Technische Universität München

Institute of Flight System Dynamics

Inertial Navigation Algorithms based on Symmetry-preserving theory

Bachelorarbeit

Author: Ferran Casanovas Bargalló

Matriculation Number: 03656625

Supervisor at TUM: Dipl.-Ing. Thomas Raffler

Supervisor at UPC: Dra.-Associate Professor Adeline de Villardi de Montlaur

September 2014

Statutory Declaration

I, Ferran Casanovas Bargalló, declare on oath towards the Institute of Flight System Dynamics of Technische Universität München, that I have prepared the present Bachelor thesis independently and with the aid of nothing but the resources listed in the bibliography.

This thesis has neither as-is nor similarly been submitted to any other university.

Garching, 30 September 2014

Ferran Casanovas Bargalló

Abstract

Nowadays, the usage of low-cost sensors on Unmanned Aerial Vehicles (UAV) has become more popular due to the increase in small UAV platforms. Some of them are used for research on new algorithms in all the different parts of flight dynamics research like flight control, navigation, data fusion, etc. Furthermore, these low-cost and small sensors are used on a huge range of new applications such as filming, photography and agricultural uses. For all of these purposes, the UAVs rely on a host of sensors to position, navigate and compute all the necessary data for the application they are designed for. The key part in that process is to obtain accurate and reliable state estimations from available measurements. Taking into account the comparably weak performance of employed low-cost sensors, algorithms to estimate the measured and non-measured variables are of primary importance. In this thesis, three Inertial Navigation Algorithms for a quadrotor UAV are implemented and compared to show its essential advantage and drawbacks. The first two algorithms are both designed based on Symmetry-preserving theory and differ from each other by the type of estimation: a non-linear observer and an Extended Kalman Filter (EKF), whereas the third one is an EKF without the Symmetry-preserving theory behind it. Besides numeric simulations, these algorithms are applied to real-time data to be able to evaluate the properties of each algorithm.

Table of Contents

List of Figures

List of Tables

Table of Acronyms

Acronym	Description
AHRS	Attitude and Heading Reference System
BIH	Bureau International de l'Heure = International Time Bureau
CTP	Conventional Terrestrial Pole
ECEF	Earth-Centered Earth-Fixed
EKF	Extended Kalman Filter
GNSS	Global Navigation Satellite System
GPS	Global Positioning System

IEKF	Invariant Extended Kalman Filter
IERS	International Earth Rotation and Reference Systems Service
IMU	Inertial Measurement Unit
INA	Inertial Navigation Algorithm / Data-fusion algorithm
INS	Integrated Navigation System
IRM	IERS Reference Meridian
IRP	IERS Reference Pole
KF	Kalman Filter
LIEKF	Left Invariant Extended Kalman Filter
MEKF	Multiplicative Extended Kalman Filter
MEMS	Micro-machined Electro-Mechanical Systems
NED	North-East-Down
RIEKF	Right Invariant Extended Kalman Filter
WGS84	World Geodetic System 1984

Table of Symbols

Symbol	Unit	Description
Attitude parameterization		
$\boldsymbol{\phi}$	rad	Roll angle.
$\boldsymbol{\theta}$	rad	Pitch angle.
$\boldsymbol{\psi}$	rad	Yaw/azimuth angle.
$\breve{\boldsymbol{q}}$	–	4-dimensional quaternion.
$\vec{\boldsymbol{n}}$	–	Normalized vector representing the axis of rotation.
$\boldsymbol{\alpha}$	rad	Angle value to rotate w.r.t $\vec{n}$ axis.
System model		
$\left(\vec{\boldsymbol{r}}^{\boldsymbol{R}}\right)_{\boldsymbol{W}}$	m	Absolute position of the reference point R, written in terms of the w-Frame.
$\left(\vec{\boldsymbol{v}}^{\boldsymbol{R}}\right)_{\boldsymbol{W}}^{\boldsymbol{W}}$	m/s	Relative velocity (that means, derived w.r.t the w-Frame) of the reference point R, written in terms of the w-Frame.
$\left(\vec{\boldsymbol{a}}^{\boldsymbol{R}}\right)_{\boldsymbol{W}}^{\boldsymbol{WW}}$	m/s^2	Relative acceleration (that means, derived twice w.r.t the w-Frame) of the reference point R, written in terms of the w-Frame.
$\left(\vec{\boldsymbol{\omega}}^{\boldsymbol{WB}}\right)_{\boldsymbol{B}}$	rad/s	Angular velocity from the b-Frame to the w-Frame, shown in the b-Frame.

$(\vec{\Psi}^{WB})$	rad	Attitude of the aircraft from the b-Frame to the w-Frame.
$(\vec{B}^{R})_{B}$	T	Relative Earth magnetic field of the reference point R, written in terms of the b-Frame.
$\boldsymbol{p, q, r}$	rad/s	Roll, pitch and yaw rotational rates.
$(\vec{g})_{W}$	m/s^2	The effective Earth's gravity, specified in the w-Frame.
Sensor models (general)		
$\vec{b}_g$	rad/s	Bias vector of the gyroscope model.
$(\vec{\omega}^{WB})_{B,true\ input}$	rad/s	Perfect angular velocity vector without noise.
$(\vec{\omega}^{WB})_{B,input}$	rad/s	Input angular velocity vector, noise model added.
$(\vec{f})_{B,true\ input}$	m/s^2	Perfect acceleration forces vector, i.e. without gravity, without noise.
$(\vec{f})_{B,input}$	m/s^2	Input acceleration forces vector, i.e. without gravity, noise model added.
a) Symmetry-preserving observer and RIEKF		
a_s	–	Scaling-factor of the accelerometer sensor model.
b) MEKF		
$\vec{b}_a$	m/s^2	Bias vector of the accelerometer model.
$\vec{v}_a$	m/s^2	White/Gaussian noise vector of the accelerometer triad.
$\vec{v}_g$	rad/s	White/Gaussian noise vector of the gyroscope triad.
$\boldsymbol{M_a}$	–	Scale factor error matrix of the accelerometer triad.
$\boldsymbol{M_g}$	–	Scale factor error matrix of the gyroscope triad.
$\vec{w}_{b_a}$	m/s^3	Process noise of the acceleration bias-random walk.
$\vec{w}_{b_g}$	rad/s^2	Process noise of the rotational rates bias-random walk.
Kalman Filter without symmetries (general)		
$\vec{x}$		State vector.
$\vec{y}$		Output vector.
$\vec{u}$		Input vector.
$\vec{\omega}$		Process noise vector.
$\boldsymbol{Q}$		Process noise covariance matrix.
$\vec{v}$		Measurement noise vector.

$\boldsymbol{R}$		Measurement noise covariance matrix.
$\boldsymbol{\Phi}$		Transition matrix.
$\boldsymbol{\Gamma}$		Input matrix.
$\boldsymbol{G}$		Process noise input matrix.
$\boldsymbol{H}$		Measurement matrix.
$\boldsymbol{P}$		State covariance matrix.
$\boldsymbol{K}$		Kalman gains (vector or a matrix, depending on the system definition and the variables to be estimated).
$\vec{\boldsymbol{f}}$		Non-linear differential system equations of a 1st order, dynamic system.
$\vec{\boldsymbol{h}}$		Non-linear measurement equation.
$\vec{\tilde{\boldsymbol{x}}}$		Estimated state vector.
$\boldsymbol{\delta\vec{x}}$		Estimation error state vector.
$\vec{\tilde{\boldsymbol{y}}}$		Estimated output vector.
$\boldsymbol{\delta\vec{y}}$		Difference between the measured expected value and the real one.
$\boldsymbol{A}$		Discretized system state matrix.
$\boldsymbol{B}$		Discretized system noise matrix.
$\boldsymbol{\delta t}$	s	Time step of the discrete-time algorithm.
MEKF		
$\vec{\boldsymbol{\psi}}_{w\tilde{w}}$	rad	Attitude error vector.
$\boldsymbol{\Psi}_{w\tilde{w}}$		Attitude error as a direction cosines matrix.
$\boldsymbol{u, v, w}$	m/s	Velocity components relative to the w-Frame.
Symmetry-preserving theory (general)		
$\boldsymbol{\varphi_g(x)}$		Local diffeomorphism depending on the state vector.
$\boldsymbol{\psi_g(u)}$		Local diffeomorphism depending on the input vector.
$\boldsymbol{\rho_g(y)}$		Local diffeomorphism depending on the output vector.
$\boldsymbol{E}$		Invariant output error vector.
$\boldsymbol{\eta}$		Invariant state error vector.
$\boldsymbol{I}$		Complete free known invariant vector.
$\boldsymbol{\Upsilon}$		Smooth function.

Symmetry-preserving observer and RIEKF

$(\vec{B})_{W,nom}$	–	Nominal Earth magnetic field (constant value).
$\vec{I}_w$	rad/s	Estimated rotational rate invariant vector.
$\vec{I}_a$	m/s^2	Estimated specific acceleration invariant vector.
$\vec{E}_v$	m/s	Invariant output error on the velocity measurement.
$\vec{E}_B$	–	Normalized invariant output error on the magnetic field measurement.
$R_g(x)$		Right translation of x by g.
$L_g(x)$		Left translation of x by g.
M		Process noise covariance matrix.
$\vec{w}(t)$		Process noise vector.
N		Measurement noise covariance matrix.
$\vec{v}(t)$		Measurement noise vector.

1 Introduction

Section 1 explains the motivation to bring to fruition this work and the main objectives of this thesis. A description of the parts of this work is also included and a small explanation of each of the parts is offered to the writer to have an overview of the contents of each of the sections.

1.1 Motivation

In the abstract of this work, the usual concept of “economical sensors” is mentioned. Such a concept is used to refer to small and lightweight Micro-machined Electro-Mechanical Systems (MEMS) inertial sensors. Recent improvements in the performance of MEMS have resulted in an increased interest in the topic of inertial navigation.

A combination of measurements and simulation is used to improve the error characteristics of inertial systems. Sensor fusion and specific constraints on the domain, i.e. domain of convergence of the algorithms, can be used to reduce errors in the Integrated Navigation System (INS). For instance, it is known from [1] that sensor fusion using magnetometers can reduce the average error in position obtained by the system.

The motivation of this work is to improve such sensor fusion algorithms to use MEMS in a more reliable manner and, by this way, enhance the acquired sensor data, which is so important for the inertial navigation systems.

1.2 Goals and objectives

Our main purposes are to implement three different types of data fusion algorithms, in this work also called INA, to compare them and conclude on the advantages and disadvantages of each one. A secondary purpose is to understand new ways of designing an observer, method that is open to lots of manners and that most of the times is not easy to perform. The design of this type of algorithm is not straightforward, because a decoupling of the different subsystems of the main system has to be used for the measurements not to interfere with the non-desired states. Another goal is to check if the results show that there is one INA better than all others in all of the aspects or, as I expected at the beginning, if one is better in an specific situation but not another. So in such a situation, each INA should be used for a specific purpose depending on the kind of system, which we are dealing with.

1.3 Overview of the chapters

In section 2 the main concepts regarding mathematical tools, reference frames, parameterization, differential geometry, symmetry and invariance theory and data fusion algorithms are explained.

Section 3 is used to clarify the types of algorithms we are working on in a general point of view.

The following three sections are based each of them on describing the symmetry-preserving observer (section 4), the Right Invariant Extended Kalman Filter (RIEKF) (section 5) and the normal Multiplicative Extended Kalman Filter (MEKF) (section 6).

In section 7, a comparison between each INA will be performed. The choice of the best gain parameters and every other tunable parameter of the algorithms done in the corresponding chapters will be assumed.

In the same chapter, real-time data instead of the true data without noise will be used. That means, we will not be able to have a reference and reliable value and a more general judgment based on our previous knowledge should be made to differentiate the correct behaviors of the incorrect ones.

To finalize, the results will be discussed and conclusions will be deduced. At the end, there is some space kept for acknowledgements and references to the used sources.

2 Principal necessary concepts

2.1 Navigation System Architecture

The different parts of a Navigation System Architecture model are:

- Inertial Measurement Unit (IMU): can be seen as the output of the inertial sensors, which of course are noisy, with some calibrations and compensations. Normally, it contains a triad of accelerometers, a triad of gyroscopes and in some cases a triad of magnetometers.
- Integrated Navigation System (INS): is the block where the INA (or data fusion block) is applied and where the states that are measured are improved and updated. It contains the inertial navigation equations, some In-flight calibration block, the synchronization for the input data at different frequencies, the integrity monitoring part, etc.
- Other sensors: sensors like a Global Navigation Satellite System (GNSS) receiver, which gives the position, velocity, time, etc. referenced to the e-Frame, and a barometer, altitude measurement, are not included inside the IMU as they are not “inertial sensors”.
- Inertial Navigation system: is similar to the INS but without the part of Data fusion. [2]

2.2 Low-cost navigation systems

Depending on the accuracy of the inertial sensors, two types of navigation systems are differenced:

- True inertial navigation systems: based on the Schuler effect thanks to very accurate inertial sensors.
- Low-cost navigation systems: based on a Flat Earth assumption and low accurate inertial sensors.

The latters can be decoupled in two types:

- Attitude and Heading Reference System (AHRS) normally completed with magnetometers.
- Aided AHRS, when they have an additional velocity and/or position sensors like a GNSS sensor. [3]

In our case, it is utilized an aided AHRS with the current inertial sensors (accelerometers and gyroscopes) complemented with magnetometers measurements and with an additional velocity measurement from the Global Positioning System (GPS) sensor.

2.3 Reference frames

In the whole work, we used three different reference frames, which are at the same time normally the most common ones. A good reference to check is [2].

2.3.1 World geodetic System 1984 (WGS84) (e-Frame)

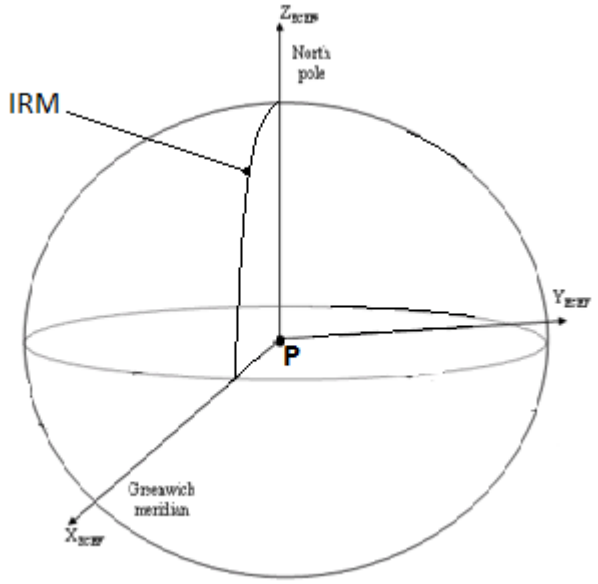

Figure 2-1: Representation of the e-Frame.

The e-Frame is centered in P (Centre of mass of the Earth). The X-axis is the intersection of the IERS Reference Meridian (IRM) and the plane passing through the origin and normal to the Z-axis. The Z-axis goes in the direction of the IERS Reference Pole (IRP). The IRM is coincident with the BIH Zero Meridian (epoch 1984.0) with an uncertainty of 0.005" and the IRP corresponds to the direction of the BIH Conventional Terrestrial Pole (CTP) at the same epoch and with the same uncertainty. The Y-axis completes a right-handed Earth-Centered Earth-Fixed (ECEF) orthogonal coordinate system.

2.3.2 Horizontal, North-Indicating Rotational Reference Frame (n-Frame) = North-East-Down Frame (NED-Frame)

The n-Frame origin is somewhere along the local vertical on the WGS84 Ellipsoid, normally is an arbitrary point of the aircraft.

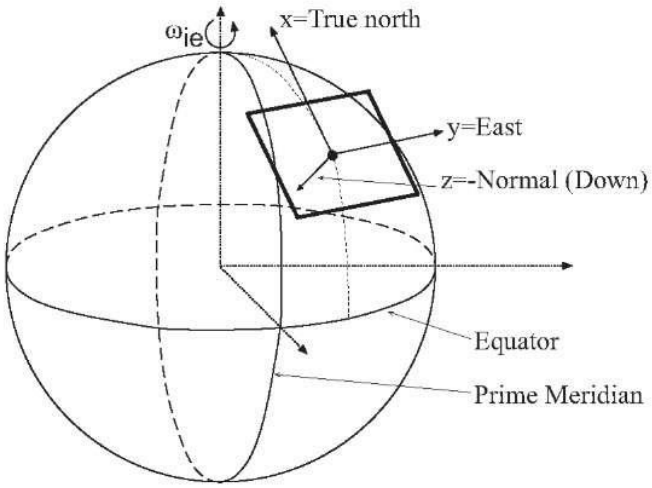

Figure 2-2: Representation of the n-Frame.

As it can be seen from the image extracted from [4], the frame of reference is rotational, i.e. a position vector is not defined and it depends on the position of the aircraft at each moment.

The X-axis and the Y-axis points to the direction of the North and the East, respectively, in the tangent plane on the WGS84 Ellipsoid. The Z-axis points down in the nadiral direction with respect of the position of the aircraft.

The data also takes into account the **World Frame (W)** referenced on the camera, but we considered that this frame is the same as the n-Frame, which will be enough for our purposes.

A brief definition of this **w-Frame** is extracted from [5]and translated:

The origin of the frame is the reference point of the camera. The Z-axis points downward in the direction of the gravity vector. The X-axis and Y-axis complete a right-handed reference system. The last degree of freedom is determined by the arrangement of the cameras.

This frame is just described because we are going to use a set of real data extracted in this frame. This means that the inputs of the INA are appropriate for the data in this frame.

Although, we considered to be like this in the real data usage, that is not a real-time usage of the INA. This means that the inputs will be slightly changed for the real-time case.

2.3.3 Body-Fixed("Strapdown") Reference Frame (b-Frame)

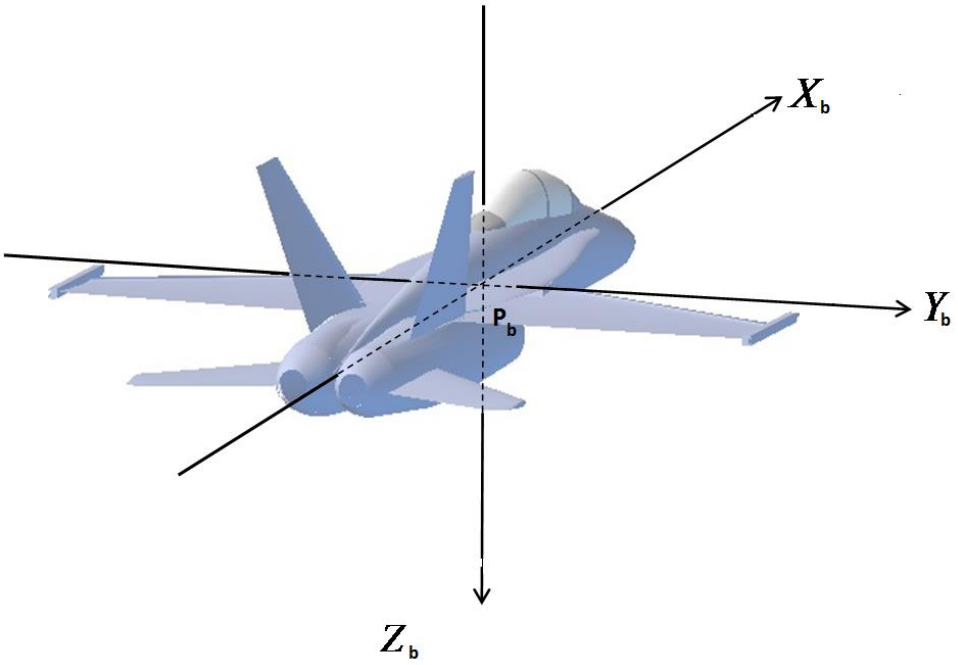

Figure 2-3: Representation of the b-Frame.

The last and more important reference frame is the one described in the picture above extracted from [6]. The origin P_b is an arbitrary reference point of the aircraft, which is not necessary the gravity center of the aircraft. The X-axis is the longitudinal roll axis of the aircraft (positive to the front). The Y-axis is the lateral pitch axis (positive to the right). And the Z-axis is the vertical yaw axis (positive downwards).

2.4 Nomenclature

2.4.1 Nomenclature of the true data

The real data used to compare the INAs is data without any kind of noise. By this way, we can also control the type of noise that we add into the sensors and define the exact output appropriate for our purposes.

The designations used in this work are based largely on the customary notation of the Institute of Flight System Dynamics. The following table describes the nomenclature and notation of the data we are going to use in the real case (this means without noise).

Variable	Description
$\left(\vec{r}^{R}\right)_{W}$	Absolute position of the reference point R, written in terms of the w-Frame.
$\left(\vec{v}^{R}\right)_{W}^{W}$	Relative velocity (that means, derived w.r.t the w-Frame) of the reference point R, written in terms of the w-Frame.

$(\vec{a}^R)_W^{WW}$	Relative acceleration (that means, derived twice w.r.t the w-Frame) of the reference point R, written in terms of the w-Frame.
$(\vec{\omega}^{WB})_B$	Angular velocity from the b-Frame to the w-Frame, shown in the b-Frame.
$(\vec{\Psi}^{WB})$	Attitude of the aircraft from the b-Frame to the w-Frame.
$(\vec{B}^R)_B$	Relative Earth magnetic field of the reference point R, written in terms of the b-Frame.
$(\vec{g})_W$	The effective Earth's gravity, specified in the w-Frame.

Table 2-1: Input data nomenclature.

Physical quantities without explicitly specified unit are always written in the International System of Units (SI). See Table of Symbols for the unit description. The data examined here is perturbed with the noise models defined in 3.3.1 for the measurements and the ones outlined in 4.1 and 6.1 for the inputs or IMU data. For this type of input data are used the sensor models described in 3.3.1 and in each of the specific algorithms.

2.4.2 Nomenclature of the real-time data

This other type of data is extracted from the real sensors of the quadrocopter and already contains real noise. In this case, we do not know exactly which kind of noise is added but is the most realistic scene that we can handle with.

The added elements used are described in the following table:

Variable	**Description**
$(\vec{v}^R)_W^E$	Relative velocity (that means, derived w.r.t the e-Frame) of the reference point R, written in terms of the w-Frame.
$(\vec{f}^R)_B$	Relative acceleration forces, without gravity vector, of the reference point R, written in terms of the b-Frame.
$(\vec{\omega}^{IB})_B$	Angular velocity from the b-Frame to the inertial-Frame (the same as w-Frame), shown in the b-Frame.
$(\vec{B}^R)_B$	Relative Earth magnetic field of the reference point R, written in terms of the b-Frame.

Table 2-2: Input real-time data nomenclature.

Physical quantities without explicitly specified unit are always written in the International System of Units (SI). See Table of Symbols for the unit description. In this case, the noise of the variables is not a model, is the real noise. This means that it is not accurately known which distortion is introduced in the used data, but the INAs have to deal with this uncontrolled noises and perturbations. For the real-time data no sensor model is used, because the data is completely real and the noise is already contained in it.

2.5 Parameterization

2.5.1 Overview

The navigation equations or derivation of the Strapdown Inertial Navigation ODEs, which define our physical system to estimate, is a key point. To determine them, a relation between Position vectors of two different coordinate systems is normally used.

A closer examination of the figure extracted from [7] that is also shown in [2] can be done like a starting point to develop the rotation matrices that will govern our physical system:

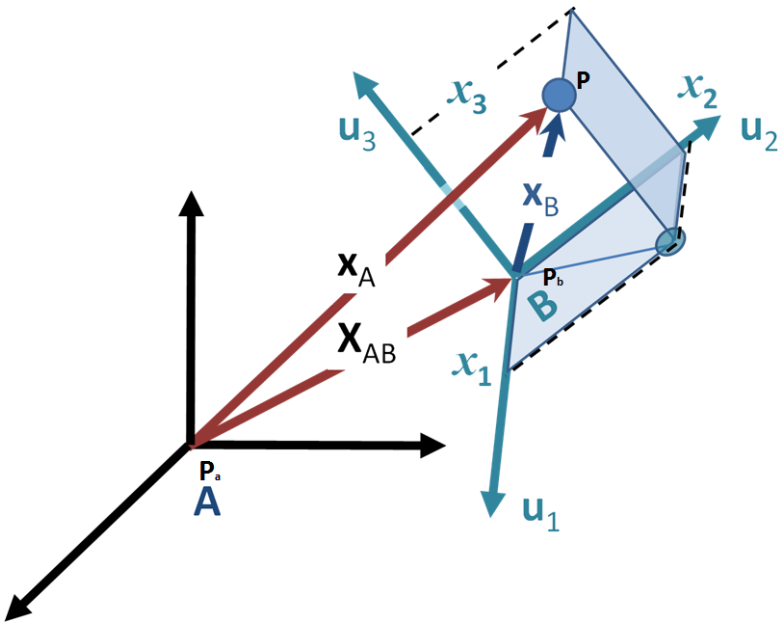

Figure 2-4: Relation between Position vectors of two different coordinate systems.

These rotation matrices display the same as the Euler angles and the quaternions. For a derivation of these tools, better refer to the literature in [2].

It has to be clarified that two different types of rotations can be performed: the active and the passive rotation.

At the former case, the rotation is performed on the vector, but at the latter case the coordinate system is rotated.

We use the passive rotation through all over this work.

2.5.2 Rotation matrix

The special orthogonal group $SO(3) = SO_3(\mathbb{R}) = SO(3, \mathbb{R})$ described afterwards at 2.6.3.5 is the one which contains all the orthogonal matrices of dimension 3x3 with the characteristics of the group defined below.

In these group are classified the rotation matrix that will be used. To show how it works, the following example performs a passive rotation of the b-Frame to the w-Frame (or what is the same to the n-Frame):

$$(\vec{r})_W = R_{WB} \cdot (\vec{r})_B \tag{2-1}$$

The derivation of the rotation matrix R_{WB} is given by the differential equation:

$$\dot{R}_{WB} = R_{WB} \cdot (\Omega^{WB})_B \tag{2-2}$$

For this rotation matrix, there are 2 different parameterizations or representations: The Euler angles and the quaternions. Both of them are explained in the following sections.

2.5.3 Euler angles

The most common parameterization is the Euler angles. They described three consecutive rotations around one different axis each of them, their order is fixed.

As an example, the rotation matrix from w-Frame to b-Frame as a concatenation of three elemental rotations is shown:

$$R_{BW} = R_x(\phi) \cdot R_y(\theta) \cdot R_z(\psi) \tag{2-3}$$

Where $= \ roll\ angle$, $\theta = \ pitch\ angle$, $\psi = \ azimut/yaw\ angle$.

