available in all its three components[22, 10, 25] is done for this algorithm, as well. Alternatively, a state-variable filter can be employed again to get an estimate of the "down" component from elevation. The other two are commonly given in the form of modulo and course, yet conversion is straightforward via the equations

$$v_N = v_{GPS} \cos \psi$$
$$v_E = v_{GPS} \sin \psi$$

[2]This actually holds in whatever reference frame

7.4.2 EKF design for the translation equations

Similarly to the AHRS, the complete non linear model is used to predict the future position and speed. This is particularly important because the update rate of this part of the algorithm is relatively low, therefore it is desirable to reduce drift caused by approximate equations as much as possible.

Linearisation of translation equations

The matrices for the EKF are obtained with the same approach used in Section 7.2 for the AHRS. The structure of the linearised process function A is as follows:

$$A = \begin{pmatrix} J_{pp} & J_{pv} & \mathbf{0}_3 \\ J_{vp} & J_{vv} & -\boldsymbol{I}_3 \\ & \mathbf{0}_{3x9} & \end{pmatrix} \tag{7.7}$$

where

$$J_{pp} = \frac{\partial P}{\partial P} = \begin{pmatrix} 0 & 0 & -\frac{v_N}{(R_m+h)^2} \\ v_E \frac{\sin(\varphi)}{((R_m+h)\cos(\varphi))^2} & 0 & -\frac{v_E}{(R_m+h)^2\cos(\varphi)} \\ 0 & 0 & 0 \end{pmatrix}$$

$$J_{pv} = \frac{\partial P}{\partial v_e^b} = T(\phi, h) R_{nb}$$

$$J_{vp} = \frac{\partial v_e^b}{\partial P} = (v_e^b)_\times R_{bn} \begin{pmatrix} -\omega_{ie}\sin(\varphi) & 0 & 0 \\ 0 & 0 & 0 \\ -\omega_{ie}\cos(\varphi) & 0 & 0 \end{pmatrix}$$

$$J_{vv} = \frac{\partial v_e^b}{\partial v_e^b} = -(\omega + \omega_{ie}^b)_\times$$

$$\begin{pmatrix} v_N \\ v_E \\ v_D \end{pmatrix} = v_e^n = R_{nb} v_e^b$$

Concerning matrices C and D, they are obtained in the same way as above and turn out to be

$$C = \begin{pmatrix} \boldsymbol{I}_6 & \mathbf{0}_{6x3} \end{pmatrix}$$

$$D = \begin{pmatrix} \mathbf{0}_3 & \mathbf{0}_3 \\ \boldsymbol{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & -\boldsymbol{I}_3 \end{pmatrix}$$

7.5 Reduced-order translational navigation EKF

This second model is a reduced version of the previous one. The main underlying assumption is that trains can only move along the direction they are heading towards, so the lateral velocity components will structurally be zero. Speed estimation along Y and Z body axes is therefore an unnecessary task and possibly a source of errors during navigation. No state reduction on the position has been carried out.

Another advantage of using a reduced model is the possibility to directly exploit odometry speed to correct velocity at a higher rate. In this framework, odometry is assumed to be always available at the same rate of MEMS sensors and is constantly used for correction. When GNSS data arrive, both position and speed are used to update the EKF state estimate.

The reduced-order model can be obtained in a straightforward way from the full-order one by taking out the differential equations for body speed and accelerometer bias along directions Y and Z. Therefore, the mechanisation equations simply write down as

$$\begin{aligned} \dot{P} &= T(\phi, h) R_{nb} \begin{pmatrix} v_{bx} & 0 & 0 \end{pmatrix}^T \\ \dot{v}_{bx} &= a_x - b_{ax} - g \sin(\theta) \\ \dot{b}_{ax} &= 0 \end{aligned}$$

where θ is the pitch angle. Notice that centrifugal effects do not appear in the equation since they are always orthogonal to the vehicle speed vector. The equations result in a reduced-order observer with 5 states.

The EKF matrices can be easily obtained from Equation (7.7) and the following ones by taking out the rows and columns of the unaccounted states.

Chapter 8

Map integration algorithm

Another important information that can be used are maps. They constrain the position of the train to a given subset of the 3D space. They can also provide information about the expected speed direction. According to their level of accuracy and completeness, maps definitely have bigger availability than satellite-based systems, hence they can be used at a higher frequency, for longer time periods and also during GNSS outages in order to limit the estimation drift.

The way maps are integrated in the navigation algorithms of this work is as additional position and speed measurements for the observers. In this way, no re-projection or algebraic constraint on the state of the estimators is introduced. In this way, complexity is kept limited and the overall algorithm is modular, meaning that GPS measurements and maps data are seen in the very same way to each one of the algorithms proposed in Chapters 6 and 7, which indeed require very few modifications. Filters can be properly tuned by simply modifying the R matrix (or other parameters, in case of non EKF-based structures) with values describing the confidence that is given to each different source of information.

8.1 Map data model

Map data are assumed to be organized as a simple 2-D matrix whose structure is depicted in Table 8.1 Rows are assumed to be already sorted by increasing arc length values in order to simplify the further steps of the algorithm. No particular assumptions are made on the geometry nor on the size of the map vector, while it has been assumed equal spacing between consecutive points.

Arc length (m)	Latitude (rad)	Longitude (rad)	Height (m)
x_1	ϕ_1	λ_1	h_1
x_2	ϕ_2	λ_2	h_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$
x_N	ϕ_N	λ_N	h_N

Table 8.1: Map data structure

8.1.1 Uncertain maps

An "uncertain" version of the maps has been prepared to evaluate the performance of the algorithm in presence of errors. This choice reflects a possible scenario in which, for example, a GPS device is used to record the train path for further use, therefore some inaccuracy is introduced by the measurements.

In this work, maps have been rendered uncertain by means of some additive noise to the ideal trajectory. It has been added in the direction normal to the ideal train path (including the vertical component). Results related to the use of uncertain (or "noisy") maps rather than ideal ones can be found in Chapter 10, as well.

8.2 Integration algorithm

The main steps of the map injection algorithm are the following:

1. Starting from the index of the last iteration solve a standard optimization problem. The cost function is the distance between each point of the map and the currently estimated position plus some user-tunable "feedforward" term, meant to compensate for computation delays. To this purpose, Matlab function `fminbnd` has been used and the problem has been properly set up for sake of computational efficiency

2. Once the minimum is found, the value is directly as a measurement for the filter without further elaboration

3. The velocity is computed by means of the finite difference [11] centred on the position of the closest map point. The vector is then scaled so that its norm matches the speed from odometry, which is assumed to be always available. Should the closest map point be close to the beginning or the end of an array, and thus have not enough elements before or after, the forward or backward finite differences are used, respectively

4. Store the array index as the starting point for the next algorithm iteration

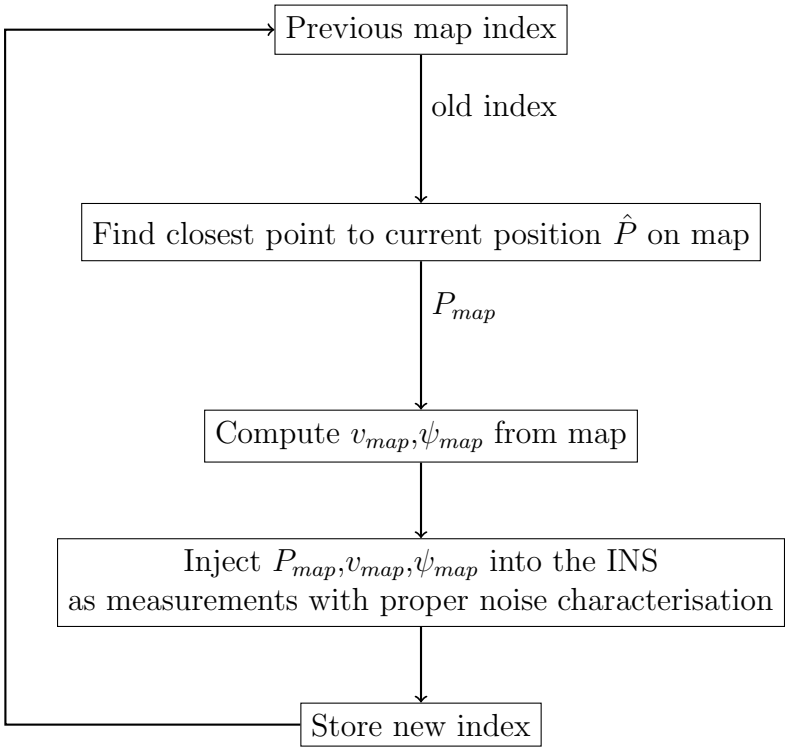

Figure 8.1: Map integration algorithm work flow

In order to keep the computational cost limited some precautions have been taken:

- The optimization problem is limited within a proper range of the last detected index. The interval is centred on the starting index and its size is proportional to the speed from odometry, plus some fixed value. Parameters are empirically tuned off-line to ensure the expected solution to fall into the search range.
- The maximum number of iterations are capped to an empirically-chosen threshold
- The map correction algorithm is run at a limited rate.

Notice that the arc length is not directly used in the algorithm while an additional measure is employed, that is the speed from the tachometer. Distance as measured from odometry is ignored here, as well as elsewhere

in the proposed inertial navigation systems. The feed-forward term on the estimated coordinates is used to project the current position a bit further to compensate for the motion and possibly choose a more suitable point for the update. It is not a key element of the algorithm but adds an additional degree of freedom for tuning.

Code of the Matlab function implementing the algorithm can be found in Appendix A.

Chapter 9

Simulation Set-up

In this chapter the implementation of the above-presented algorithms via different simulation set-ups is discussed. Simulation results are then discussed in Chapter 10.

Models have been implemented in the Matlab-Simulink environment. The main functional blocks of each simulation are the plant model, the INS algorithm and the GPS availability model. The last one's purpose is to simulate the arrival of valid GNSS data or absence of satellite signal according to a user-defined time schedule, in order to test the algorithm under several conditions of external measurements. The actual map update rate during periods of availability is set by the algorithm itself.

The general scheme in Figure 9.1 is related to all the developed algorithms.

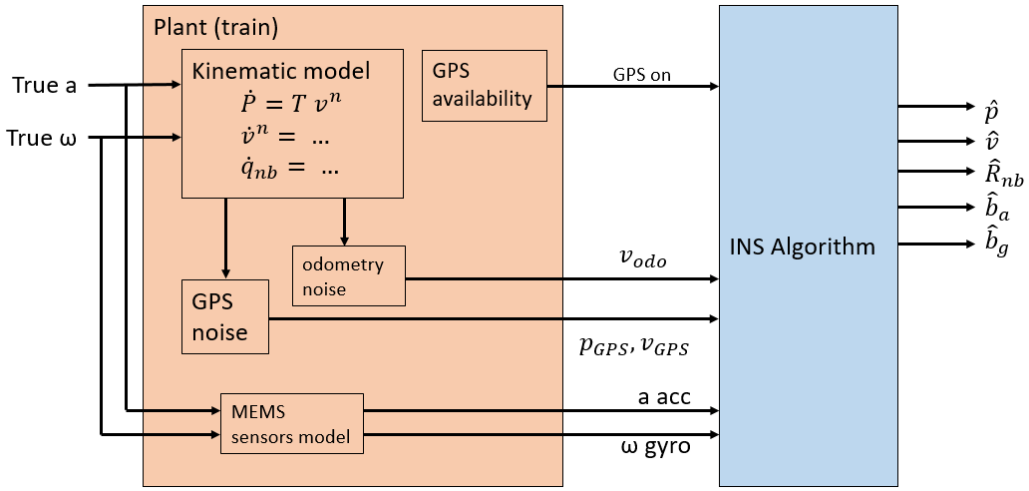

Figure 9.1: General simulation scheme

9.1 Plant model

The physical plant was recreated in Simulink in continuous time. Geographical coordinates, NED frame speed and quaternion representation have been used in the model according to Equations (2.32), (6.2) and (6.6). GPS position and speed are output and noise is added to the true values, if required for the simulation. The latter is assumed to be available in all its three components as well in as modulo plus course.

In addition, MEMS sensors model is present, whose equations are reported in Equations (3.1) and (3.2). As mentioned in 3, their bias is modelled as a constant additive disturbance. Odometry speed noise model is given in Equation (3.3).

All the inertial navigation algorithms receive measurements from the plant subsystem and output to the user all their estimates. As an additional step, pre-filtering of accelerometers and gyroscope is used. This is a common practice in inertial navigation in order to end up with better signals from the sensors. The two filters are designed as simple low-pass filters whose parameters are collected in Table 10.1.

9.2 Single-Stage INS algorithm

In order to validate the algorithm structure, simulations for the single-stage algorithm from Chapter 6 have been carried out on a continuous-time hybrid model to avoid discretization-related errors. The core of the single-state INS algorithm does not differ much from a standard EKF structure. As it can be seen from Figure 9.2, EKF update is triggered by new available GNSS data properly correcting the internal state estimate; the standard prediction equations are used otherwise. Moreover, heading from GNSS is not directly used and speed from odometry is ignored.

The prediction model is based on Equations (2.32), (6.2) and (6.6). Proper transformations similar to those presented in Chapter 2 and that can be found in [10, 25] are used to convert the internal quaternion representation into RPY angles in order to use the KF filter equations of Chapter 6. The *a posteriori* RPY angles estimate is then converted back to re-initialise the integrators. Unlike the double-state formulations, it is not necessary to further elaborate GPS measurements since the internal speed representation is in NED frame.

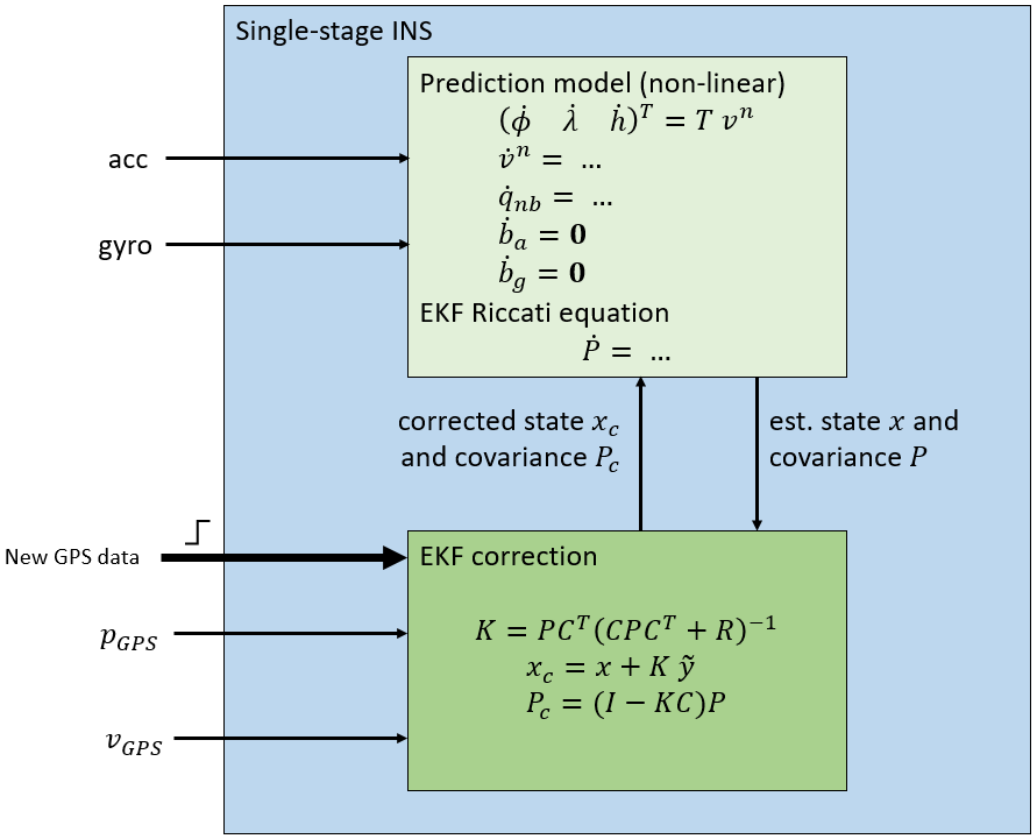

Figure 9.2: Single-stage INS scheme

9.3 Double-Stage INS algorithms

The algorithms that can be used to build several variation of double-stage inertial navigation algorithms have been directly implemented in discrete-time. A standard forward Euler discretization has been used for the conversion from the continuous-time formulation, while the exact expression from Equation (2.13) has been used for improved accuracy on quaternions[1]. Figure 9.3 shows the basic scheme of the tested variants, which have been made up as

1. EKF-based AHRS + Full-order translation EKF
2. EKF-based AHRS + Reduced-order translation EKF
3. NLO-based AHRS + Full-order translation EKF
4. NLO-based AHRS + Reduced-order translation EKF

It is important to notice the standard interconnection between the two systems and the estimates that are exchanges, as well as the map injection algorithm discussed in Chapter 8, which takes position estimate and odometry speed as inputs and outputs geographical coordinates, 3D velocity and heading to both modules in the form of measurements. GNSS update is activated by the external availability model, while map-based update is triggered by an internally-generated signal at a user-chosen rate. Such signal can be also completely deactivated to evaluate algorithm performance in different conditions. Results obtained with several combinations of GNSS and maps availability are presented in Chapter 10, too.

As remarked in Section 7.4, velocity measurements are provided to the translation EKF in NED coordinates and rotated via the estimated attitude $\hat{R}_{bn}$ coming from the used AHRS algorithm. Whenever the reduced-order model from Section 7.5 is used, then odometry speed is used also in the functional block as well as the "reduced" model equations.

The stages' structure is the same of the EKF presented in the single-state algorithm in Figure 9.2, with the addition of the state-variable filter of Figure 7.1 - implemented in discrete-time as well - for computing the compensated accelerations for the AHRS.

Non-linear observer

The NLO-based AHRS structure is quite different from the other stages. It is based on the equations presented in Section 7.3 and represented in Figure 9.4.

[1] They are actually implemented in the NLO-based AHRS only

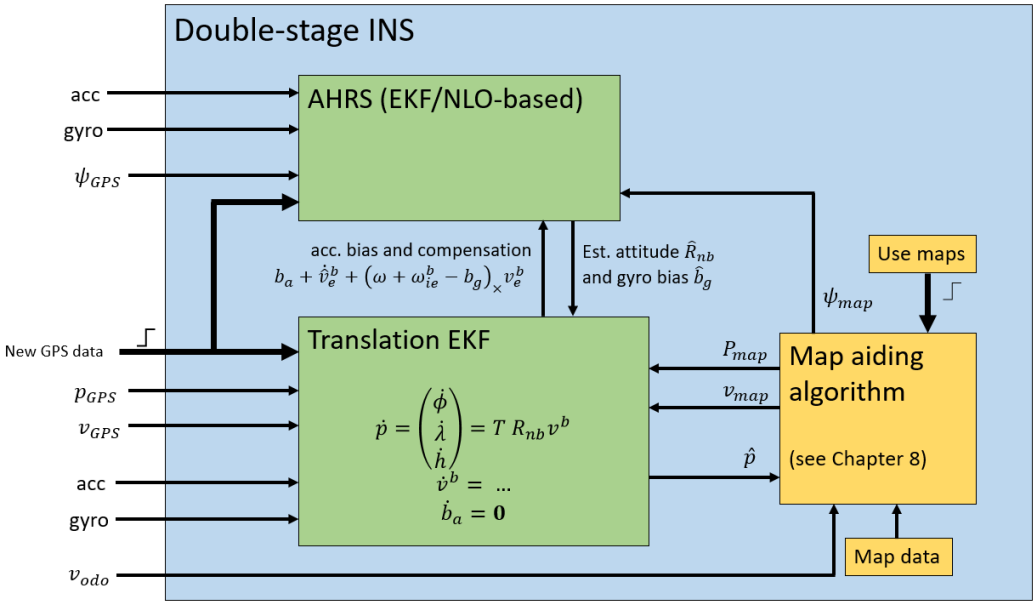

Figure 9.3: Double-stage INS scheme

The "Non linear observer" block implements the dynamical part of the filter performing attitude and gyroscope bias estimation with the PI-like control action performed by ω_m, which is built according to Equation (7.5) by comparing the estimated and measured reference vectors.

Measurements v_1 and v_2 correspond to the true North and local gravity vectors and are generated by the third functional block. The additional single-state observer used to generate "fictitious" values of v_1 between new GNSS or maps samples is also present there. Eventually, just like the Extended Kalman Filter-based structure, the accelerometer compensation term from the translational EKF is an additional input and the estimate attitude and gyroscope bias are outputs to be fed back to the other stage for position, velocity and accelerometer bias prediction.

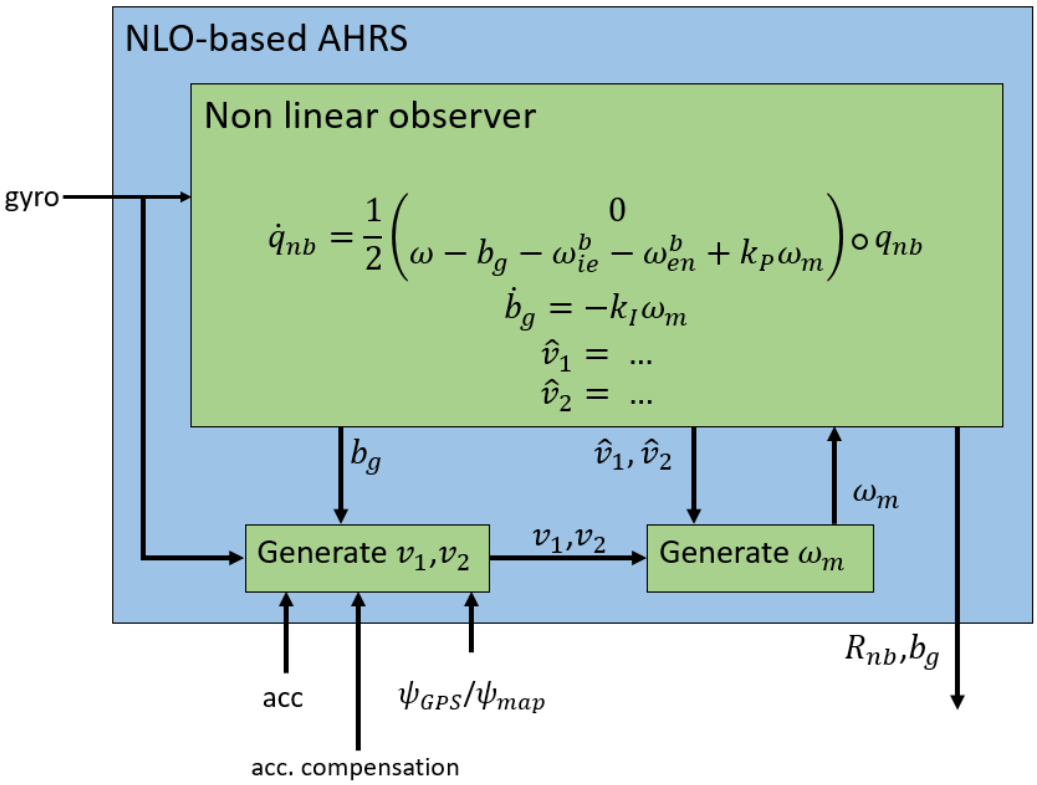

Figure 9.4: NLO-based AHRS scheme

Chapter 10

Simulation Results

The most relevant results from the simulation models described in Chapter 9 are reported in thie following sections. Several combinations of algorithms have been used in order to understand the different performance levels under ideal and more realistic conditions. Map integration algorithm from Chapter 8 has been also tested. Some of the simulation parameters are contained in Table 10.1

Parameter	Value
GPS data rate	1 Hz
Map-based update rate	1 Hz
Accelerometers, gyroscopes and odometry data rate	100 Hz
INS update frequency[1]	1000 Hz
Distance between map points	5m along arc length
Accelerometer filter bandwidth	10 Hz
Gyroscope filter bandwidth	5 Hz
Accelerometer noise power	10^{-3} $(m/s^2)^2/Hz$
Gyroscope noise power	10^{-4} $(rad/s)^2/Hz$
Odometry noise power	10^{-3} $(m/s)^2/Hz$

Table 10.1: Simulation parameters for all models

The choice of filters' bandwidth has been empirical based on previous simulations results and [18] in order to search for a trade-off between output accuracy and reduced noise in the slow-dynamics framework of trains.

[1] Only for algorithms already implemented in discrete-time, i.e. all those developed as two stages

10.1 Single-Stage INS algorithm

The simulation was performed at a constant speed of 10 m/s towards East and constant attitude without sensors and GPS noise and sensors biases. The single-stage EKF was perfectly initialized speed and attitude and run without any bias estimation, while a small position error was introduced. GPS update occurs after 5 seconds since simulation start without no further outage. Simulations results are shown in Figures 10.1 to 10.4

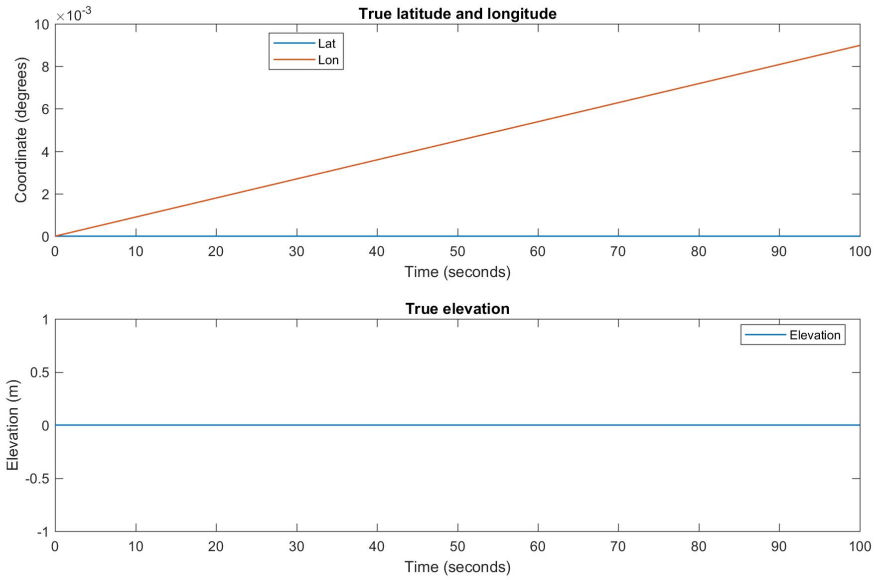

Figure 10.1: True position, ideal conditions

The peak that can be noticed in the down velocity error at 5 seconds is due to the GPS update that "reveals" the elevation error to the filter. From that moment on, the estimate position and speed tend to the true values with a residual error of about 2 meters along North and East[1], which is due to the tuning of the EKF measurement matrix. The following tests were performed with a smaller R matrix and resulted in lower steady-state position errors, as expected.

[1] An angle of 10^{-4} degrees corresponds to about 11.1 meters along any direction at the Equator, since parallel circles decrease in size as latitude's modulo increases

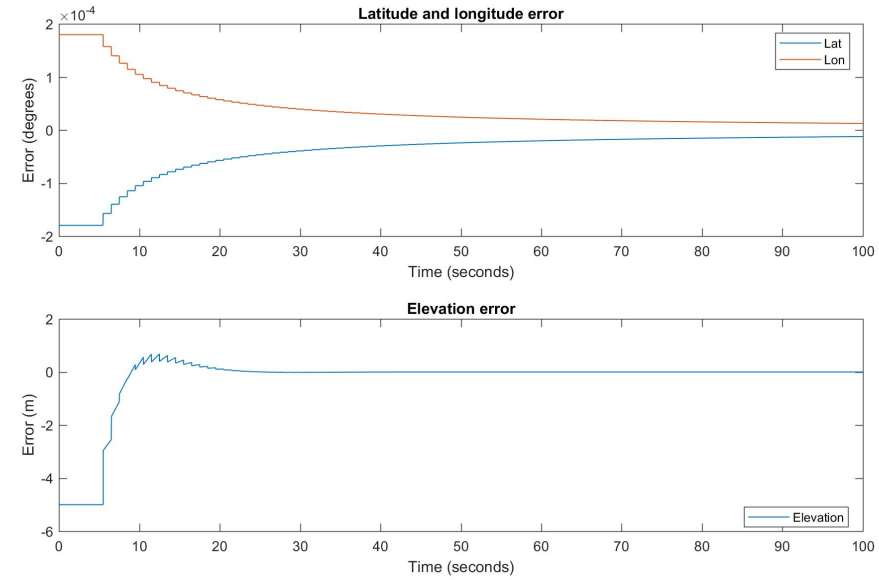

Figure 10.2: Position error, ideal conditions

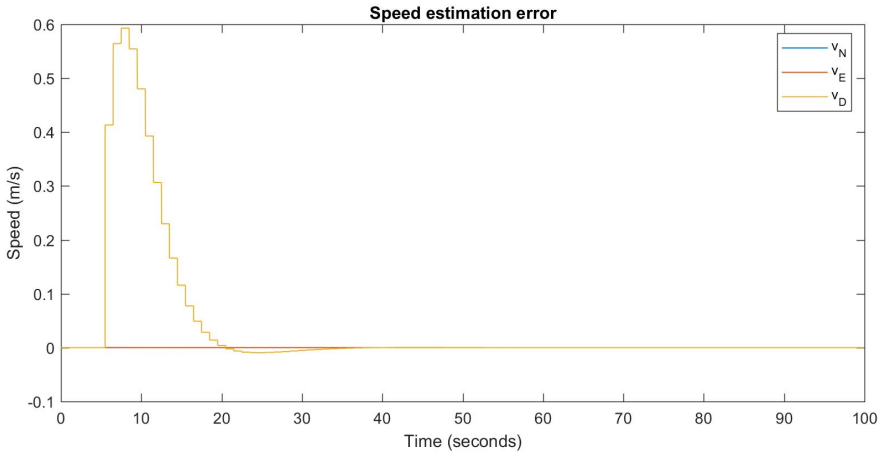

Figure 10.3: Velocity error, ideal conditions

With sensor bias

The following results are related to the same initial condition and trajectory of the first test with the only addition of a fixed accelerometer and gyroscope bias to the readings. Bias estimation has been disabled in this run in order to test the algorithm's robustness to such disturbances e.g. in case of wrong estimation. Maximum bias amplitude along each direction is 0.1 m/s^2 for

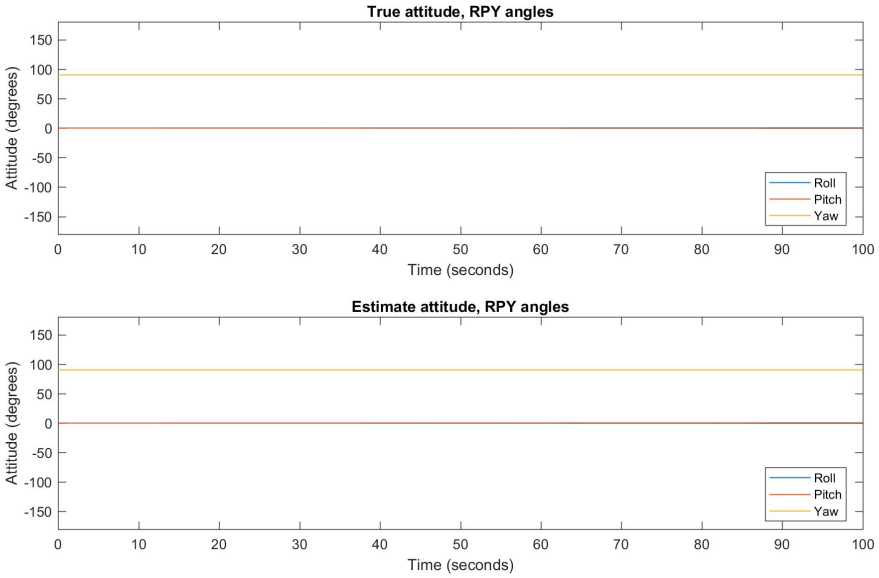

Figure 10.4: Attitude, RPY angles, ideal conditions

accelerometers and 0.1 rad/s for gyroscopes.

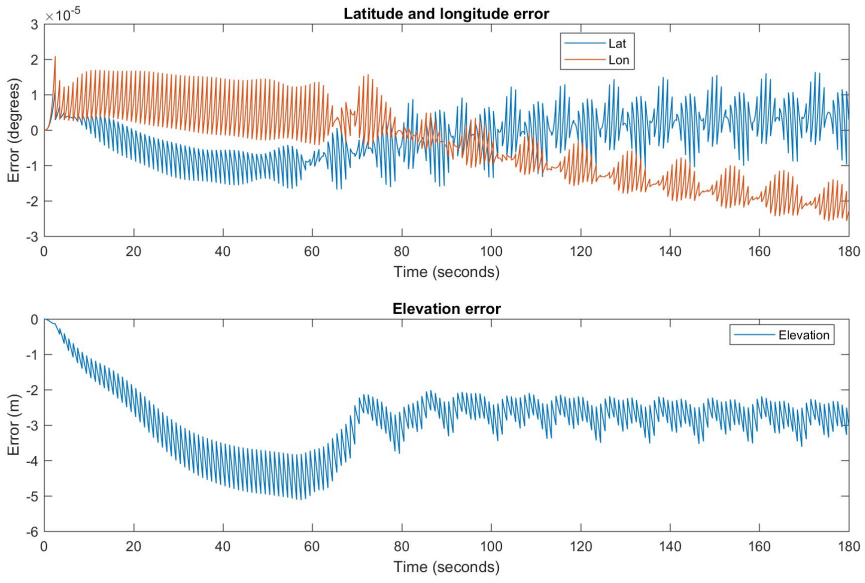

Figure 10.5: Position error, MEMS bias only

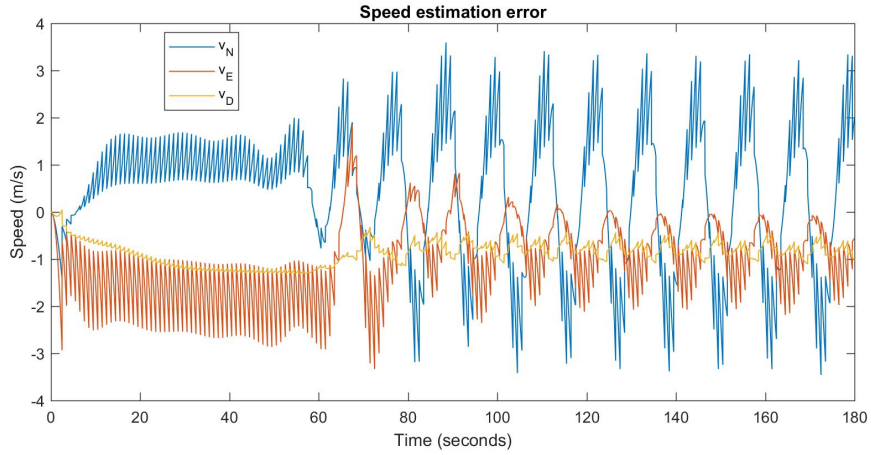

Figure 10.6: Velocity error, MEMS bias only

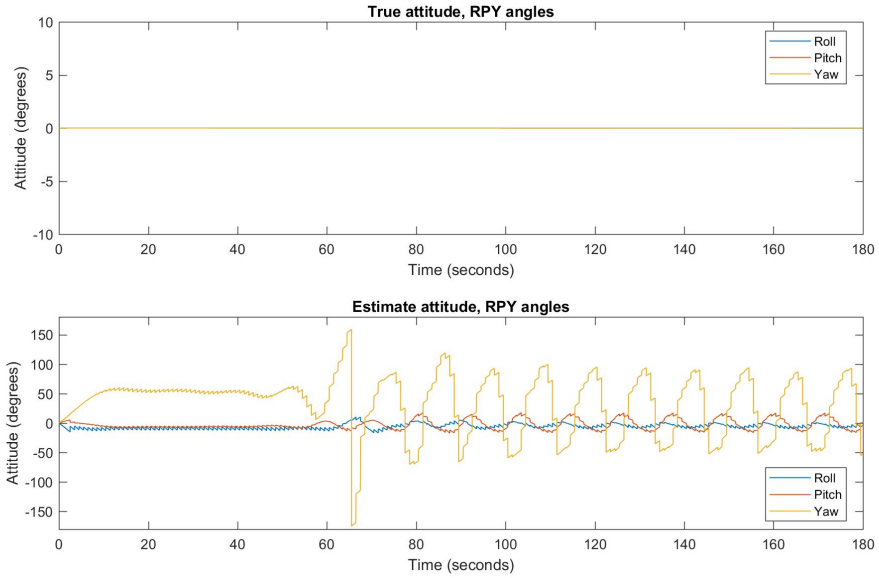

Figure 10.7: Attitude, RPY angles, MEMS bias only

Figures 10.5 and 10.6 show how the position and speed error (caused by wrong accelerometer readings) is kept contained by GPS updates. However, the EKF could not completely correct for the attitude error induced by the gyroscope bias, as seen from Figure 10.7, which results unbounded on the yaw component. Attitude errors are one of the most relevant source of errors during dead reckoning, hence it is crucial to keep them as low as possible.

Another simulation has been carried out with the same initial values and

motion type, its results being presented in Figures 10.8 to 10.10. Both accelerometer and gyroscope bias are estimated and compensated in this run. It can be appreciated from Figure 10.8 that only two gyroscope bias converge to the true value, while the Z-axis estimation reaches a wrong value without anyway becoming unstable. This clearly has the effect of increasing the attitude error, as can be seen in Figure 10.10, similarly to the previous simulation. This implies big speed and positioning errors during *dead reckoning*. On the other hand, position and speed estimation still converge to the true values. Limited convergence is achieved for accelerometer bias, too.

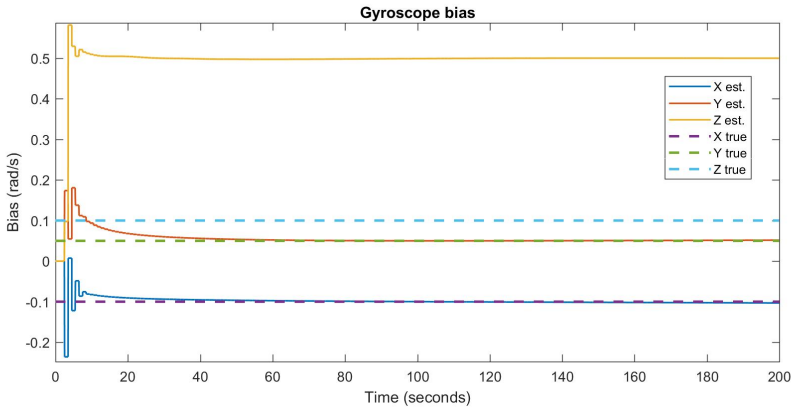

Figure 10.8: True and estimated gyroscope bias, MEMS bias plus estimation

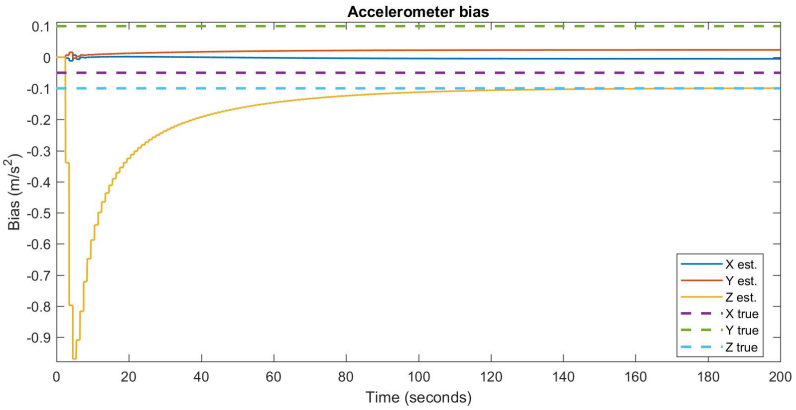

Figure 10.9: True and estimated accelerometer bias, MEMS bias plus estimation

As it can be seen from all the above simulations, the single-stage algorithm shows partial convergence towards the true values. As outlined also in the

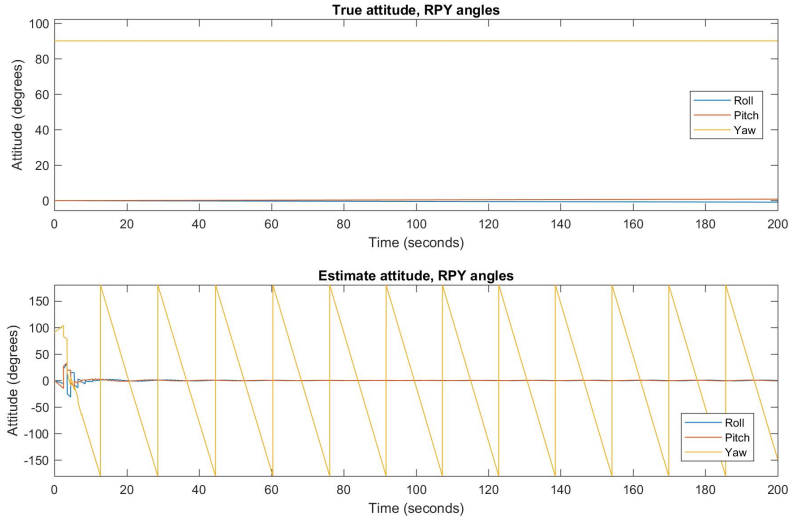

Figure 10.10: Attitude, MEMS bias plus estimation

introduction of Chapter 7, the full set of informations given by the sensors is not completely exploited. In particular, the accelerometer is used explicitly only in the speed dynamics, while it can be used also to get an estimate of the attitude and correct it at a higher rate than GNSS.

Because of the convergence issues, no test with map integration has been carried out for two main reasons: first, map information has the same structure as GNSS data, so observability is left unchanged; secondly, injected information cannot be more accurate than GNSS since they are only based on proprioceptive sensors and their action can also bring the system away from a correct estimate (see also the "ringing" effect documented in the Section 10.2 as well as Figure 10.13 for an insight on the phenomenon).

10.2 Double-Stage INS algorithms

The simulation results related to the two-stages INS algorithm are reported in this section. Four variations have been tested, according to the type of AHRS (EKF-based or non-linear estimator) and the order of the navigation EKF (full or reduced). The map integration algorithm described in Chapter 8 is implemented on all the four variants.