Each of the above rotation matrices are defined as:

$$R_x(\phi) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{pmatrix}, \qquad R_y(\theta) = \begin{pmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{pmatrix},$$
$$R_z(\psi) = \begin{pmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2-4}$$

Which yields to:

$$R_{BW} = \begin{pmatrix} \cos\theta \cdot \cos\psi & \cos\phi \cdot \sin\psi & -\sin\theta \\ \sin\phi \cdot \sin\theta \cdot \cos\psi - \cos\phi \cdot \sin\psi & \sin\phi \cdot \sin\theta \cdot \sin\psi + \cos\phi \cdot \cos\psi & \cos\phi \cdot \sin\theta \\ \cos\phi \cdot \sin\theta \cdot \cos\psi + \sin\phi \cdot \sin\psi & \cos\phi \cdot \sin\theta \cdot \sin\psi - \sin\phi \cdot \cos\psi & \cos\phi \cdot \cos\theta \end{pmatrix}$$

The inverse rotation is performed by the transpose of the matrix above, due to the orthogonal property of the group of matrices mentioned in 2.6.3.2.

$$R_{WB} = R_z(-\psi) \cdot R_y(-\theta) \cdot R_x(-\phi) = {R_{Bw}}^t \tag{2-5}$$

Which is the same as:

$$R_{WB} = \begin{pmatrix} \cos\theta \cdot \cos\psi & \sin\phi \cdot \sin\theta \cdot \cos\psi - \cos\phi \cdot \sin\psi & \cos\phi \cdot \sin\theta \cdot \cos\psi + \sin\phi \cdot \sin\psi \\ \cos\theta \cdot \sin\psi & \sin\phi \cdot \sin\theta \cdot \sin\psi + \cos\phi \cdot \cos\psi & \cos\phi \cdot \sin\theta \cdot \sin\psi - \sin\phi \cdot \cos\psi \\ -\sin\theta & \sin\phi \cdot \cos\theta & \cos\phi \cdot \cos\theta \end{pmatrix}$$

As we discuss in the2.5.4, there are some singularities for this parameterization when the pitch angle is equal to $\pm 90°$. This problem expands for the three Euler angles, because the calculation of the roll and yaw angles depends on the pitch angle.

For this reason, another parameterization is stated in the following section and used with all the observers, i.e. the quaternion representation of a rotation.

A good illustrated reference to check a different definition of the Euler angles can be found in [6].

2.5.4 Quaternions

The definition of the quaternion and the operations with this mathematical tool, have to be defined in an appropriate manner at the beginning to use them in the correct way.

The quaternions were first described in 1843 by the Irish mathematician William Rowan Hamilton on the 16th of October of 1843. The theory developed by Sir. Hamilton uses 3 complex elements and can be seen as an extension of the complex set of numbers denoted by $\mathbb{C}$.

The quaternion group is specified by $\mathbb{H}$, and is defined as:

$$\mathbb{H} = \{\boldsymbol{q} = q_0 + q_1 i + q_2 j + q_3 k \,|q_t \in \mathbb{R}\, where\ t \in \{0,1,2,3\}, \\ i^2 = j^2 = k^2 = i \cdot j \cdot k = -1\} \quad \textbf{(2-6)}$$

It can be also represented in a form of a vector as:

$$\breve{q} = \begin{pmatrix} q_0 \\ \vec{q} \end{pmatrix} = \begin{pmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{pmatrix} \quad \textbf{(2-7)}$$

Another way of defining the quaternion for us to be easier to understand is:

$$\breve{q} = \begin{pmatrix} q_0 \\ \vec{q} \end{pmatrix} = \begin{pmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{pmatrix} = \begin{pmatrix} \cos(\alpha/2) \\ \vec{n} \cdot \sin(\alpha/2) \end{pmatrix} \quad \textbf{(2-8)}$$

Where $\vec{n}$ is an element of the set $\mathbb{R}^3$ that means is a 3-element vector $\vec{n} \in \mathbb{R}^3$. This vector defines the axis of the rotation. Moreover, α is an angle showing the degrees of rotation with respect to the axis $\vec{n}$.

Taking into account the approximation for **small angles** $\begin{matrix} \boldsymbol{\cos(\alpha/2) \longrightarrow 1} \\ \boldsymbol{\sin(\alpha/2) \longrightarrow \alpha/2} \end{matrix}$ the equation above yields to:

$$\breve{q} = \begin{pmatrix} 1 \\ \vec{n} \cdot \alpha/2 \end{pmatrix} \quad \textbf{(2-9)}$$

Furthermore, the operations of this mathematical set must be also defined:

Norm

$$||\breve{q}|| = \sqrt{{q_0}^2 + {q_1}^2 + {q_2}^2 + {q_3}^2} \tag{2-10}$$

Multiplication

Viewing $\boldsymbol{q}$ and $\boldsymbol{p}$ as 2 different elements of the set $\mathbb{H}$, the quaternion multiplication $*$writes:

$$\begin{aligned}\breve{p} * \breve{q} &= \begin{pmatrix} p_0 q_0 - \overrightarrow{p^t} \cdot \vec{q} \\ p_0\vec{q} + q_0\vec{p} + \vec{p} \times \vec{q} \end{pmatrix} = \begin{pmatrix} p_0 & -\overrightarrow{p^t} \\ \vec{p} & p_0 I_3 + [\vec{p}\times] \end{pmatrix} \cdot \begin{pmatrix} q_0 \\ \vec{q} \end{pmatrix} \\ &= \begin{pmatrix} p_0 & -p_1 & -p_2 & -p_3 \\ p_1 & p_0 & -p_3 & p_2 \\ p_2 & p_3 & p_0 & -p_1 \\ p_3 & -p_2 & p_1 & p_0 \end{pmatrix} \cdot \begin{pmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{pmatrix}\end{aligned} \tag{2-11}$$

Conjugate

The conjugate of this mathematical tool is:

$$\overline{\breve{q}} = \begin{pmatrix} q_0 \\ -\vec{q} \end{pmatrix} = \begin{pmatrix} q_0 \\ -q_1 \\ -q_2 \\ -q_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \cdot \begin{pmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{pmatrix} \tag{2-12}$$

Inverse

The inverse of the quaternion can be seen as: [8]

$$\breve{q}^{-1} = \frac{\overline{\breve{q}}}{||\breve{q}||^2} \; in\; case\; ||\breve{q}|| = 1 \xrightarrow{yields} \breve{q}^{-1} = \overline{\breve{q}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \cdot \begin{pmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{pmatrix} \tag{2-13}$$

It is also important to designate the identity element of the set $\mathbb{H}$, $\breve{e} = \begin{pmatrix} 1 \\ \vec{0} \end{pmatrix}$and the property or equality of the inverse of the multiplication between 2 quaternions: $(\breve{p} * \breve{q})^{-1} = \breve{q}^{-1} * \breve{p}^{-1}$.

Quaternions representing rotations

The definition of a unit quaternion is the one with the norm equal to 1: $||\breve{\boldsymbol{q}}|| = \mathbf{1}$.

This property is very useful because enables us to represent the attitude of a rigid body as an alternative to the rotation matrix in a three-dimensional space. The advantage of using quaternions instead of Euler angles is that the formers do not have singularities and can only be represented as one specific Euler angle. At the same time the drawback is that a quaternion and its negative represents the same rotation, thing that does not happen with the rotation matrix as there is only one rotation matrix for each existing rotation.

Besides, any vector $\vec{v} \in \mathbb{R}^3$ can be used in a quaternion form like: $\breve{v} = \begin{pmatrix} 0 \\ \vec{v} \end{pmatrix}$. From now, we systematically identify vectors $\vec{v} \in \mathbb{R}^3$with the corresponding quaternion $\breve{v} \in \mathbb{H}$.

$$\begin{pmatrix} 0 \\ (\vec{v})_W \end{pmatrix} = \breve{q}_{WB} * \begin{pmatrix} 0 \\ (\vec{v})_B \end{pmatrix} * (\breve{q}_{WB})^{-1} = \breve{q}_{WB} * \begin{pmatrix} 0 \\ (\vec{v})_B \end{pmatrix} * \breve{q}_{BW} \quad \textbf{(2-14)}$$

And this equals to:

$$\begin{pmatrix} 0 \\ (\vec{v})_W \end{pmatrix} = R(\breve{q}_{WB}) \cdot \begin{pmatrix} 0 \\ (\vec{v})_B \end{pmatrix} \quad \textbf{(2-15)}$$

Where $R(\breve{q}_{WB}) \in SO(3)$.

Quaternion differential equation

The time derivative of a quaternion which represents a rotation is given by the following relationship. On the basis of the simple calculation and the absence of singularities that the quaternion provides to us, it is ideal for use in real-time calculation tools. The preservation of the norm has to be ensured, though.

$$\dot{\breve{q}}_{WB} = \frac{1}{2} \cdot \breve{q}_{WB} * \begin{pmatrix} 0 \\ (\vec{\omega}^{WB})_B \end{pmatrix} \quad \textbf{(2-16)}$$

2.5.5 Transformations

This part states all the transformations between quaternions, Euler angles and rotation matrices that will be used in this thesis.

Rotation matrix from quaternion

The rotation matrix calculated from the quaternion $\breve{q}$ writes:

$$R(\breve{q}) = \begin{pmatrix} {q_0}^2 + {q_1}^2 - {q_2}^2 - {q_3}^2 & 2(q_1 \cdot q_2 - q_0 \cdot q_3) & 2(q_1 \cdot q_3 + q_0 \cdot q_2) \\ 2(q_1 \cdot q_2 + q_0 \cdot q_3) & {q_0}^2 - {q_1}^2 + {q_2}^2 - {q_3}^2 & 2(q_2 \cdot q_3 - q_0 \cdot q_1) \\ 2(q_1 \cdot q_3 - q_0 \cdot q_2) & 2(q_2 \cdot q_3 + q_0 \cdot q_1) & {q_0}^2 - {q_1}^2 - {q_2}^2 + {q_3}^2 \end{pmatrix} \quad \textbf{(2-17)}$$

Euler angles from quaternion

A closer examination of the rotation matrix can be made taking into account the relationship:

$$R_{WB} = R(\breve{q}_{WB}) = R_3(\psi) \cdot R_2(\theta) \cdot R_1(\phi)$$

$$= \begin{pmatrix} {q_{WB_0}}^2 + {q_{WB_1}}^2 - {q_{WB_2}}^2 - {q_{WB_3}}^2 & 2\left(q_{WB_1} \cdot q_{WB_2} - q_{WB_0} \cdot q_{WB_3}\right) & 2\left(q_{WB_1} \cdot q_{WB_3} + q_{WB_0} \cdot q_{WB_2}\right) \\ 2\left(q_{WB_1} \cdot q_{WB_2} + q_{WB_0} \cdot q_{WB_3}\right) & {q_{WB_0}}^2 - {q_{WB_1}}^2 + {q_{WB_2}}^2 - {q_{WB_3}}^2 & 2\left(q_{WB_2} \cdot q_{WB_3} - q_{WB_0} \cdot q_{WB_1}\right) \\ 2\left(q_{WB_1} \cdot q_{WB_3} - q_{WB_0} \cdot q_{WB_2}\right) & 2\left(q_{WB_2} \cdot q_{WB_3} + q_{WB_0} \cdot q_{WB_1}\right) & {q_{WB_0}}^2 - {q_{WB_1}}^2 - {q_{WB_2}}^2 + {q_{WB_3}}^2 \end{pmatrix} \quad \textbf{(2-18)}$$

$$= \begin{pmatrix} \cos\theta \cdot \cos\psi & \sin\phi \cdot \sin\theta \cdot \cos\psi - \cos\phi \cdot \sin\psi & \cos\phi \cdot \sin\theta \cdot \cos\psi + \sin\phi \cdot \sin\psi \\ \cos\theta \cdot \sin\psi & \sin\phi \cdot \sin\theta \cdot \sin\psi + \cos\phi \cdot \cos\psi & \cos\phi \cdot \sin\theta \cdot \sin\psi - \sin\phi \cdot \cos\psi \\ -\sin\theta & \sin\phi \cdot \cos\theta & \cos\phi \cdot \cos\theta \end{pmatrix}$$

So the easiest element to calculate is theta θ selecting the element R_{31}:

$$\theta = asin\left(-2\left(q_{WB_1} \cdot q_{WB_3} - q_{WB_0} \cdot q_{WB_2}\right)\right)$$
$$\Downarrow$$
$$unique\ for\ \theta \in \left[-\frac{\pi}{2}; \frac{\pi}{2}\right] \quad \textbf{(2-19)}$$

From the elements $R_{32}$and R_{33} , also phi ϕ can be determined:

$$\sin\phi = \frac{2\left(q_{WB_2} \cdot q_{WB_3} + q_{WB_0} \cdot q_{WB_1}\right)}{\cos\theta}, \cos\phi = \frac{{q_{WB_0}}^2 - {q_{WB_1}}^2 - {q_{WB_2}}^2 + {q_{WB_3}}^2}{\cos\theta}$$

$$\phi = atan2\left(2\left(q_{WB_2} \cdot q_{WB_3} + q_{WB_0} \cdot q_{WB_1}\right), {q_{WB_0}}^2 - {q_{WB_1}}^2 - {q_{WB_2}}^2 + {q_{WB_3}}^2\right) \quad \textbf{(2-20)}$$
$$\Downarrow$$
$$\phi \in [-\pi; \pi)\ unique\ for\ \theta \in \left(-\frac{\pi}{2}; \frac{\pi}{2}\right)\ and\ non-unique\ for\ \theta \in \pm\frac{\pi}{2}$$

With the *atan2* function, the singularity can be treated.

Going on with the elements $R_{11}$and R_{21} , also phi ϕ can be determined:

$$\sin\psi = \frac{2\left(q_{WB_1} \cdot q_{WB_2} + q_{WB_0} \cdot q_{WB_3}\right)}{\cos\theta}, \cos\psi = \frac{{q_{WB_0}}^2 + {q_{WB_1}}^2 - {q_{WB_2}}^2 - {q_{WB_3}}^2}{\cos\theta}$$

$$\psi = atan2\left(2\left(q_{WB_1} \cdot q_{WB_2} + q_{WB_0} \cdot q_{WB_3}\right), {q_{WB_0}}^2 + {q_{WB_1}}^2 - {q_{WB_2}}^2 - {q_{WB_3}}^2\right) \quad \textbf{(2-21)}$$
$$\Downarrow$$
$$\psi \in [-\pi; \pi)\ unique\ for\ \theta \in \left(-\frac{\pi}{2}; \frac{\pi}{2}\right)\ and\ non-unique\ for\ \theta \in \pm\frac{\pi}{2}$$

Quaternion from Euler angles

The transformation from Euler angles to quaternion directly gives to us the quaternion representing the rotation to the World frame (W), or Earth frame (E) in case there is no flat earth assumption, with respect to the Body frame (B) due to the Euler angles are the angles that relate these 2 frames.

$$\breve{q}_{WB} = \begin{pmatrix} \cos\frac{\phi}{2}\cdot\cos\frac{\theta}{2}\cdot\cos\frac{\psi}{2} + \sin\frac{\phi}{2}\cdot\sin\frac{\theta}{2}\cdot\sin\frac{\psi}{2} \\ \sin\frac{\phi}{2}\cdot\cos\frac{\theta}{2}\cdot\cos\frac{\psi}{2} - \cos\frac{\phi}{2}\cdot\sin\frac{\theta}{2}\cdot\sin\frac{\psi}{2} \\ \cos\frac{\phi}{2}\cdot\sin\frac{\theta}{2}\cdot\cos\frac{\psi}{2} + \sin\frac{\phi}{2}\cdot\cos\frac{\theta}{2}\cdot\sin\frac{\psi}{2} \\ \cos\frac{\phi}{2}\cdot\cos\frac{\theta}{2}\cdot\sin\frac{\psi}{2} - \sin\frac{\phi}{2}\cdot\sin\frac{\theta}{2}\cdot\cos\frac{\psi}{2} \end{pmatrix} \quad \textbf{(2-22)}$$

2.5.6 Skew-symmetric matrix

The definition of this mathematical tool is extracted from [2] and it reads:

The "*vector skew(veck)*-operator" is a special operator that can be applied on skew-symmetric matrices $\{\boldsymbol{\Omega} \in \mathbb{R}^{33} | \boldsymbol{\Omega}^T = -\boldsymbol{\Omega}\}$:

$$veck(\boldsymbol{\Omega}) = veck\begin{pmatrix} 0 & -\omega_z & -\omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix} = \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} = \vec{\omega} \quad \textbf{(2-23)}$$

The $veck^{-1}$-operator creates a skew-symmetric matrix from a vector:

$$[(\vec{\omega})\times] = veck^{-1}(\vec{\omega}) = veck^{-1}\begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} = \begin{pmatrix} 0 & -\omega_z & -\omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix} = \boldsymbol{\Omega} \quad \textbf{(2-24)}$$

With the $veck^{-1}$-operator, an outer product can be expressed by a matrix multiplication:

$$\vec{\omega}\times\vec{a} = veck^{-1}(\vec{\omega})\cdot\vec{a} = \boldsymbol{\Omega}\cdot\vec{a} \xrightarrow{yields}$$

$$\vec{\omega}\times\vec{a} = \begin{pmatrix} \omega_y a_z - \omega_z a_y \\ \omega_z a_x - \omega_x a_z \\ \omega_x a_y - \omega_y a_x \end{pmatrix} = \begin{pmatrix} 0 & -\omega_z & -\omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix}\cdot\begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} = \boldsymbol{\Omega}\cdot\vec{a} \quad \textbf{(2-25)}$$

2.5.7 Inner product

The inner product is also named as scalar product or dot product and is denoted by $\langle\vec{a},\vec{c}\rangle$.

$$\langle\vec{a},\vec{c}\rangle = \left\langle\begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix},\begin{pmatrix} c_x \\ c_y \\ c_z \end{pmatrix}\right\rangle = a_x\cdot c_x + a_y\cdot c_y + a_z\cdot c_z \quad \textbf{(2-26)}$$

2.6 Differential geometry, symmetry and invariance concepts

Symmetries have been used in control theory for feedback design and optimal control in some other works, but in this case, these symmetries are going to be used for observer and filter design. The purpose is to design an algorithm which preserves these symmetries and improve the results, which are the estimations of the variables. Invariant systems, observers and errors will be defined.

First of all, some definitions on the branch of mathematics called ***Differential Geometry*** need to be clarified and refreshed with good examples for the reader to understand how the symmetries are implemented.

This mathematical branch studies the spaces up to isometries. Isometry is a term to describe a rotation, a translation, a reflection, a glide (which is a combination of a reflection and a translation) or an identity map.

In this section, all the necessary concepts of differential geometry and others will be defined to be used later on. Firstly, the basic concepts will be stated and, secondly, Lie groups and Lie algebras are going to be explained to make the comprehension more straightforward.

2.6.1 Mapping(Map)

A map is a way of associating unique objects to every element in a given set. So a map $\boldsymbol{f}$: $\boldsymbol{A} \rightarrow \boldsymbol{B}$ from A to B is a function $\boldsymbol{f}$ such that for every element $\boldsymbol{a} \in \boldsymbol{A}$, there is a unique object $\boldsymbol{f}(\boldsymbol{a}) \in \boldsymbol{B}$. The terms function and mapping are synonymous for map. [9]

The term refers to a *function* with a special structure or a *morphism that*generalizes the idea of a function. Normally, it is a function with a specific property.

In many branches of mathematics, the term **map** is used to mean a function, sometimes with a specific property of particular importance to that branch. For instance, a "map" is a *continuous function* in topology, a *linear transformation* in linear algebra, etc.

An alternate definition can be found on [2]: Let X and Y be sets and $A \subset X$ any subset of X. A mapping (or transformation or function) M of A into Y is obtained by associating to every $x \in A$ a single image $y \in Y$, shorthand $y = Mx$.

2.6.2 Group

A **group (G)** is a finite or infinite set of elements together with an operation (called the **group operation**) that combines any two of its elements to form a third element satisfying the so called group axioms or properties: **closure, associativity, identity property and inverse property.**

In that way a set is also a group under this specific operation. For instance, let us state the elements A, B and C with the product operation between A and B denoted as $A \cdot B$. This product is also a group, in case it satisfies the following properties:

2.6.2.1 Closure

A set has a closure under an operation if performance of that operation on members of the set always produces a member of the set. It is also said that the set is closed under the operation. (i.e. if $\boldsymbol{A, B} \in \boldsymbol{G}$, then the product $\boldsymbol{A \cdot B} \in \boldsymbol{G}$).

2.6.2.2 Associativity

The operation is associative, i.e. for all $A, B, C \in G \rightarrow (A \cdot B) \cdot C = A \cdot (B \cdot C)$.

2.6.2.3 Identity

There is an identity element for every element $A \in G$ such that $A \cdot I = I \cdot A = A$

2.6.2.4 Inverse

There must be an inverse for every element $A \in G$ such that $A \cdot A^{-1} = A^{-1} \cdot A = I$

The study of groups is known as group theory. The groups with a finite number of elements are called finite groups and the number of elements is the group order. It exists the infinite groups as well.

A simple example of a finite group is the symmetric group S_n, which is the group of permutations of n objects. An example for an infinite group is the set of integers under addition (operation of the group). [10]

2.6.3 Types of groups

Some important types of groups that might be used during the work are:

2.6.3.1 General Linear group = $\boldsymbol{GL(n) = GL_n(\mathbb{R}) = GL(n, \mathbb{R})}$:

Is the set of nxn invertible matrices and the operation is the ordinary matrix multiplication.

2.6.3.2 Orthogonal group $\boldsymbol{O(n) = O_n(\mathbb{R}) = O(n, \mathbb{R})}$:

Is the set of nxn orthogonal matrices and the operation is the ordinary matrix multiplication. An orthogonal matrix A and is the one that complies $A \cdot A^T = I$ and, in particular, $A^{-1} = A^T$.

2.6.3.3 Euclidean group $\boldsymbol{\mathcal{E}(n) = \mathcal{E}_n(\mathbb{R}) = \mathcal{E}(n, \mathbb{R})}$:

is the group of rotations and translations. Taking into account that the Euclidean n-space is denoted by $\mathbb{R}^n$, i.e., $\mathbb{R}^1$is the set of real numbers.

2.6.3.4 Special Liner group $\boldsymbol{SL(n) = SL_n(\mathbb{R}) = SL(n, \mathbb{R})}$:

is the group of matrices with determinant equal to 1.

2.6.3.5 Special Orthogonal group $\boldsymbol{SO(n) = SO_n(\mathbb{R}) = SO(n, \mathbb{R})}$:

Is the subgroup of $O(n)$ with determinant equal to 1. [11]

2.6.4 Bijective function

Is a function $\boldsymbol{f}$ between the elements of two sets X and Y, where one element of one set is paired exactly with another element of the other set, and vice versa, (i.e.$\boldsymbol{f: X \rightarrow Y}$ where X and Y are sets.)

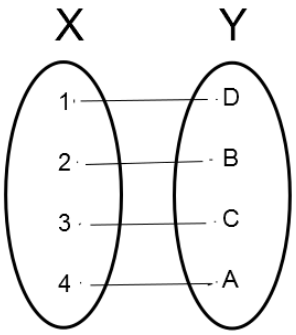

Figure 2-5: Bijective function representation.

2.6.5 Surjective Function

Is a function $\boldsymbol{f}$ between the elements of two sets X and Y, where every element y of Y has corresponding element x in X such that, $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{y}$. But the function $\boldsymbol{f}$may map more than one element of X to the same element of Y.

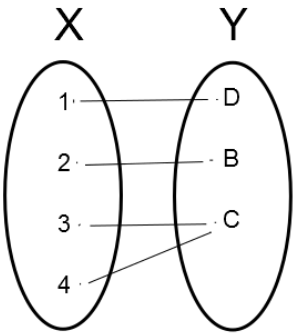

Figure 2-6: Surjective function representation.

2.6.6 Category

A category consists of three things: a collection of objects, for each pair of objects a collection of morphisms (sometimes called "arrows") from one to another, and a binary operation defined on compatible pairs of morphisms called composition. The category must satisfy an identity axiom and an associative axiom which is analogous to the monoid axioms.

In most concrete categories over sets, an object is some mathematical structure (e.g. a group, vector space or smooth manifold) and a morphism is a map between two objects.

One usually requires morphisms to preserve the mathematical structure of the objects. So if the objects are all groups, a good choice for a morphism would be a group homomorphism. Similarly, for vector spaces, one would choose linear maps, and for differentiable manifolds, one would choose differentiable maps. [12]

2.6.7 Morphism

A morphism is a map between two objects in an abstract category. For instance, in *Set theory,* a morphism is a function as well as, in *Group theory* a morphism is a group homomorphism.

2.6.7.1 Homomorphism

Is a general morphism.

2.6.7.2 Monomorphism

A morphism $\boldsymbol{f}: \boldsymbol{Y} \rightarrow \boldsymbol{X}$ in a category is a monomorphism if, for any two morphisms $u, v: Z \rightarrow Y \; and \; fu = fv$ implies that$u = v$.

2.6.7.3 Epimorphism

A morphism $\boldsymbol{f}: \boldsymbol{Y} \rightarrow \boldsymbol{X}$ in a category is an epimorphism if, for any two morphisms$u, v: X \rightarrow Z \; and \; uf = vf$ implies $u = v$.

2.6.7.4 Isomorphism

A bijective morphism is called an isomorphism (if there is an isomorphism between two objects, then it is said that they are isomorphic).

2.6.7.5 Endomorphism

A surjective morphism from an object to itself is called an endomorphism.

2.6.7.6 Automorphism

An isomorphism between an object and itself is called an automorphism. [13]

2.6.8 Topological space

Is a set X together with a collection of open subsets $\mathcal{T}$ that satisfies the four subsequent statements:

a) The empty set $\emptyset \in \mathcal{T}$.
b) $X \in \mathcal{T}$.
c) The intersection of a finite number of sets in $\mathcal{T} \in \mathcal{T}$.
d) The union of an arbitrary number of sets in $\mathcal{T} \in \mathcal{T}$.

Alternatively, $\mathcal{T}$ may be defined to be the closed sets rather than the open sets, in which case the two latter conditions become:

c) The intersection of an arbitrary number of sets in $\mathcal{T} \in \mathcal{T}$.
d) The union of a finite number of sets in $\mathcal{T} \in \mathcal{T}$. [14]

2.6.9 Manifold

A manifold is a topological space that is locally Euclidean. To illustrate this idea, consider the ancient belief that the Earth was flat as contrasted with the modern evidence that is round. The discrepancy arises essentially from the fact that on the small scales that we see, the Earth does indeed look flat.

In general, any object that is nearly “flat” on small scales is a manifold. More concisely, any object that can be charted is a manifold. [15]

To understand this concept, is better to show some examples of the most basic manifolds: [16]

Example 1

Let k, m, n be positive integers. Throughout we denote by:

$$|x| := \sqrt{{x_1}^2 + {x_2}^2 + \cdots + {x_n}^2} \tag{2-27}$$

the Euclidean norm of a vector $x = (x_1, \cdots, x_n) \in \mathbb{R}^n$. The basic example of a manifold of dimension $m$is the Euclidean space $M = \mathbb{R}^n$itself.

Example 2

Another example is the unit sphere$\in \mathbb{R}^{m+1}$:

$$S^m := \{x \in \mathbb{R}^{m+1} \mid |x| = 1\} \tag{2-28}$$

In this particular case for $m = 1$ we obtain the unit circle:

$$S^1 \subset \mathbb{R}^2 \cong \mathbb{C} \tag{2-29}$$

Other examples

Other examples are the groups:

$$SL(n, \mathbb{R}) := \{A \in \mathbb{R}^{nxn} \mid \det(A) = 1\}$$
$$O(n) := \{A \in \mathbb{R}^{nxn} \mid \mathrm{A} \cdot \mathrm{A}^{\mathrm{t}} = I\} \tag{2-30}$$

2.6.10 Diffeomorphism

Given two manifolds $M\ and\ N$, a differentiable map $\boldsymbol{f}: \boldsymbol{M} \rightarrow \boldsymbol{N}$ is a diffeomorphism is it is a bijection and its inverse $\boldsymbol{f^{-1}}: \boldsymbol{N} \rightarrow \boldsymbol{M}$ is differentiable as well. To sum up the concept, a diffeomorphism is a map between two manifolds which is differentiable and has a differentiable inverse. [16]

Or as defined in [2]: A mapping (M) is diffeomorphic if it is a continuously differentiable bijective mapping $\boldsymbol{M}: \boldsymbol{X} \rightarrow \boldsymbol{Y}$whose inverse is also continuously differentiable.

2.6.11 Homeomorphism

A mapping (M) is homeomorphic if it is a continuous bijective mapping $\boldsymbol{M}: \boldsymbol{X} \rightarrow \boldsymbol{Y}$whose inverse is also continuous. [2]

2.6.12 Smooth Manifold

A smooth manifold or a differentiable manifold is a topological manifold (or a manifold) that is infinitely differentiable C^{∞}. Every smooth manifold is a manifold, but not necessarily vice versa.

And all the above definitions bring us to the position of understanding the definition that define a symmetries and invariances, which are the key point on the new symmetry-preserving observer and filter.

2.6.13 Lie Group

Basically, a Lie group is a smooth manifold and a group obeying the group properties and that satisfies the additional condition that the group operations are differentiable (smooth).

A more formal definition follows from the paper [17]:

A lie group is a group G, equipped with a manifold structure such that the following group operations are smooth:

$$Multiplication: G \times G \to G, \qquad (g_1, g_2) \mapsto g_1 \cdot g_2$$
$$Inverse: G \to G, \qquad g \mapsto g^{-1} \tag{2-31}$$

A morphism of Lie groups G, G' is a morphism of groups $\phi: G \to G'$ that is smooth.

After all these important definitions that succeed one to each other the theory of symmetry-preserving observers will be stated in the same way as it is done in [3].

2.7 Symmetry-preserving observers theory

The construction of the generic observer and the Kalman filter comes directly from the symmetry-preserving (or invariant) observer theory introduced in [18]. They briefly recall here the main ideas and definitions of this previous work, completed with an additional result (for further details, see [18]).

2.7.1 Invariant systems and equivariant outputs

Definition: Let G be a Lie Group with identity e and $\sum$ an open set (or more generally a manifold). An action of a transformation group $(\phi_g)_{g \in G}$ on $\sum$ is a smooth map

$$(g_1, \xi) \in G \times \sum \mapsto \phi_g(\xi) \in \sum \tag{2-32}$$

such that:

- $\phi_g(\xi) = \xi \; for \; all \; \xi$
- $\phi_{g_2} \circ \phi_{g_1}(\xi) = \phi_{g_2 g_1}(\xi) \; for \; all \; g_1, g_2, \xi$

By construction $\phi_g$is a diffeomorphism on $\sum$ for all g. The transformation group is local if $\phi_g(\xi)$is defined only for g in a neighborhood of$e$. In this case the transformation law $\phi_{g_2} \circ \phi_{g_1}(\xi) = \phi_{g_2 g_1}(\xi)$is defined only when it makes sense. It is considered only local transformation groups. "For all g" thus means "for all g around$e$", and "for all ξ" means "for all ξ in some neighborhood".

Consider now the smooth output system:

$$\begin{aligned} \dot{x} &= f(x,u) \\ y &= h(x,u) \end{aligned} \qquad \textbf{(2-33)}$$

Where x belongs to an open subset $\mathcal{X} \subset \mathbb{R}^n$, u to an open subset $\mathcal{U} \subset \mathbb{R}^m$and y to an open subset $\mathcal{Y} \subset \mathbb{R}^p$, where$p \leq n$.
We assume the signals $u(t), y(t)$to be known (y is measured, and u is measured or known as a control input).

*Consider also the local group of transformations on$\mathcal{X} \times \mathcal{U}$ defined by *$(X,U) = \left(\varphi_g(x), \psi_g(u)\right)$, where φ_g and ψ_g are local diffeomorphisms.*

Definition: The system $\dot{x} = f(x,u)$is G-invariant if $f\left(\varphi_g(x), \psi_g(u)\right) = D\varphi_g(x) \cdot f(x,u)\ for\ all\ g,x,u.$
With $(X,U) = \left(\varphi_g(x), \psi_g(u)\right)$, the property also reads $\dot{X} = f(X,U)$, i.e., the system is left unchanged by the transformation.

Definition: The output $y = h(x,u)$is G-equivariant if there exists a transformation group $(\rho_g)_{g \in G}$ on $\mathcal{Y}$ such that $h\left(\varphi_g(x), \psi_g(u)\right) = \rho_g\left(h(x,u)\right)\ for\ all\ g,x,u.$
With $(X,U) = \left(\varphi_g(x), \psi_g(u)\right)$, the property also reads $Y = \rho_g(y)$, the definition reads $Y = h(X,U)$. The two previous definitions can be illustrated by the commutative diagram:

$$\begin{array}{ccc} T\mathcal{X} & \xrightarrow{D\varphi_g} & T\mathcal{X} \\ f \uparrow & & f \uparrow \\ \mathcal{X} \times \mathcal{U} & \xrightarrow{\varphi_g \times \psi_g} & \mathcal{X} \times \mathcal{U} \\ h \downarrow & & h \downarrow \\ \mathcal{Y} & \xrightarrow{\varrho_g} & \mathcal{Y}. \end{array}$$

Figure 2-7: Diagram describing the operations, extracted from [18].

2.7.2 Invariant preobservers

Definition (Preobserver): *The system $\dot{\tilde{x}} = F(\tilde{x},u,y)$ is a preobserver of the system* (2-33) *if $F\left(x,u,h(x)\right) = f(x,u)\ for\ all\ x,u.$*

An observer is then a preobserver such that $\tilde{\boldsymbol{x}}(\boldsymbol{t}) \to \boldsymbol{x}(\boldsymbol{t})$(possibly only locally).

Definition: *The preobserver $\dot{\tilde{x}} = F(\tilde{x}, u, y)$ is G-invariant if $F\left(\varphi_g(\tilde{x}), \psi_g(u), \rho_g(y)\right) = D\varphi_g(\tilde{x}) \cdot F(\tilde{x}, u, y)\ for\ all\ g, \tilde{x}, u, y.$*
The property also reads $\dot{\tilde{X}} = F\left(\tilde{X}, U, Y\right)$, i.e., the system is left unchanged by the transformation.

The key idea to build an invariant preobserver is to use an invariant output error.

Definition: *The smooth map $(\tilde{x}, u, y) \mapsto E(\tilde{x}, u, y) \in \mathcal{Y}$is an invariant output error if:*

- $the\ map\ y \mapsto E(\tilde{x}, u, y)\ is\ invertible\ for\ all\ \tilde{x}, u.$
- $E\left(\tilde{x}, u, h(\tilde{x}, u)\right) = 0\ for\ all\ \tilde{x}, u.$
- $E\left(\varphi_g(\tilde{x}), \psi_g(u), \rho_g(y)\right) = E(\tilde{x}, u, y)\ for\ all\ \tilde{x}, u, y.$

The first and second properties mean E is an "output error", i.e., it is zero if and only if $h(\tilde{x}, u) = y$; the third property, which also reads $E\left(\tilde{X}, U, Y\right) = E(\tilde{x}, u, y)$, defines invariance.
Similarly, the key idea to study the convergence of an invariant preobserver is to use an invariant state error.

Definition: *The smooth map $(\tilde{x}, x) \mapsto \eta(\tilde{x}, x) \in \mathcal{X}$is an invariant state error if:*

- $is\ a\ diffeomorphism\ on\ \mathcal{X} \times \mathcal{X}.$
- $\eta(x, x) = 0\ for\ all\ x.$
- $\eta\left(\varphi_g(\tilde{x}), \psi_g(x)\right) = \eta(\tilde{x}, x)\ for\ all\ \tilde{x}, x.$

Then they state the two main results – based on the Cartan moving frame method – in the special case where $g \mapsto \varphi_g(x)$ is a free and transitive action, see [18]*for the general case. The moving frame $x \mapsto \gamma(x)$ is obtained by solving for g the so-called normalization equation $\varphi_g(x) = c$ for some arbitrary constant c; in other words $\varphi_{\gamma(x)}(x) = c$. The function$\psi_g$may also be used in the normalization equation$\left(\varphi_g(x), \psi_g(u)\right) = c.$*

Theorem: *The general form of any invariant preobserver is:*

$$F(\tilde{x}, u, y) = f(\tilde{x}, u) + \sum_{i=1}^{n} (L_i(E, I) \cdot E)\omega_i(\tilde{x}) \qquad \textbf{(2-34)}$$

Where:

- $\omega_i, i = 1, \dots, n,$ *is the invariant vector field defined by*

  $$\omega_i(\tilde{x}) = \left[D\varphi_{\gamma(\tilde{x})}(\tilde{x})\right]^{-1} \cdot \frac{\partial}{\partial x_i}$$

  With $\frac{\partial}{\partial x_i}$ the i^{th} canonical vector field on $\mathcal{X}$.

- E *is the invariant error defined by*

$$E(\tilde{x}, u, y) = \rho_{\gamma(\tilde{x})}\big(h(\tilde{x}, u)\big) - \rho_{\gamma(\tilde{x})}(y) \quad \textbf{(2-35)}$$

- *I is the (complete) invariant defined by*

$$I(\tilde{x}, u) = \psi_{\gamma(\tilde{x})}(u) \quad \textbf{(2-36)}$$

- *$L_i, i = 1, \dots, n, is\ a\ 1 \times p$matrix with entries possibly depending on E and I, and can be freely chosen.*

Theorem: *The error system reads $\dot{\eta} = \Upsilon(\eta, I)$for some smooth function Υ, where η is the invariant state error defined by*

$$\eta(\tilde{x}, x) = \varphi_{\gamma(\tilde{x})}(\tilde{x}) - \varphi_{\gamma(x)}(x) \quad \textbf{(2-37)}$$

This result greatly simplifies the convergence analysis of the preobserver, since the error equation is autonomous but for the "free" known invariant I. Indeed for a general nonlinear (not invariant) observer the error equation depends on the trajectory $t \mapsto \big(x(t), u(t)\big)$ of the system, hence is in fact of dimension $2n + m$, whereas the dimension of the invariant state error equation is only $2n + m - dim(G)$. To simplify, we will use from now on the term "observer" instead of "preobserver".

3 INAs types and overview

The INAs are the focus of this work; more specifically the three different algorithms that are introduced in this work are classified in two main types: observers and Kalman Filters (KF). Chapter 3 describes the types of algorithms in a general point of view that are going to be used and the considerations of the systems that have to be taken into account to apply these algorithms to real systems. Each of the three algorithms will be explained in detail in the following posterior chapters.

Furthermore, descriptions of the measured data and the known inputs of the system will be stated and the assumptions or simplifications taken into account according to our purposes will be explained.

So taking an overview of the actual chapter, the information will be displayed in the following order.

First of all, the technical terms used for these algorithms will be clarified and the assumptions or simplifications with respect to the real world will be denoted.

Secondly, the input data used will be shown and some verification regarding the system will be performed to check whether it is observable or not.

Finally, the types of INAs will be explained in a general way to have a look on how they work and which are the main differences that we must expect from them, when comparing the results.

3.1 Technical terms and assumptions

3.1.1 Flat Earth Model

This study was developed under the assumption of a Flat Earth Model, which means we considered the Earth to be flat. This consideration is not far from the reality as we are working with algorithms for small UAVs, which range is normally very small compared with big commercial aircrafts. The endurance of these systems is also a key point regarding the small range, because is not a first consequence of that but it also plays an important role as it is clear that with 10 – 20 minutes of endurance no long flight can be conducted.

Coming back to our assumption of a Flat Earth, the main consequence of this assumption is that: $R \to \infty$. So it is assumed the Earth's mean radius as the infinite value of the Earth's radius, which is $R = Earth's\ mean\ radius = 6371.009\ km$.

At the same time, this step brings us to consider a constant gravity vector pointing to the positive part of Z-axis in the n-Frame 2.3.2.

$$(\vec{g}^R)_W(z) = (\vec{g}^R)_n(z) = \frac{g_0}{\left(1 - \frac{z}{R}\right)^2} \qquad \textbf{(3-1)}$$

Where $= 6371.009\ km$, $g_0 = standard\ gravitational\ acceleration\ = 9.80665\ m/s^2$and $Z = -h$. Yielding to:

$$(\vec{g}^R)_W(h) = (\vec{g}^R)_n(h) = \frac{g_0}{\left(1+\frac{h}{R}\right)^2} \quad \textbf{(3-2)}$$

Introducing the Flat Earth model assumption:

$$(\vec{g}^R)_W(z) = (\vec{g}^R)_n(z) = g_0 \quad \textbf{(3-3)}$$

And this brings us to a constant gravity problem.

3.1.2 Notation concepts of the system

A more in depth identification of each of the parameters and variables used in the INAs algorithms should be stated before advancing on the work, because the terms used in these algorithms are most of the times unknown if the readers are not related with this field. It is also important to clarify them, because, depending on the author, the nomenclature tends to change from one person to another.

Completing our definition of a system in (2-33), and taking this notation as a first reference the preceding system is extended to the linearized matrix form:

$$\begin{aligned} \dot{\vec{x}}(t) &= A \cdot \vec{x}(t) + B \cdot \vec{u}(t) + G \cdot \vec{w}(t) \\ y(t) &= H \cdot \vec{x}(t) + \vec{v}(t) \end{aligned} \quad \textbf{(3-4)}$$

On the one hand, the first equation is the system equation, which represents the behavior of the whole system and the variables we want to estimate. On the other hand, the second equation is the output equation and describes how the measurements are introduced or used in our system. Then the variables of the system are defined in the following table:

Variable	Name	Description
$\vec{x}$	State vector	Vector of variables that are going to be estimated.
$\vec{u}$	Input vector	Vector of the inertial measured variables, normally the IMU data. In most of the cases is composed of the accelerometers, gyroscopes and gravity measurements.
A	State/transition matrix	Matrix which converts the variables to be estimated into the same differentiated variables.
B	Input matrix	Matrix that converts the inputs to the correct form to be suited into the system equation.
$\vec{w}$	Process noise vector	Vector of the same length as the State vector, which is formed of random Gaussian variables. Represents the uncertainties of the model.
$\vec{y}$	Output vector	Vector of the same size as the Measurement vector.
H	Measurement matrix	Matrix that inputs the measurements into the output equation to improve the estimated states.
$\vec{v}$	Measurement noise	Vector of the same length as the output vector, which is formed of random Gaussian variables. Denotes the noise of the measurements.

$\boldsymbol{G}$	Noise matrix	Matrix which adapts the size of the noise vector to the size of the system equations. Normally if $\vec{w}$ is of the length defined above this matrix is not necessary.

Table 3-1: Description of the different variables of our system and output equations.

The noise matrices will be explain more in detail for each of the KFs, but here they are stated in a general form. Inside the three following chapters the specific nomenclature of each INA will be explained and stated. Though, always regarding to the general description described in this chapter. Even though some of the parameters can be slightly different in each of these three posterior chapters, the appropriate explanation and comparison for each of the variables will be examined more in detail for each case.

Differentiation of nomenclature is introduced in this work to clarify and separate each of the cases and to show why each of the algorithms is different.

3.2 Observability

The condition of observability is of paramount importance for the KF to be able to estimate the states. So, if and only if the system is observable, we will be able to estimate the states of our system with the KF algorithm.

A system is observable if the initial state can be obtained ("observed") from the knowledge of the input variables and the output equation.

The observability definition is introduced from [19] and adapted for our system defined in (3-4) as:

$$\begin{gathered} y(0) = H \cdot x(0) \\ y(1) = H \cdot x(1) = H \cdot A \cdot x(0) \\ \vdots \\ y(n-1) = H \cdot A^{n-1} \cdot x(0) \end{gathered} \tag{3-5}$$

Which can be transformed to:

$$\begin{pmatrix} y(0) \\ y(1) \\ \vdots \\ y(n-1) \end{pmatrix} = \begin{pmatrix} H \\ H \cdot A \\ \vdots \\ H \cdot A^{n-1} \end{pmatrix} \cdot x(0) \tag{3-6}$$

From here, it is obtained the observability matrix M_{obs} :

$$M_{obs} = \begin{pmatrix} H \\ H \cdot A \\ \vdots \\ H \cdot A^{n-1} \end{pmatrix} \tag{3-7}$$

And this matrix has a unique solution if: $(\boldsymbol{M_{obs}}) = \boldsymbol{n}$. So the system is observable if and only if the observability matrix has the same rank as the number of states (n). The rank can be checked by calculating the determinant of M_{obs} and if:

$$\boldsymbol{det(M_{obs}) \neq 0 \xrightarrow{yields} rank(M_{obs}) = n} \quad \textbf{(3-8)}$$

In case the system is non-observable, it means that the length of the input vector $\vec{u}$is less than the length of the output vector $\vec{y}$**,** which signifies the number of measurements is lower than the number of variables we want to estimate.

3.3 Inputs of the system

Taking into account the definitions above, here the inputs and measurements used in our three cases are going to be outlined. It is a key point to use the same variables performing the identical roles for each of the algorithms; otherwise the comparison between them would not have any significant.

First of all, it is appropriate to differentiate between a measurement and an input. The measurements are the data that we use in the output equation to improve the estimations of the system. These measurements are used to improve the estimations of the state variables. So, for instance, if we choose the velocity of the GPS sensors like a measurement, this velocity will be used to correct the error between the estimations and the real variables of the state vector using gain terms. On the other hand, the input data is normally the data coming from the IMU, which is used in the system equation but not as a correction to enhance the estimation of the states.

To check the condition of observability in our specific case, we first have to differentiate the inputs and the measurements that will be used and then check for the condition to be accomplished.

Regarding the above definitions, our system will have the following input vector $\vec{u}$:

$$\vec{u} = \begin{pmatrix} (\vec{f}^R)_B \\ (\vec{\omega}^{WB})_B \end{pmatrix} \quad \textbf{(3-9)}$$

The gravity would be another variable of the input vector but as we considered the assumption of a Flat Earth Model 3.1.1, it becomes a constant.

The State vector $\vec{x}$ will be defined in each part, as it is slightly different for each one of the algorithms. And finally the measurement matrix depends on two measurements: The velocity $(\vec{v}^R)_W^W$ from the GPS sensor and the normalized Earth magnetic field $\overline{(\vec{B}^R)}_W$ measured with the magnetometers in the w-Frame, rotated to the b-Frame and normalized.

$$\vec{y} = \begin{pmatrix} (\vec{v}^R)_W^W \\ \overline{(\vec{B}^R)}_{B,input} \end{pmatrix} \quad \textbf{(3-10)}$$

The considered system is denoted as aided-AHRS, because it is aided with the GPS velocity measurements $\left(\vec{v}^{R}\right)_{W}^{W}$.

3.3.1 Sensor models

The proposed sensor models are different for each of the INAs regarding the IMU sensors, i.e. accelerometers and gyroscopes. Moreover, the sensor models for the velocity from the GPS sensor and for the magnetometers are the same throughout all of this work.

For this reason, these two latter sensor models will be outlined here and the IMU sensor model will be developed on the specific chapter of each INA.

The data extracted from these sensors is going to be treated as the measurements of the INAs. Then, by deduction, the two measurements are velocity and magnetic field, as it is explained in the previous section.

First of all, it is important to define the magnetometer sensor model. It is used this normalized magnetic field in the b-Frame proposed like:

$$\overline{\left(\vec{B}^{R}\right)}_{B,input} = \left\| \breve{q}_{EB}{}^{-1} * \begin{pmatrix} 0 \\ \left(\vec{B}^{R}\right)_{W} \end{pmatrix} * \breve{q}_{EB} \right\| \tag{3-11}$$

The GPS sensor extracts the velocity in the w-Frame and is utilized directly in this frame of reference without any rotation of the data, so the sensor model is nothing different from: $\left(\vec{v}^{R}\right)_{W}^{W}$.

The measurements are perturbed using a Gauss-Markov noise model, which is the most realistic in comparison with the real sensor noise.

3.4 Types of INA

Evaluating the different methods to estimate the values of the state variables of a dynamic system like the one we are considering, we will deal with two different types of algorithms: observers and KF.

Both of them are utilized for the same purpose but with the latter method, the algorithm takes into account stochastic theory to estimate the states. More specifically, it considers that the system is excited by stochastic (random) disturbances and stochastic (random) measurement noise.

Regarding the implementation type of the methods, both of them can be implemented in a continuous-time way or in a discrete-time way. The latter is probably the best option to implement, as it is less time-consuming and is specifically suited for computer calculations.

It is also a main consideration to differentiate between the types of dynamical system we are dealing with, there are mainly two types: the linear systems and non-linear systems.

Normally, the observers and KFs are designed for linear systems but some modifications can be used for using them to estimate the states of non-linear systems, as this is our case.

3.4.1 Observer

In control theory, an observer is a system that provides an estimate of the state variables of a system, from measurements of the output and known data as an input of the real system, i.e. the IMU data or the input variables. In most common cases, the physical state of the system cannot be directly determined, here is when the observer performs its role. The internal states (i.e. the states between the initial time and the final time) are determined with an observer with the help of the system outputs and the known inputs. The simplest example is, for instance, a vehicle in a tunnel: the rates and velocities at the entrance and exit of the tunnel can be directly observed, but the exact states inside the tunnel can only be estimated.

At this point is when the property of observability has a paramount importance. Only in the case a system is observable 3.2, it is possible to totally reconstruct the system states from the measurements and input data.

An observer is an algorithm with a similar structure than the KF, but it is calculated from specified estimator error dynamics, or in other words, how fast and stable you want the estimates to converge to the real values (assuming you could measure them) [20]. The theory and implementation of an observer is more straightforward than a KF. Though at the same time it is not easy to design for systems with more than one measurement, which is our case.

To have a general overview of how a general observer works, the following Figure 3-1. It is important to notice that this figure extracted from [20] is not exactly the same as our observer block diagram, but it is just to have an idea of how these kinds of algorithms work.

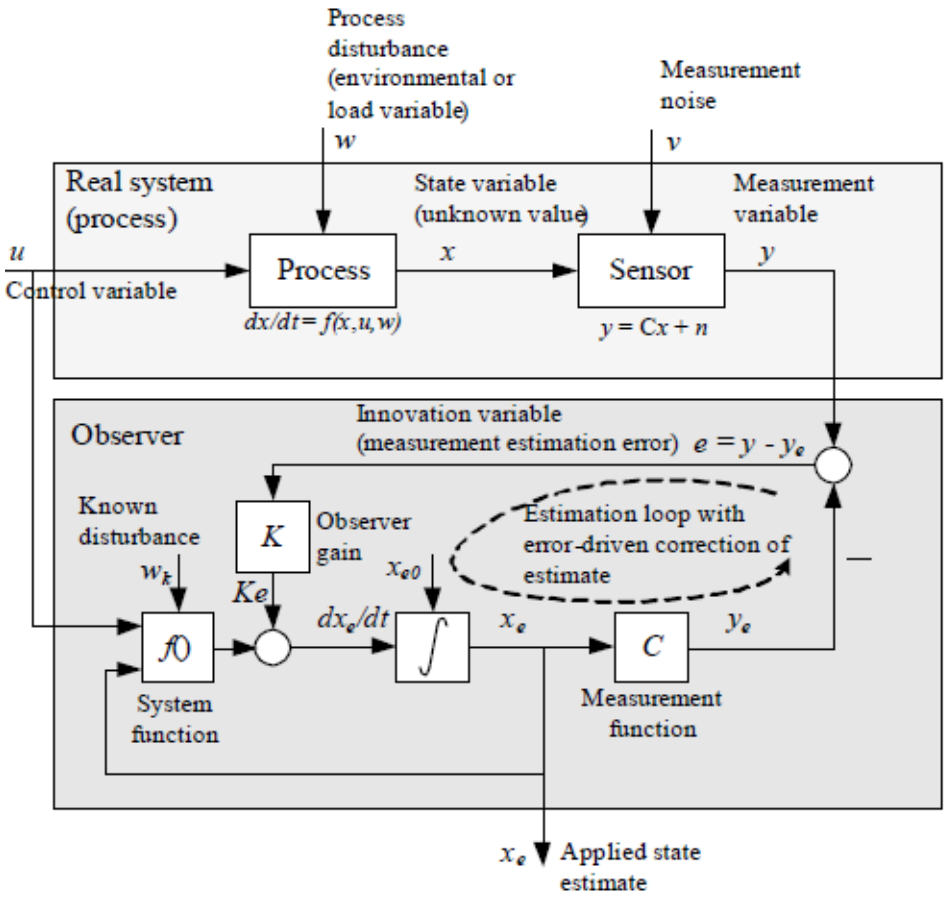

Figure 3-1: General continuous-time observer block diagram.

The notation of this figure is neither the one that we will use for this thesis but it is straightforward to guess what every variable represents. In such a way, the variables with an $e$as sub-index denote the estimation of the variable, the equality in our notation is: $\tilde{x} = x_e$.

Using our notation the error equation is defined as:

$$\vec{\boldsymbol{e}}_{\boldsymbol{x}}(\boldsymbol{error}) = \vec{\boldsymbol{x}} - \vec{\tilde{\boldsymbol{x}}} \tag{3-12}$$

So the observer bases the calculation of the estimation on the error, calculated with the previous values of the real variable and the estimation of the same variable and uses the tuneable gains to give more or less importance to the correction term to be added or subtracted depending on each case.

The conclude with the explanation of a general observer, the most important point to tune the behaviour in each case are the gains elements, which are the responsible of adjusting the way the observer estimates the system variables.

3.4.2 Kalman Filter (KF)

The KF is an algorithm first developed by Rudolf E. Kalman on 1960. It is similar to an observer and the main difference is that uses stochastic theory as the main source to estimate the state variables and that utilizes two steps: **the prediction step and the correction step.** The separation in two different steps enhances the results of the estimated states compared with the case of the observer. The KF is performed in a closed-loop cycle.

The principal advantage with respect to observers is that the Kalman filter is more straightforward to design for systems with more than one measurement. The design of an observer is not simple because the measurements we want to use have to be decoupled in a kind of way to affect only a specific variable. For instance, in our case, the magnetometer measurement is used to improve the estimation of the yaw angle ψ, but it is designed in a way that it only affects this angle and not the other attitude angles, which will not be a positive affect for their estimations at all.

Extracting the definition from [19]: "*The KF is a state estimator which produces an optimal estimate in the sense that the mean value of the sum of the estimation errors gets a minimal value."*

Taking into account the same error definition as for the observer (3-12), the definition above is the same as saying that the KF minimizes the following sum of squared errors:

$$E\left(\sum_{i=1}^{size(\vec{x})} {e_{x_i}}^T \cdot e_{x_i}\right) = E\left({e_{x_1}}^2 + \cdots + {e_{x_n}}^2\right) \tag{3-13}$$

That is the reason why sometimes the KF is also named the least mean-square method. This algorithm assumes that the system, which states are going to be estimated, is excited by white random disturbances (called process noise) and that the measurements contain a

similar type of Gaussian noise (called measurement noise). It is important to note that there must be at least one real measurement in a KF. The main feature of the KF is that is only designed for linear systems, and for our case is not appropriate. So, in this thesis is used the modification of the normal KF called EKF, which works for nonlinear systems too.

The main difference between the KF and the EKF is that the EKF uses a linearized version of the system equations around the estimated trajectory, which are non-linear, to be implemented in the KF algorithm and, by this way, utilize the same algorithm for the same purposes. The EKF is the most popular solution for non-linear systems in engineering applications.

Furthermore, it is important to note that we use the EKF with another additional modification that reduces the amount of calculations for the attitude estimation. This other modification is the MEKF, which is the EKF for nonlinear models with another definition of the error.

Since in this case it is used the quaternion representation, the error equation changes from a subtraction to a quaternion multiplication:

$$\vec{e}_q(\boldsymbol{error}) = \breve{\boldsymbol{q}} * \tilde{\breve{\boldsymbol{q}}}^{-1} \tag{3-14}$$

Only for the attitude estimation, the other variables utilize the definition of the error as in (3-12). Such a change is performed because it would not have any sense a subtraction between quaternions.