Each simulation has been run for 200 seconds. The benchmark path is composed in this way:

1. The train is stopped for the first 35 seconds

2. 25 seconds of constant acceleration at $1\frac{m}{s^2}$

3. After reaching the speed of $25\frac{m}{s}$ at $T = 60$ seconds, the train turns counter-clockwise to perform a circle of radius about 600 meters

10.2.1 Double-EKF, full-order algorithm

Without white noise

In the following test all measures are assumed to be noise-free, including maps. A small bias is added to all the three gyroscope axes and assumed to be partially known at the beginning, while accelerometer bias is absent and not estimated. Moreover, a period of GPS and maps outages has been modelled to determine quantitatively how the amplitude of the estimation error grows with time. Figure 10.11 shows how the outages have been arranged. Therefore, there are 30 seconds of "complete" dead reckoning, GPS is un-

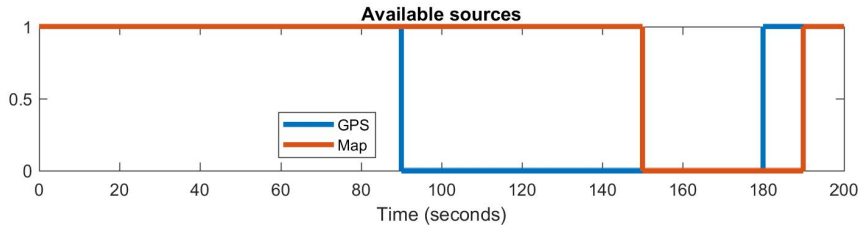

Figure 10.11: GPS and maps availability during the simulations

available for 90 seconds and one minute of navigation is based only on maps. These availability structure is shared among all the tests of this subsection, unless otherwise specified.

As shown in Figures 10.12 and 10.13 the position estimation accuracy is very high and contained within about 4 meters throughout *dead reckoning*. The "ringing" effect on the error is due to the relatively high map granularity of 5 meters and to the consequent, unavoidable discrepancy with GPS readings; as a matter of fact, this occurs only in the periods where GPS and maps are used jointly, that is in the first 90 seconds and in the last ten. Moreover, its "shape" may change but the maximum amplitude is limited and related to map density. Hence, because of the nature of this effect, using map aiding at a higher frequency will not cancel it out and coarse granularity will worsen the estimation.

Ringing amplitude has already been reduced to these values by using a bigger R matrix when the map-based update occurs, which is also quite desirable for the use of uncertain maps. In addition, using a denser map can

further improve the situation. Another solution is to keep the algorithm run continuously, in order to keep it up to date, but to inject corrections either only when GPS is not available or at a lower frequency.

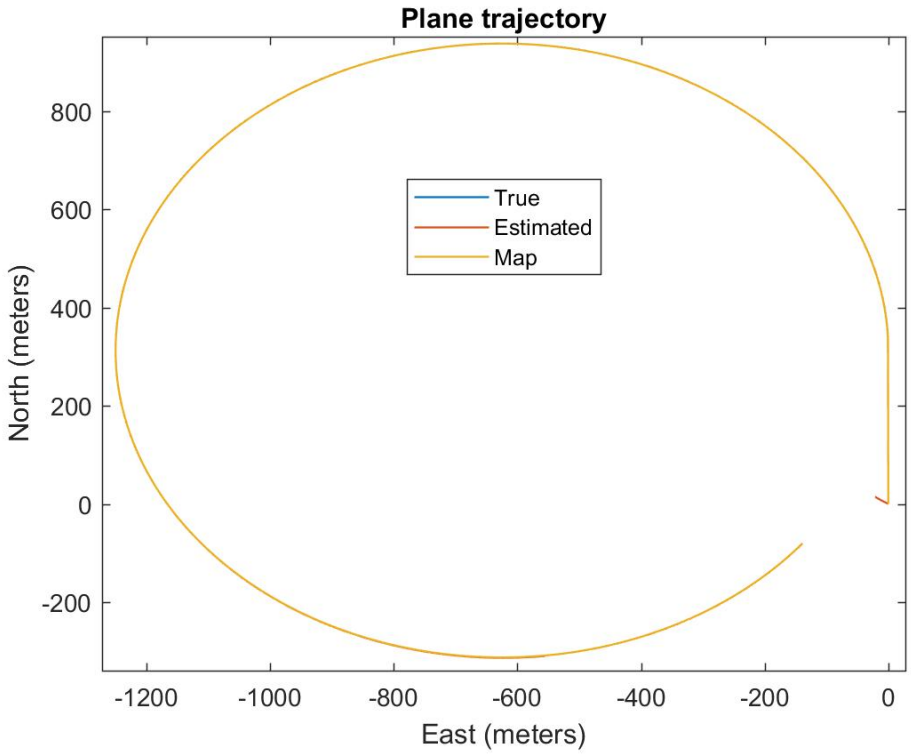

Figure 10.12: True and estimated trajectory, no noise

Height estimation is also accurate and prone to the very same ringing effect. The maximum error reached during dead reckoning is less than 2 meters and mainly related to a (small) attitude error in the pitch component.

Speed estimation is also accurate. The error along the Z component is due to the sudden change of motion and the subsequent attitude error, related to the speed of the state-variable filter compensating the accelerometer readings. A proof of this statement can be seen from Figure 10.16

Roll and pitch errors are influenced by accelerometers compensation. In case they were not present, a constant error would be present in the attitude, with negative effects during dead reckoning, especially along the "pitch" component. It is also possible to notice how the yaw error decreases progressively with time, because of the gyroscope bias estimation that gradually reaches the true values. This is an important aspect since the algorithm will be

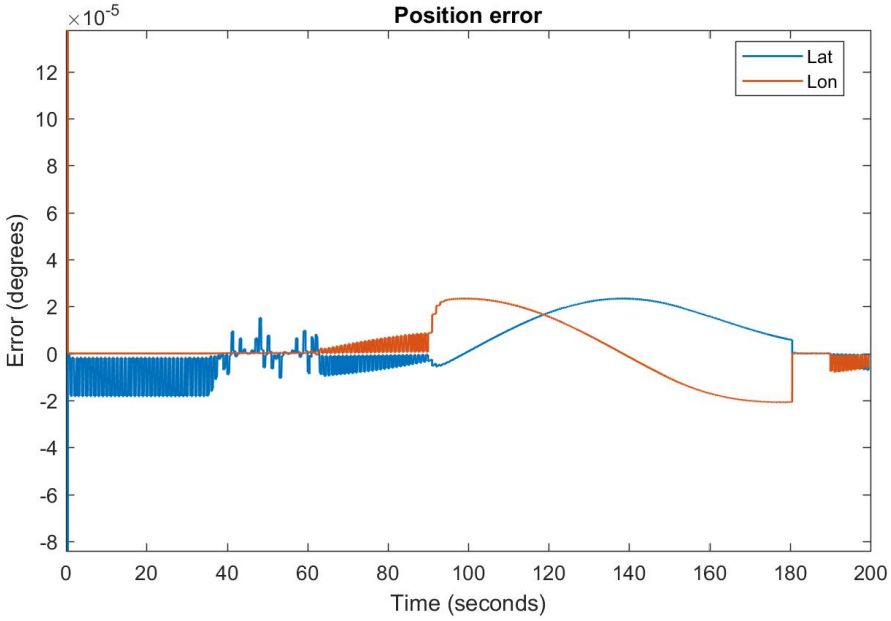

Figure 10.13: Position error in degrees, no noise

better initialised when dead reckoning will start and drift will be contained.

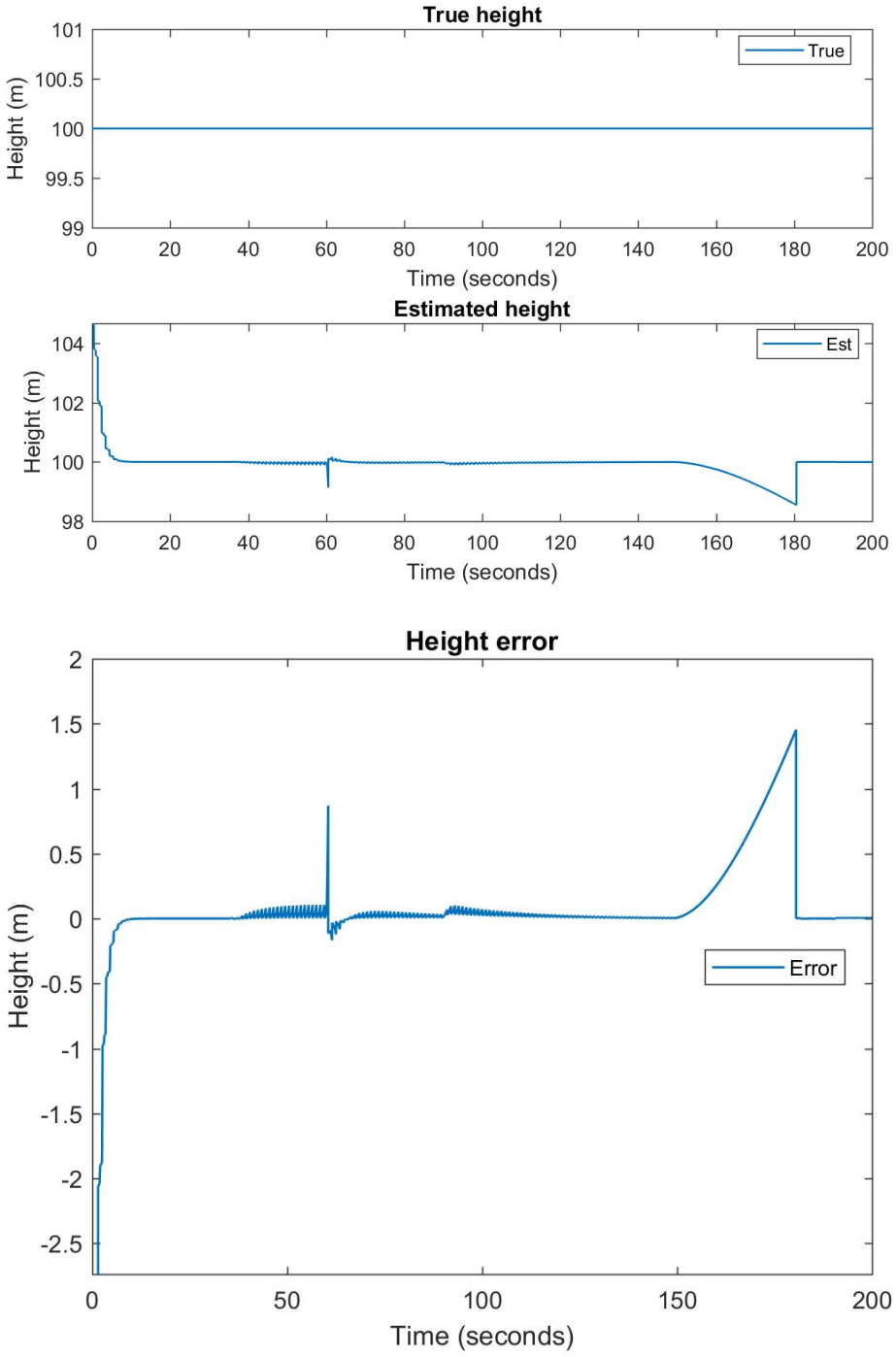

Figure 10.14: Estimated height and error, no noise

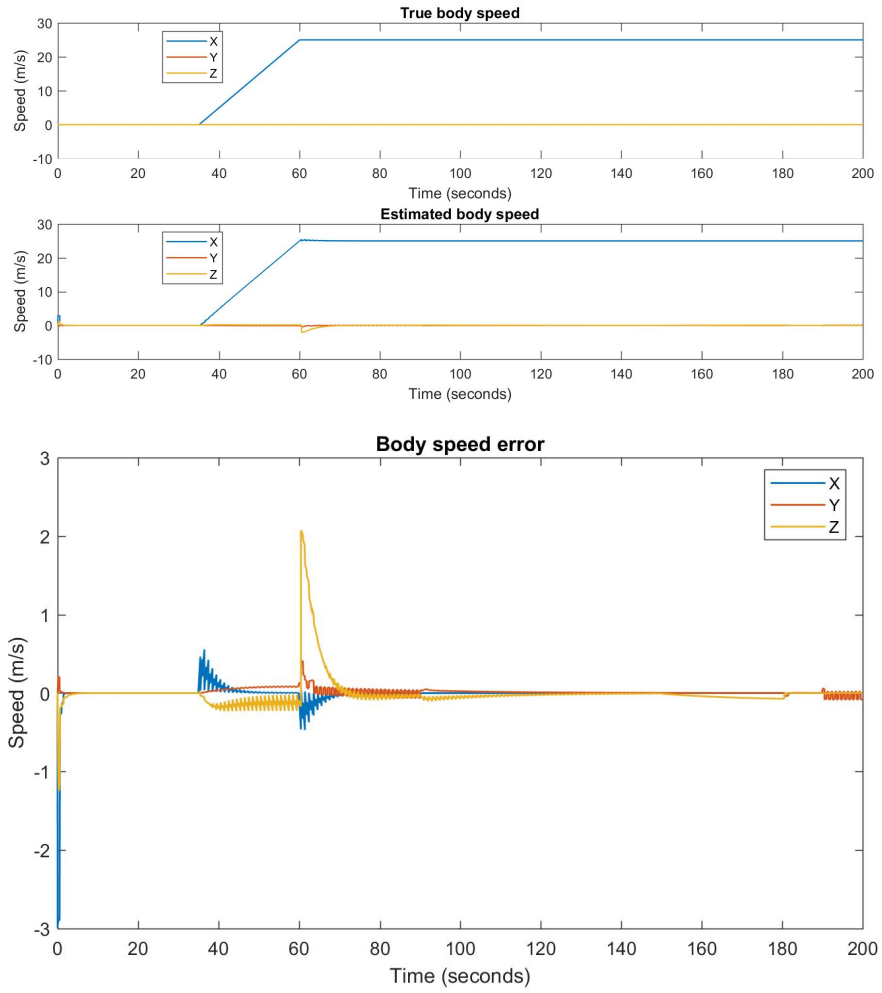

Figure 10.15: True and estimated body speed and error, no noise

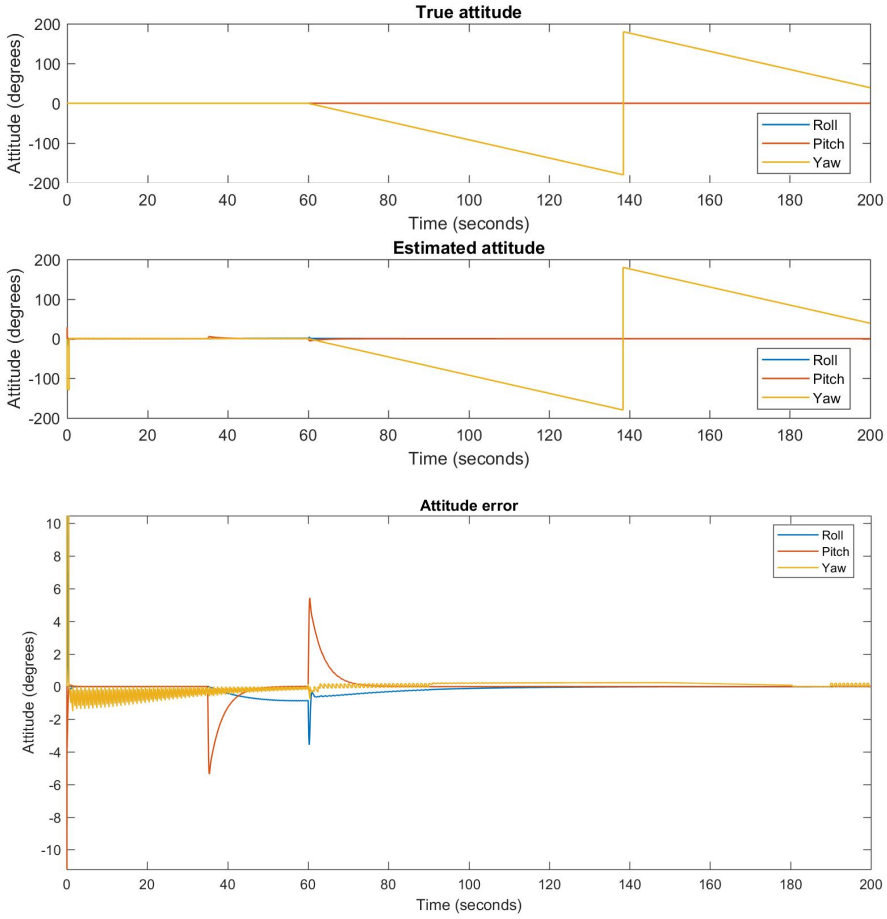

Figure 10.16: True and estimated attitude and error, no noise

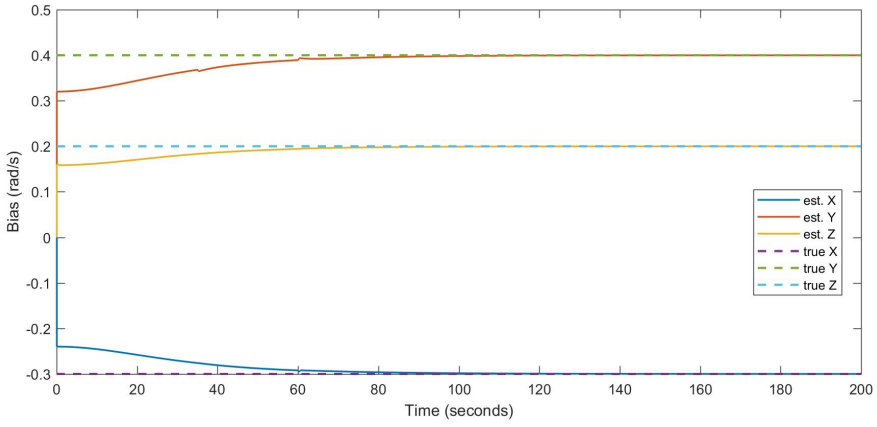

Figure 10.17: Gyroscope bias, no noise

Noise on MEMS sensors only

Starting from the previous simulation, sources of disturbances are incrementally introduced. Figures 10.18 to 10.26 show the result of the double EKF algorithm on the same benchmark, with added white noise only on MEMS sensors. GPS, odometry and maps still provide ideal measurements to the algorithm.

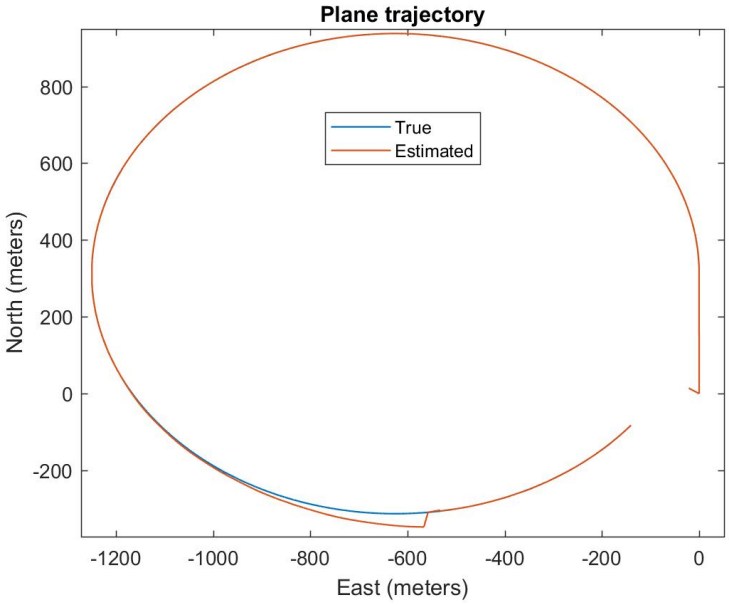

Figure 10.18: True and estimated trajectory, MEMS noise only

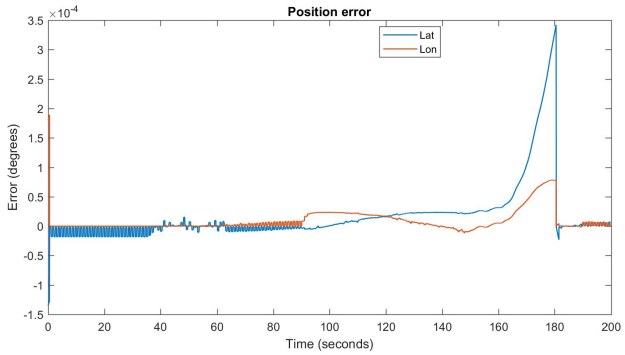

Figure 10.19: Position error in meters, MEMS noise only

It can be easily seen that some non-negligible error is introduced by noise during dead reckoning without maps. In particular, the final position error af-

ter 30 seconds of navigation without any external aid turns out to be of about 40 meters on the plane according to Figure 10.20, most of it being related to a heading estimate drift of about 5-6 degrees, as seen from Figure 10.25, rather than of a wrong body speed estimate on the corresponding axes (see Figure 10.23).