To have a clear idea of how an EKF is illustrated as a block diagram, the following figure extracted from [19] is shown:

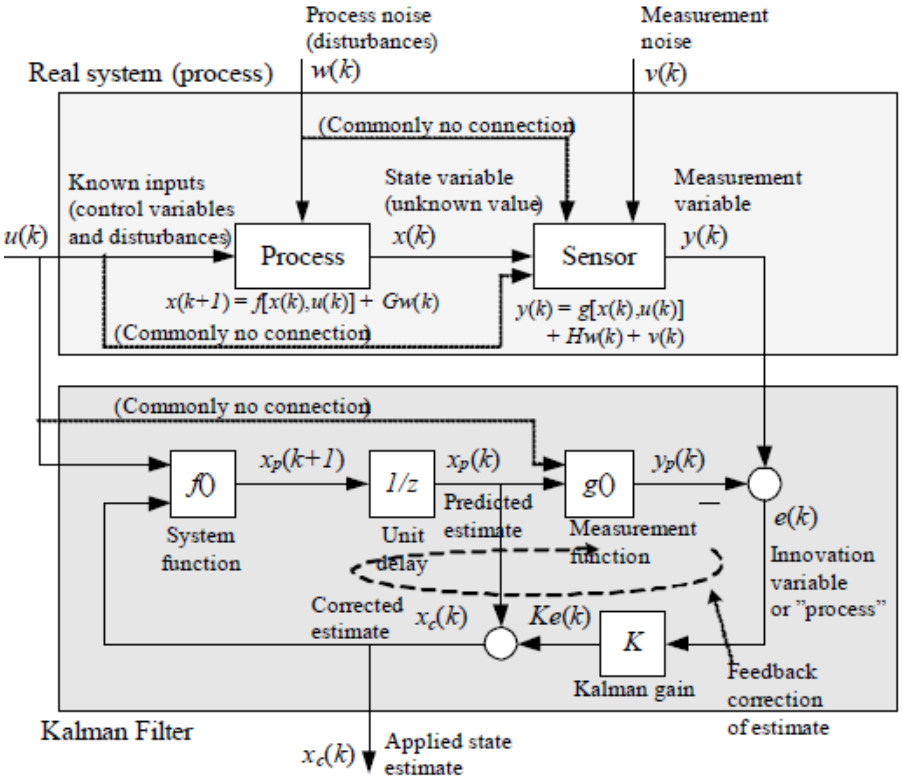

Figure 3-2: Block diagram of a general time-discrete EKF.

The notation issue is the same as for Figure 2-1. It has to be noticed that the EKF uses two steps and the Kalman gains ($\boldsymbol{K}$) are not tuneable, so it means that are calculated by the algorithm based on the stochastic values, such as the covariance matrix ($\boldsymbol{P}$), the Process noise matrix ($\boldsymbol{Q}$) or the Measurement noise matrix ($\boldsymbol{R}$). These stochastic matrices used with the KF are defined in the following table:

Variable	Description
$\boldsymbol{P}$	State covariance matrix, measure of how the difference between the error on the states and its expectation value change together.
$\boldsymbol{Q}$	Input covariance matrix or process noise matrix, measure of how the difference between the error on the inputs and its expectation value change together.
$\boldsymbol{R}$	Measurement covariance matrix or measurement noise matrix, measure of how the difference between the error on the measurements and its expectation value change together.

Table 3-2: Stochastic variables of the KF.

These stochastic matrices will be properly identified for each of the KFs defined in chapters 5 and 6. The governing equations of the KF will be outlined later on for each of the two KFs that we use.

As I highlighted before, it is important to notice that this diagram shows a general overview of how an EKF works, but it is not important the same as the INAs that will be introduced in the following chapters.

It is a priority to see that like in the observer case, we can use the KF as a continuous or a discrete-time process. In our following cases, each one will be appropriately identified. The equations of each type of INA will be stated in the corresponding chapters. Though, a good reference to check the development of the equations of the KF algorithm is [21].

4 Symmetry-preserving observer

In this chapter, the invariant observer will be outlined and the advantages and especial nomenclature of the algorithm developed in detail. There will be an explanation of the correction part of the observer using the errors and gains to enhance the estimates. The choice of the tuneable parameters, which in this particular case are mainly the gains, is going to be explained. Finally, the discussions on the behaviour of this algorithm are stated.

4.1 Inertial sensor models

It is important to clarify the accelerometer sensor model:

$$\left(\vec{f}\right)_{B,input} = a_s \cdot \breve{q}_{EB}{}^{-1} * \begin{pmatrix} 0 \\ (\vec{g})_W \end{pmatrix} * \breve{q}_{EB} \tag{4-1}$$

Also the gyroscope sensor model:

$$(\vec{\omega}^{WB})_{B,input} = (\vec{\omega}^{WB})_{B,true\ input} - \vec{b}_g \tag{4-2}$$

The acceleration is understood as a rotation of the gravity vector multiplied by a scaling factor a_s. Although the model is defined like that, we use the accelerometer data in the corresponding b-frame as input to the system as it is more realistic. The gyroscope sensor model, instead of the former, takes into account the biases of the triad. This is also used with the RIEKF, but not with the MEKF. With the latter, a different accelerometer and gyroscope sensor definitions are used. It is also comprehended a scaling factor but in a different way and for both sensor models. In the same manner, biases are considered for both models as well.

This sensor models also applies for the RIEKF, but not for the MEKF.

4.2 Observer equations and nomenclature

The definition of the system is the main point to design afterwards the observer algorithm. It is important to notice that the definition of this system is the same of the one used for the RIEKF 5, but is slightly different from the used MEKF 6.

The following equations define the states that are going to be estimated (this is the first equation defined in (3-4)):

$$\begin{aligned} \dot{\breve{q}}_{EB} &= \frac{1}{2}\breve{q}_{EB} * \left(\begin{pmatrix} 0 \\ (\vec{\omega}^{WB})_{B,input} \end{pmatrix} - \begin{pmatrix} 0 \\ \vec{b}_g \end{pmatrix} \right) \\ \left(\dot{\vec{v}}\right)_W^W &= (\vec{g})_W + \frac{1}{a_s}\breve{q}_{EB} * \begin{pmatrix} 0 \\ \left(\vec{f}\right)_{B,input} \end{pmatrix} * \breve{q}_{EB}{}^{-1} \end{aligned} \tag{4-3}$$

$$\dot{\vec{b}}_g = 0$$

$$\dot{a}_s = 0$$

It can be seen that it is desired to estimate the attitude in a quaternion form, the velocity in the w-Frame, the gyro biases and the scaling factor of the accelerometers sensor model. After defining the system equation as stated above, the output equation that show the measurements taken into account as an aid to improve the estimates is outlined as:

$$\vec{y}_V = (\vec{v})^W_{W,meas}$$

$$\vec{y}_B = \left\| \breve{q}_{EB}{}^{-1} * \begin{pmatrix} 0 \\ (\vec{B})_{W,meas} \end{pmatrix} * \breve{q}_{EB} \right\| \tag{4-4}$$

This equation is the same as the second equation in (3-4), the differences are that here the equations are decoupled and in the reference they are shown in a matrix form.

The value of $(\vec{\boldsymbol{B}})_{\boldsymbol{W,meas}} = (\vec{\boldsymbol{B}})_{\boldsymbol{W,nom}} = (\mathbf{1} \quad \mathbf{0} \quad \mathbf{1})^{\boldsymbol{T}}$ is used for all the algorithms for the simulations with the true data trajectories. For the case of the real-time data, the direct measurement is used instead of rotating the nominal value with the corresponding estimated attitude and adding some noise as explained in 3.3.1.

Like the explanation in 3.4.1 introduce, the observer uses errors extracted from the measurements to correct the estimates of the states and obtain better results. In the case of the observer, these error equations are multiplied by some gain factors to give a certain amount of importance to the measurements. In the invariant observer, the following are the error equations utilized for the observer to correct the states with the measurements of the velocity and magnetic field. In this case, these errors are defined in an invariant manner using the theory in 2.7. The equations outline an invariant error definition:

$$\vec{E}_V = \tilde{\vec{y}}_V - \vec{y}_V = \left(\tilde{\vec{v}}\right)^W_W - (\vec{v})^W_{W,meas}$$

$$\vec{E}_B = (\vec{B})_{W,meas} - \tilde{\breve{q}}_{EB} * \begin{pmatrix} 0 \\ \vec{y}_B \end{pmatrix} * \tilde{\breve{q}}_{EB}{}^{-1} \tag{4-5}$$

$$= (\vec{B})_{W,meas} - \tilde{\breve{q}}_{EB} * \left\| \breve{q}_{EB}{}^{-1} * \begin{pmatrix} 0 \\ (\vec{B})_{W,meas} \end{pmatrix} * \breve{q}_{EB} \right\| * \tilde{\breve{q}}_{EB}{}^{-1}$$

The definition of the invariant error will be of paramount importance for the design of the observer as will be seen on the following lines. Previously, the rotational rate and specific acceleration vectors are displayed:

$$\vec{I}_w = \tilde{\breve{q}}_{EB} * \left(\begin{pmatrix} 0 \\ (\vec{\omega}^{WB})_{B,input} \end{pmatrix} - \begin{pmatrix} 0 \\ \tilde{\vec{b}}_g \end{pmatrix} \right) * \tilde{\breve{q}}_{EB}{}^{-1} \tag{4-6}$$

$$\vec{I}_a = \frac{1}{\tilde{a}_s}\tilde{\bar{q}}_{EB} * \begin{pmatrix} 0 \\ (\vec{f})_{B,input} \end{pmatrix} * {\tilde{\bar{q}}_{EB}}^{-1}$$

The variables expressed above are described in the following table:

Variable	Description
$\vec{E}_V$	Invariant output error in velocity measurement, in w-Frame.
$\vec{E}_B$	Invariant output error in magnetic field measurement, in w-Frame.
$\vec{I}_w$	Estimated rotational rate vector, in w-Frame.
$\vec{I}_a$	Estimated specific acceleration vector, in w-Frame.

Table 4-1: Invariant variables of the observer's correction term.

These terms are used in the observer correction step and are multiplied by the gains to obtain the most suitable possible response. It is recommended to notice that they are functions of the estimates and the measurements; as a consequence they are known variables used for the construction of the observer.

Another characteristic to take into account is that the two latter variables, in the way they are defined, show the type of trajectory the UAV is performing. For instance, the case where $\dot{\vec{I}}_a = \mathbf{0} \xrightarrow{yields} \vec{I}_a = \boldsymbol{constant}$ and $\vec{I}_w = \mathbf{0}$, denotes the aircraft is in level-flight.

As it is investigated before, the concept of invariance of these error equations allow the system correction terms to be decoupled into the following four different subsystems.

The longitudinal subsystem:

$$\begin{pmatrix} \delta\dot{\tilde{\eta}}_2 \\ \delta\dot{\tilde{v}}_1 \\ \delta\dot{\tilde{\beta}}_2 \end{pmatrix} = \begin{pmatrix} 0 & kl_v(g)_{W_3} & -\frac{1}{2} \\ -2k(g)_{W_3} & -m_v & 0 \\ 0 & -kn_v(g)_{W_3} & 0 \end{pmatrix} \begin{pmatrix} \delta\tilde{\eta}_2 \\ \delta\tilde{v}_1 \\ \delta\tilde{\beta}_2 \end{pmatrix} \quad \textbf{(4-7)}$$

The lateral subsystem:

$$\begin{pmatrix} \delta\dot{\tilde{\eta}}_1 \\ \delta\dot{\tilde{v}}_2 \\ \delta\dot{\tilde{\beta}}_1 \end{pmatrix} = \begin{pmatrix} 0 & -kl_v(g)_{W_3} & -\frac{1}{2} \\ 2k(g)_{W_3} & -m_v & 0 \\ 0 & kn_v(g)_{W_3} & 0 \end{pmatrix} \begin{pmatrix} \delta\tilde{\eta}_1 \\ \delta\tilde{v}_2 \\ \delta\tilde{\beta}_1 \end{pmatrix} \quad \textbf{(4-8)}$$

The vertical subsystem:

$$\begin{pmatrix} \delta\dot{\tilde{v}}_3 \\ \delta\dot{\alpha} \end{pmatrix} = \begin{pmatrix} -m_v & k(g)_{W_3} \\ -k(g)_{W_3} o_v & 0 \end{pmatrix} \begin{pmatrix} \delta\tilde{v}_3 \\ \delta\alpha \end{pmatrix} \tag{4-9}$$

The heading subsystem:

$$\begin{aligned} \begin{pmatrix} \delta\dot{\tilde{\eta}}_3 \\ \delta\dot{\tilde{\beta}}_3 \end{pmatrix} = & \begin{pmatrix} -2k^2(g)_{W_3}{}^2 l_B(\tilde{B})_{W,meas_1}{}^2 & -\frac{1}{2} \\ 2k^2(g)_{W_3}{}^2 n_B(\tilde{B})_{W,meas_1}{}^2 & 0 \end{pmatrix} \begin{pmatrix} \delta\tilde{\eta}_3 \\ \delta\tilde{\beta}_3 \end{pmatrix} \\ & + \begin{pmatrix} 2k^2(g)_{W_3}{}^2 l_B(\tilde{B})_{W,meas_3}(\tilde{B})_{W,meas_1} \\ -2k^2(g)_{W_3}{}^2 n_B(\tilde{B})_{W,meas_3}(\tilde{B})_{W,meas_1} \end{pmatrix} \delta\tilde{\eta}_1 \end{aligned} \tag{4-10}$$

It is important to highlight that the nomenclature on the decoupling of the system into the four different subsystems above is not the same that it is used on this thesis. Thus, with the purpose of clarifying this important part that is directly extracted from [3], the following table displays the changes in nomenclature that apply only for the equations (4-7) to (4-10).

Variable	Description
$\breve{\boldsymbol{\eta}} = \begin{pmatrix} \mathbf{0} \\ \vec{\boldsymbol{\eta}} \end{pmatrix}$	Vector representing the attitude error, in the quaternion form.
$\vec{\boldsymbol{v}}$	Vector representing the velocity error.
$\vec{\boldsymbol{\beta}}$	Vector representing the gyro bias error.
$\boldsymbol{\alpha}$	Vector representing the accelerometer scale-factor error.

Table 4-2: Invariant error system variables.

These variables are defined in **chapter 4 – page 63 of** [3] and denote the definition of the invariant error system to decouple the system in the four equations above. Without this structure the system could not be decoupled and the gains defined to give more importance in determined measurements would be very difficult to tune. The next table give displays some other important variables utilized in the equations above.

Variable	Description
$\breve{\boldsymbol{\eta}}_{\mathbf{0}}$	Quaternion representing a frame rotation of the vector $\vec{\boldsymbol{I}}_{\boldsymbol{a}}$ to be vertical.
$(\tilde{\boldsymbol{B}})_{\boldsymbol{W,meas}}$	Vector representing the Earth magnetic field, rotated by $\breve{\boldsymbol{\eta}}_{\mathbf{0}}$.
$\boldsymbol{k}$	Positive constant that relates the $(\vec{\boldsymbol{g}})_{\boldsymbol{W}}$ with $\vec{\boldsymbol{I}}_{\boldsymbol{a}}$.
$\tilde{\vec{\boldsymbol{\eta}}}$	Vector representing the attitude error, rotated by $\breve{\eta}_0$.
$\tilde{\vec{\boldsymbol{v}}}$	Vector representing the velocity error, rotated by $\breve{\boldsymbol{\eta}}_{\mathbf{0}}$.
$\tilde{\vec{\boldsymbol{\beta}}}$	Vector representing the gyro bias error, rotated by $\breve{\boldsymbol{\eta}}_{\mathbf{0}}$.

Table 4-3: Variables of the decoupled systems.

The meaning of each of the parameters above is not of particular importance. It is shown to understand that the gains act in different parts of the observer. For further details, it is recommended to check the reference [3].

Regarding the subsystems outlined before, the final gains to change the influence, which each of the measurements have on the estimates, are displayed:

$$\begin{aligned} L_v\vec{E}_V &= -l_v \cdot \vec{I}_a \times \vec{E}_V & L_B\vec{E}_B &= l_B \cdot \langle (\vec{B})_{W,meas} \times \vec{E}_B\,, \vec{I}_a \rangle \cdot \vec{I}_a \\ M_v\vec{E}_V &= -m_v \cdot \vec{E}_V & M_B\vec{E}_B &= 0 \\ N_v\vec{E}_v &= n_v \cdot \vec{I}_a \times \vec{E}_V & N_v\vec{E}_B &= -n_B \cdot \langle (\vec{B})_{W,meas} \times \vec{E}_B\,, \vec{I}_a \rangle \cdot \vec{I}_a \\ O_v\vec{E}_v &= o_v \cdot \langle \vec{I}_a, \vec{E}_V \rangle & O_B\vec{E}_B &= 0 \end{aligned} \quad \textbf{(4-11)}$$

These are the terms applied to the system equations to create the observer. That is the reason why it is said that the implementation of an observer is usually straightforward, but not the design of such an observer. A proper discussion of the gains and their effects on the estimation of the states will be done in the following subsection 4.3.

After all that, the observer equations are outlined with the following equations:

$$\dot{\tilde{q}}_{EB} = \frac{1}{2}\,\tilde{q}_{EB} * \left(\begin{pmatrix} 0 \\ (\vec{\omega}^{WB})_{B,input} \end{pmatrix} - \begin{pmatrix} 0 \\ \vec{\tilde{b}}_g \end{pmatrix} \right) + \begin{pmatrix} 0 \\ L_v\vec{E}_V + L_B\vec{E}_B \end{pmatrix} * \tilde{q}_{EB} + \\ + \lambda \left(1 - \left\|\tilde{q}_{EB}\right\|^2\right) * \tilde{q}_{EB}$$

$$\left(\dot{\vec{v}}\right)_W^W = (\vec{g})_W + \frac{1}{\tilde{a}_s}\tilde{q}_{EB} * \begin{pmatrix} 0 \\ (\vec{f})_{B,input} \end{pmatrix} * \tilde{q}_{EB}^{\,-1} + \left(M_v\vec{E}_V + M_B\vec{E}_B\right) \quad \textbf{(4-12)}$$

$$\dot{\vec{\tilde{b}}}_g = \tilde{q}_{EB}^{\,-1} * \begin{pmatrix} 0 \\ N_v\vec{E}_V + N_B\vec{E}_B \end{pmatrix} * \tilde{q}_{EB}$$

$$\dot{\tilde{a}}_s = \tilde{a}_s\left(O_v\vec{E}_V + O_B\vec{E}_B\right)$$

It is easily seen how straightforward is to identify on each of the equations the correction part and the propagation part, which is the part illustrated on the system equations. The term added in the quaternion equation denoted as $\lambda\left(1 - \left\|\tilde{q}_{EB}\right\|^2\right) * \tilde{q}_{EB}$ is a well-known trick to maintain the norm of the quaternion equal to 1. The constant λ is set to 1, but this is not the unique option.

4.3 Selection of the tuneable parameters and results

Knowing how an observer works and how is designed this specific invariant observer, the only point that lasts in the way to obtain the best possible estimation of the states is to tune the gains. Commonly, this is not straightforward because the parameters can depend on more than one measurement at a time. This is the reason why the gains of this observer are defined in this decoupled form, this manner permits the system to be decoupled but what is more important to decouple the effect of each measurement on each of the variables.

The idea is to rely on the magnetic field measurement as less as possible. By this way, the only variable that is affected by the magnetic field measurement is the yaw angle and the corresponding bias, as it can be seen in the heading subsystem (4-10).

It is also proved that when the observer estimations converge, the following correction terms become zero:

$$N_v\vec{E}_V + N_B\vec{E}_B = 0 \qquad O_v\vec{E}_V = 0 \tag{4-13}$$

This convergence result leads to:

$$\vec{I}_a \times \vec{E}_V = 0 \qquad \langle \vec{I}_a, \vec{E}_V \rangle = 0 \tag{4-14}$$

So even if the model is wrong, for example a perturbation is currently affecting the magnetic field, the observer equations ensure $\left(\tilde{\vec{v}}\right)_W^W = (\vec{v})_{W,meas}^W$ once the observer has converged. For further details, it is recommended to check the reference [3].

With this decoupled system structure, the results in [3] show that each of the subsystems locally asymptotically converge for every "smooth" trajectory; i.e. when $\dot{\hat{\eta}}_0$, $\dot{\vec{I}}_{\boldsymbol{a}}$ and $\vec{\boldsymbol{I}}_{\boldsymbol{w}}$ can be neglected, because of first order terms.

The main difference that an observer has when comparing to a KF is that the gains are fixed, this means that a correct choice of the gains is the key point for the observer to estimate correctly the states. Basically, we can consider an observer as a Kalman Filter when it has converged. The gains of the KF in such a phase become more or less constant. Thus, the best way of choosing the most appropriate gains is to try to guess which ones are the steady-state values.

Another point to take into account when using an observer is that because the gains are fixed, the algorithm is less flexible and, hence, it is better to adapt the gains for every different trajectory to estimate.

To make the most suitable choice, I have used two different trajectories to compare the best gain configuration for each of them. Then, it is also explained the most appropriate gain configuration for an experimental case, where there is no defined trajectory and where the UAV can follow an infinite amount of trajectories during the same flight.

To start dealing with the gains parameters defined in [3], the following table displays each one of them:

Gain coefficient	Description
$\boldsymbol{l_v}$	Gain coefficient to tune the first two attitude angle $\phi\ and\ \theta$ estimations, using the velocity measurement.
$\boldsymbol{n_v}$	Gain coefficient to tune the first two gyro bias

	element estimations corresponding to the attitude angles $and\ \theta$, using the velocity measurement.
$\boldsymbol{l_B}$	Gain coefficient to tune the yaw angle ψ estimation, using the magnetic field measurement.
$\boldsymbol{n_B}$	Gain coefficient to tune the last gyro bias element estimation corresponding to the yaw angle ψ, using the magnetic field measurement.
$\boldsymbol{m_v}$	Gain coefficient to tune the three velocity element estimations, using the velocity measurement.
$\boldsymbol{o_v}$	Gain coefficient to tune the scaling factor estimation, using the velocity measurement.

Table 4-4: Gain parameters of the observer.

As it is shown, there are 6 gain coefficients to tune. From these gain coefficients, only two of them are using the magnetic field as a measurement to correct the variables that they estimate. All the other 4 coefficients are applied to the velocity measurement.

After some test checks and manipulations of the gain configuration, I have noticed that it is desirable to use a specific gain configuration for each type of trajectory. This will improve the estimation in this specified trajectory.

Moreover, depending on how smooth the estimation is desired to be, the gain configuration should be different. The smoothness is not the most accurate factor to define the type of estimation, but I think that after showing the results what I mean with this term will be understood.

The observer has been tested in a quite representative amount of trajectories. However, I have selected two of them to make a comparison. For each of these two trajectories a certain gain configuration has been thought to be most accurate for our purposes. The next table displays the values of the two selected gain configuration:

Gain Coefficient	Trajectory 1 StateLog_Seq6	Trajectory 2 StateLog_Seq6_2
$\boldsymbol{l_v}$	$2.4e^{-3}$	$3e^{-2}$
$\boldsymbol{n_v}$	$3e^{-2}$	$3e^{-2}$
$\boldsymbol{l_B}$	$9e^{-4}$	$4.5e^{-4}$
$\boldsymbol{n_B}$	$8e^{-3}$	$5e^{-3}$
$\boldsymbol{m_v}$	0.9	1.2
$\boldsymbol{o_v}$	$2e^{-3}$	$1e^{-3}$

Table 4-5: Gain parameter values for the 2 trajectories.

With these two gain configurations, I tried to adapt the system more specifically for each of the cases and improve the estimation of the gyro biases and of the scaling factor. In [3] two different gain configurations are used for a simulated trajectory that is not very turbulent and for a specific part of a real case trajectory. These other two gain configurations are commented more in advance.

It is also fundamental the selection of the initial values for the observer as a starting point for the observer algorithm. These initial values are shown in the next figure and are selected far from the true values to show the convergence properties of the observer:

Variable to estimate	Real initial values for trajectories 1 and 2	Initial values of the observer for Trajectories 1 and 2
$\boldsymbol{q_0[rad]}$	1	$\cos(\pi/3)$
$\boldsymbol{q_1[rad]}$	0	$\sin(\pi/3)/\sqrt{3}$
$\boldsymbol{q_2[rad]}$	0	$-\sin(\pi/3)/\sqrt{3}$
$\boldsymbol{q_3[rad]}$	0	$\sin(\pi/3)/\sqrt{3}$
$\boldsymbol{u}\ [\boldsymbol{m/s}]$	0	10
$\boldsymbol{v}\ [\boldsymbol{m/s}]$	0	-10
$\boldsymbol{w}\ [\boldsymbol{m/s}]$	0	5
$\boldsymbol{b_{g_x}}\ [°/\boldsymbol{s}]$	1	1
$\boldsymbol{b_{g_y}}\ [°/\boldsymbol{s}]$	2	2
$\boldsymbol{b_{g_z}}\ [°/\boldsymbol{s}]$	1	1
$\boldsymbol{a_s}$	1.1	1.1

Table 4-6: Initial true and estimated values.

Regarding the values of the Table 4-5 that have been chosen, the results are shown. The following five figures are the estimation of the variables of the trajectory 1 with the gains of the table above:

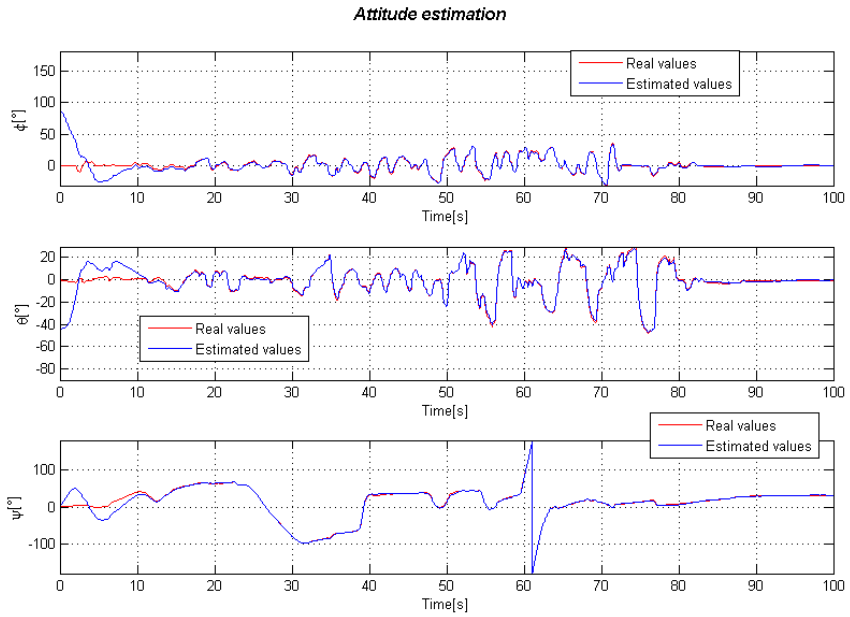

Figure 4-1: Attitude estimation for trajectory 1.

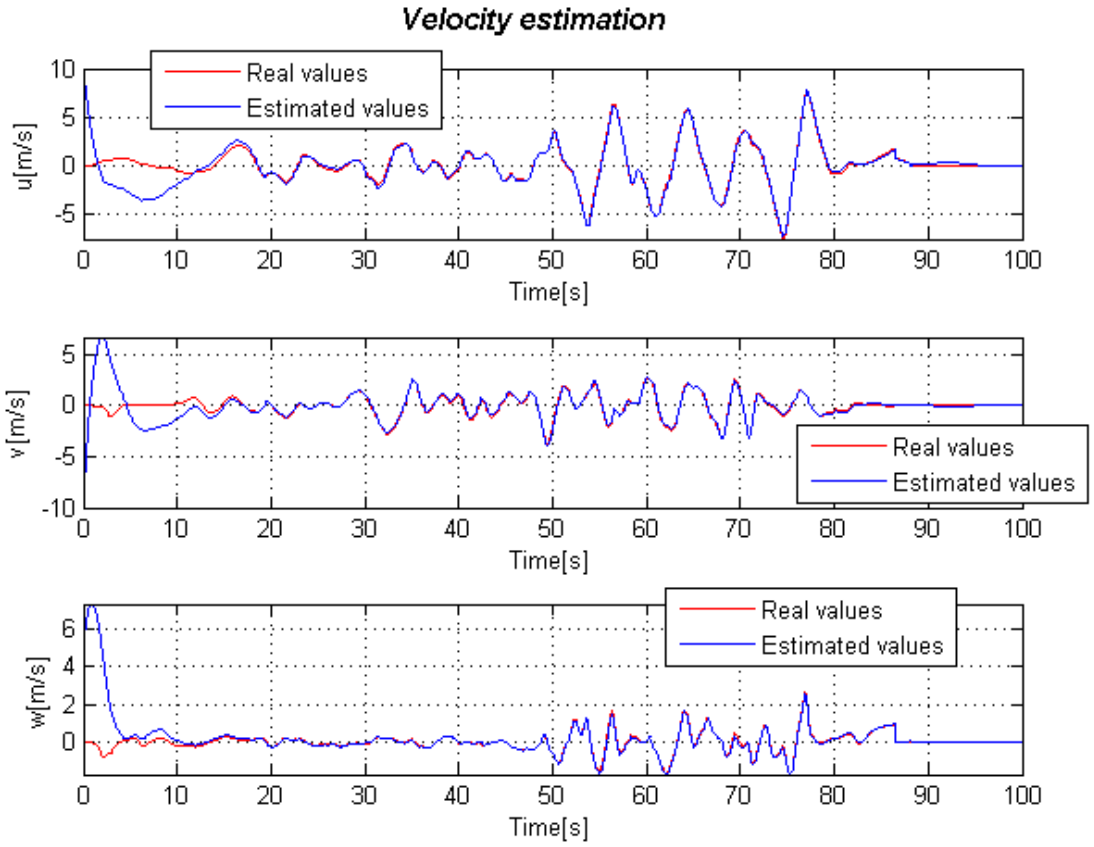

Figure 4-2: Velocity estimation for trajectory 1.

The initial attitude angles and velocity elements are set far away from the real initial values to test the convergence of the observer, as shown in Table 4-6. In the particular case of the attitude angles are initialized at an angle of $\boldsymbol{\pi/3\ rad}$ far from the real initial position. By this way, it is proved that the observer converges more than locally. This conclusion is extracted in the paper [18], it is literally said that: "*The convergence is far from being only local. We suspect much stronger stability. We conjecture that such non-linear invariant observer is almost globally convergent.*"

It is also important to notice that: *"One can expect local convergence around every trajectory of the system and not only in its equilibrium points or "slowly-varying trajectories". The global behavior tends to be better and the region of attraction larger (compared e.g. to a Luenberger observer). Furthermore, the invariance property of this observer is often desirable from an engineering point of view."*

Moreover, the gyro biases and the scaling factor are initialized with the same initial values as the real ones. The following two figures show how these two estimated variables convergence to the real values:

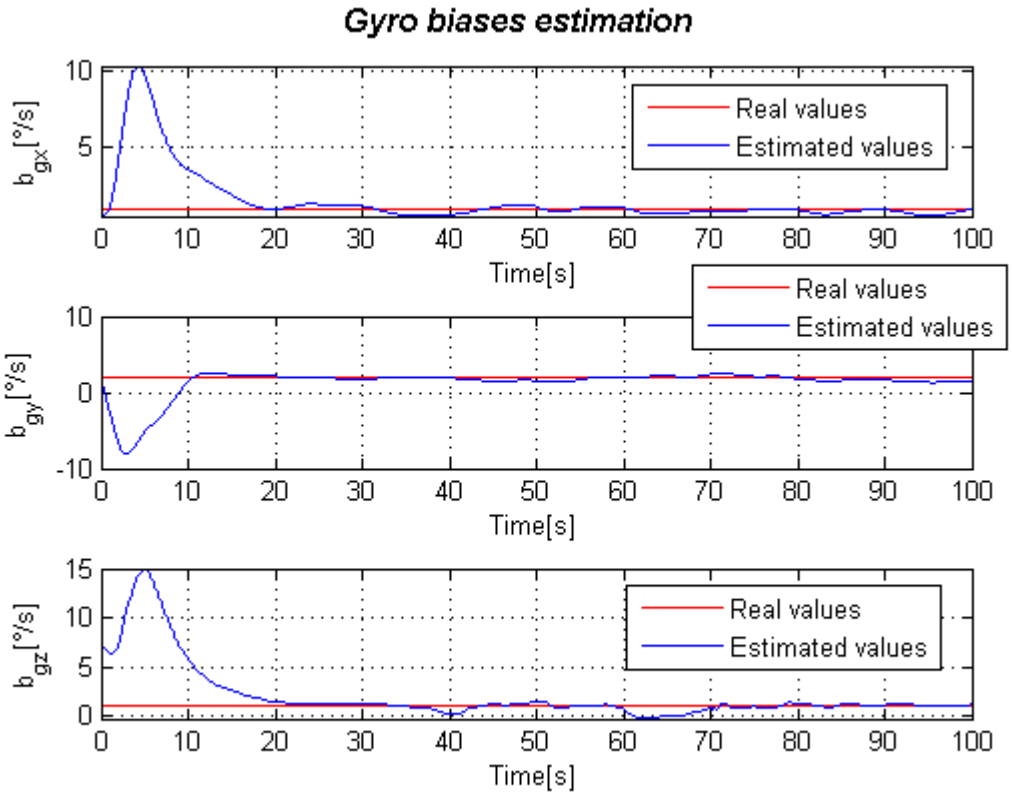

Figure 4-3: Gyro biases estimation for trajectory 1.

For the case of the scaling factor, it can be seen that the estimation is turbulent. This is because of the election of the gains.

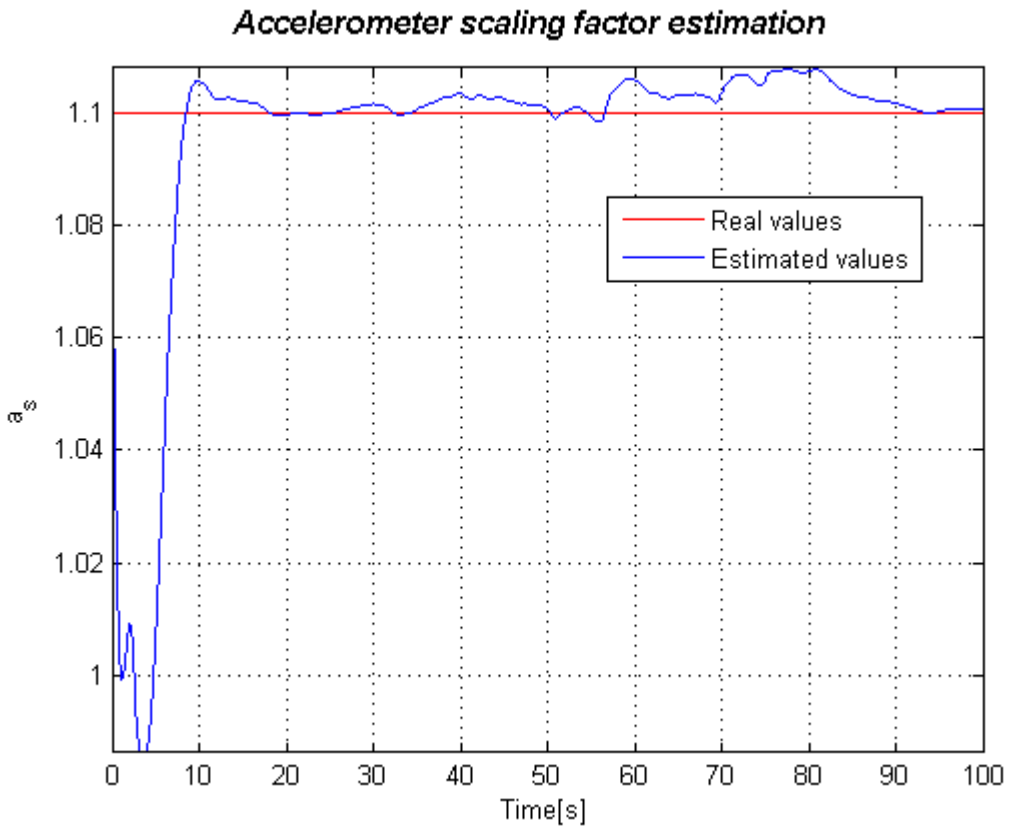

Figure 4-4: Scaling factor estimation for trajectory 1.

The smooth behavior that I was talking about some paragraphs above can be observed in the attitude error between the real values and the estimated ones.

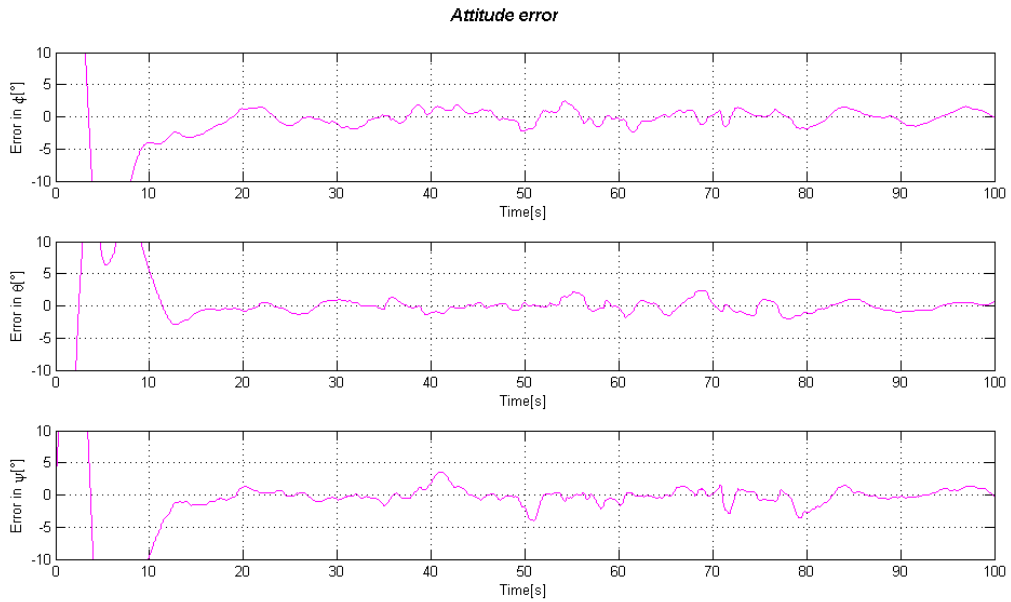

Figure 4-5: Attitude error on the estimation for trajectory 1.

The same plots for the trajectory 2 with the appropriate selected gains shown in Table 4-5 are displayed in the following five figures:

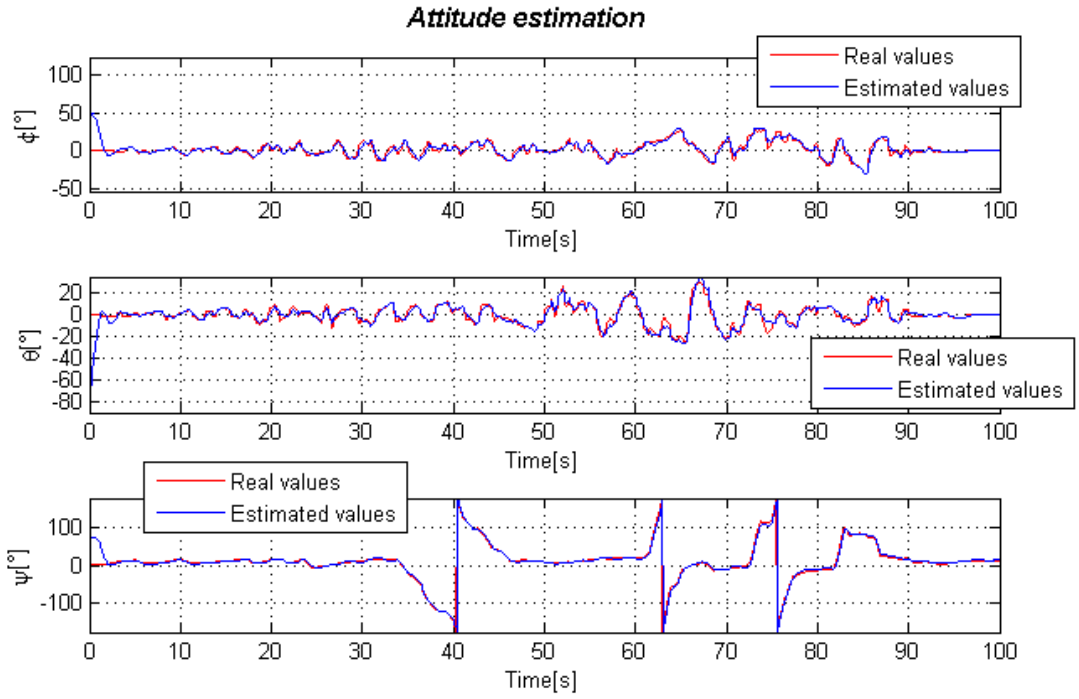

Figure 4-6: Attitude estimation for trajectory 2.

The attitude estimation in this case in comparison with the results of trajectory 1 is preferable, because the convergence is really faster at the beginning than in the previous case. There are some points on the estimation of ϕ and θ, which seems not to follow the real values properly but it is not bad as this trajectory changes more abruptly than the first one.

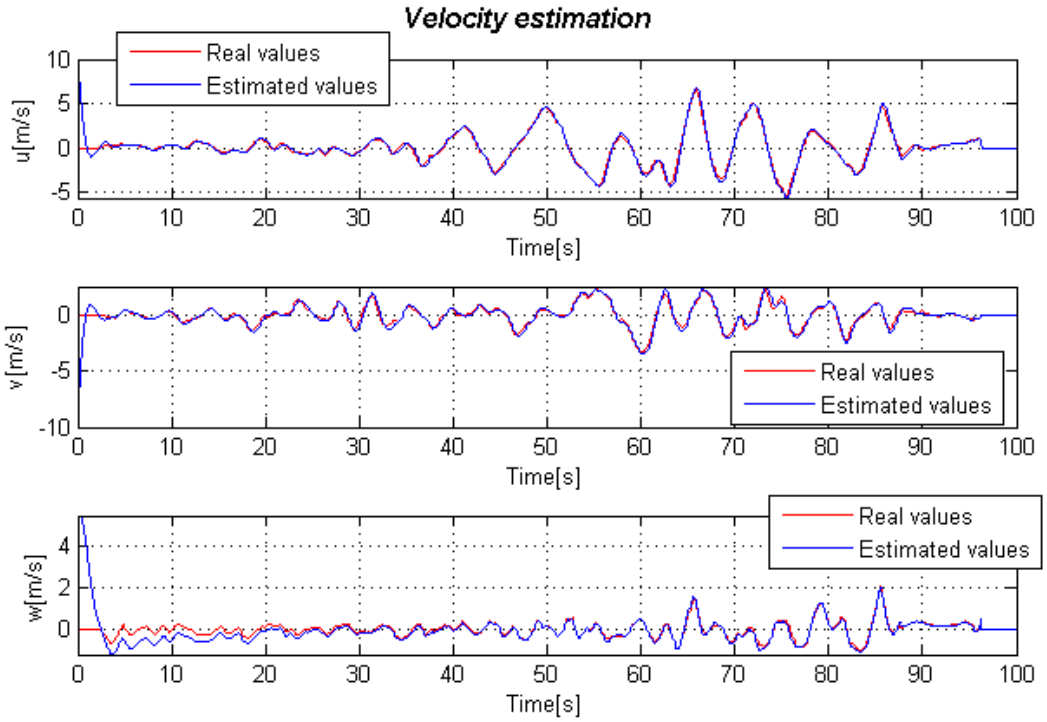

Figure 4-7: Velocity estimation for trajectory 2.

The velocity estimation shows a scaling between the estimation and the real value for the Z-axis component till the estimation is stabilized at about 30 seconds.

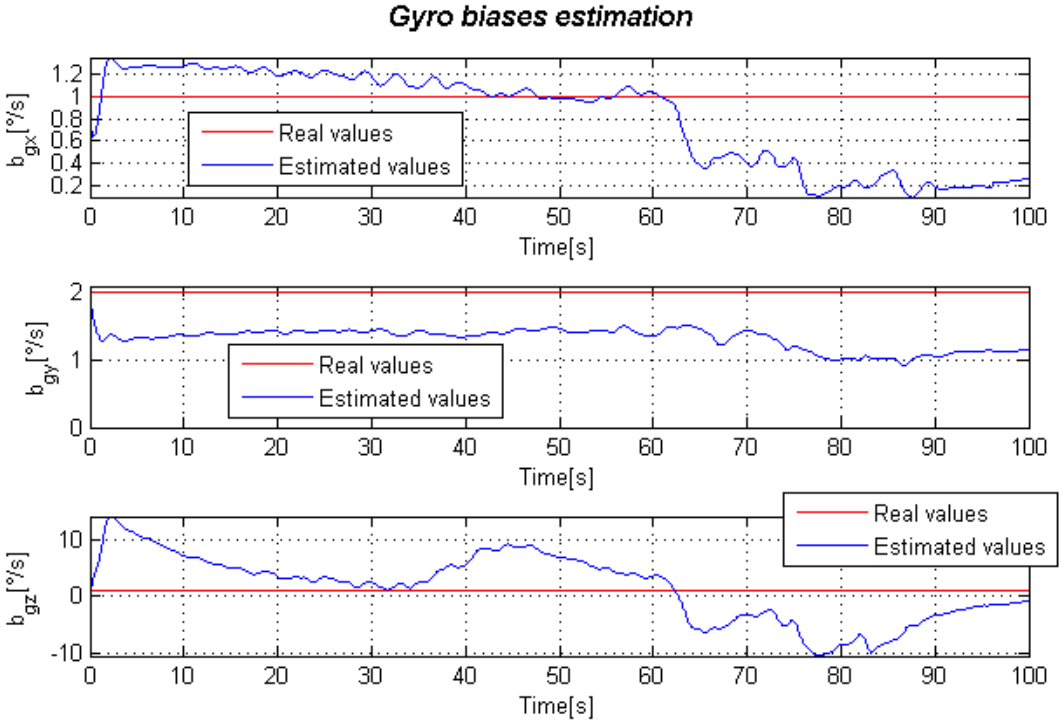

Figure 4-8: Gyro biases estimation for trajectory 2.

Also the gyro bias estimations in this second trajectory are slightly different from the ones in trajectory 1. In the previous case, it can be seen that the estimation of the biases makes a sharp increase at the beginning and converge afterwards. In this case the result is preferable because there is no this sudden difference at the beginning.

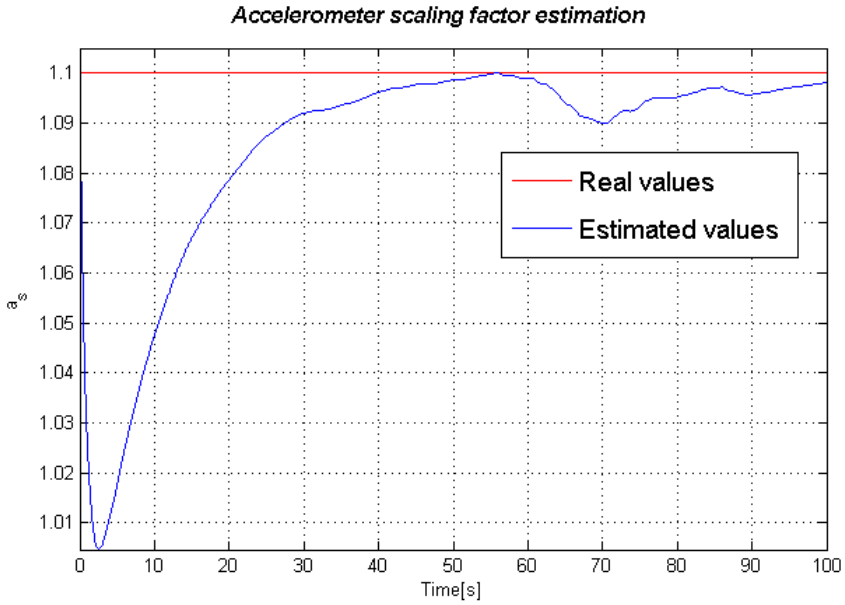

Figure 4-9: Scaling factor estimation for trajectory 2.

Regarding the scaling factor, it is similar than the previous case. A closer examination of the attitude error can be performed and the same turbulence as before is illustrated.

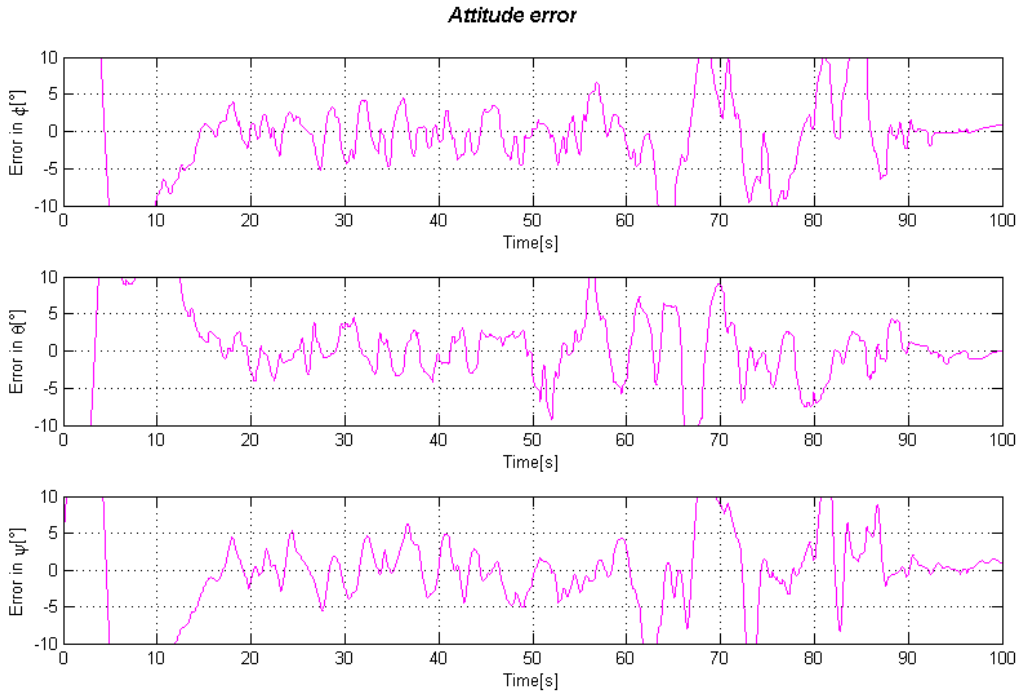

Figure 4-10: Attitude error on the estimation for trajectory 2.

After analyzing these two cases, we are going to choose the second trajectory and use the gain configuration of the thesis [3] to compare the two gain selections. The gain configuration stated in [3] is used for an experimental case, which means that is better in a general point of view and it is not adapted for a specific trajectory. In the next table, it is displayed the gain configuration for the experimental case:

Gain Coefficient	Gain configuration for Trajectory 2
$\boldsymbol{l_v}$	$2.8e^{-3}$
$\boldsymbol{n_v}$	$4e^{-5}$
$\boldsymbol{l_B}$	$7e^{-3}$
$\boldsymbol{n_B}$	$1e^{-4}$
$\boldsymbol{m_v}$	0.9
$\boldsymbol{o_v}$	$9.4e^{-5}$

Table 4-7: Gain configuration for an experimental case, extracted from [3].

Taking a closer look, it can be observed that the gains, in general, are slightly smaller than in the case of Table 4-5 for the trajectory 2. Selecting smaller gains normally tends to make the estimation smoother and without sudden changes. The best improvement of this gain configuration is the biases estimation. The following plot shows the biases estimation for such a case:

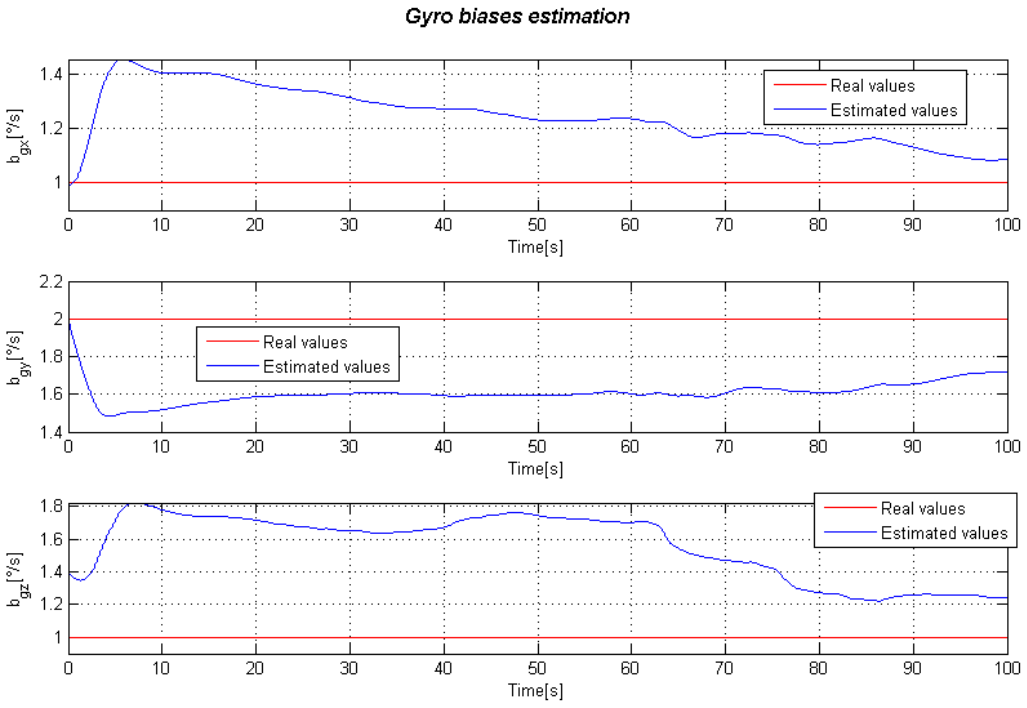

Figure 4-11: Gyro biases estimation for trajectory 2, experimental case.