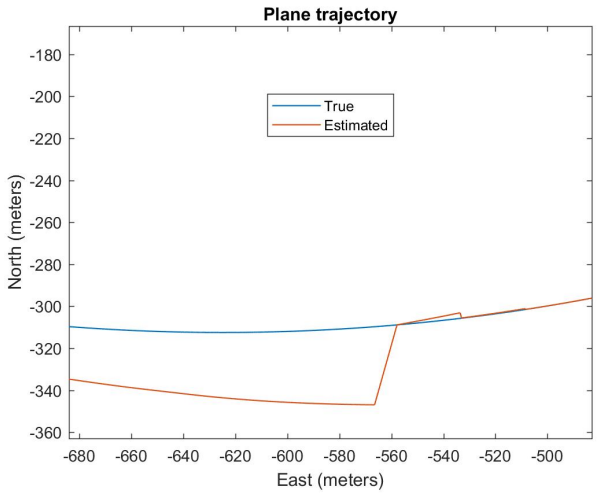

Figure 10.20: Detail of true and estimated plane trajectory, MEMS noise only

The height error is manly related to the Z component of the estimated speed. A possibility to reduce this effect is via a reduced-order observer, whose results are discussed later in this chapter.

Concerning attitude, the noisy estimate is mainly related to the sensors noise, but the error still remains bounded within less than 10 degrees. Accelerometers are more relevant on roll and pitch, while yaw is more influenced by gyroscope noise.

Under these noisy conditions, gyroscope bias estimates are still correct and very close to their actual values.

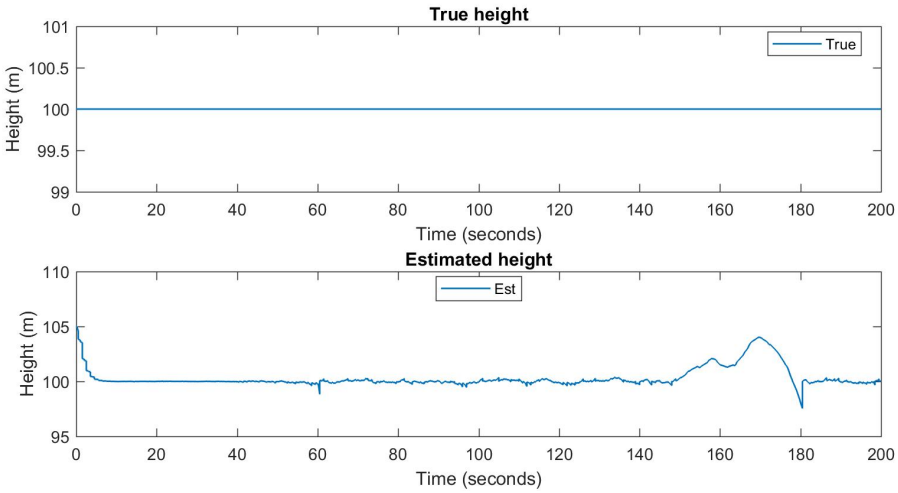

Figure 10.21: True and estimated height, MEMS noise only

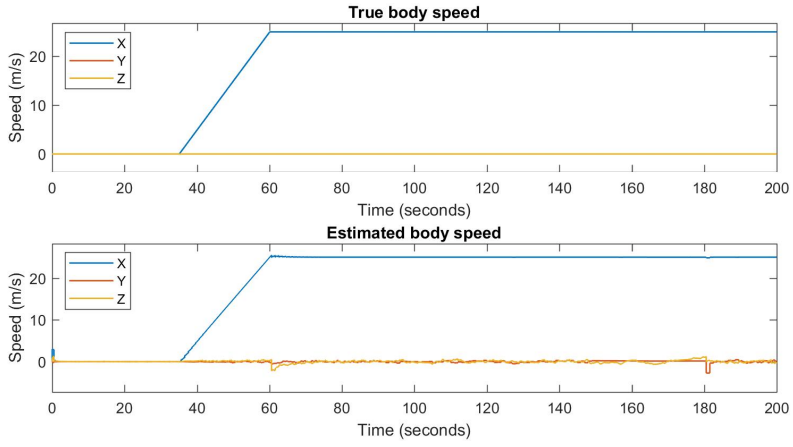

Figure 10.22: True and estimated body speed, MEMS noise only

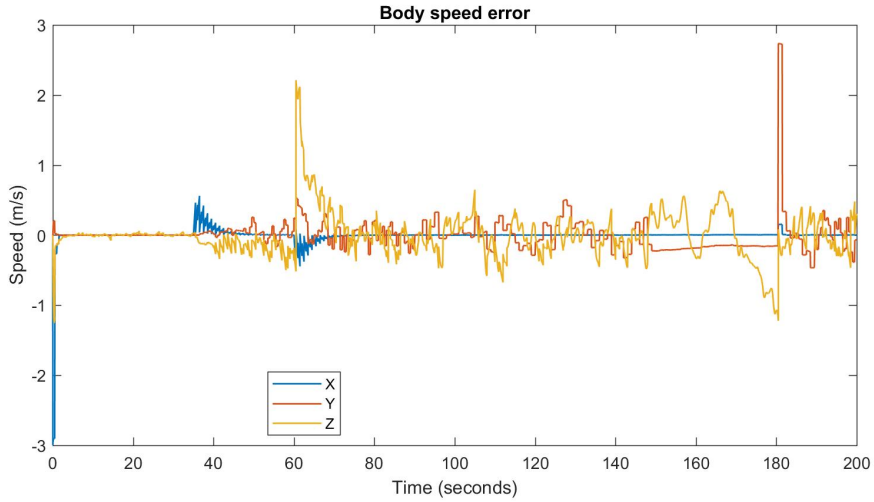

Figure 10.23: Body speed estimation error, MEMS noise only

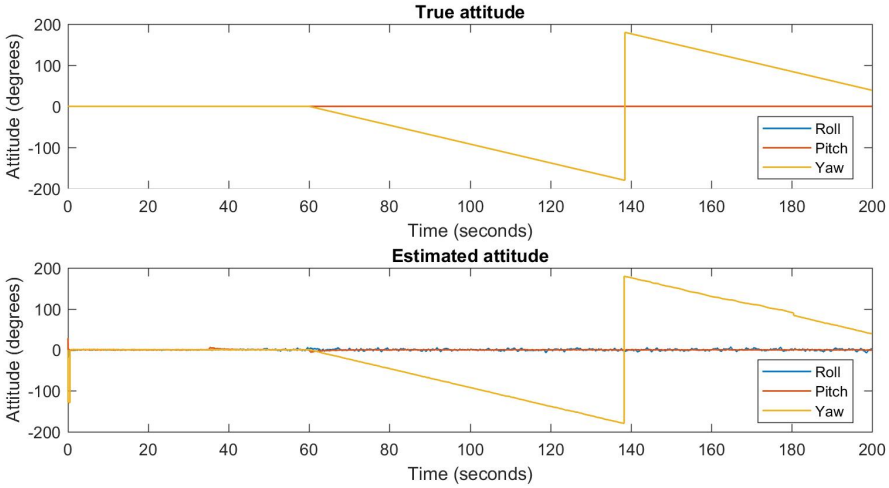

Figure 10.24: True and estimated attitude, MEMS noise only

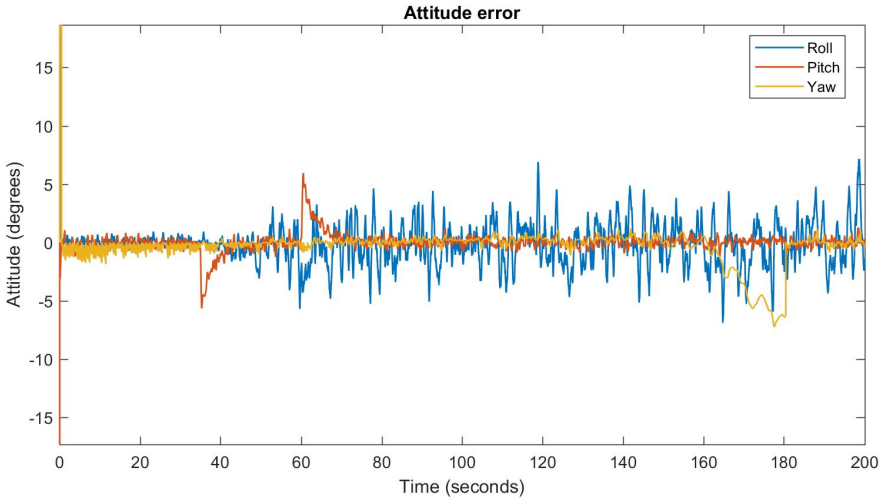

Figure 10.25: Attitude estimation error, MEMS noise only

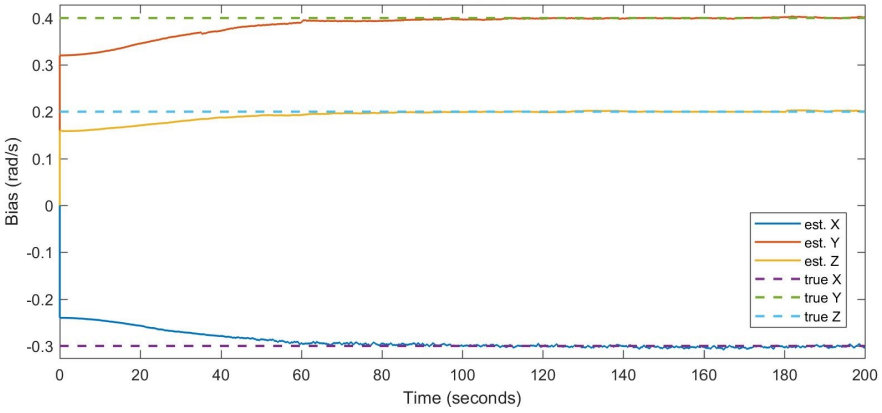

Figure 10.26: Gyroscope bias, MEMS noise only

Noise on all sources

The same benchmark and algorithm have been used in another simulations with noise added on MEMS sensors, GPS, odometry and maps. Starting estimates and periods of measures' availability are left unchanged.

As expected, position and height estimates are less accurate throughout the simulation due to the noise added on GPS and maps, even if the maximum amplitude is not bigger with respect to the previous one. However, positioning error after the period of dead reckoning is strongly dependant on the state of the filter after the last received measurement.

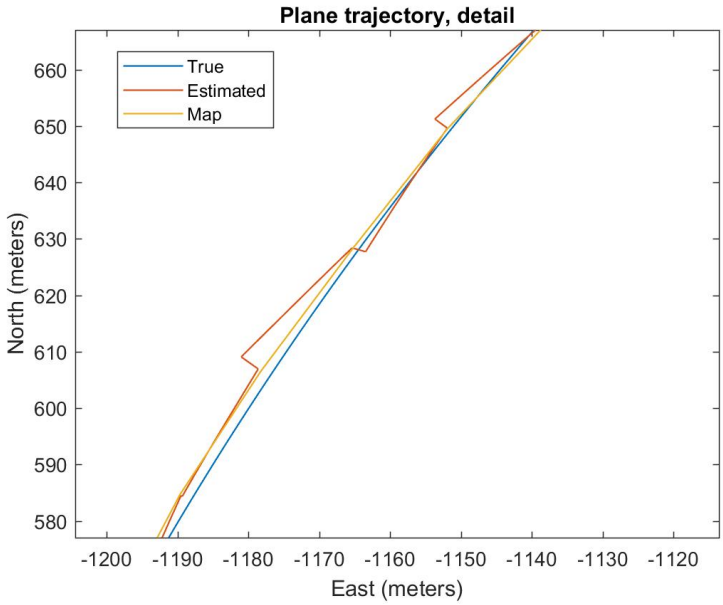

Figure 10.27: Detail of true and estimated trajectory, all noises

Speed and attitude estimates are obviously worse when eteroceptive sensors (including maps) become noisy. In particular, an important error is present on the axes orthogonal to the speed of the vehicle, while the X component presents a smaller error. Also heading becomes very noisy when maps are not ideal and when the vehicle is at a stop, as can be seen from Figure 10.32.

The spikes on the yaw attitude are due to map inaccuracy and sign change in odometry speed at stop; in fact, the effect disappears as the train starts moving. Moreover, it can be noticed that the "ringing" effect present in the previous two simulation is almost completely covered by GNSS noise and thus not distinguishable (see Figures 10.13 and 10.19). However, the overall

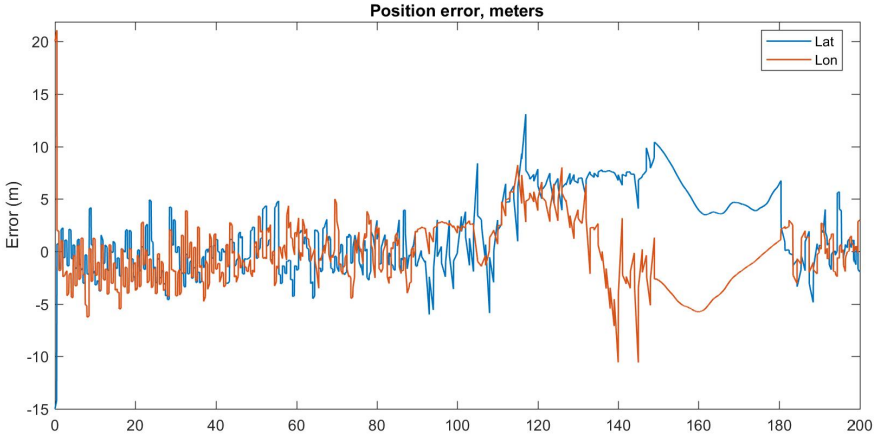

Figure 10.28: Position error in meters, all noises

performance level in presence of noise is considered acceptable in relation with the noise level and similar to the performance levels depicted in Section 5.1.

In addition, gyroscope bias estimates converge to the true values in 60-80 seconds since simulation start, with a small residual error due to noise, as seen from Figure 10.33.

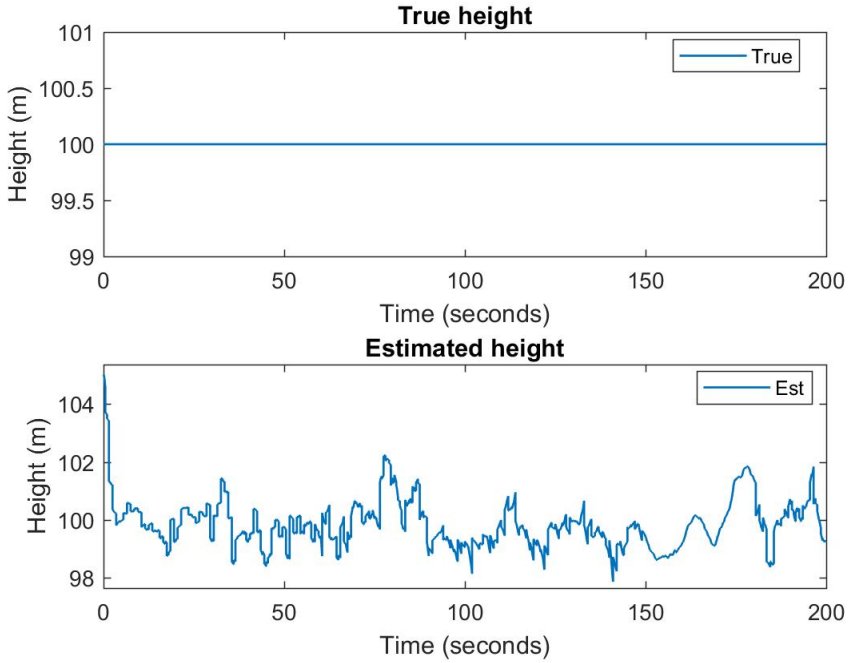

Figure 10.29: Estimated height, all noises

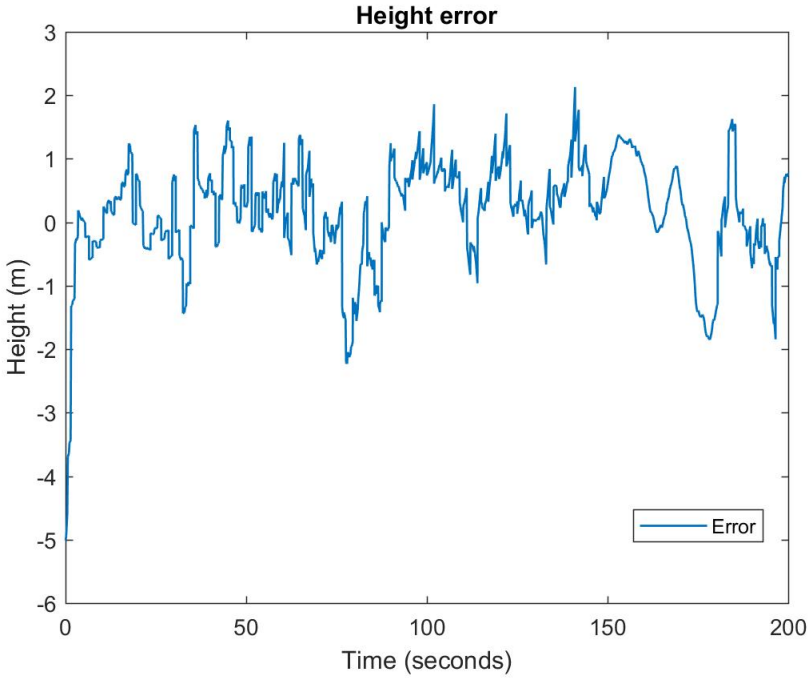

Figure 10.30: Height error, all noises

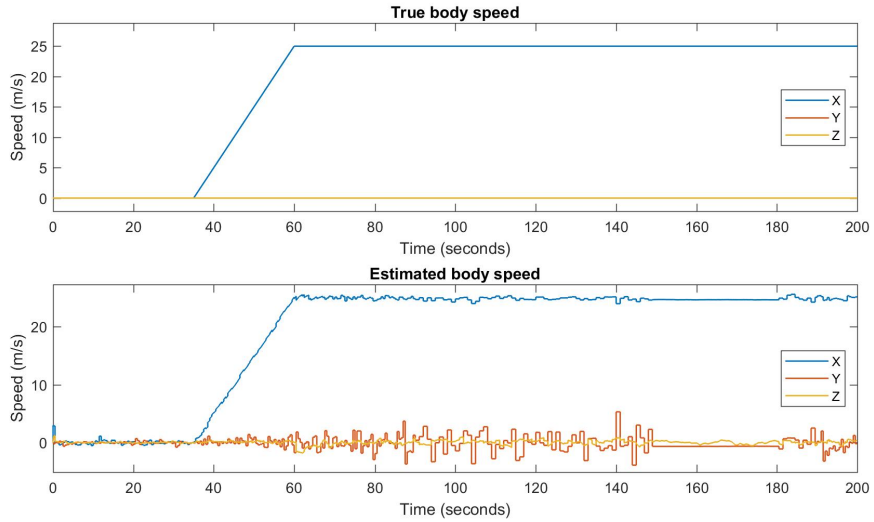

Figure 10.31: True and estimated body speed, all noises

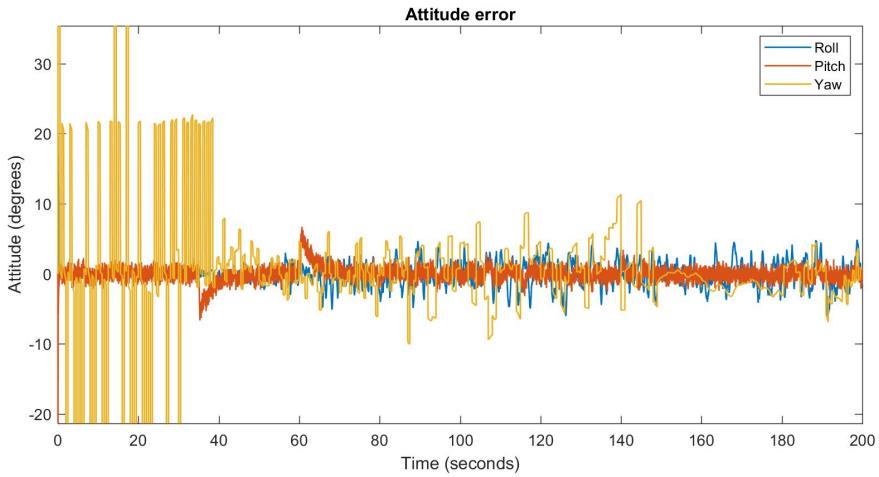

Figure 10.32: Attitude error, RPY angles, all noises

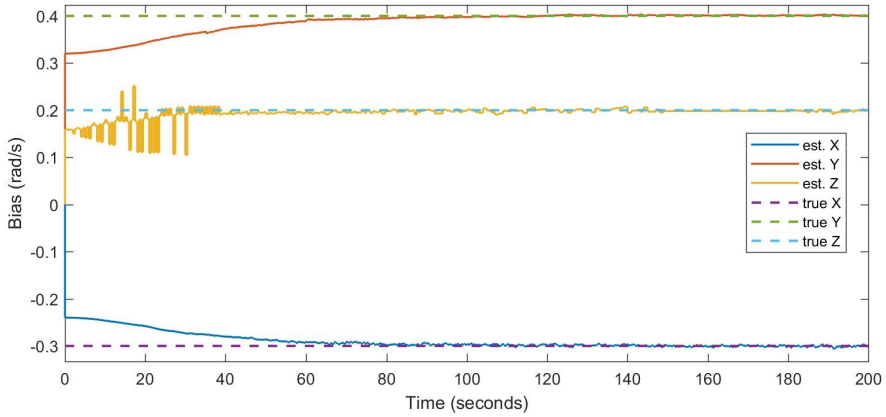

Figure 10.33: Gyroscope bias, all noises

To provide an additional comparison, the following simulation has been carried out by assuming maps to be always available, so only a GPS outage is present between 90 and 180 seconds. The other conditions were unchanged. The maximum position error is reduced with respect to the previous case from 10 to a about 5 meters, but - most importantly - it is almost completely localized along the map trajectory, according to Figure 10.34. Speed measurement from odometry plays a crucial role in determining the positioning error.

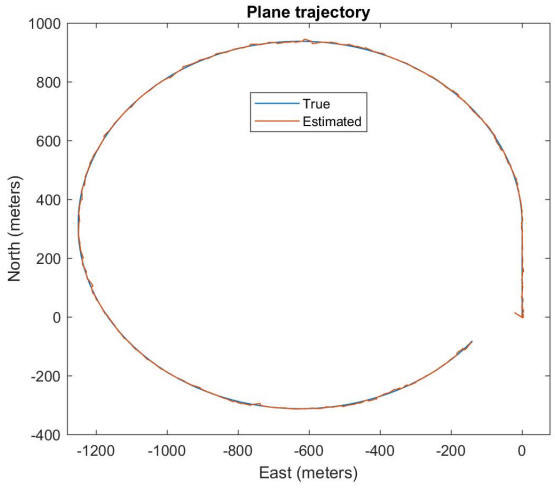

Figure 10.34: Detail of true and estimated trajectory, all noises, map always available

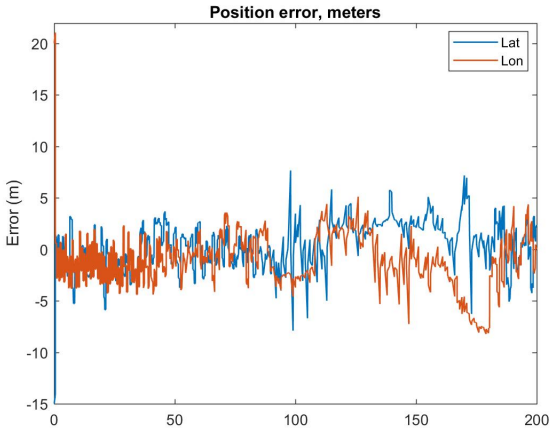

Figure 10.35: Position error in meters, all noises, map always available

10.2.2 Double-EKF, reduced-order algorithm

The reduced-order version of the above algorithm has been tested to evaluate its performance by exploiting the kinematic constrain of the vehicle.

Without white noise

Here are reported the results of a simulation run with the reduced-order observer algorithm for the navigation dynamics. All noises are absent in this first test, with the exception of gyroscope bias. Accelerometer bias is not accounted in the estimation

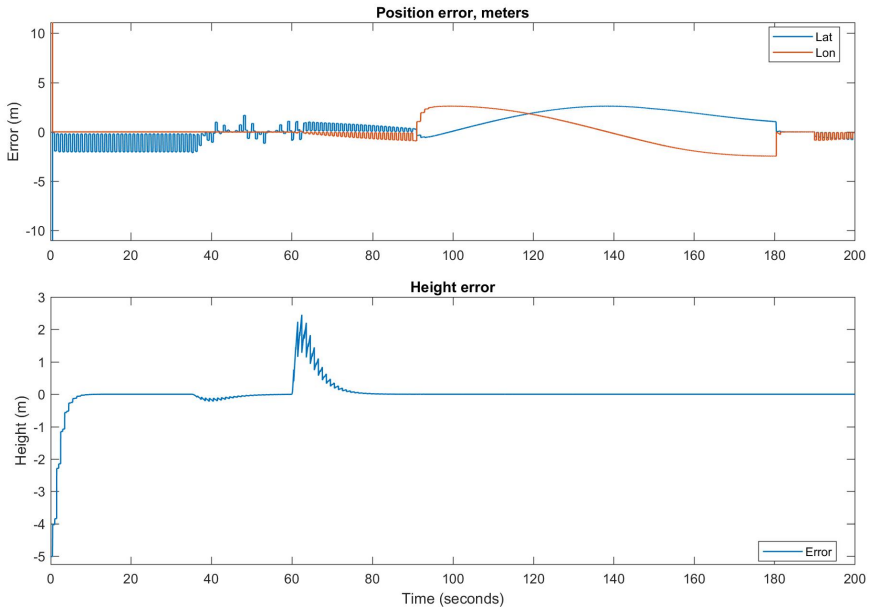

Figure 10.36: Position error in meters, no noise

It is easy to see that the same positioning performance are achieved for latitude and longitude, while height error results to be more limited because of the reduced order with respect to the full-order observer (see fig. Figure 10.14). The same performance were noticed for what concerns speed accuracy along body axis X.

Since the AHRS algorithm is the same of the full-order model, no relevant difference in the performance was noticed even with the usual interconnection used to compensate the true acceleration and centripetal effects. Moreover, gyroscope bias estimation still converges to the true value with the same timings of the previous tests, as shown in Figures 10.17, 10.26 and 10.33.

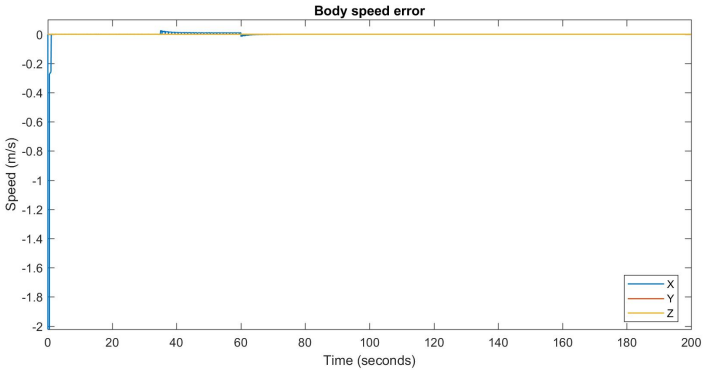

Figure 10.37: Body speed error, no noise

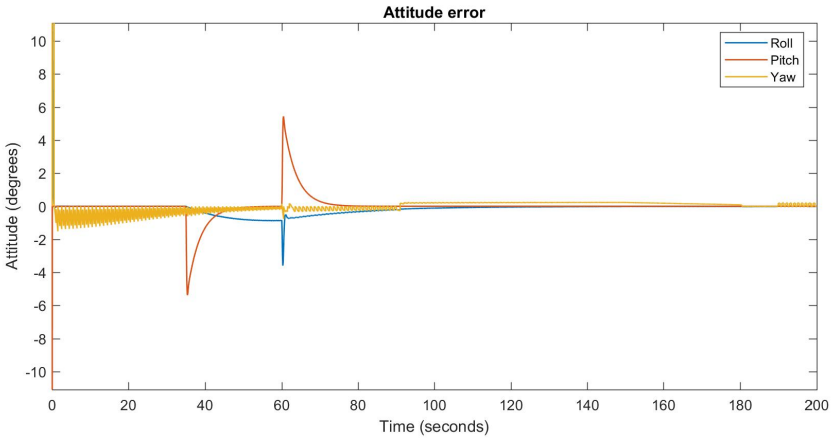

Figure 10.38: Attitude error, no noise

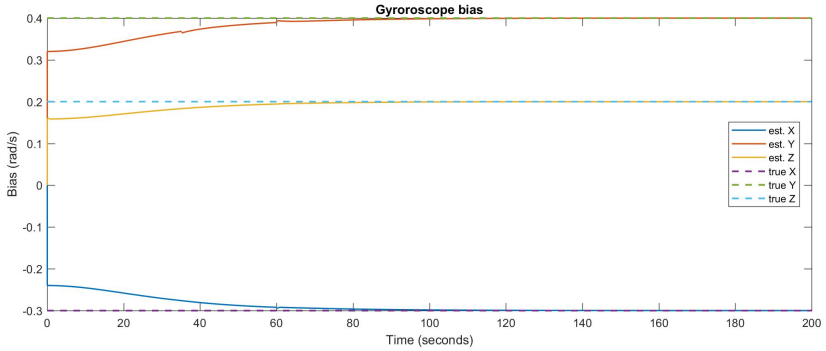

Figure 10.39: Gyroscope bias, no noise

Noise on all sources

Two simulations have been carried out for comparison with the full-oder model in order to understand the advantages and drawbacks of a simplified structure. In the first test with noise, GPS and maps are made available according to Figure 10.11 and noise is present on all possible sources. Once again, accelerometer bias is absent and not estimated.

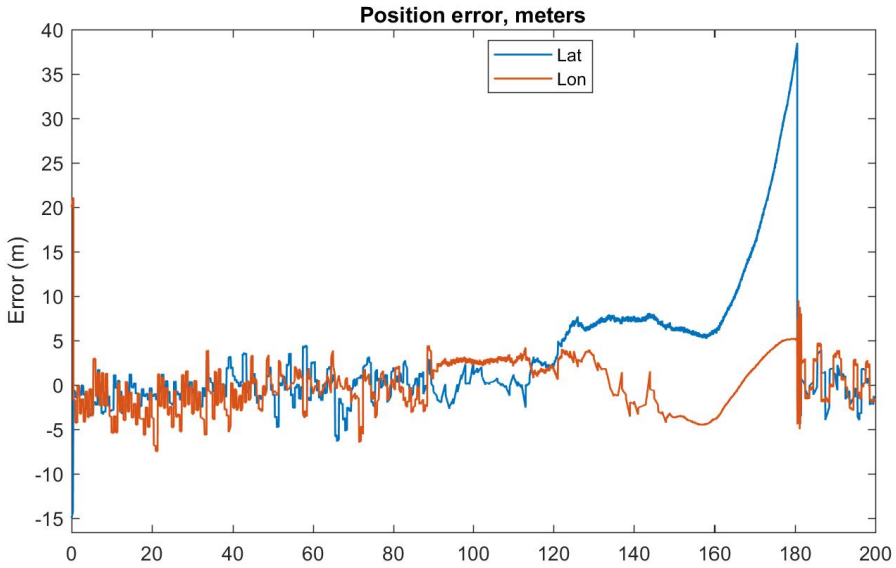

Figure 10.40: Lateral position error in meters, all noises

The benefits of the reduced-order observer can be noticed in the speed estimation. As shown in Figure 10.40, the elevation error is more limited thanks to the assumption on the train motion. On the other hand, the cause of the residual error is to be found on the attitude error, induced by accelerometers noise.

The plane position error results to be bigger in this run. However, the cause of this drift is to be found on the AHRS because of a small residual error in bias estimation at the beginning of the dead reckoning phase which is not corrected by any source for 30 seconds. It can be appreciated in Figures 10.42 and 10.43 and whose effect are visible on the overall plane trajectory depicted in Figure 10.44.

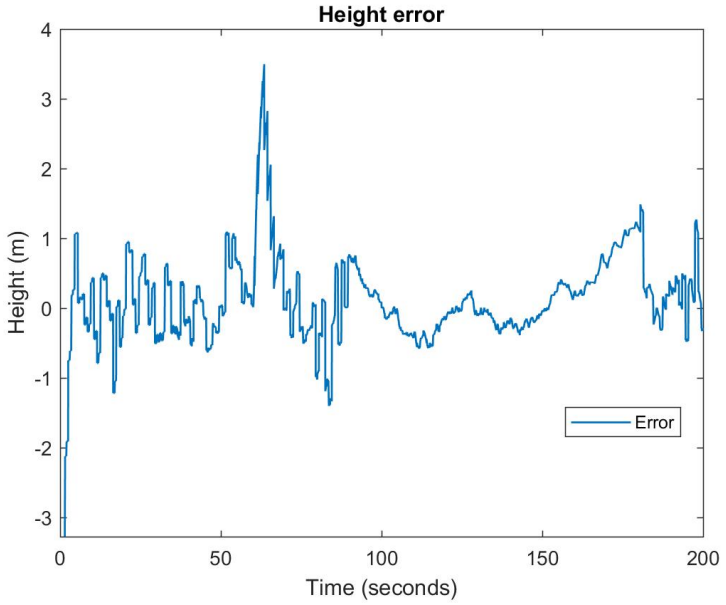

Figure 10.41: Height error in meters, all noises

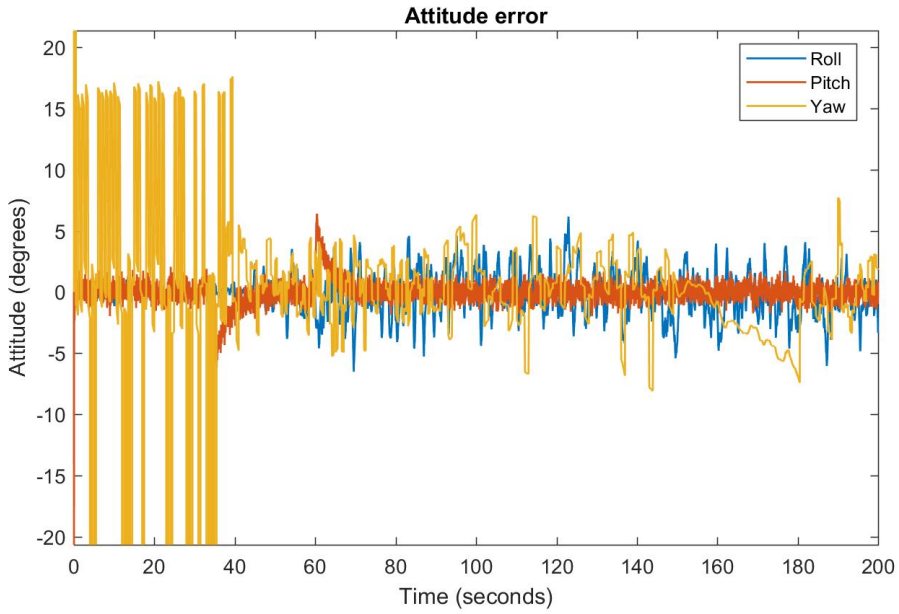

Figure 10.42: Attitude error, all noises

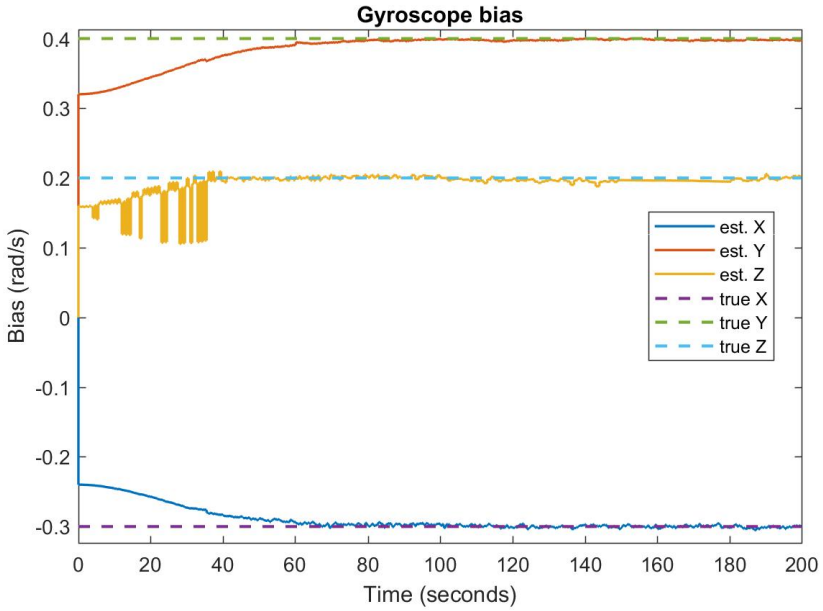

Figure 10.43: Gyroscope bias, all noises

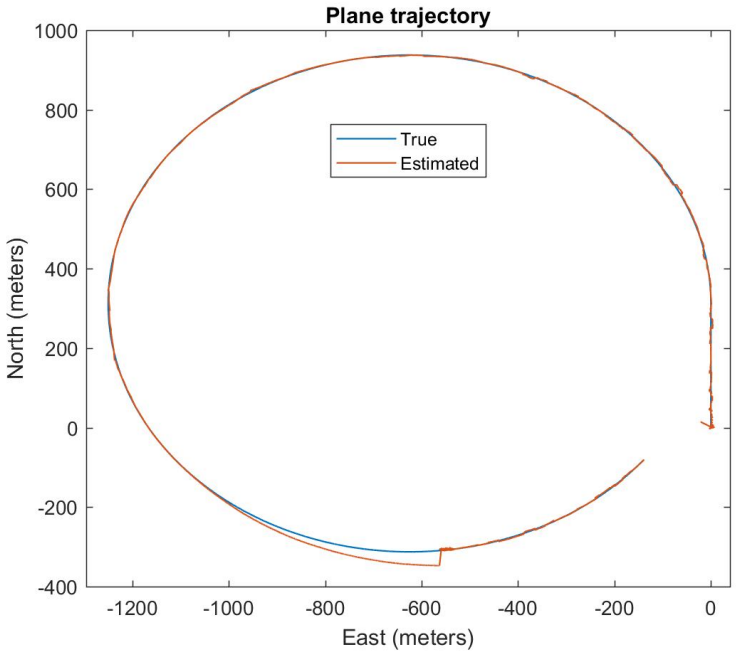

Figure 10.44: True and estimate plane trajectory, all noises

In the second run, maps are assumed to be available at all times. Then gyro bias estimates converge and the heading error remains bounded because of the continuous correction. The residual position error is due to the noise on odometry readings and map uncertainties as it is again all contained along the train's expected path given by the map.

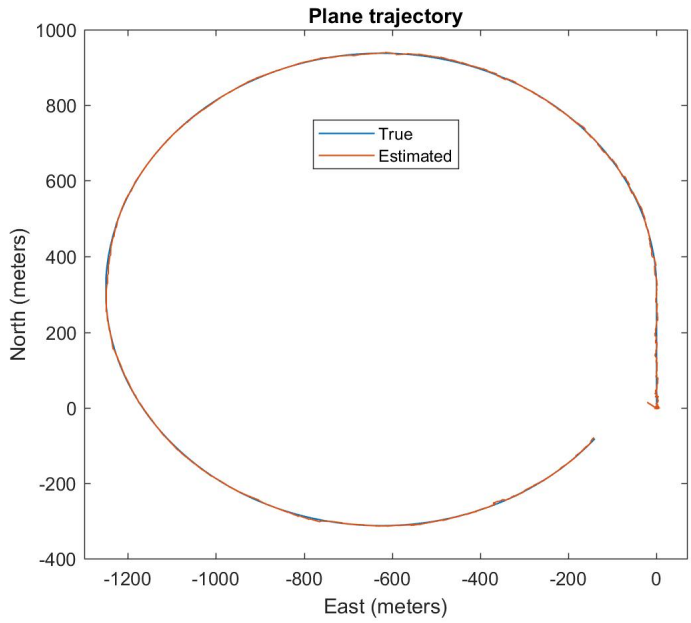

Figure 10.45: True and estimate plane trajectory, all noises, map always present

In order to provide a useful comparison, in the following plots uncertainties on maps and odometry have been removed in order to better appreciate which is the major cause. In the simulation reported in Figure 10.47 odometry has no noise, while in Figure 10.48 the ideal maps and noisy odometry were used.

It is possible to see that both uncertainties have more or less the same effect on the performance degradation. Nevertheless, another crucial role is played by the filters initialisation before dead reckoning, which is not exact and affected by GNSS' and sensors' non-idealities. Another important factor is map granularity, which may lead to an additional drift according to the position of the closest point to the estimation. However, the maximum error along each direction is not greater than 8-10 meters at any time in this benchmark.