As it can be seen, the biases converge very slowly to the real values and it is probably needed more time for them to reach the real values. It is clear that the response is smoother, though. The concept of smoothness can also be seen on the following attitude error plot:

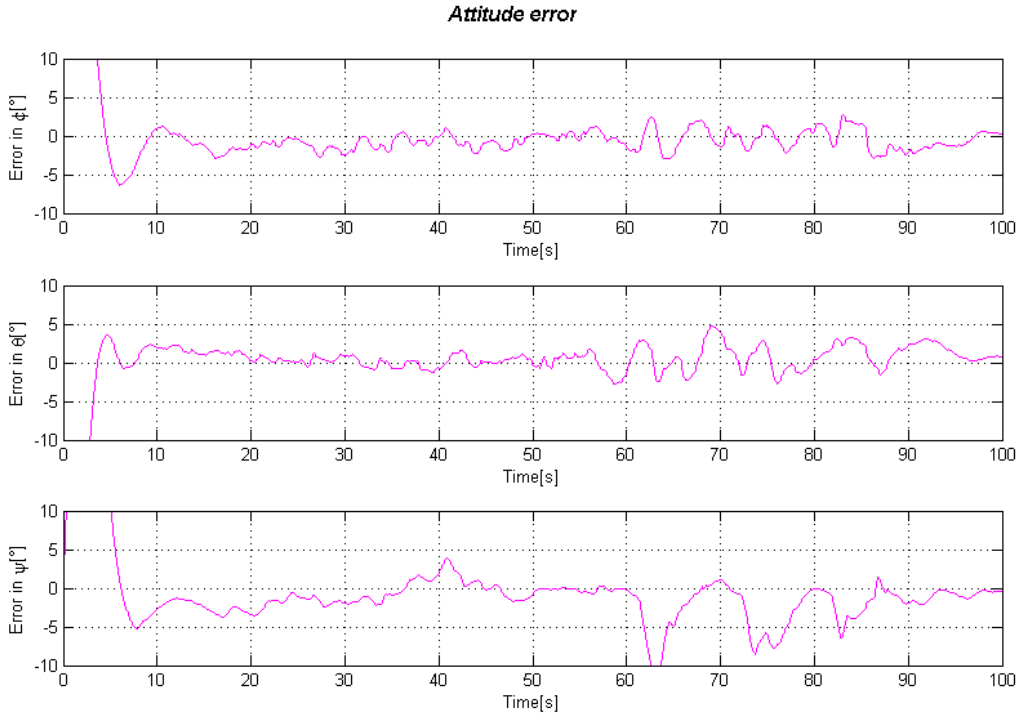

Figure 4-12: Attitude error on the estimation for trajectory 2, experimental case.

That is the reason why the experimental gain configuration is going to be used in the following chapters to make a comparison between the estimations of this observer and the KFs that will be explained in chapters 5 and 6.

5 Right Invariant Extended Kalman Filter (RIEKF)

Going on with the INAs' definitions, chapter 5 defines the RIEKF, its advantages and specific characteristics. As it is explained in the previous chapter, the specific nomenclature of this KF will be stated. The explanation compared with the normal system equations introduced in section 3.1.2 will be done. Finally, the breakthroughs of this algorithm will be concluded and the behaviour of the estimations evaluated in detail.

First of all, the equations of the RIEKF will be explained and then the implementation of the system to these equations will be shown. This KF type is implemented in a linearized continuous-time way and this means that the algorithm is more time-consuming, but this is not a problem in this case as this same algorithm can be discretized and implemented in a discrete-time way to enhance its efficiency. To see an easier implementation of the time-discrete algorithm, it is recommended to check [22]. In this thesis, it is implemented like that because the work is based on the PhD thesis in [3], and the algorithm is defined in a time-continuous manner.

Other interesting related papers that show easier examples to implement of other IEKF versions like Left IEKF (LIEKF) can be found in [23] and [24]. These examples show how the invariances are used in an EKF for systems with fewer variables to estimate than in our case.

5.1 Symmetry-preserving observers theory for IEKF

In this section, the idea is to extend the general definitions of the symmetry-preserving observers theory introduced in 2.7 and adapt them to the case of the IEKF as explained in [24]. This part is directly extracted from [24] and clarifies the equations that preserve invariance properties for the specific case of the EKF. With these clarifications, the equations used in the following section are more straightforward to comprehend.

As it is developed in [24] and using the same nomenclature as in 2.7, the general definitions stated in 2.7.2 can be adapted to the following equations:

"We now state the two main results in the special case where $g \mapsto \varphi_g(x)$ is invertible (i.e. when G is of dimension n), see [18] *for the general case. $\mathcal{X}$ can then be (locally) identified with G; if $\mathcal{X} = G$ from start with globally defined transformations (as in the example treated in this paper), all computations are moreover global. The group action coincides with left translations L_g, i.e. $\varphi_g(x) = L_g(x)$; right translations R_g write $R_g(x) = x \cdot g = \varphi_x(g)$.*

Theorem 1: *A symmetry-preserving observer reads*

$$\dot{\tilde{\hat{x}}}(t) = f\left(\tilde{\hat{x}}(t), \vec{u}(t)\right) + \mathrm{D}L_{\tilde{\hat{x}}(t)}(e) \cdot K \cdot \left(\rho_{\tilde{\hat{x}}(t)^{-1}}\left(\vec{y}(t)\right) - \rho_{\tilde{\hat{x}}(t)^{-1}}\left(h\left(\tilde{\hat{x}}(t), \vec{u}(t)\right)\right)\right) \qquad \textbf{(5-1)}$$

Where the matrix gain K may depend only on the invariant quantity $\tilde{I} := \psi_{\tilde{\hat{x}}(t)^{-1}}(u)$ and of the invariant output error $\rho_{\tilde{\hat{x}}(t)^{-1}}\left(\vec{y}(t)\right) - \rho_{\tilde{\hat{x}}(t)^{-1}}\left(h\left(\tilde{\hat{x}}(t), \vec{u}(t)\right)\right)$.

Instead of using the usual "linear" error $\tilde{\vec{x}} - \vec{x}$, we can now use an invariant error, with the remarkable following property.

Theorem 2: *The error system for the invariant state error $\vec{\eta} := \vec{x}^{-1} \cdot \tilde{\vec{x}}$ reads $\dot{\vec{\eta}} = \Upsilon(\vec{\eta},\ \tilde{I})$.*

Explicitly, after using repeatedly invariance properties,

$$\begin{aligned}
&\dot{\vec{\eta}}(t) = \mathrm{D}L_{\vec{x}(t)^{-1}}\left(\tilde{\vec{x}}(t)\right) \cdot \dot{\tilde{\vec{x}}}(t) + \mathrm{D}R_{\tilde{\vec{x}}(t)}(\vec{x}(t)^{-1}) \cdot \dot{\vec{x}(t)}^{-1} \\
&= \mathrm{D}L_{\vec{x}(t)^{-1}}\left(\tilde{\vec{x}}(t)\right) \cdot \dot{\tilde{\vec{x}}}(t) - \mathrm{D}R_{\tilde{\vec{x}}(t)}(\vec{x}(t)^{-1}) \\
&\cdot \left(\mathrm{D}L_{\vec{x}(t)^{-1}}(e) \cdot \mathrm{D}L_{\vec{x}(t)^{-1}}\left(\vec{x}(t)\right) \cdot \dot{\vec{x}}(t)\right) \\
&= \mathrm{D}L_{\vec{\eta}(t)}(e) \cdot f(e, \tilde{I}) - \mathrm{D}R_{\vec{\eta}(t)}(e) \cdot f\left(e, \psi_{\vec{\eta}(t)}(\tilde{I})\right) + \mathrm{D}L_{\vec{\eta}(t)}(e) \cdot K \\
&\cdot \left(h(\vec{\eta}(t)^{-1}, \tilde{I}) - h(e, \tilde{I})\right)
\end{aligned} \tag{5-2}$$

This result greatly simplifies the convergence analysis, since the error equation is autonomous but for the "free" known invariant $\tilde{I}$. For a general (not symmetry-preserving) nonlinear observer the error equation depends on the trajectory $t \mapsto \left(x(t), u(t)\right)$ of the system, hence is in fact of around equilibrium points (the linearization of the error equation around an equilibrium point is autonomous) to the much wider class of the so-called permanent trajectories characterized by the fact that $\tilde{I}$ is constant along them."

In our case, these permanent trajectories are defined as the trajectories that have the specific acceleration $\vec{I}_a$ and rotational rate $\vec{I}_w$ vectors constant along them.

5.2 RIEKF equations and nomenclature

The nomenclature used in this type of KF is the same as in the case of the observer in 4, so my purpose in that is to identify the usual variables of the KF nomenclature and match them with the variables of Table 3-1and Table 3-2.

Firstly, the system and output equations are defined:

$$\begin{aligned}
\dot{\vec{x}}(t) &= f\left(\vec{x}(t), \vec{u}(t)\right) + M \cdot \vec{w}(t) \\
\vec{y}(t) &= h\left(\vec{x}(t), \vec{u}(t)\right) + N \cdot \vec{v}(t)
\end{aligned} \tag{5-3}$$

The following table displays the corresponding matrices for the preceding definitions on Table 3-2 and their relation with this new definition:

Variable	Description
$\boldsymbol{P}$	State covariance matrix, measure of how the difference between the error on the states and its expectation value change together.
$\boldsymbol{Q{\sim}M}$	Input covariance matrix or process/driving noise matrix, measure of how the difference

	between the error on the inputs and its expectation value change together.
$\boldsymbol{R}\sim\boldsymbol{N}$	Measurement covariance matrix or measurement noise matrix, measure of how the difference between the error on the measurements and its expectation value change together.

Table 5-1: Stochastic variables for the RIEKF nomenclature.

With the previous system and output equations definitions, the KF equations are grouped in one step. Instead of using a prediction – correction sequence, both phases are unified in one unique step that can be identified as the equivalent of the two step normal sequence.

Unique step:

$$K = P \cdot C^T \cdot (N \cdot N^T)^{-1}$$

$$\dot{\tilde{\vec{x}}}(t) = f\left(\tilde{\vec{x}}(t), \vec{u}(t)\right) + K \cdot \left(\vec{y}(t) - h\left(\tilde{\vec{x}}(t), \vec{u}(t)\right)\right) \quad \textbf{(5-4)}$$

$$\dot{P} = \mathrm{A} \cdot P + P \cdot A^T + M \cdot M^T - P \cdot C^T \cdot (N \cdot N^T)^{-1} \cdot C \cdot P = \mathrm{A} \cdot P + P \cdot A^T + M \cdot M^T - K \cdot C \cdot P$$

Where A and C are:

$$A = \left.\frac{\partial\left(f\left(\vec{x}(t), \vec{u}(t)\right)\right)}{\partial\left(\vec{x}(t)\right)}\right|_{\vec{x}=\tilde{\vec{x}}} \qquad C = \left.\frac{\partial\left(h\left(\vec{x}(t), \vec{u}(t)\right)\right)}{\partial\left(\vec{x}(t)\right)}\right|_{\vec{x}=\tilde{\vec{x}}} \quad \textbf{(5-5)}$$

Finally, the only definition that lasts for this algorithm to work correctly is the one of the matrices A, C, M and N and the vector of Kalman gains (K):

$$A = \begin{pmatrix} 0_{3\times3} & 0_{3\times3} & -\frac{1}{2}I_3 & 0_{3\times1} \\ -2\cdot\left[\tilde{\vec{I}}_a\times\right] & 0_{3\times3} & 0_{3\times3} & -\left[\tilde{\vec{I}}_a\times\right] \\ 0_{3\times3} & 0_{3\times3} & \left[\tilde{\vec{I}}_w\times\right] & 0_{3\times1} \\ 0_{1\times3} & 0_{1\times3} & 0_{1\times3} & 0 \end{pmatrix} \quad \textbf{(5-6)}$$

$$C = \begin{pmatrix} 0_{3\times3} & I_3 & 0_{3\times3} & 0_{3\times1} \\ 2\cdot\left[\left(\vec{B}\right)_{W,meas}\times\right] & 0_{3\times3} & 0_{3\times3} & 0_{3\times1} \end{pmatrix} \quad \textbf{(5-7)}$$

$$M = Diag\left(M_{\tilde{q}_{EB}} \quad M_{(\vec{v})_W^W} \quad M_{b_g} \quad M_a\right) \quad \textbf{(5-8)}$$

$$N = Diag\begin{pmatrix} N_{(\vec{v})_W^W} & N_{(\vec{B})_W} \end{pmatrix} \quad \textbf{(5-9)}$$

$$K = -\begin{pmatrix} K_{\breve{q}_{EB}} & K_{(\vec{v})_W^W} & K_{b_g} & K_a \end{pmatrix}^T \quad \textbf{(5-10)}$$

As it is explained in [3]: *"The proposed filter uses a geometrically adapted correction term based on an invariant output error. For that to make sense from a stochastic point of view, we assume the driving and measurement noise enter the system in an invariant way.*

The main benefit of the IEKF is that the matrices A and C are constant on a much larger set of trajectories than the equilibrium points as is the case of the EKF. Informally, this means the IEKF should in general converge at least around any slowly-varying permanent trajectory, rather than just around any slowly-varying equilibrium point for the EKF."

Then, in accordance with [3], the two main specific features of the RIEKF are:

- Symmetry-preserving structure: Rotations, translations and scaling in the appropriated frames leave the error system unchanged.
- Larger expected domain of convergence: the matrices A and C used for computing the Kalman gains are constant not only in level flight but also on every permanent trajectory defined by constant $\vec{I}_w$ and $\vec{I}_a$.

It is also important to notice that the computation of the kalman gains only depends on the matrix C and consequently on the output equations, more precisely on the magnetic field measurement, but not directly on the system equations. Nevertheless, it has to be clarified that it depends indirectly on the state covariance matrix, which represents the system equations, and on the N matrix which represents the noise of the inputs in the system equations.

Proceeding with the development and taking into account the definitions of the symmetry-preserving theory stated in section 2.7.2, the second equation of (5-4) become:

$$\dot{\tilde{\vec{x}}}(t) = f\left(\tilde{\vec{x}}(t), \vec{u}(t)\right) + \mathrm{D}L_{\tilde{\vec{x}}(t)} \cdot K \cdot \left(\rho_{\tilde{\vec{x}}(t)^{-1}}\left(\vec{y}(t)\right) - \rho_{\tilde{\vec{x}}(t)^{-1}}\left(h\left(\tilde{\vec{x}}(t), \vec{u}(t)\right)\right)\right) \quad \textbf{(5-11)}$$

Where $L_{\tilde{\vec{x}}(t)}$ denotes the left multiplication by $\tilde{\vec{x}}(t)$, i.e. $L_{\tilde{\vec{x}}(t)}(z) = z \cdot \tilde{\vec{x}}(t)$ and where $\mathrm{D}L_g f(x) = f\left(L_g(x)\right)$.

Then if the system is decoupled in one equation for each state variable, the corresponding equations to (5-3) are separated in the following form:

$$\dot{\breve{q}}_{EB} = \frac{1}{2}\breve{q}_{EB} * \left(\begin{pmatrix} 0 \\ (\vec{\omega}^{WB})_{B,input} \end{pmatrix} - \begin{pmatrix} 0 \\ \vec{b}_g \end{pmatrix}\right) + M_{\breve{q}_{EB}} \cdot \omega_{\breve{q}_{EB}} * \breve{q}_{EB}$$

$$\left(\dot{\vec{v}}\right)_W^W = (\vec{g})_W + \frac{1}{a_s}\breve{q}_{EB} * \begin{pmatrix} 0 \\ (\vec{f})_{B,input} \end{pmatrix} * \breve{q}_{EB}{}^{-1} + M_{(\vec{v})_W^W} \cdot \omega_{(\vec{v})_W^W} \quad \textbf{(5-12)}$$

$$\dot{\vec{b}}_g = \breve{q}_{EB}{}^{-1} * \begin{pmatrix} 0 \\ M_{b_g} \cdot \omega_{b_g} \end{pmatrix} * \breve{q}_{EB}$$

$$\dot{a}_s = a_s \cdot M_a \cdot \omega_a$$

Here, it is shown how the driving noise or process noise is introduced to the system equations for them to preserve the invariance. By the same form, the observation noise is modelled in the output equations in a similar way:

$$\vec{y}_V = (\vec{v})^W_{W,meas} + N_{(\vec{v})^W_W} \cdot v_{(\vec{v})^W_W}$$

$$\vec{y}_B = \breve{q}_{EB}{}^{-1} * \left(\begin{pmatrix} 0 \\ (\vec{B})_{W,nom} \end{pmatrix} + \begin{pmatrix} 0 \\ N_{(\vec{B})_W} \cdot v_{(\vec{B})_W} \end{pmatrix} \right) * \breve{q}_{EB} \quad \textbf{(5-13)}$$

The system and output equations are defined in the same way as in the observer case shown in (4-3) and (4-4), except of the difference that Gaussian noise is introduced to define the system in a stochastic model as the KF equations state.

Applying the same process as before but for the equation in (5-4), or which is the same for (5-11), the filter estimated variable equations look like:

$$\dot{\tilde{\breve{q}}}_{EB} = \frac{1}{2}\, \tilde{\breve{q}}_{EB} * \left(\begin{pmatrix} 0 \\ (\vec{\omega}^{WB})_{B,input} \end{pmatrix} - \begin{pmatrix} 0 \\ \widetilde{\vec{b}_g} \end{pmatrix} \right) + \begin{pmatrix} 0 \\ K_{\breve{q}_{EB}} \cdot \vec{E} \end{pmatrix} * \tilde{\breve{q}}_{EB} + \\ + \lambda \left(1 - \left\| \tilde{\breve{q}}_{EB} \right\|^2 \right) * \tilde{\breve{q}}_{EB}$$

$$\left(\dot{\tilde{\vec{v}}} \right)^W_W = (\vec{g})_W + \frac{1}{\tilde{a}_s} \tilde{\breve{q}}_{EB} * \begin{pmatrix} 0 \\ (\vec{f})_{B,input} \end{pmatrix} * \tilde{\breve{q}}_{EB}{}^{-1} + \left(K_{(\vec{v})^W_W} \cdot \vec{E} \right) \quad \textbf{(5-14)}$$

$$\dot{\widetilde{\vec{b}_g}} = \tilde{\breve{q}}_{EB}{}^{-1} * \begin{pmatrix} 0 \\ K_{b_g} \cdot \vec{E} \end{pmatrix} * \tilde{\breve{q}}_{EB}$$

$$\dot{\tilde{a}}_s = \tilde{a}_s \cdot K_a \cdot \vec{E}$$

Which are similar equations as in the observer case but with different gain definitions. The invariant output error outlines:

$$\vec{E} = \begin{pmatrix} \tilde{\vec{y}}_V - \vec{y}_V \\ (\vec{B})_{W,nom} - \tilde{\breve{q}}_{EB} * \begin{pmatrix} 0 \\ \vec{y}_B \end{pmatrix} * \tilde{\breve{q}}_{EB}^{-1} \end{pmatrix}$$

$$= \begin{pmatrix} (\tilde{\vec{v}})_W^W - (\vec{v})_{W,meas}^W \\ (\vec{B})_{W,nom} - \tilde{\breve{q}}_{EB} * \breve{q}_{EB}^{-1} * \begin{pmatrix} 0 \\ (\vec{B})_{W,nom} \end{pmatrix} * \breve{q}_{EB} * \tilde{\breve{q}}_{EB}^{-1} \end{pmatrix} \quad \textbf{(5-15)}$$

This is exactly the same error definition as in the observer case stated in (4-5). The definition of the accelerometers and gyroscopes sensor models is the same as in (4-1) and (4-2).

It is also important to notice that the specific acceleration $\vec{I}_a$ and rotational rate $\vec{I}_w$ vectors are the same as in (4-6). That is the reason why in [3] the RIEKF is considered the same as an observer because it is designed in the same way, but its principle to estimate is based on stochastic theory, which is the main difference with the observer.

5.3 Selection of the tuneable parameters and results

The RIEKF works differently from the symmetry-preserving observer because the gains are not a tuneable parameter; instead they are parameters that change constantly due to the KF algorithm. The RIEKF is based on the principle of stochastic theory, which means that the tuneable parameters are the covariance matrices and all the other stochastic parameters.

The favourable point is that the RIEKF is designed with the same symmetry-preserving theory like the observer, which means the resulting estimated variables can be easily compared. Moreover, it is not straightforward to compare the tuning of the parameters because they have different meanings.

The previous section 4.3 discusses in detail each one of the gain parameters of the symmetry-preserving observer and their corresponding meanings. Hence, in this section we will do the same but developing the meaning of the tuneable variables in this case.

First of all, it is interesting to describe each one of the tuneable variables on the following table:

Variable	Description
$\boldsymbol{P_0}$	Initial state covariance matrix, used to overcome the initial error values. Normally the terms of the covariance matrix must be bigger than the corresponding initial errors to overcome them and converge to the correct values.
$\boldsymbol{M}$	Input covariance matrix or process/driving noise matrix, used for the filter to pay more attention to the estimations. If this matrix is increased, more noise will be added on the

	system but the estimated values will be closer to the real ones. This matrix represents the rate of change of the variables to estimate.
$\boldsymbol{N}$	Measurement covariance matrix or measurement noise matrix. It represents the error introduced with the measurements. With this parameter, the influence of the measurements can be tuned.

Table 5-2: Physical meaning of the stochastic parameters.

Tuning these parameters, it can be achieved the same effect as tuning the gains for the observer. Even though, the stochastic variables are not as straightforward as the gain parameters to tune. The advantage is that each row of the matrices represents one of variable to estimate and makes easier to know which of the matrix elements to tune.

The initial values of the variables to estimate are the same as for the observer case and they are displayed in Table 4-6. So the important values to define here are the initial covariance matrices described above.

Matrix	Initial values
$\boldsymbol{P_0}$	$Diag\left(rad(25)^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 10^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad rad(3)^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 0.5^2\right)$
$\boldsymbol{M}$	$Diag\left(0.5 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 0.0524 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 0.01 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 0.1\right)$
$\boldsymbol{N}$	$Diag\left(1^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 0.2^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T\right)$

Table 5-3: Values of the covariance matrices.

In this case, the matrices $P_0 \; and \; M$ are of dimension 10x10. The purpose of selecting these certain values is arbitrary. In our case, we found that the estimations for trajectory 2, explained on the previous chapter, perform better with these initial coefficients. To tune properly the RIEKF, it is important to take into account Table 5-2 and change the values according to what are your preferences at the time of the simulation.

In this case it is only contemplated trajectory 2 because the behaviour of the estimations does not change that much from one to another trajectory. That is one advantage compared with the observer, the RIEKF is better for general cases because of the adaptive gain parameters. In some trajectories it is even better than the specific gain configuration of the observer. A closer examination will be performed now with the simulations for the initial data displayed above:

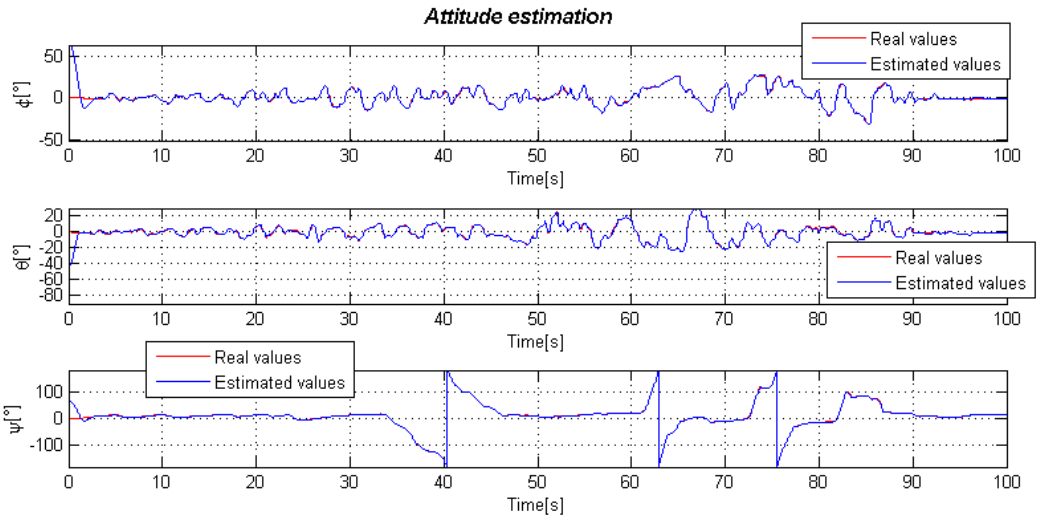

Figure 5-1: Attitude estimation with the RIEKF for trajectory 2.

As it can be seen, the initial error for the estimated values is reasonably big but the algorithm is capable of converging really fast, in about 2 or 3 seconds, without any necessity of initial alignment. That is one of the breakthroughs of this specific algorithm, the symmetries which are used to design it make the algorithm to accept more initial error for smaller values of P_0. This property of increased efficiency and faster convergence with a greater accuracy for the throughout estimation process are some other advantages of this interesting algorithm. Compared with Figure 4-6 the convergence response is similar but the overall accuracy is really better in this case.

The attitude estimation if often the most crucial part of these algorithms that use the attitude for the estimation of the other variables. This is the cause, why revising this estimation should be the first step for every engineer in this field of study.

To take a look of the improvement of accuracy, it is appropriate to observe the error in the attitude estimation. Firstly, the same plot with the same axis adjustment but for the RIEKF algorithm is released:

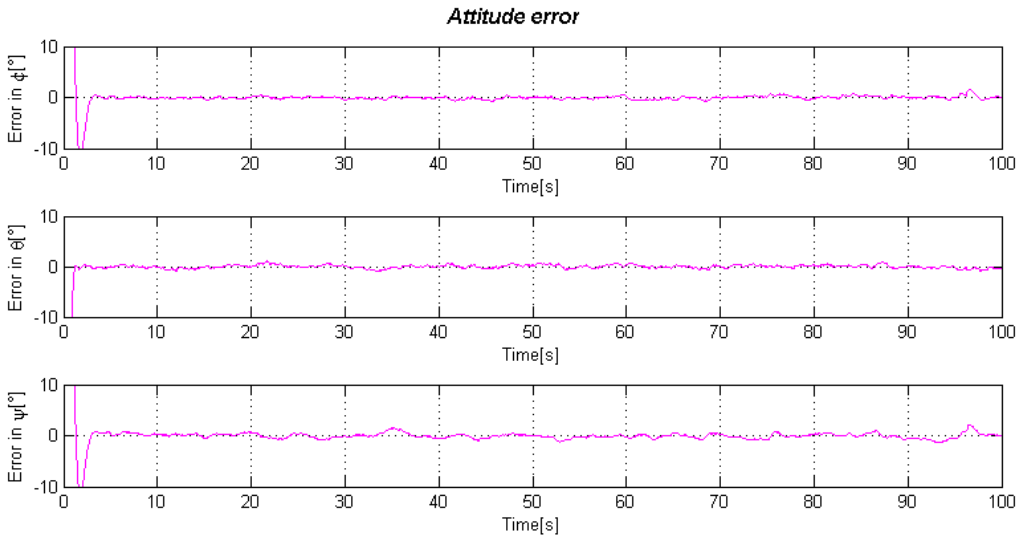

Figure 5-2: Attitude error on the estimation for trajectory 2.