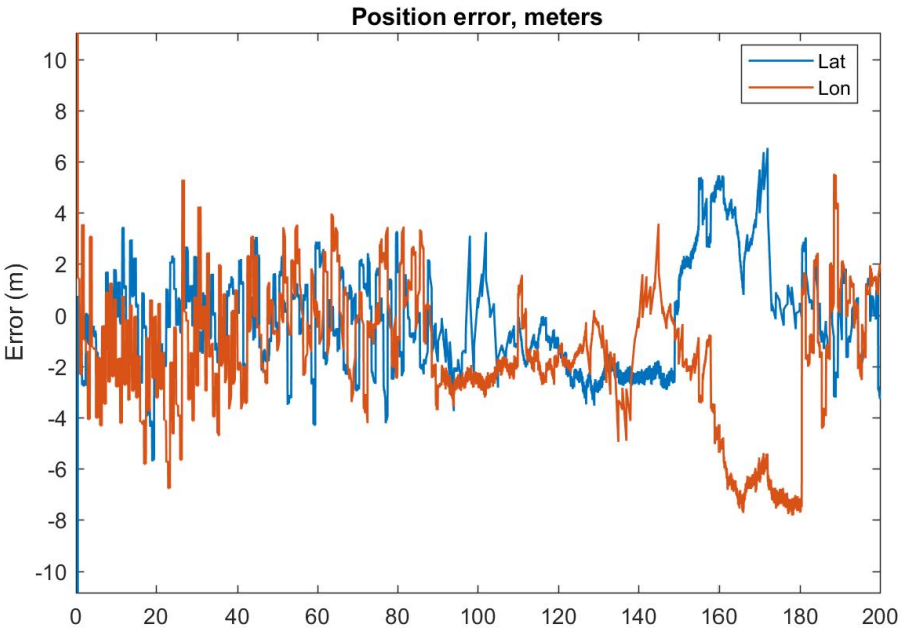

Figure 10.46: Position error in meters, all noises, map always present

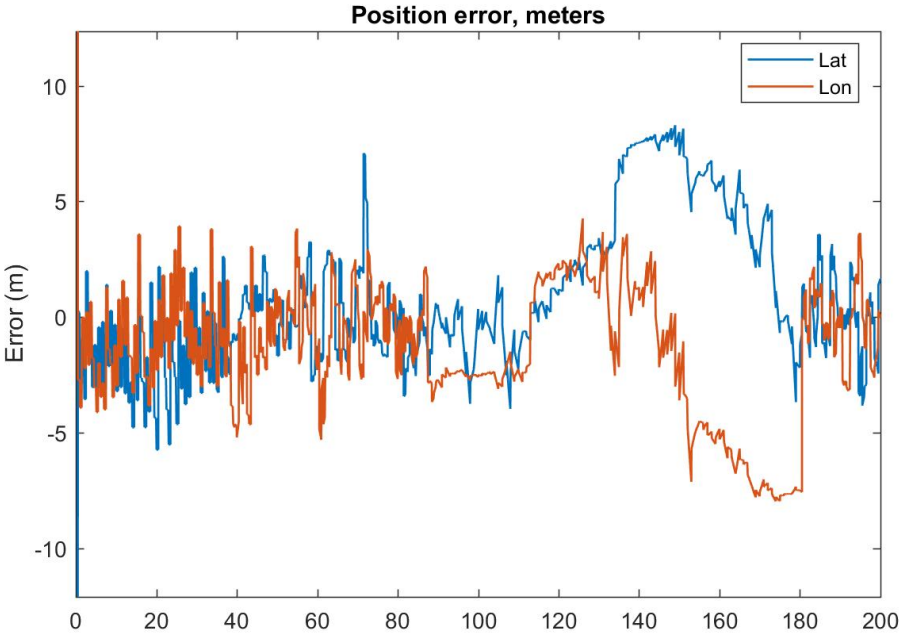

Figure 10.47: Position error in meters, ideal odometry reading

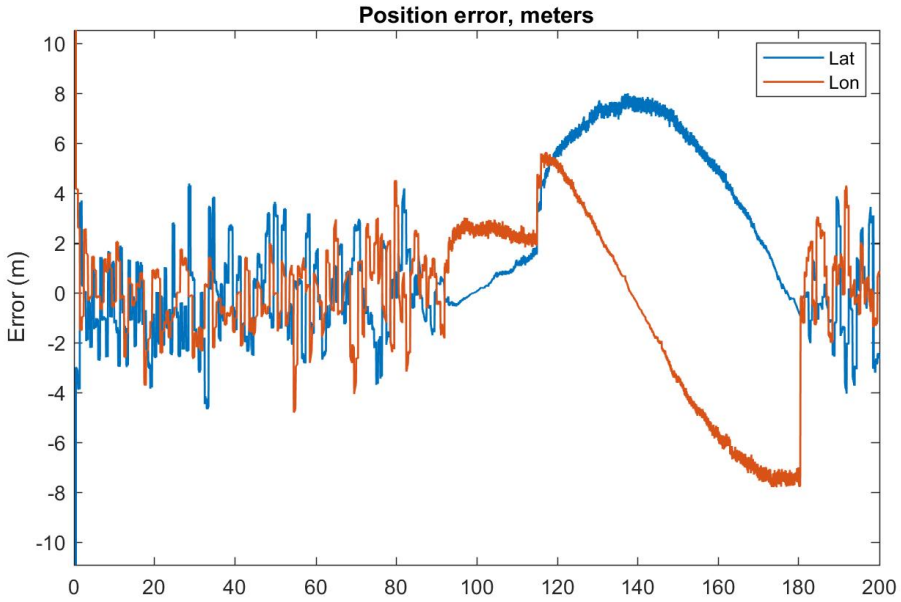

Figure 10.48: Position error in meters, ideal maps

10.2.3 NLO-EKF, full-order algorithm

The following two algorithm differ from the previous ones in the AHRS section, which employes the nonlinear observer designed in Section 7.3 and implemented according to the scheme in Chapter 9.

Without white noise

The first simulation is completely without noise, except for some partially known gyroscope bias. The source availability diagram is the same as the previous simulations (see figure 10.11) in order to provide useful information for a comparison on the same benchmark and conditions. Results are presented in Figures 10.49 to 10.53.

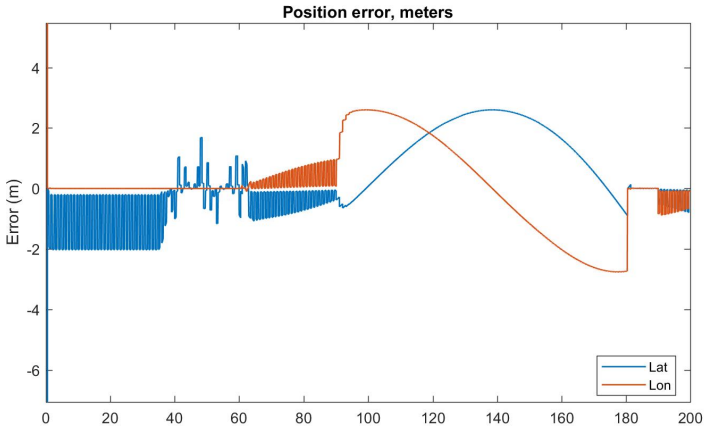

Figure 10.49: Position error in meters, no noise

As seen in Figures 10.49 and 10.50, the positioning error is still limited, even if the results on the elevation are slightly worse in this case. The same ringing effect due to the coarse map granularity of 5 meters is present also in this case.

No relevant differences were noticed on the trend of body speed estimates. In addition, the quality of attitude estimation is as good as in the EKF case. Finally, Figure 10.53 shows how the gyroscope bias estimates converge to the true values with a "time constant" similar to the EKF-based AHRS, thus proving the excellent convergence properties of the NLO.

On overall, it is possible to appreciate that the estimation performance are very similar to those achieved by the double EKF-based algorithm on the same ideal conditions. It can be easily noticed that the effect of a wrong initialisation of the algorithm is rapidly compensated and corrected by the

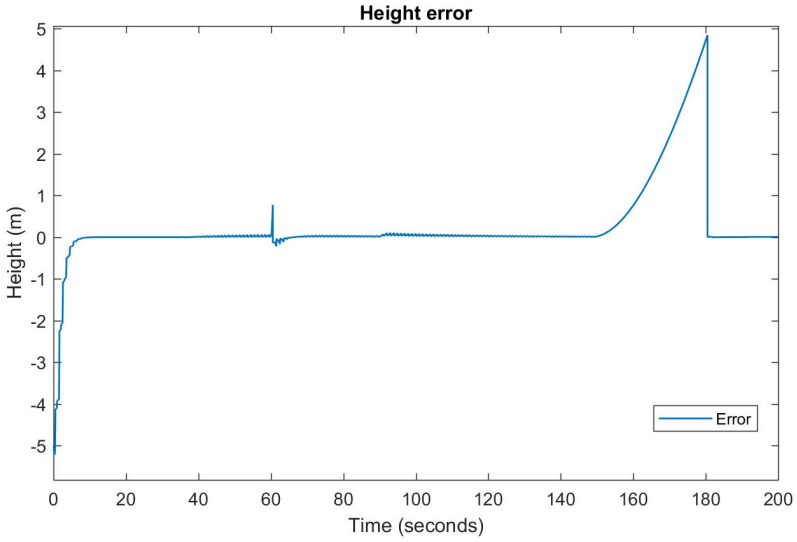

Figure 10.50: Height estimation error, no noise

first measurements, especially for what concerns attitude since because of the accelerometers' data rate. This behaviour is shared by all the four two-stage algorithms proposed in this work.

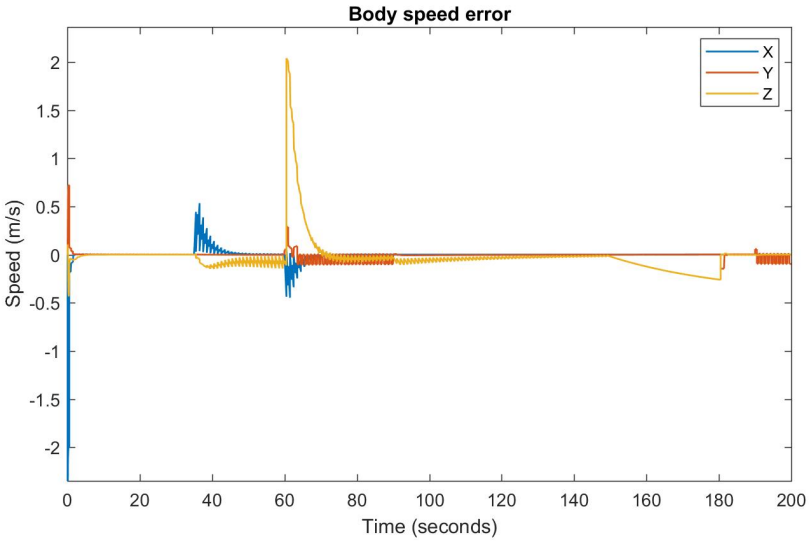

Figure 10.51: Body speed error, no noise

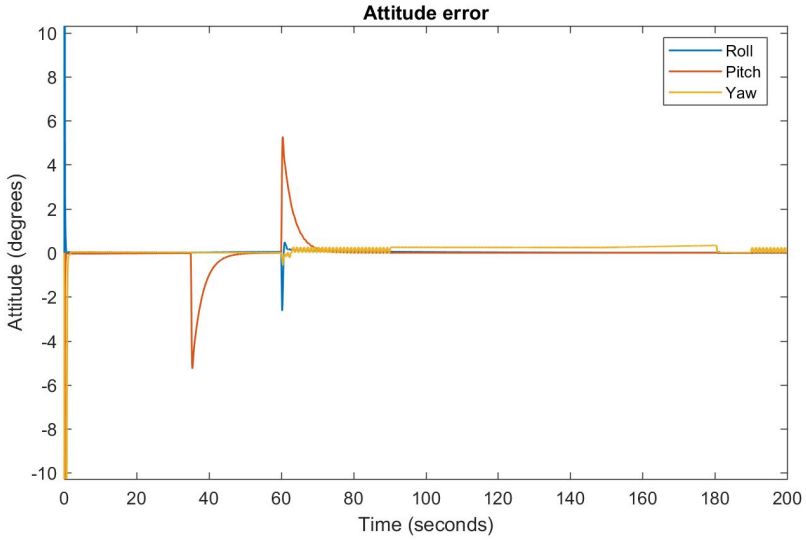

Figure 10.52: Attitude error, no noise

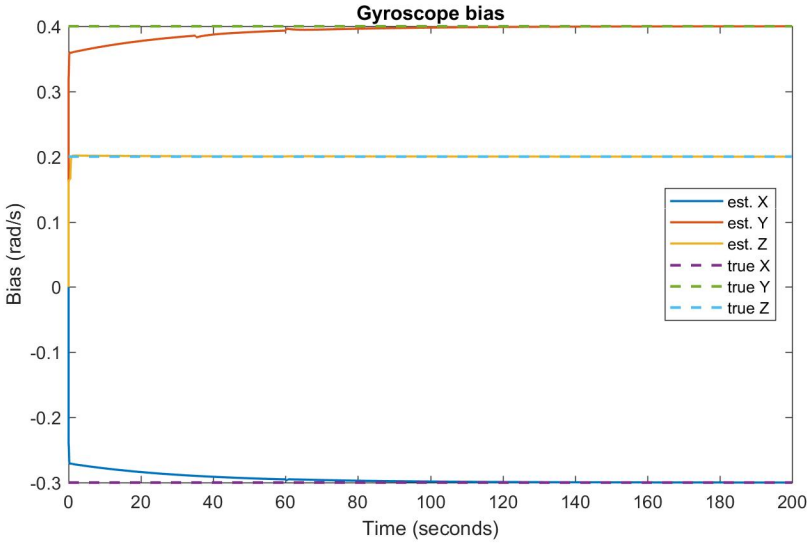

Figure 10.53: Gyroscope bias, no noise

MEMS noise only

The following simulation was carried out by adding a white noise disturbances to the accelerometers and gyroscope, their spectral density being the same as in the previous cases. As expected, errors induced by the full-order model are related to the body speed along the vertical axis, causing an important elevation error of 35 meters after 30 seconds of total dead reckoning, as shown in Figures 10.54 and 10.56.

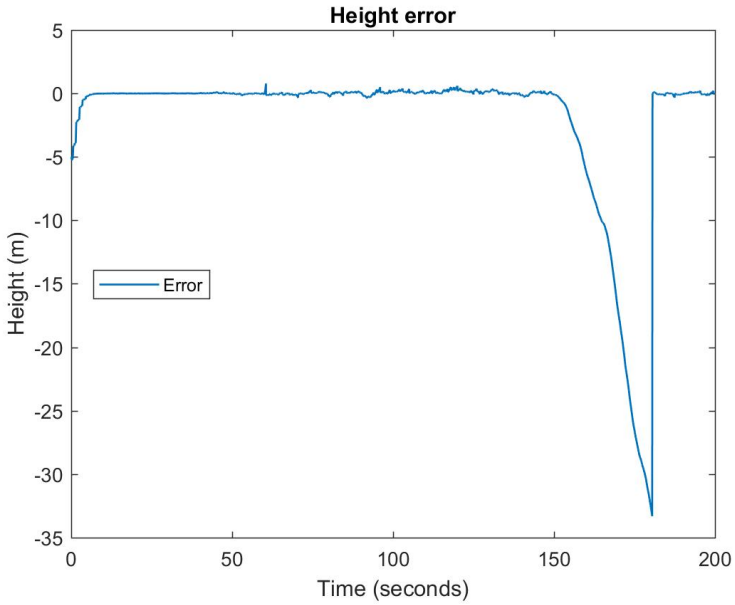

Figure 10.54: Height error, MEMS noise only

On the other hand, positioning error on the East-North plane are mostly related to some drift in the attitude, especially in the yaw component. The overall error along each component is less than 20 meters during dead reckoning (from T=150 s to T=180 s) resulting in about 24 meters of distance on the plane and 45 meters in 3D[2] ; however, it should be pointed out that such values tend to vary between runs because of the stochastic disturbances and the amplifying effect of integrators with respect to small perturbations of the initial state during dead reckoning.

Nevertheless, estimation of gyroscope non-idealities performed by the NLO is still accurate and converges to the true values, as shown in Figure 10.59.

[2]Mainly driven by elevation error, see fig 10.54

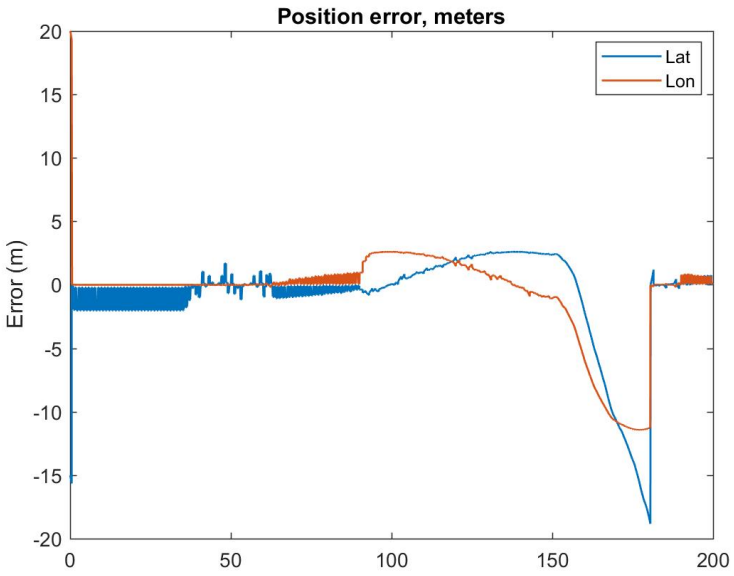

Figure 10.55: Position error, MEMS noise only

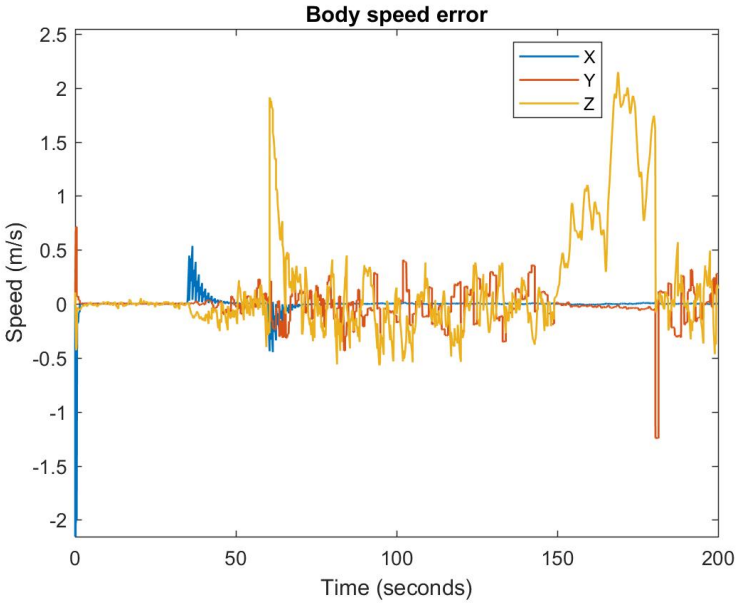

Figure 10.56: Body speed error, MEMS noise only

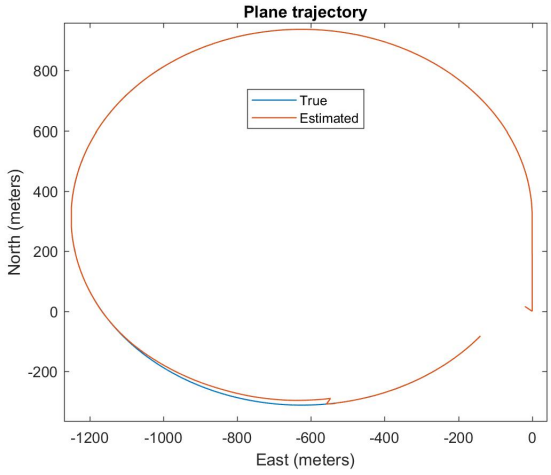

Figure 10.57: True and estimated trajectory, MEMS noise only

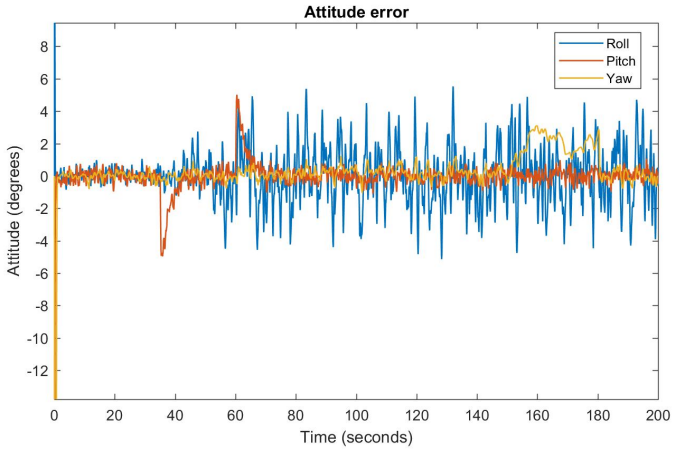

Figure 10.58: Attitude error, MEMS noise only

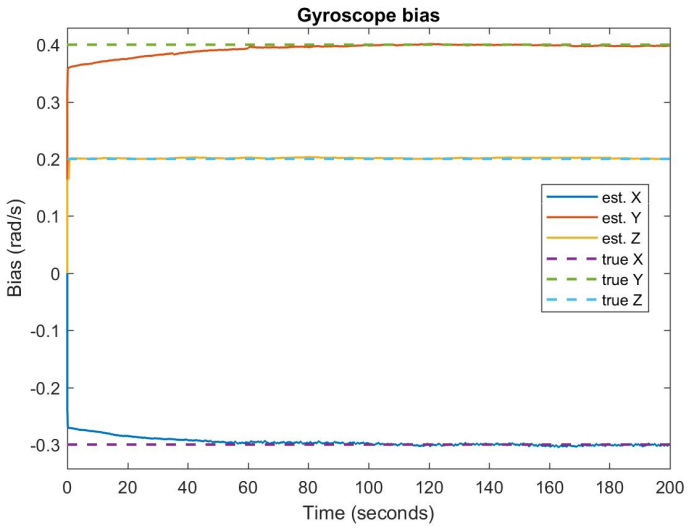

Figure 10.59: Gyroscope bias, MEMS noise only

All noise sources

The following results are obtained from a simulation where GPS, maps and odometry noise were present together with MEMS'. As expected, positioning errors on all components tend to be greater. The main cause is related to errors on the body speed estimate, as seen from Figure 10.62.