It is straightforward to see that the error is smaller compared with Figure 4-10. An adjustment for the y-axis has been performed on the following plot to display the exact difference of accuracy:

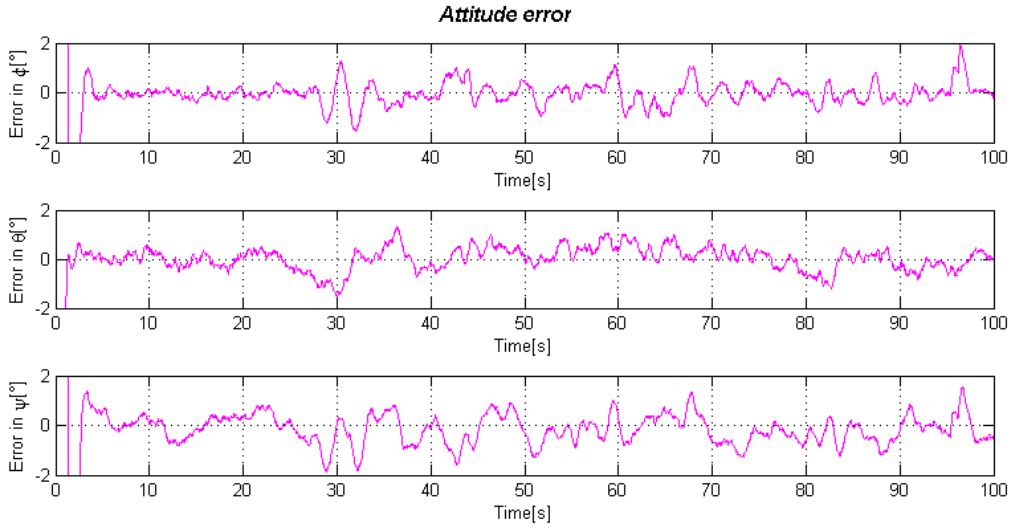

Figure 5-3: Adjustment on the y-axis for Figure 5-2.

Obviously, this algorithm needs more computational time to extract the same result and that is one of the clear drawbacks, but the comparison between algorithms will be discussed in chapter 7. That was only a closer examination for the reader to start noticing some of the differences.

Proceeding with the results for the RIEKF, it is also interesting to see how the estimation of the other states is obtained. The velocity is normally another important state to estimate in most of the cases:

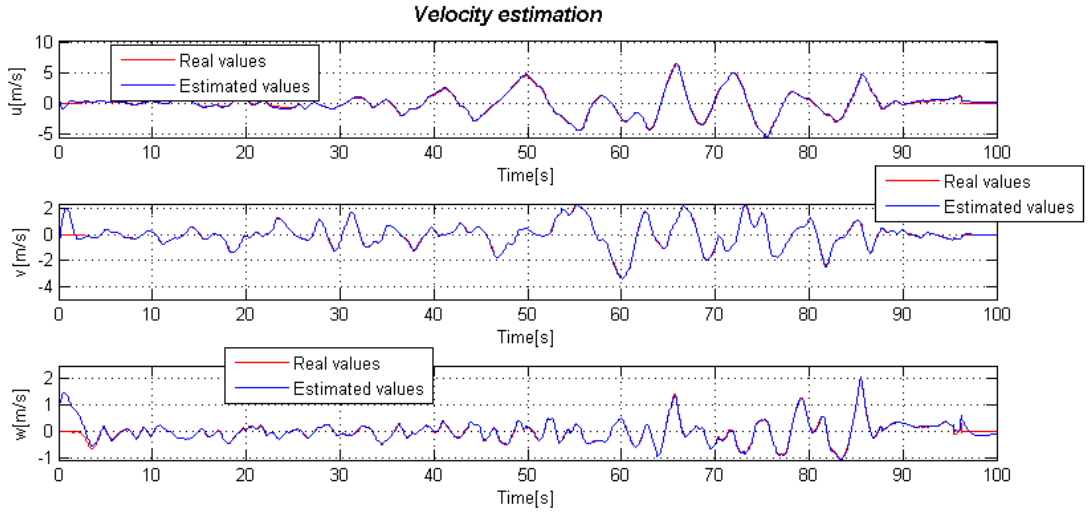

Figure 5-4: Velocity estimation with the RIEKF for trajectory 2.

An important characteristic that is solved in this case is the time of convergence for the z-axis, which in Figure 4-7 needs about 20 seconds to be corrected for the algorithm. The accuracy is also enhanced.

Moreover, the gyro biases estimation is really improved. Taking a fast overview, it can be seen that the estimation is smoother, without sudden changes, and closer to the real values for each one of the axis.

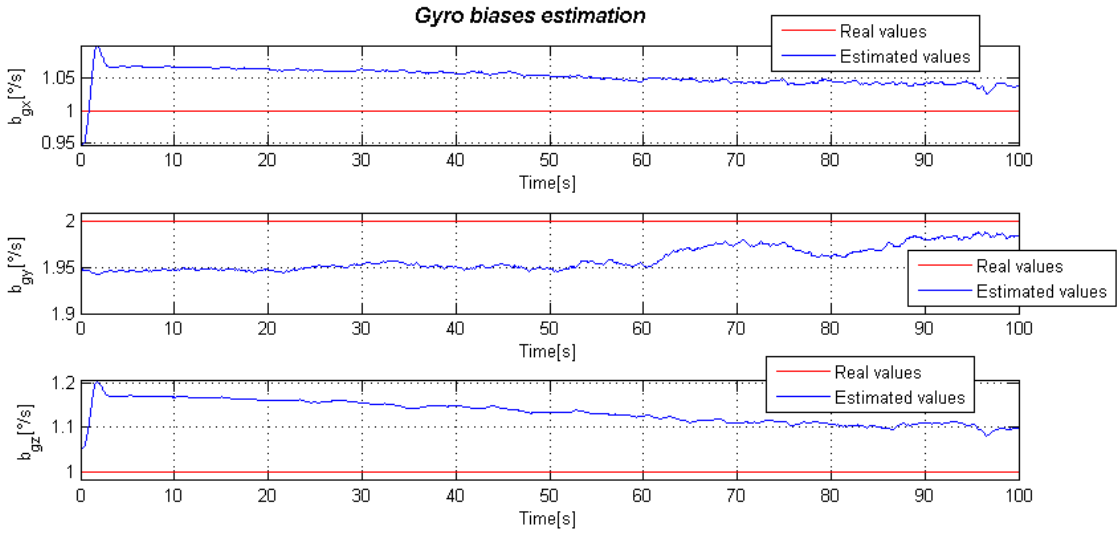

Figure 5-5: Gyro biases estimation with the RIEKF for trajectory 2.

Regarding the accelerometer scaling factor estimation two abrupt deviations occur at the beginning and at the end of the time period but the overall behaviour is reasonable for a correct estimation.

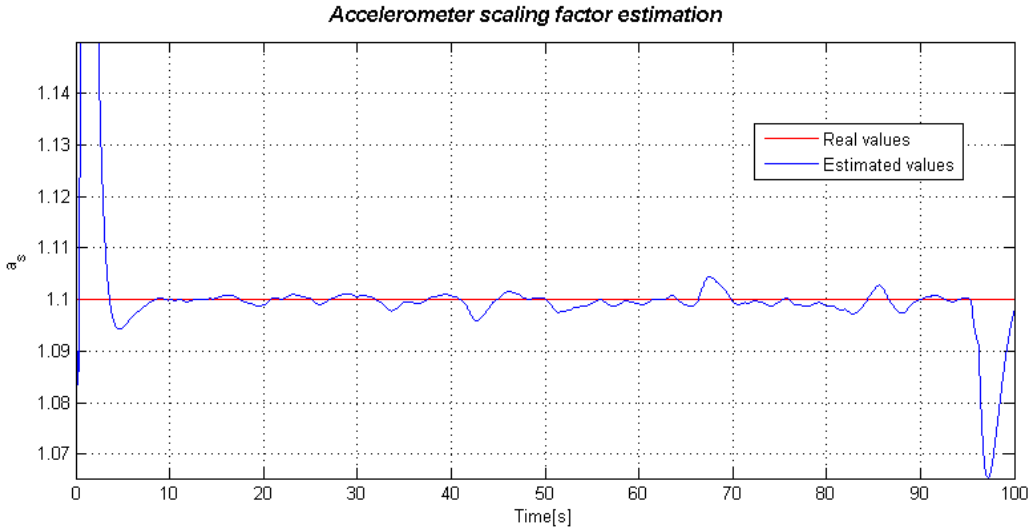

Figure 5-6: Accelerometer scaling factor estimation with the RIEKF for trajectory 2.

From a general point of view, the RIEKF is closer to the symmetry-preserving observer than to the MEKF. The RIEKF and the MEKF are different because of the different implementation, the former is computed in time-continuous mode and the latter is implemented in discrete-time mode. The equations that govern Kalman Filter algorithms are slightly different in such cases. A good example of this is shown in [25], where both implementations are defined, the respective equations stated and the different characteristics discussed.

6 Multiplicative Extended Kalman Filter (MEKF)

To finalize with the last of the algorithms, chapter 6 defines the MEKF without invariance properties, its advantages and specific characteristics. As it is explained in the previous chapter, the specific nomenclature of this KF will be stated. The explanation compared with the normal system equations stated in section 3.1.2 will be done. Finally, the breakthroughs of this algorithm will be concluded and the behaviour of the estimations evaluated in detail.

First of all, the equations of the MEKF will be stated and then the implementation of the system to these equations will be shown. This KF is implemented in a discrete-time way and the corresponding equations are discretized for the algorithm to be more efficient.

6.1 Inertial sensor models

The inertial sensor models for the MEKF are defined in a different way compared with the ones of the observer and RIEKF. They are extracted from [2]. It is important to notice that the sensor models considered in here are a little bit more complex but in the algorithm estimations it is used the most simplified way to bring the models closer to make a meaningful comparison between them. The sensor model of the accelerometer displayed in (4-1) contemplates only a scaling factor and no biases, but for this case the model is stated:

$$\left(\vec{f}\right)_{B,input} = \left(\vec{f}\right)_{B,true\ input} + \vec{b}_a + \begin{pmatrix} M_{a_{xx}} & 0 & 0 \\ 0 & M_{a_{yy}} & 0 \\ 0 & 0 & M_{a_{zz}} \end{pmatrix} \cdot \left(\vec{f}\right)_{B,true\ input} + \vec{v}_a \tag{6-1}$$

It takes into account biases, a scale factor matrix and an input noise vector (with a white mean 0 noise distribution, which is the most realistic for this case). The scale factor cannot be directly compared because of the big difference in the definition in the previous accelerometer sensor model.

The gyroscope sensor model seems to be more similar as the one taken into account before. The main difference is that the gyro biases are added not subtracted as in the previous case. Also it is important to notice that an input noise vector is considered. In this case both sensor models are equally defined:

$$\left(\vec{\omega}^{WB}\right)_{B,input} = \left(\vec{\omega}^{WB}\right)_{B,true\ input} + \vec{b}_g + \begin{pmatrix} M_{g_{xx}} & 0 & 0 \\ 0 & M_{g_{yy}} & 0 \\ 0 & 0 & M_{g_{zz}} \end{pmatrix} \cdot \left(\vec{\omega}^{WB}\right)_{B,true\ input} + \vec{v}_g \tag{6-2}$$

In the usage, the biases and the scale factor error are considered zero for the models to be closer to the ones of the previous INAs.

The only part that can be hardly compared is the estimation of the gyro biases considered in both gyroscope models.

6.2 General equations and nomenclature

It is directly stated the discrete-time KF system definition for the MEKF, as it is the utilized one (extracted from [25]):

$$\begin{aligned} \vec{x}_{k+1} &= \Phi_k \cdot \vec{x}_k + \Gamma_k \cdot \vec{u}_k + G_k \cdot \vec{\omega}_k \\ \vec{y}_k &= H_k \cdot \vec{x}_k + \vec{v}_k \end{aligned} \quad \textbf{(6-3)}$$

The normally distributed random variables with mean 0 and covariance matrix will be defined in this work as:

$$\begin{aligned} \vec{\omega}_k &\sim (0, Q_k) \\ \vec{v}_k &\sim (0, R_k) \end{aligned} \quad \textbf{(6-4)}$$

The matrices $Q_k \; and \; R_k$ respectively represent the manner how the system process noise $\vec{\omega}_k$ and the output measurement noise $\vec{v}_k$ are introduced into the problem. In case, these variables are diagonal matrices, it means that the variables are Gaussian or white noise variables with mean equal to 0, which is the case that applies for the system defined to estimate.

It is important to identify the specific nomenclature of this discrete-time KF, ad hoc the following table displays the new introduced elements:

Variables	Description
$\boldsymbol{P^- \; and \; \vec{x}^-}$	Predicted variables are represented with a superscripted minus sign.
$\boldsymbol{P^+ \; and \; \vec{x}^+}$	Corrected variables are represented with a superscripted plus sign.
$\boldsymbol{P_k \; and \; \vec{x}_k}$	Variables in the time step k.
$\boldsymbol{P^T}$	Transpose of matrix P.
$\boldsymbol{\delta\vec{x}}$	Vector of the computed errors of the state vector $\vec{x}$.
$\boldsymbol{\tilde{\vec{x}}}$	Vector of estimated values of the state vector $\vec{x}$.

Table 6-1: MEKF specific nomenclature.

This MEKF uses two steps to enhance the estimations of the considered states.

Prediction:

In the prediction step, both the state vector and the covariance matrix are predicted using the value of the state vector $\vec{x}_k^+$, the input vector $\vec{u}_k$ and the covariance matrix P_k^+ in the previous time step k.

$$\vec{x}_{k+1}^{-} = \Phi_k \cdot \vec{x}_k^{+} + \Gamma_k \cdot \vec{u}_k$$
$$P_{k+1}^{-} = \Phi_k \cdot P_k^{+} \cdot {\Phi_k}^T + \mathrm{G}_k \cdot Q_k \cdot {\mathrm{G}_k}^T \quad \textbf{(6-5)}$$

Correction:

This second step utilizes the predicted values obtained previously to finally extract the Kalman gains, which are used to compute the corrected values of the state vector and the covariance matrix.

$$K_k = P_{k+1}^{-} \cdot {\mathrm{H}_k}^T \cdot \left(\mathrm{H}_k \cdot P_{k+1}^{-} \cdot {\mathrm{H}_k}^T + R_k\right)^{-1}$$
$$\vec{x}_{k+1}^{+} = \vec{x}_{k+1}^{-} + K_k \cdot (\vec{y}_k - \mathrm{H}_k \cdot \vec{x}_{k+1}^{-}) \quad \textbf{(6-6)}$$
$$P_{k+1}^{+} = (I - K_k \cdot \mathrm{H}_k) \cdot P_{k+1}^{-}$$

As it is seen in Figure 3-2, the algorithm of the KF works separately of the propagation of the system equations and is applied afterwards to correct the errors of this propagation thank to the corrected state vector extracted from the filter. This structure allows executing the different parts of the diagram at different rates, to save some computational time, for example, or to force the algorithm to run faster. A simpler and more clarifying diagram is extracted from [5] and modified corresponding to this English work:

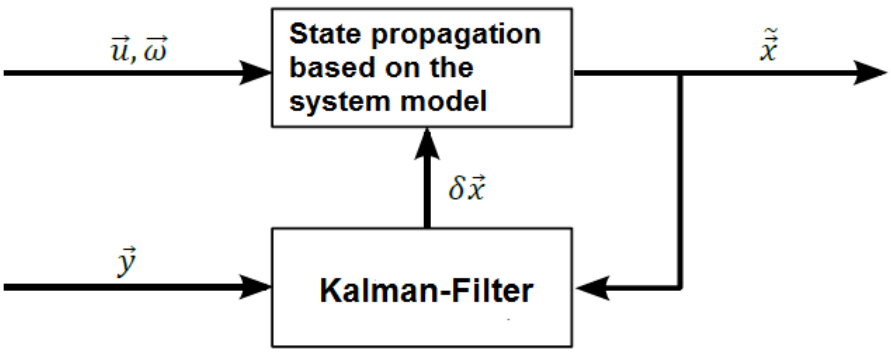

Figure 6-1: Closed-loop general error State-space KF

As shown in (6-3), the non-linear system model used is defined with the following inputs and output equations:

$$\vec{x}_{k+1} = \vec{f}(\vec{x}_k, \vec{u}_k, \vec{\omega}_k)$$
$$\vec{y}_k = \vec{h}_k(\vec{x}_k) + \vec{v}_k \quad \textbf{(6-7)}$$

The real value of the state vector is defined as it follows:

$$\vec{x} = \tilde{\vec{x}} + \delta\vec{x} \quad \textbf{(6-8)}$$

In our case, it has to be stated that the same rate is used in both blocks of the diagram. The propagation of the model and the correction of the states performed by the KF algorithm use the same frequency, which means for every time step there is a estimation for each propagation time-step.

To completely clarify all the variables used in the KF equations, it is important to define the matrices Φ and G:

$$\Phi = \left.\frac{\partial\left(\delta\dot{\vec{x}}\right)}{\partial(\delta\vec{x})}\right|_{\vec{x}=\tilde{\vec{x}}}$$
$$\mathrm{G} = \left.\frac{\partial\left(\delta\dot{\vec{x}}\right)}{\partial(\vec{\omega})}\right|_{\vec{x}=\tilde{\vec{x}}} \quad \textbf{(6-9)}$$

As it can be seen, these matrices are the Jacobi matrices of the error state vector differentiated with respect to itself Φ and the same vector differentiated with respect to the process noise vector $\vec{\omega}$. The measured values of the output equation have to be processed, because it is not the absolute value the one which is used for the estimation. The difference between the expected and the measurement values with the non-linear measurement equation and the estimated system state is calculated and used for this purpose:

$$\delta\vec{y} = \tilde{\vec{y}} - \vec{h}\left(\tilde{\vec{x}}\right) \quad \textbf{(6-10)}$$

Finally, the measurement matrix for the correction of the steps of the filter is the linearization of the measurement equation:

$$\mathrm{H} = \left.\frac{\partial\vec{h}(\vec{x})}{\partial\vec{x}}\right|_{\vec{x}=\tilde{\vec{x}}} \quad \textbf{(6-11)}$$

The computation of the Kalman gain equation shown above in (6-6) performs and inversion, which can be computationally problematic in embedded real-time systems for the big number of calculations that can be performed in only one time-step. For solve this situation, the correction step is calculated as a series of individual measurements that are consequently computed. This brings the matrix inversion to be calculated as a scalar division, which saves a lot of computation time.

In comparison, both of the filters use a similar pattern but each of them has some advantages and some drawbacks. For instance, the MEKF has a smaller range of convergence than the RIEKF with similar values. That means that with a big initial error the covariance matrix for the state to estimate has to be bigger than in the case of the RIEKF. This is because of the convergence properties of each one of the KFs.

In a practical application, this can be solved with an initial alignment to reduce the initial errors and then the MEKF works in a correct way, without diverging. The main advantage of the RIEKF is this bigger convergence property that allows us to estimate the states without the initial alignment for a bigger set of trajectories and bigger values of initial errors.

6.3 Selected MEKF approach equations

As it is outlined in 2.2, an aided AHRS is examined. Both in chapters 4 and 5 and also in this chapter, the system estimates the attitude, velocity and gyro biases. The main difference is

that in this case, instead of estimating the scaling factor of the accelerometers sensor model, it also estimates the accelerometer biases.

The estimated state vector $\tilde{\vec{x}}$ and the error state vector $\delta\vec{x}$, in this case, are defined:

$$\tilde{\vec{x}} = \begin{pmatrix} \tilde{\vec{q}}_{EB} \\ (\tilde{\vec{v}}^R)_W^W \\ \tilde{\vec{b}}_a \\ \tilde{\vec{b}}_g \end{pmatrix}, \qquad \delta\vec{x} = \begin{pmatrix} \vec{\psi}_{\tilde{W}W} \\ (\delta\vec{v}^R)_W^W \\ \delta\vec{b}_a \\ \delta\vec{b}_g \end{pmatrix} \tag{6-12}$$

To bring closer each one of the INAs, the accelerometer biases will be removed (set equal to 0) to make easier to evaluate them.

The data flow diagram of the selected MEKF is displayed on the following figure substituting the general vectors of the Figure 6-1, shown previously.

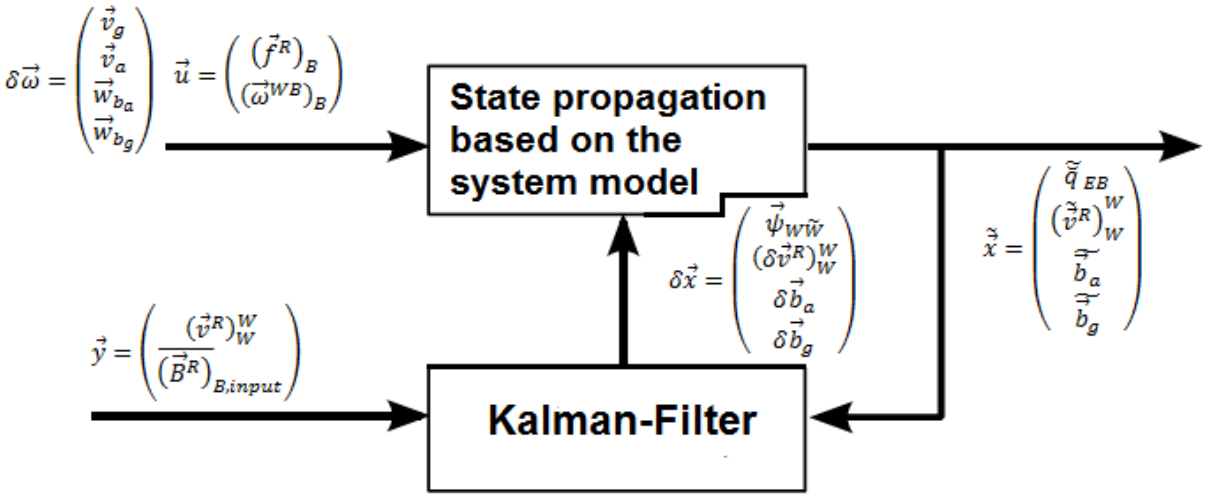

Figure 6-2: Flux diagram of the selected MEKF

6.3.1 Development of the system equations

In this subsection the development of the equations will be outlined and the equations used in the MEKF will be identified.

In this case, the system model uses the rotation matrix R_{WB} inside the filter part to get the error, but at the same time uses the quaternion form for correct this error of the final estimated variable and to avoid the singularities of the Euler angles. That is the reason why in Figure 6-2 the error state vector contains a three element vector for the attitude and the state vector represents the attitude with the quaternion form.

6.3.1.1 System equations

The following are the equations of the system, which are going to be treated and developed to obtain the corresponding form for the MEKF.

The attitude equation will be represented in the form of a rotation matrix and also in the quaternion form:

$$\dot{R}_{WB} = R_{WB} \cdot (\Omega^{WB})_B$$

$$\dot{\vec{q}}_{WB} = \frac{1}{2} \cdot \vec{q}_{WB} * \begin{pmatrix} 0 \\ (\vec{\omega}^{WB})_B \end{pmatrix} = \frac{1}{2} \cdot \begin{pmatrix} q_{WB,0} & -q_{WB,1} & -q_{WB,2} & -q_{WB,3} \\ q_{WB,1} & q_{WB,0} & -q_{WB,3} & q_{WB,2} \\ q_{WB,2} & q_{WB,3} & q_{WB,0} & -q_{WB,1} \\ q_{WB,3} & -q_{WB,2} & q_{WB,1} & q_{WB,0} \end{pmatrix} \cdot \begin{pmatrix} 0 \\ p \\ q \\ r \end{pmatrix} \quad \textbf{(6-13)}$$

The second system equation is the velocity equation:

$$\left(\dot{\vec{v}}^R\right)_W^W = R_{WB} \cdot \left(\vec{f}\right)_B + (\vec{g})_W \quad \textbf{(6-14)}$$

To finalize we have the two bias equations, for the accelerometer biases and for the gyroscopes biases, respectively:

$$\begin{aligned} \dot{\vec{b}}_a &= \vec{w}_{b_a}, \qquad \vec{w}_{b_a} \sim N(0, Q_{b_a}) \\ \dot{\vec{b}}_g &= \vec{w}_{b_g}, \qquad \vec{w}_{b_g} \sim N\left(0, Q_{b_g}\right) \end{aligned} \quad \textbf{(6-15)}$$

As it can be seen, the ideal equation would be $\dot{\vec{b}}_a = 0$, but the noise model of the MEKF algorithm is modeled with a random white noise variable.

6.3.1.2 Error equations

These equations have to be assessed with the introduction of the error variables on the real system. The only equation which is defined differently due to its special considerations is the attitude error:

$$R_{WB} = R_{W\widetilde{W}} \cdot R_{\widetilde{W}B} \quad \textbf{(6-16)}$$

It is considered as an error rotation matrix between the real w-Frame and the “estimated w-Frame”.

Proceeding with the definitions, the error of the other variables is defined as an addition of the error term to the estimated value to obtain the real value:

$$\begin{aligned} (\vec{v})_W^W &= \left(\tilde{\vec{v}}\right)_W^W + (\delta\vec{v})_W^W \\ \vec{b}_a &= \tilde{\vec{b}}_a + \delta\vec{b}_a \\ \vec{b}_g &= \tilde{\vec{b}}_g + \delta\vec{b}_g \end{aligned} \quad \textbf{(6-17)}$$

It is also of paramount importance to outline the error on the accelerometers, which consists of white noise and the biases together:

$$\left(\vec{f}\right)_B = \left(\tilde{\vec{f}}\right)_B + \left(\delta\vec{f}\right)_B \quad \textbf{(6-18)}$$

And with the In-flight calibration, the following applies:

$$\left(\delta\vec{f}\right)_B = \delta\vec{b}_a + \vec{v}_a, \qquad \vec{v}_a \sim N\left(0, Q_{v_a}\right) \tag{6-19}$$

Analogously, the same model defines the gyroscopes error model:

$$(\vec{\omega}^{WB})_B = \left(\tilde{\vec{\omega}}^{WB}\right)_B + (\delta\vec{\omega}^{WB})_B \tag{6-20}$$

and

$$(\delta\vec{\omega}^{WB})_B = \delta\vec{b}_g + \vec{v}_g, \qquad \vec{v}_g \sim N\left(0, Q_{v_g}\right) \tag{6-21}$$

6.3.1.3 Non-linear error differential equations:

To obtain the equations for the MEKF, it has to be combined both the error and the system equations.