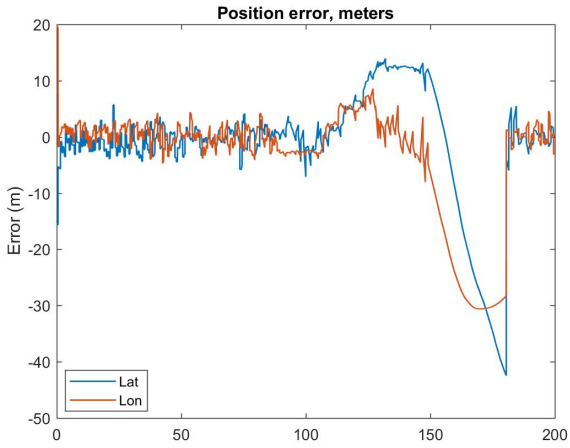

Figure 10.60: Position error, all noises enabled

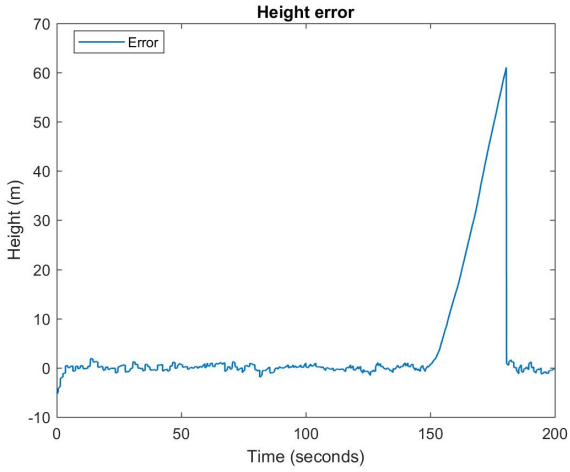

Figure 10.61: Elevation error, all noises enabled

Attitude error is limited, even if the estimation is noisy. Once again, error on roll and pitch components is constantly bounded thanks to the

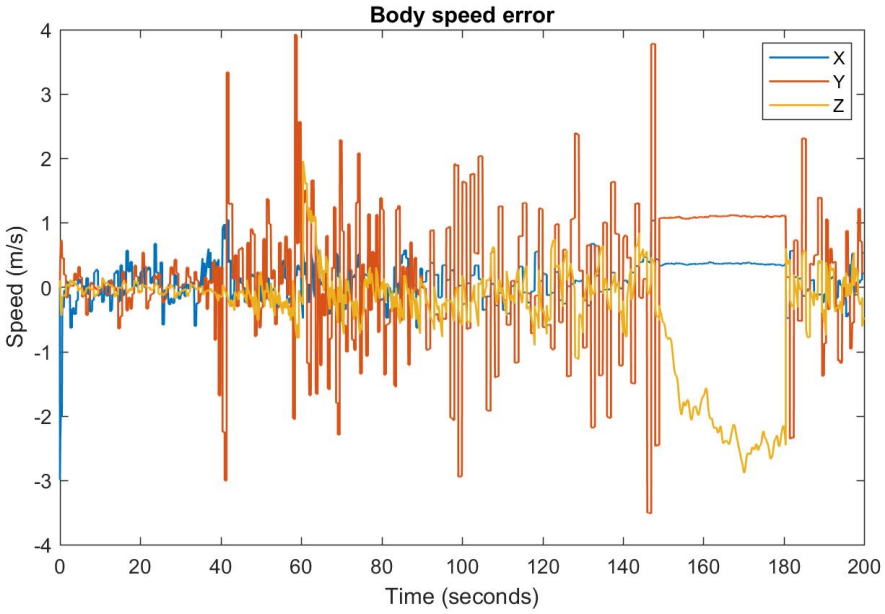

Figure 10.62: Body speed error, all noises enabled

continuous accelerometer-based correction and the NLO-based AHRS shows performances that are very similar to the EKF-based one.

As already observer in Figures 10.53 and 10.59, gyroscope bias estimation shown in Figure 10.64 is still very accurate, thus reducing drift during unavailability of both GNSS and map aiding.

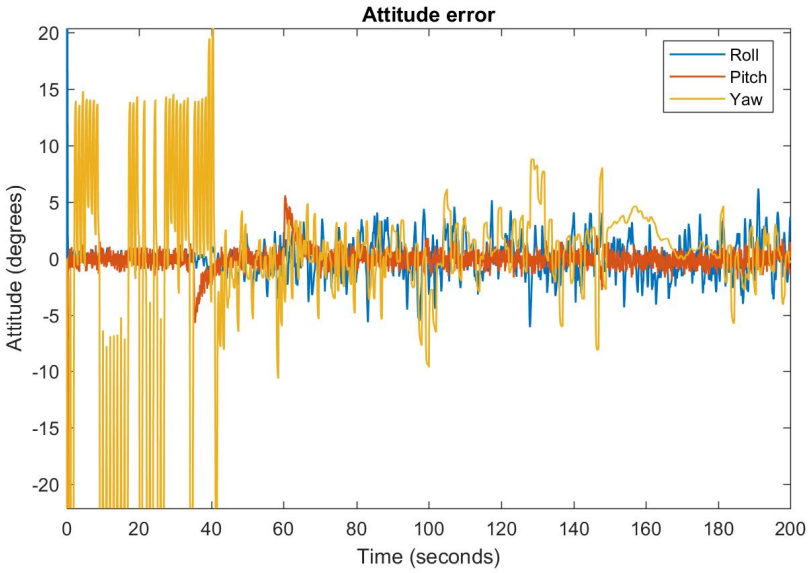

Figure 10.63: Attitude error, all noises enabled

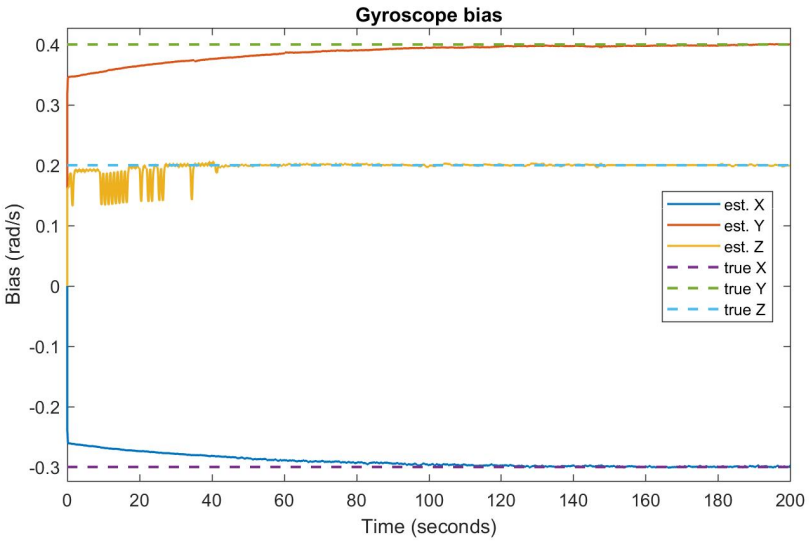

Figure 10.64: Gyroscope bias, all noises enabled

10.2.4 NLO-EKF, reduced-order algorithm

In this sections the results obtained with the INS algorithm based on the NLO and a reduced-order navigation observer are presented.

Ideal case, no noise

As previously done with the other three variants, results in ideal, white-noise free, simulations are presented.

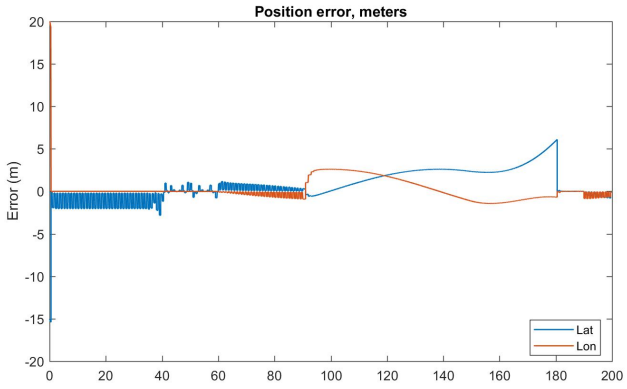

Figure 10.65: Position error, no noise

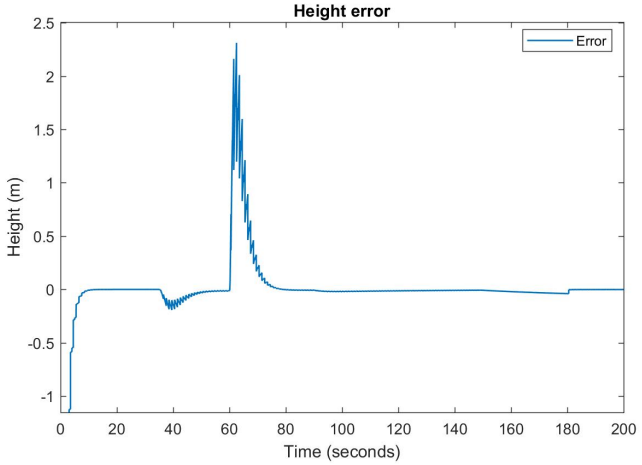

Figure 10.66: Elevation error, no noise

The same "ringing" effect shown in the previous simulations under ideal conditions is present also here in the intervals when both GNSS and maps are

used. Gyroscope bias estimation converges as well with the usual dynamics that came out in the results reported in the previous sections. Finally, attitude estimation errors are due to the body accelerations and compensated by the algorithm structure. Once again, the big starting attitude error is immediately corrected by the observer.

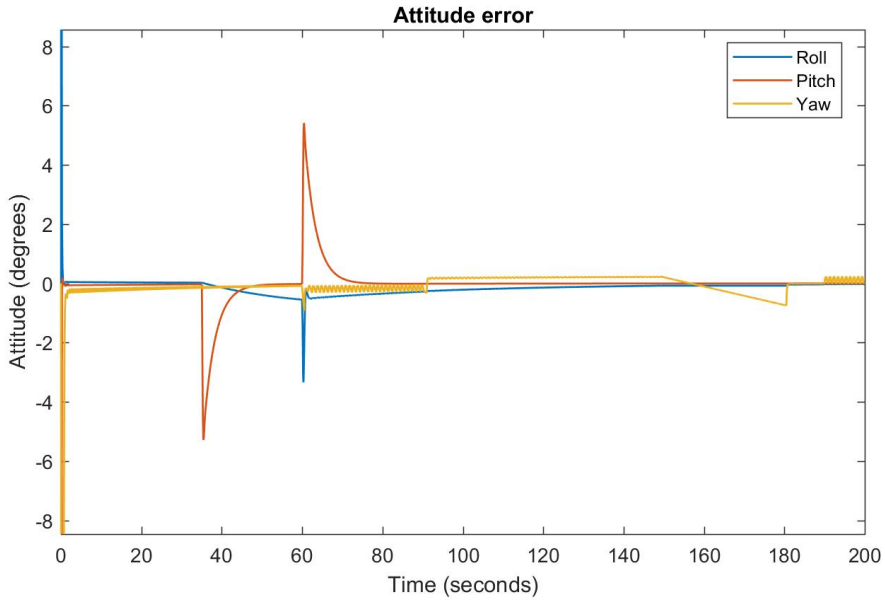

Figure 10.67: Attitude error, RPY angles, no noise

MEMS noise only

Similarly to what has been reported for the other simulations, using a reduced-order navigation observer greatly reduces elevation drift. The error reported in Figure 10.68 is limited to less than one meter at the end of dead reckoning phase (that is at T=180 seconds).

Figure 10.70 allows to appreciate the "shape" of the estimation drift on the North-East plane at the end of dead reckoning.

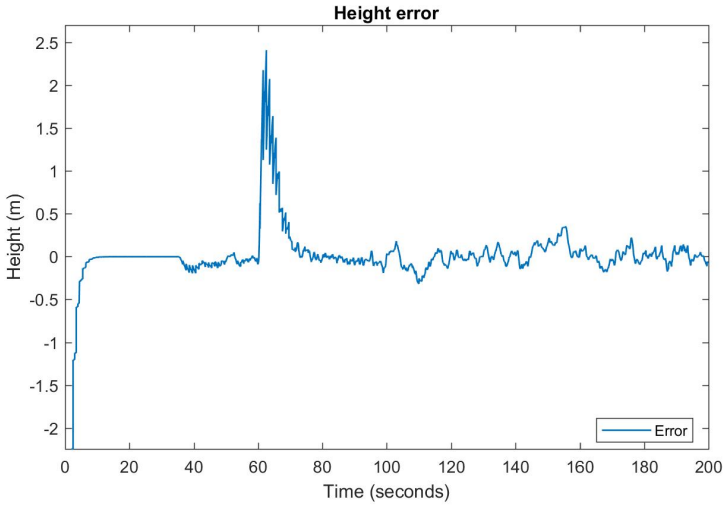

Figure 10.68: Elevation error, MEMS noise only

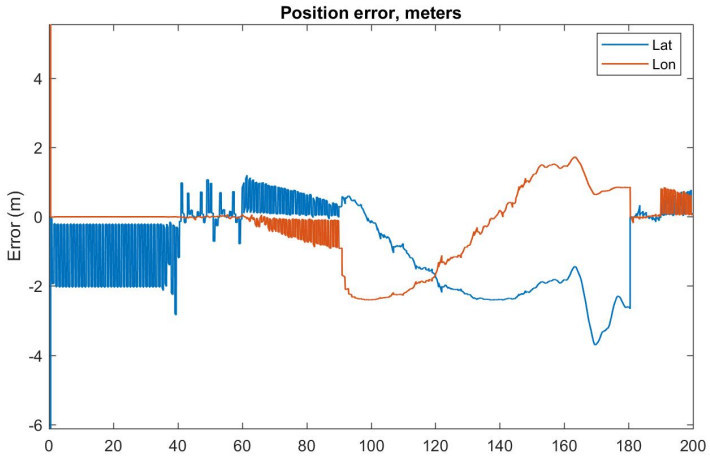

Figure 10.69: Position error, MEMS noise only

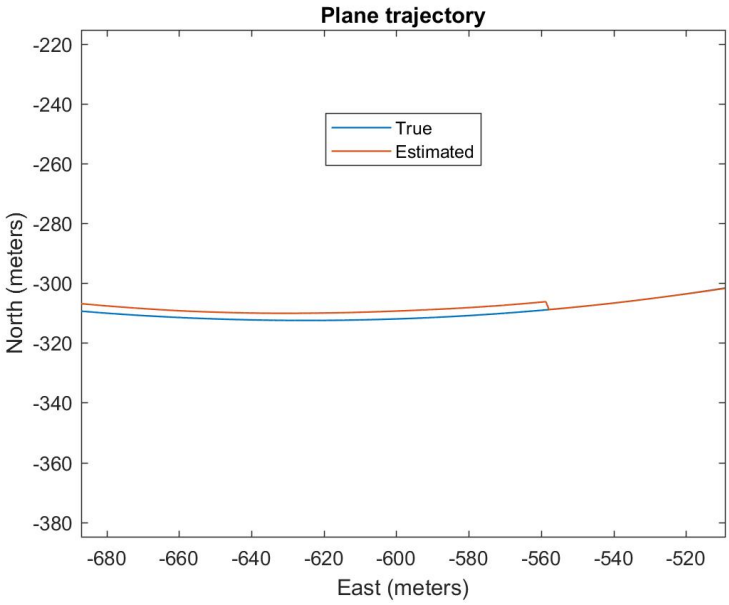

Figure 10.70: Detail of true and estimated trajectory, MEMS noise only

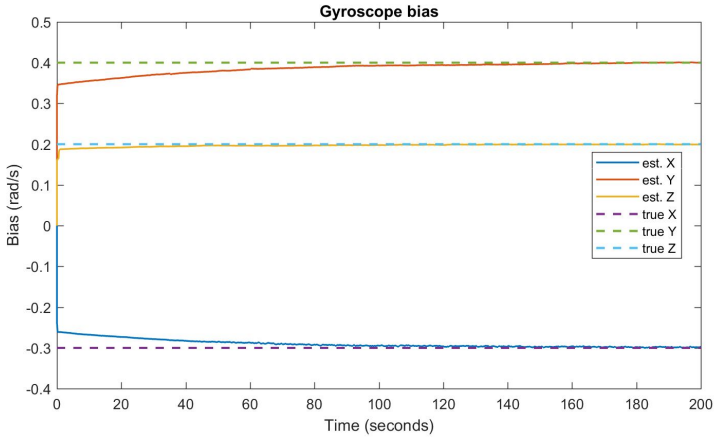

Figure 10.71: Gyroscope bias, MEMS noise only

All noise sources

Figures 10.72 to 10.77 show the result of a simulation run in which all possible noise sources were added.

With the aid of Figures 10.11 and 10.74, it is possible to appreciate from the plane trajectory in Figure 10.75 that the position error during the "map-only" phase (from 90 to 150 seconds) is again contained within a small range of the true train trajectory, its magnitude being about 22 meters along arc length. This phenomenon occurs throughout all the simulations in this section.

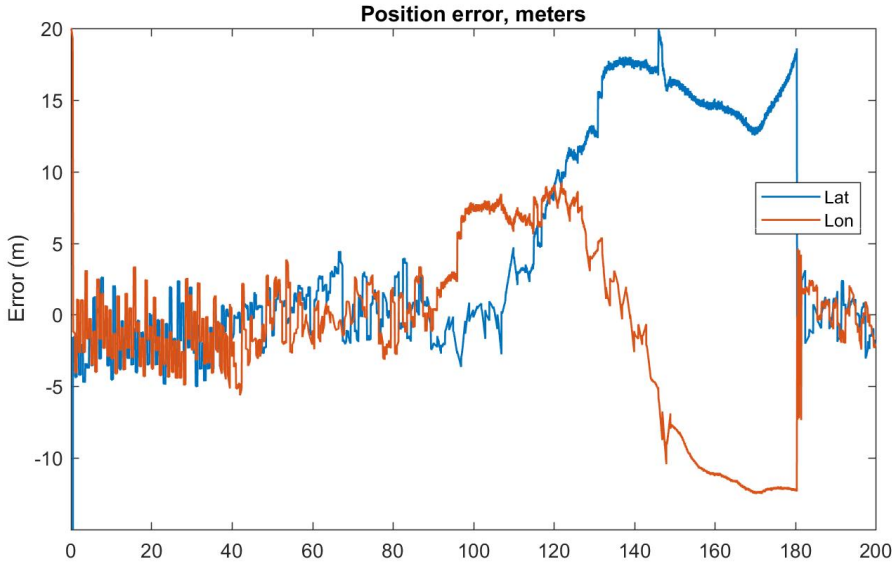

Figure 10.72: Position error, all noises enabled

From Figures 10.76 and 10.77 it is possible to appreciate the quality of gyroscope bias estimation and the boundedness of the attitude error, which is confined within a range of $\pm$ 5 degrees in roll and pitch, basically.

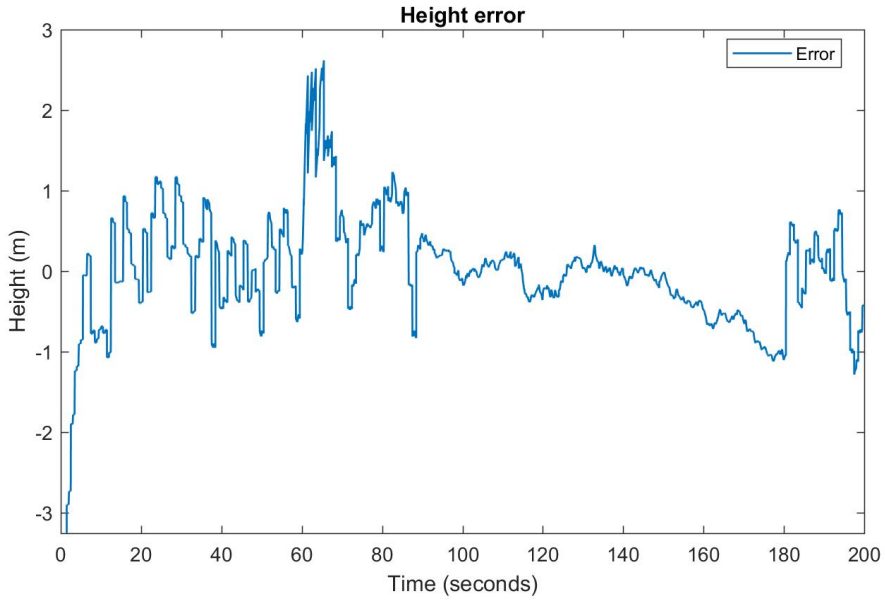

Figure 10.73: Elevation error, all noises enabled

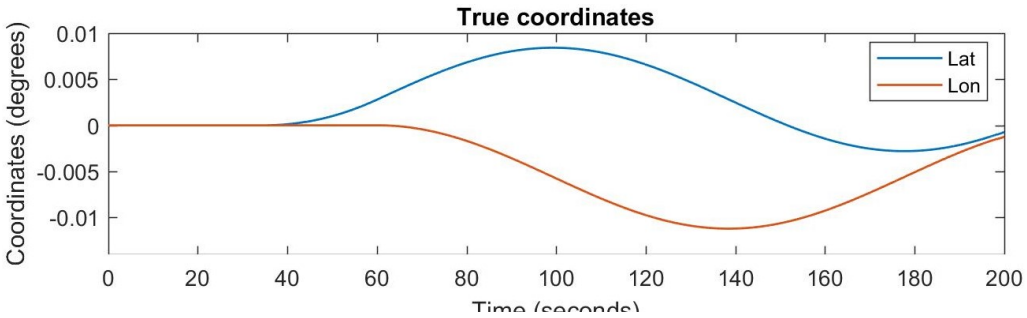

Figure 10.74: True coordinates, all noises enabled

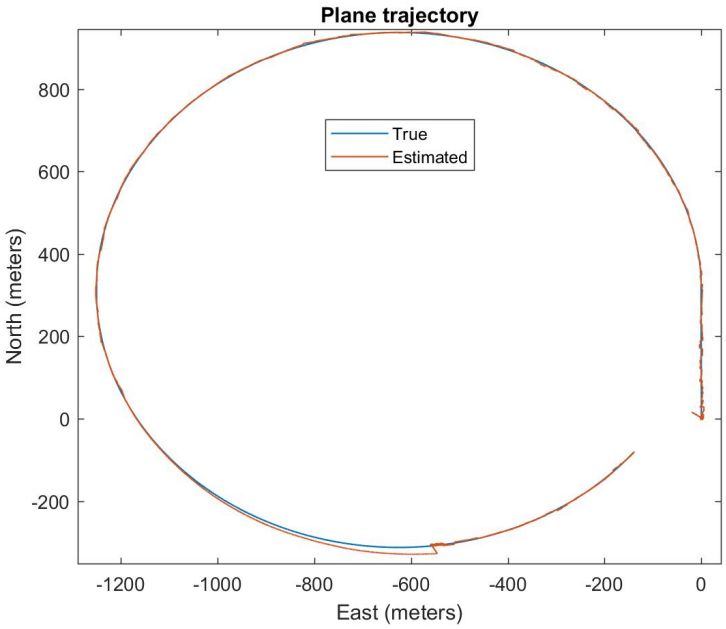

Figure 10.75: Plane trajectory, all noises enabled

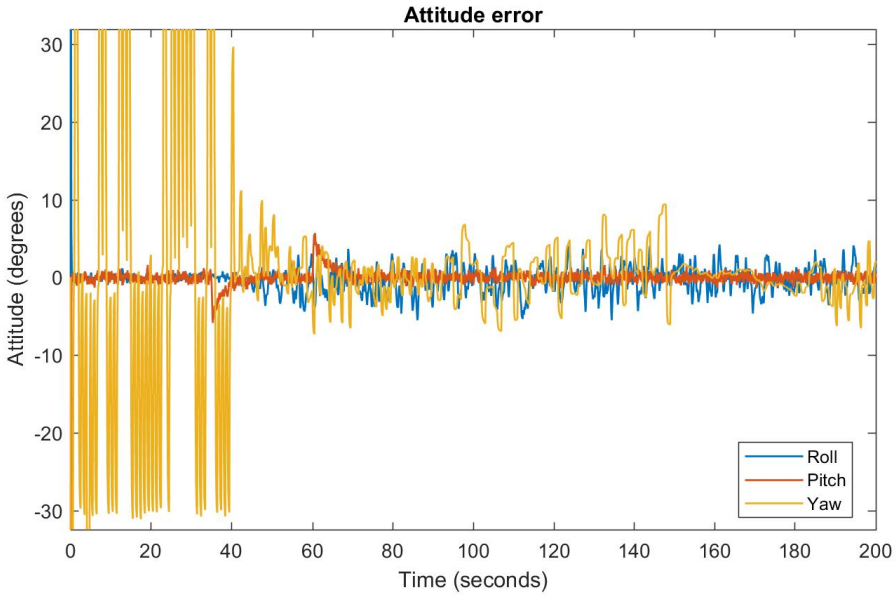

Figure 10.76: Attitude error, all noises enabled

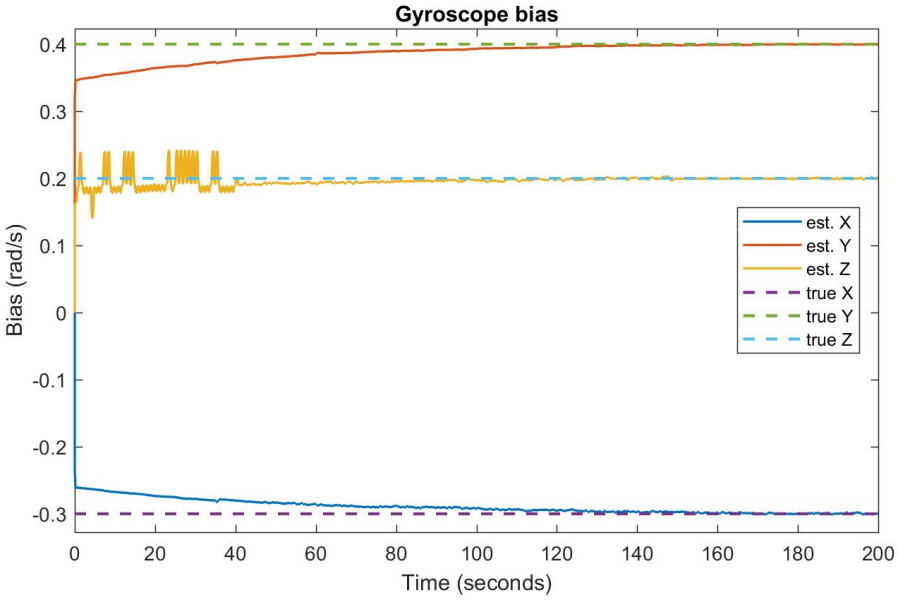

Figure 10.77: Gyroscope bias, all noises enabled

10.3 Accelerometer bias estimation

As seen in the simulation results of this chapter, the proposed two-stage algorithms are able to estimate asymptotically gyroscope bias with a good convergence speed. On the other hand, accelerometer biases show their effect both on attitude and speed/position dynamics of the observer, therefore it is more difficult to estimate the latter ones correctly. As a matter of fact, wrong accelerometer readings provide to the AHRS a wrongly-oriented gravity vector, thus leading to a wrong estimated orientation. As shown in Figure 10.78, the phenomenon of wrong attitude estimation affects both the EKF-based and the NLO-based AHRS algorithms in very similar ways.

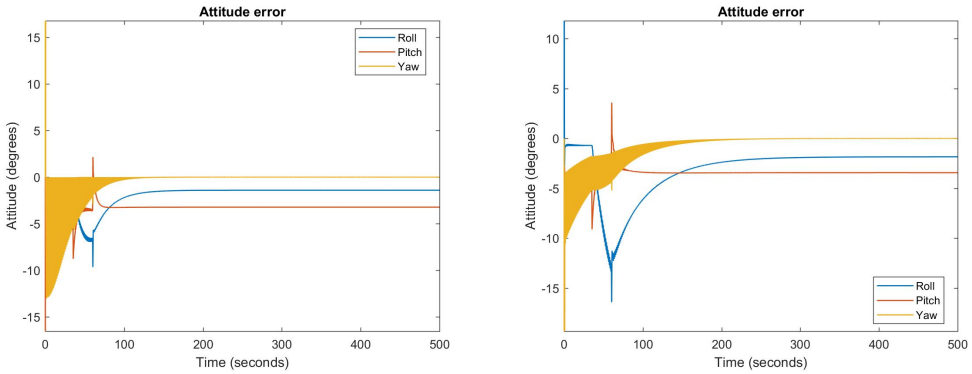

Figure 10.78: Attitude error, EKF-based (left) and NLO-based (right) AHRS

Even if the lateral position error is not as big as one would expect, that's because of the error in the body speed estimation that compensates for it. In fact, speed measurements from GPS are wrongly rotated into the body frame, and then the resulting velocity is rotated back by the opposite rotation when resolving for position.

It must be also pointed out that the navigation EKF estimates accelerometer bias mainly by looking at the speed error and adjusting their estimates accordingly. Hence, even if a wrong body speed partially compensates for the wrong attitude, bias estimation does not converge to the true values, as shown in Figure 10.79.

As a further demonstration, if he navigation EKF is provided the true attitude, bias estimates converge to their true value, as shown in Figure 10.80

However, there is a big error in the elevation component, as expected. The errors along all directions will become very large during extended periods of both GNSS and map outages.

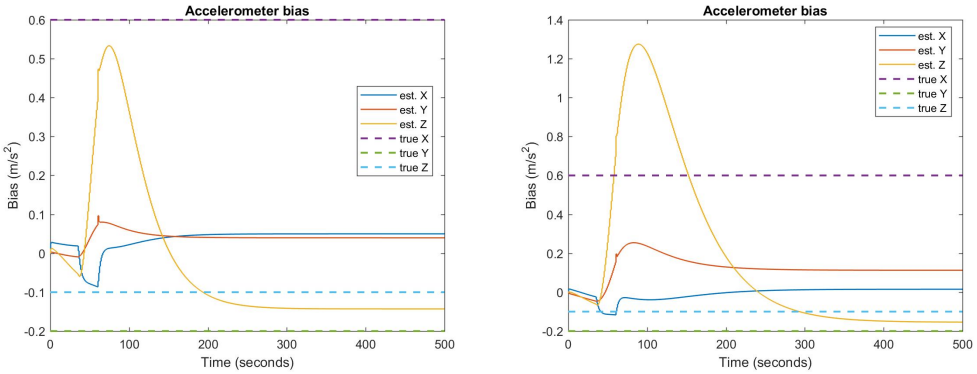

Figure 10.79: Accelerometer bias, double-EKF (left) and EKF-NLO (right), full-order algorithm

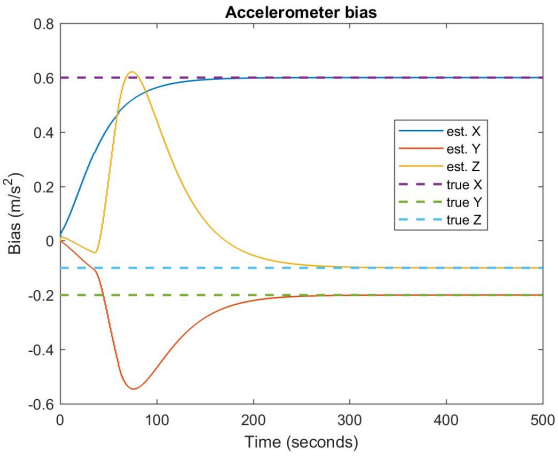

Figure 10.80: Accelerometer bias estimate with true attitude

All the two-stage algorithms presented in Chapter 7 present a subsystem whose dynamics are much faster than the other. In this case, the AHRS converges in very little time to an estimate of the relative orientation and can be considered as a static system with an algebraic input/output relationship. Singular perturbation analysis is a valid tool that can be applied to this framework to study the stability of the interconnection by relying on the results provided by the Thikonov theorem[16, Chapter 11]. The main goal of the analysis could be understanding if the presented algorithms can be adapted to reach convergence of all the estimates or if the two-stage structure is not suitable to this purpose.

10.4 Summary of the results

Concerning the single-EKF structure and its already outlined limitations, its convergence domain turned out to be insufficient for practical applications and showed poor convergence properties for attitude and MEMS bias. As a matter of fact, a non-null unobservable subspace is also present during normal train manoeuvring, that is when there are no sudden accelerations or fast attitude changes.

A number of observability analyses following different approaches are present in literature. It has been shown that the unobservable state space is strongly related to the type of motion the vehicle is subjected to and is generally a linear combination of attitude and MEMS sensors bias[12, 24]. As a result, attitude and frequent acceleration changes are required in order for the overall state space to be completely observable. This also motivates the estimation behaviour outlined in Figure 10.8, in which in particular the gyroscope bias along the direction of the specific force is unobservable during constant linear motion.

On the other hand, the four proposed two-stage algorithms show good performances during dead reckoning. Using a reduced-order model brings to some improvement in position and speed estimation, in particular along the vertical axis, which appears to be the most sensible to errors and prone to drift; however, it has been observed that the advantage brought by the reduced-order structure can be less appreciated in noisy conditions as well as complete dead reckoning.

Both developed AHRS algorithms are able to correctly estimate and compensate for gyroscope biases during the interconnection with both translational model observers, making them quite reliable. As a matter of fact, they could also work by their own without particular problems, provided that proper compensation of vehicle acceleration is performed where dictated from the particular working conditions.

Even if a correct accelerometer bias estimation is not achieved, simulations proved better convergence properties for gyroscope bias and attitude with the two-module algorithms. A final consideration that can be done is that the proposed two-stage structure is not the only possibility to design two interconnected observers; however, it has the advantage of using accelerometers in a more structured way instead of using only GNSS information as measurements.

Chapter 11

Conclusions

Several INS algorithms were proposed in this work. They all make use of a subset of sensors commonly used in IMUs. Simulations have proven that using the accelerometer to provide information on the attitude allows to increase the rate of some information and to converge more rapidly to the correct value. The reduced-order structure for navigation equation, developed to exploit the train kinematic constraints, provided better positioning accuracy during dead reckoning. The modified NLO structure showed to have the same performance of a standard EKF structure, its main advantage being much lower computational load with respect to the other solution and global convergence in all practical situations. Nevertheless, using a two-stage structure allows for greater modularity and flexibility.

Moreover, addition of the map integration algorithm proved to be very useful to improve positioning during GNSS outages. The strongest result obtained is the containment of the position error along the train path and of the speed estimate along suitable values for the vehicle. Simulation results during dead reckoning with or without maps turned out to be very satisfactory and suitable for non-safety critical tasks, such as on-board information to passengers.

The problem of on-line compensation of sensor bias has been addressed. No problems were found for gyroscopes, while some issues arose when dealing with accelerometers because of their usage in several block of the two-stage algorithm presented in Chapter 7.

Future development for this work may be focused on the study of the interconnection stability concerning accelerometer bias estimation. Another possible topic can be related to machine learning applied to maps. If the train runs the same path several times it is possible to collect information from the IMU and adjust the map according to the found discrepancy and the level of confidence of data, as well as fill possible gaps in the trajectory

and improve data granularity, which turned out to be an important factor to have good navigation performance.

Appendix A

MATLAB code for map aiding

```
function [yaw,pos,spd,new_idx] = map_point(v_o,map,x_hat,R_m,
    Rnb,idx_old)
% structure of "map" variable:
% ascissa lat lon h (above Earth centre)
opt_rng = round(10 + 0.3*v_o); % search range depends on
    vehicle speed, plus some margin
st = max(1,idx_old-opt_rng/2); % abbrev. start
fi = min(st+opt_rng,length(map)); % abbrev. finish

t_ff = 0.001; % use to tune feedforward term
x_add = t_ff*diag([ 1/R_m 1/( R_m*cos(x_hat(1)) ) 1])*Rnb*[1
    0 0]'*v_o; % feedforward term

fun = @(i) distance_map(round(i), x_hat+x_add, map, R_m,
    x_hat(1));
opt=optimset('MaxIter',25,'Display','iter','FunValCheck','on'
    ,'TolX',1e-3);
% MaxIter and TolX are tuned by hand
[idx, D] = fminbnd(fun,st,fi,opt);
idx = round(idx);
% output new index and position
new_idx = idx(1);
pos = map(idx(1),2:4)';
% at leat one assignment must be guaranteed
spd = [0 0 0]';

% use finite differences method
st = max(1,new_idx-4);
fi = min(new_idx+4,length(map));
% define coefficients for central difference
coeff6 = [ -1/60 3/20 -3/4 0 3/4 -3/20 1/60];
coeff8 = [ 1/280 -4/105 1/5 -4/5 0 4/5 -1/5 4/105 -1/280];

```

```
% use coeff6 or coeff8 according to the desired level of
    accuracy

if st > (new_idx-4) % if true it means st has been shifted
    because of the beginning of map array
     % use forward difference, I'm at the beginning of the
        array
     coeff6 = [ -49/20 6 -15/2 20/3 -15/4 6/5 -1/6]; %
        coefficients FW difference
     coeff8 = [ -49/20 6 -15/2 20/3 -15/4 6/5 -1/6 0 0];
     st = new_idx;
     fi = st + 8;
else
if fi < (new_idx+4) % similar to the previous if but for the
    end of map array
     % use backward difference, I'm at the end of the array
     coeff6 = [ 1/6 -6/5 15/4 -20/3 15/2 -6 49/20]; %
        coefficients BW difference
     coeff8 = [ 0 0 1/6 -6/5 15/4 -20/3 15/2 -6 49/20];
     fi = new_idx;
     st = fi - 8;
end
end

spd = map(st:fi,2:4)'*coeff8'; % division by step size is
    omitted as scaling will occur later
spd(1) = spd(1)*R_m;
spd(2) = spd(2)*R_m*cos(pos(1));

spd = spd/norm(spd)*v_o; % scale according to odometry speed
yaw = atan2(spd(2),spd(1)); % heading (for AHRS...)
```

There follows the `distance_map` function, which is used to compute the distance between the estimated position and some point on the map:

```
function d=distance_map(index, x_hat, map, R_m, lat)
QD = diag([R_m R_m*cos(lat) 1]); % to convert latitude/
    longitude into meters
if (index>0 && index<=length(map)) % consistency check
dist=sqrt(((x_hat - map(floor(index),2:4)')')*QD*(x_hat - map
    (floor(index),2:4)')); % round index to be sure it's
    integer
else
     dist=realmax;
end
d=dist; % return distance
```

Bibliography

[1] NAVSTAR GPS user equipment introduction, September 1996.

[2] Advanced Navigation. Company website. http://www.advancednavigation.com.au.

[3] Silvère Bonnabel Alex Barrau. The Invariant Extended Kalman Filter as a Stable Observer. *IEEE Transactions on Automatic Control, Vol. 62, No. 4*, pages 1797–1812, April 2017.

[4] Pedro Batista, Carlos Silvestre, Paulo Oliveira, and Bruno Cardeira. Accelerometer calibration and dynamic bias and gravity estimation: Analysis, design, and experimental evaluation. *IEEE Transactions on Control Systems Technology, Vol. 19, No. 5*, pages 1128–1137, September 2011.

[5] S. Beeby, G. Ensel, M. Kraft, and N. White. *MEMS mechanical sensors*. Artech house inc., 2004.

[6] R. S. Bucy and P.D. Joseph. *Filtering for Stochastic Processes with Applications to Guidance*. John Wiley & Sons, 1698.

[7] Damien Douxchamps. Own website. https://damien.douxchamps.net/research/imu/.

[8] Garry A. Einicke. *Smoothing, Filtering and Prediction: Estimating the Past, Present and Future*. InTech, 2012.

[9] European Space Agency (ESA). What is Galileo? http://www.esa.int/Our_Activities/Navigation/Galileo/What_is_Galileo.

[10] Jay A. Farrell. *Aided Navigation, GPS with High Rate Sensors*. McGraw-Hill, 2008.

[11] Richard W Hamming. *Numerical Methods for Scientists and Engineers*. Dover Publications, Inc., New York, NY, USA, 2nd edition, 1986.

[12] Sinpyo Hong, Man Hyung Lee, Ho-Hwan Chun, Sun-Hong Kwon, and Jason L. Speyer. Observability of error states in GPS/INS integration. *IEEE Transactions on Vehicular Technology, Vol. 54, No. 2*, pages 731–743, March 2005.

[13] A. H. Jazwinski. *Stochastic Processes and Filtering Theory*. New York: Academic, 1972.

[14] Simon J. Julier and Jeffrey K. Uhlmann. Unscented Filtering and Nonlinear Estimation. *Proceedings of the IEEE, Vol. 92, No. 3*, March 2004.

[15] R. E. Kalman. A new approach to linear filtering and prediction problems. *ASME Journal of Basic Engineering*, 1960.

[16] Hassan K. Khalil. *Nonlinear Systems*. Pearson, 3rd edition, 2001.

[17] Robert Mahony, Taker Hamel, and Jean-Michel Pflimlin. Nonlinear complementary filters on the special orthogonal group. *IEEE Transactions on Automatic Control, Vol. 53, No. 5*, pages 1203–1218, June 2008.

[18] Jan Rohac, Jakob M. Hansen, Mushfiqul Alam, Martin Sipos, Tor A. Johansen, and Thor I. Fossen. Validation of nonlinear integrated navigation solutions. *Annual Reviews in Control*, 2017.

[19] SBG Systems. Company website. https://www.sbg-systems.com/.

[20] SBG Systems. Ellipse test result. https://www.sbg-systems.com/docs/Ellipse_Test-Automotive.pdf.

[21] Diego Emilio Serrano. Design and analysis of MEMS gyroscopes. In *IEEE Sensors 2013*, 2013.

[22] Sharma and Kumar et al. Accurate navigation of a uav using kalman filter based gps-ins integration. *5th Symposium on Applied Aerodynamics and Design of Aerospace Vehicles*, 2011.

[23] Walter Stockwell. Bias stability measurement: Allan Variance.

[24] Yonggang Tang, Yuanxin Wu, Meiping Wu, Wenqi Wu, Xiaoping Hu, and Lincheng Shen. INS/GPS integration: Global observability analysis. *IEEE Transactions on Vehicular Technology, Vol. 58, No. 3*, pages 1129–1142, March 2009.

[25] David H. Titterton and John L. Weston. *Strapdown Inertial Navigation Technology*. The Institution of Electrical Engineers, 2nd edition, 2004.

[26] Trenitalia. Odometria SCMT - principi generali dell'algoritmo per il calcolo della velocità stimata in caso di pattinamento o slittamento degli assi di misura, 2002.

[27] Trimble. Company website. http://www.trimble.com.

[28] VectorNav. Company website. http://www.vectornav.com/.

[29] Eric A. Wan and Rudolph Van Der Merwe. The unscented kalman filter. In *Kalman Filtering and Neural Networks*, pages 221–280. Wiley, 2001.

[30] Greg Welch and Gary Bishop. An introduction to the kalman filter. Technical report, University of North Carolina at Chapel Hill, Department of Computer Science, 1995.

[31] Oliver J. Woodman. An introduction to inertial navigation. Technical report, University of Cambridge, August 2007.