The attitude error differential equation is used with the rotation matrix form and developed:

$$\dot{R}_{WB} = R_{WB} \cdot (\Omega^{WB})_B \tag{6-22}$$

And substituting the previous error and differential equations:

$$\frac{\partial}{\partial t}(R_{W\tilde{W}} \cdot R_{\tilde{W}B}) = R_{W\tilde{W}} \cdot R_{\tilde{W}B} \cdot \left(\left(\tilde{\Omega}^{WB}\right)_B + (\delta\Omega^{WB})_B\right)$$
$$\dot{R}_{W\tilde{W}} \cdot R_{\tilde{W}B} + R_{W\tilde{W}} \cdot \dot{R}_{\tilde{W}B} = R_{W\tilde{W}} \cdot R_{\tilde{W}B} \cdot \left(\left(\tilde{\Omega}^{WB}\right)_B + (\delta\Omega^{WB})_B\right) \tag{6-23}$$

Following the same development properties, it can be concluded that:

$$\dot{R}_{\tilde{W}B} = R_{\tilde{W}B} \cdot \left(\tilde{\Omega}^{WB}\right)_B \tag{6-24}$$

Using this new term in the equation above:

$$\dot{R}_{W\tilde{W}} \cdot R_{\tilde{W}B} + R_{W\tilde{W}} \cdot R_{\tilde{W}B} \cdot \left(\tilde{\Omega}^{WB}\right)_B = R_{W\tilde{W}} \cdot R_{\tilde{W}B} \cdot \left(\left(\tilde{\Omega}^{WB}\right)_B + (\delta\Omega^{WB})_B\right)$$
$$\dot{R}_{W\tilde{W}} \cdot R_{\tilde{W}B} = R_{W\tilde{W}} \cdot R_{\tilde{W}B} \cdot (\delta\Omega^{WB})_B \tag{6-25}$$
$$\dot{R}_{W\tilde{W}} = R_{W\tilde{W}} \cdot R_{\tilde{W}B} \cdot (\delta\Omega^{WB})_B \cdot {R_{\tilde{W}B}}^T$$

And with a similar development, it can be found the error differential velocity equation:

$$\left(\dot{\tilde{\vec{v}}}\right)_W^W + \left(\delta\dot{\vec{v}}\right)_W^W = R_{W\tilde{W}} \cdot R_{\tilde{W}B} \cdot \left(\left(\tilde{\vec{f}}\right)_B + \left(\delta\vec{f}\right)_B\right) + (\vec{g})_W \tag{6-26}$$

And after substituting the boundary condition:

$$R_{\tilde{W}B} \cdot \left(\vec{f}\right)_B + (\vec{g})_W + \left(\delta\dot{\vec{v}}\right)_W^W = R_{W\tilde{W}} \cdot R_{\tilde{W}B} \cdot \left(\left(\tilde{\vec{f}}\right)_B + \left(\delta\vec{f}\right)_B\right) + (\vec{g})_W \tag{6-27}$$

Yielding to:

$$\begin{aligned}\left(\delta\dot{\vec{v}}\right)_W^W &= R_{W\tilde{W}} \cdot R_{\tilde{W}B} \cdot \left(\left(\tilde{\vec{f}}\right)_B + \left(\delta\vec{f}\right)_B\right) - R_{\tilde{W}B} \cdot \left(\vec{f}\right)_B \\ \left(\delta\dot{\vec{v}}\right)_W^W &= \left(R_{W\tilde{W}} - I_3\right) \cdot R_{\tilde{W}B} \cdot \left(\tilde{\vec{f}}\right)_B + R_{W\tilde{W}} \cdot R_{\tilde{W}B} \cdot \left(\delta\vec{f}\right)_B\end{aligned} \tag{6-28}$$

Finally, the two last error equations are the sensor accelerometers and gyroscopes biases:

$$\begin{aligned}\frac{\partial}{\partial t}\left(\tilde{\vec{b}}_a + \delta\vec{b}_a\right) &= \vec{w}_{b_a} \\ \frac{\partial}{\partial t}\left(\tilde{\vec{b}}_g + \delta\vec{b}_g\right) &= \vec{w}_{b_g}\end{aligned} \tag{6-29}$$

After a pair of changes these two equations yield to:

$$\begin{aligned}\delta\dot{\vec{b}}_a &= \vec{w}_{b_a}, \qquad \vec{w}_{b_a} \sim N\left(0, Q_{b_a}\right) \\ \delta\dot{\vec{b}}_g &= \vec{w}_{b_g}, \qquad \vec{w}_{b_g} \sim N\left(0, Q_{b_g}\right)\end{aligned} \tag{6-30}$$

6.3.1.4 Linearized error differential equations:

Using the equations previously developed on the section above and assuming some transformations, the equations will be linearized to be used afterwards for the MEKF. The error state vector $\delta\vec{x}$ and the process noise error vector $\delta\vec{\omega}$ are defined:

$$\delta\vec{x} = \begin{pmatrix} \vec{\psi}_{W\tilde{W}} \\ \left(\delta\vec{v}^R\right)_W^W \\ \delta\vec{b}_a \\ \delta\vec{b}_g \end{pmatrix}, \qquad \delta\vec{\omega} = \begin{pmatrix} \vec{v}_g \\ \vec{v}_a \\ \vec{w}_{b_a} \\ \vec{w}_{b_g} \end{pmatrix} \tag{6-31}$$

The biases equations are already linear; hence, they do not need to be linearized. Going on with the other equations, the first to linearize is the attitude equation. It is assumed that the error rotation matrix $R_{W\tilde{W}}$ contains small rotations, which brings us to the following simplification for small angles:

$$R_{W\tilde{W}} = \left(I_3 + \Psi_{W\tilde{W}}\right) = \left(I_3 + \left[\vec{\psi}_{W\tilde{W}} \times\right]\right) \tag{6-32}$$

Using this simplification, it is straightforward to extent the equation to:

$$\dot{\Psi}_{W\tilde{W}} = \left(I_3 + \Psi_{W\tilde{W}}\right) \cdot \left(\delta\Omega^{WB}\right)_{\tilde{W}} = \left(\delta\Omega^{WB}\right)_{\tilde{W}} + \Psi_{W\tilde{W}} \cdot \left(\delta\Omega^{WB}\right)_{\tilde{W}} \tag{6-33}$$

Since higher order differential errors terms are not taken into account, the equations yields to:

$$\dot{\Psi}_{W\tilde{W}} \approx \left(\delta\Omega^{WB}\right)_{\tilde{W}} \tag{6-34}$$

Removing the $veck^{-1}$ – operator of both sides:

$$\dot{\vec{\psi}}_{W\tilde{W}} = (\delta\vec{\omega}^{WB})_{\tilde{W}} = R_{\tilde{W}B} \cdot (\delta\vec{\omega}^{WB})_B \tag{6-35}$$

Finally substituting the corresponding rotational rate equivalent:

$$\dot{\vec{\psi}}_{W\tilde{W}} = (\delta\vec{\omega}^{WB})_{\tilde{W}} = R_{\tilde{W}B} \cdot \left(\delta\vec{b}_g + \vec{v}_g\right) \tag{6-36}$$

Now, the same process with the velocity variable has to be developed starting from equation (6-28) and substituting the assumption (6-32) yields to:

$$\left(\delta\dot{\vec{v}}\right)_W^W = \Psi_{W\tilde{W}} \cdot R_{\tilde{W}B} \cdot \left(\tilde{\vec{f}}\right)_B + (I_3 + \Psi_{W\tilde{W}}) \cdot R_{\tilde{W}B} \cdot \left(\delta\vec{f}\right)_B \tag{6-37}$$

Where the following term can be considered zero, because of a higher order error term:

$$\Psi_{W\tilde{W}} \cdot R_{\tilde{W}B} \cdot \left(\delta\vec{f}\right)_B \approx 0 \tag{6-38}$$

Assumption that brings the previous equation to the result:

$$\left(\delta\dot{\vec{v}}\right)_W^W = \left[\vec{\psi}_{W\tilde{W}} \times\right] \cdot R_{\tilde{W}B} \cdot \left(\tilde{\vec{f}}\right)_B + R_{\tilde{W}B} \cdot \left(\delta\vec{f}\right)_B \tag{6-39}$$

And isolating the state variables leads to:

$$\left(\delta\dot{\vec{v}}\right)_W^W = -\left[\left(R_{\tilde{W}B} \cdot \left(\vec{f}\right)_B\right) \times\right] \cdot \vec{\psi}_{W\tilde{W}} + R_{\tilde{W}B} \cdot \left(\delta\vec{f}\right)_B \tag{6-40}$$

Finally, according to (6-19) the final equation for the velocity estimation is:

$$\left(\delta\dot{\vec{v}}\right)_W^W = -\left[\left(R_{\tilde{W}B} \cdot \left(\vec{f}\right)_B\right) \times\right] \cdot \vec{\psi}_{W\tilde{W}} + R_{\tilde{W}B} \cdot \left(\delta\vec{b}_a + \vec{v}_a\right) \tag{6-41}$$

The linearized error equations used for the MEKF in matrix form, i.e. equivalent to the system equations, are:

$$\begin{pmatrix} \dot{\vec{\psi}}_{\widetilde{W}W} \\ (\delta\dot{\vec{v}}^R)_W^W \\ \delta\dot{\vec{b}}_a \\ \delta\dot{\vec{b}}_g \end{pmatrix} = \begin{pmatrix} 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & R_{\widetilde{W}B} \\ -\left[\left(R_{\widetilde{W}B}\cdot\left(\vec{f}\right)_B\right)\times\right] & 0_{3\times3} & R_{\widetilde{W}B} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \end{pmatrix} \cdot \begin{pmatrix} \vec{\psi}_{\widetilde{W}W} \\ (\delta\vec{v}^R)_W^W \\ \delta\vec{b}_a \\ \delta\vec{b}_g \end{pmatrix} + \begin{pmatrix} R_{\widetilde{W}B} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & R_{\widetilde{W}B} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & I_3 & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & I_3 \end{pmatrix} \cdot \begin{pmatrix} \vec{v}_g \\ \vec{v}_a \\ \vec{w}_{b_a} \\ \vec{w}_{b_g} \end{pmatrix} \quad \textbf{(6-42)}$$

Where matrices $A\ and\ B$ are:

$$A = \begin{pmatrix} 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & R_{\widetilde{W}B} \\ -\left[\left(R_{\widetilde{W}B}\cdot\left(\vec{f}\right)_B\right)\times\right] & 0_{3\times3} & R_{\widetilde{W}B} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \end{pmatrix} \quad \textbf{(6-43)}$$

$$B = \begin{pmatrix} R_{\widetilde{W}B} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & R_{\widetilde{W}B} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & I_3 & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & I_3 \end{pmatrix}$$

6.3.1.5 Discretization

The matrices $A\ and\ B$ are discretized using the following approximation in time:

$$\Phi = I + A\cdot\delta t + A^2\cdot\frac{\delta t^2}{2} \qquad \Gamma = B\cdot\delta t + A\cdot B\cdot\frac{\delta t^2}{2} \quad \textbf{(6-44)}$$

The second order terms can be neglected to reduce the computational time of the algorithm. Adopting these two equations, the system equation in the discretized form is formulated:

$$\delta\vec{x}_{k+1} = \Phi_k\cdot\delta\vec{x}_k + \Gamma_k\cdot\delta\vec{\omega}_k \quad \textbf{(6-45)}$$

From here, the matrices for the MEKF are extracted and used to predict the future states in time.

6.3.2 Development of the measurement equations

The final step is to get the measurement equations sued in the correction step of the MEKF algorithm. It is known that these equations correct the whole state vector improving the predicted values with the data of the measured variables, in this case the velocity and the magnetic field. The output equation is used in two steps, one for each measurement to save time of computation and build a more efficient algorithm. The result is the same but this two-step correction will be stated for the velocity and for the magnetic field.

6.3.2.1 Velocity

The output equation regarding the velocity measurement is subtracted directly, because the estimation and the measured values are at the same frame:

$$\vec{y}_v = (\vec{v}^R)^W_{W,meas} - \left(\tilde{\vec{v}}\right)^W_W \tag{6-46}$$

Then the measurement matrix is selected in a manner to choose only the velocity from the error state vector $\delta\vec{x}$ to be corrected:

$$H_v = (0_{3\times3} \quad I_3 \quad 0_{3\times3} \quad 0_{3\times3}) \tag{6-47}$$

Consequently, the error output equation for the velocity is:

$$\delta\vec{y}_v = \vec{y}_v - H_v \cdot \vec{x} \tag{6-48}$$

Additionally, it is important to define the measurement covariance matrix R_v :

$$R_v = \begin{pmatrix} \sigma_u^2 & 0 & 0 \\ 0 & \sigma_v^2 & 0 \\ 0 & 0 & \sigma_w^2 \end{pmatrix} \tag{6-49}$$

6.3.2.2 Magnetic field

The magnetic field output equation for the correction step is more complex but it only depends on the reference frame of the measurement. In our case, the measurement of the magnetic field is contemplated in the b-Frame, which means the nominal magnetic field has to be rotated from the w-Frame to the b-Frame. The following development shows the previous equations for this second case:

$$\vec{y}_B = \left(\vec{B}\right)_{B,meas} - R_{BW} \cdot \left(\vec{B}\right)_{W,nom} = \left(\vec{B}\right)_{B,meas} - R_{B\tilde{W}} \cdot R_{\tilde{W}W} \cdot \left(\vec{B}\right)_{W,nom} \tag{6-50}$$

Examining the equation (6-32) and knowing that the inverse of a rotation matrix is the same as the transpose, we obtain:

$$R_{W\tilde{W}}{}^T = R_{\tilde{W}W} = \left(I_3 - \left[\vec{\psi}_{W\tilde{W}} \times\right]\right) \tag{6-51}$$

And then substituting into the previous equation:

$$\begin{aligned} \vec{y}_B &= \left(\vec{B}\right)_{B,meas} - R_{B\tilde{W}} \cdot \left(I_3 - \left[\vec{\psi}_{W\tilde{W}} \times\right]\right) \cdot \left(\vec{B}\right)_{W,nom} \\ \vec{y}_B &= \left(\vec{B}\right)_{B,meas} - R_{B\tilde{W}} \cdot \left(\vec{B}\right)_{W,nom} + R_{B\tilde{W}} \cdot \left[\vec{\psi}_{W\tilde{W}} \times\right] \cdot \left(\vec{B}\right)_{W,nom} \\ \vec{y}_B &= \left(\vec{B}\right)_{B,meas} - R_{B\tilde{W}} \cdot \left(\vec{B}\right)_{W,nom} - R_{B\tilde{W}} \cdot \left[\left(\vec{B}\right)_{W,nom} \times\right] \cdot \vec{\psi}_{W\tilde{W}} \end{aligned} \tag{6-52}$$

Taking a closer examination to the vector $\vec{\psi}_{W\tilde{W}}$ and knowing that is defined as:

$$\vec{\psi}_{W\tilde{W}} = \begin{pmatrix} \delta\phi \\ \delta\theta \\ \delta\psi \end{pmatrix} \tag{6-53}$$

The measurement matrix can be calculated from the differential equation of $\vec{y}_B$ with respect to :

$$\frac{\partial \vec{y}_B}{\partial \psi} = \frac{\partial}{\partial \psi}\left(\left(\vec{B}\right)_{B,meas} - R_{B\widetilde{W}} \cdot \left(\vec{B}\right)_{W,nom} - R_{B\widetilde{W}} \cdot \left[\left(\vec{B}\right)_{W,nom} \times\right] \cdot \vec{\psi}_{W\widetilde{W}}\right) \tag{6-54}$$

Where the considered nominal magnetic field is assumed to be $\left(\vec{\boldsymbol{B}}\right)_{\boldsymbol{W,nom}} = (\mathbf{1} \quad \mathbf{0} \quad \mathbf{1})^{\boldsymbol{T}}$, so the last cross product is equal to:

$$\left[\left(\vec{B}\right)_{W,nom} \times\right] \cdot \vec{\psi}_{W\widetilde{W}} = \begin{vmatrix} i & j & k \\ 1 & 0 & 1 \\ \delta\phi & \delta\theta & \delta\psi \end{vmatrix} = \begin{pmatrix} -\delta\theta \\ \delta\phi - \delta\psi \\ \delta\theta \end{pmatrix} \tag{6-55}$$

This brings the previous equation to the result:

$$\frac{\partial \vec{y}_B}{\partial \psi} = \frac{\partial}{\partial \psi}\left(\left(\vec{B}\right)_{B,meas} - R_{B\widetilde{W}} \cdot \left(\vec{B}\right)_{W,nom} - R_{B\widetilde{W}} \cdot \begin{pmatrix} -\delta\theta \\ \delta\phi - \delta\psi \\ \delta\theta \end{pmatrix}\right) = R_{B\widetilde{W}} \cdot \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \tag{6-56}$$

The corresponding measurement matrix is equal to:

$$H_B = \begin{pmatrix} 0_{3\times2} & \dfrac{\partial \vec{y}_B}{\partial \psi} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \end{pmatrix} \tag{6-57}$$

It has the same dimensions as equation (6-47). As it is said before, the measurement matrix could be integrated in one step and use at the correction step to get the same results as we are obtaining with this decoupled option.

Finally, the equations to use inside the MEKF are:

$$\delta\vec{y}_B = \vec{y}_B - H_B \cdot \vec{x} \tag{6-58}$$

Additionally, the measurement covariance matrix for the magnetic field measurement R_B :

$$R_B = \begin{pmatrix} \sigma_{B_x}^2 & 0 & 0 \\ 0 & \sigma_{B_y}^2 & 0 \\ 0 & 0 & \sigma_{B_z}^2 \end{pmatrix} \tag{6-59}$$

6.4 Selection of the tuneable parameters and results

The tuneable parameters in this last type of algorithm are similar to the ones displayed in Table 5-2. A difference to take into account is that the MEKF estimates twelve elements instead of the ten-element estimation of the RIEKF. The algorithm in here is designed to estimate the biases of both accelerometer and gyroscope sensors but it does not pay attention to the scaling factor of the accelerometer. Features like the latter and the fact that

the MEKF is not designed for preserving the invariances of the system which acts on, are some of the points that make a difficult job to compare this third algorithm with the latter ones; comparison, which will be performed on the following chapters.

First of all, the following table will display the possible tuneable parameters to take into account for the simulation step:

Matrix	Initial values
$\boldsymbol{P_0}$	$Diag\left(rad(10)^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 10^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 0.4^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad rad(3)^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T\right)$
$\boldsymbol{Q}$	$Diag\left(rad(3)^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 0.5^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 0.01^2 \cdot \begin{pmatrix}0\\0\\1\end{pmatrix}^T \quad rad(1)^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T\right)$
$\boldsymbol{R_v}$	$1^2 \cdot I_3$
$\boldsymbol{R_B}$	$0.2^2 \cdot I_3$

Table 6-2: Selected tuneable parameters for the simulation.

For the reason above outlined, the matrices $P_0\ and\ Q$ are of dimension 12x12. The description of these variables can be correctly identified with the explanations in Table 5-2. The alteration that has to be applied due to the change of nomenclature is $Q \sim M$ and $R \sim N$, as stated in Table 5-1.

Differently from the case of matrix N for the RIEKF, in this case, the noise measurement matrix $R \sim N$ is decoupled in two 3x3 matrices for each one of the measurements. The purpose of this decoupling is to calculate the correction step of the MEKF in two separate steps for each one of the measurements and, by this way, improve the computational efficiency of the algorithm, which means decrease the required time to get the same result. Thank to this implementation and to the discrete-time mode used, the computing time for this KF algorithm is smaller and then the MEKF is more efficient.

The MEKF uses for this simulation completely different initial values for the estimations that suit exemplifies in an appropriate manner to show the characteristics of this algorithm clearer. The following table displays the initial values for the estimated variables to start the algorithm:

Variable to estimate	Real initial values of the MEKF for Trajectory 2	Estimated initial values of the MEKF for Trajectory 2
$\boldsymbol{\phi}[°]$	0	10
$\boldsymbol{\theta}[°]$	0	-30
$\boldsymbol{\psi}[°]$	0	60
$\boldsymbol{u}\ [\boldsymbol{m/s}]$	0	10

$\boldsymbol{v}\ [\boldsymbol{m/s}]$	0	-10
$\boldsymbol{w}\ [\boldsymbol{m/s}]$	0	5
$\boldsymbol{b_{a_x}}\ [\boldsymbol{m/s^2}]$	-0.25	0
$\boldsymbol{b_{a_y}}\ [\boldsymbol{m/s^2}]$	0.1	0
$\boldsymbol{b_{a_z}}\ [\boldsymbol{m/s^2}]$	0.2	0
$\boldsymbol{b_{g_x}}\ [°/\boldsymbol{s}]$	1	0
$\boldsymbol{b_{g_y}}\ [°/\boldsymbol{s}]$	2	0
$\boldsymbol{b_{g_z}}\ [°/\boldsymbol{s}]$	1	0

Table 6-3: Initial true and estimated state values for the MEKF.

In this case, the biases are initialized far from the real values to see the quantity of needed time to converge due to the fact that the convergence issue of the gyro biases for the symmetry-preserving observer and the RIEKF is the pending estimation to enhance.

The property of accuracy is still present on the MEKF attitude and velocity estimations, the different behavior here is the convergence of the estimation curves. It is easily seen that the initial error is quickly corrected leaving a small scaled difference between the real and the estimated values which is finally corrected in about 20 seconds.

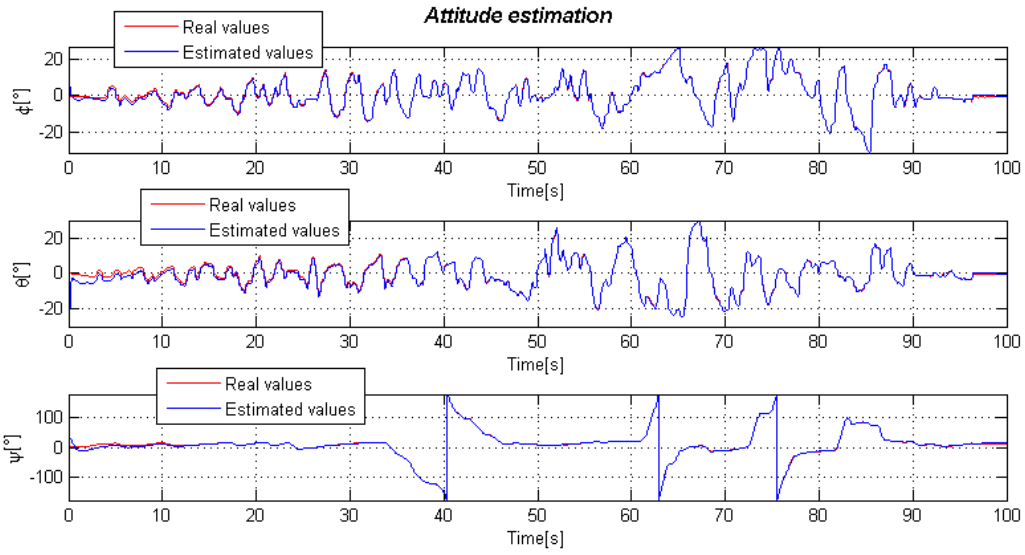

Figure 6-3: Attitude estimation with the MEKF for trajectory 2.

This statement can be checked on the following plot, which is an adjustment of the axis of the plot to focus on the initial correction.

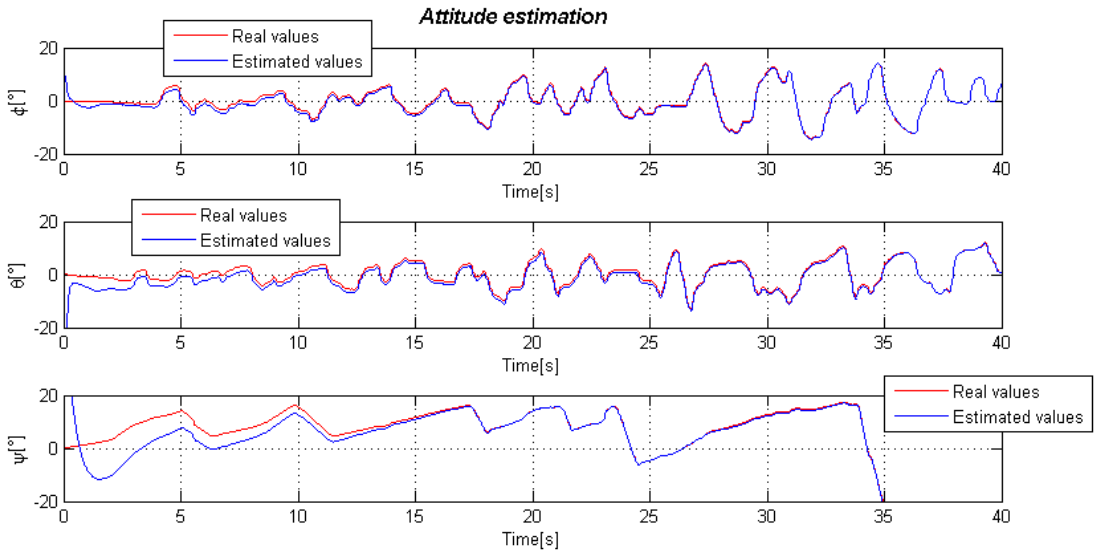

Figure 6-4: Adjustment on the x-axis for Figure 6-3.

The interesting property is the smoothness of this algorithm. An advantage of the MEKF compared with the RIEKF is that the estimations are smoother, i.e. no sudden changes take place on the estimation curves. This can be proved with the error of estimation for the attitude. Compared with Figure 5-3, even the amplified plot is smoother than the non-amplified one for the symmetry-preserving observer (Figure 4-12).

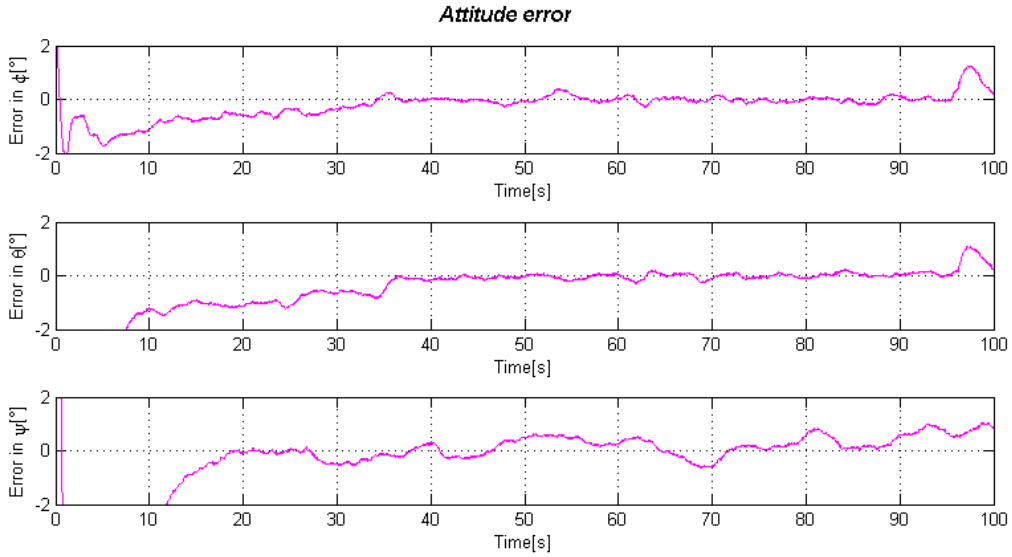

Figure 6-5: Attitude error on the estimation for trajectory 2.

For the velocity estimation, the convergence issue in the first 20 seconds is not relevant.

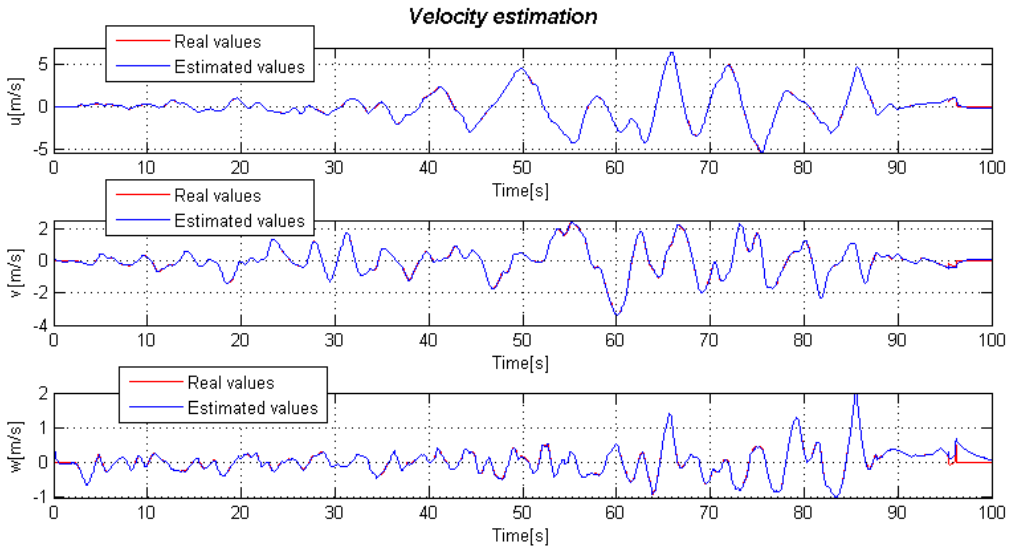

Figure 6-6: Velocity estimation with the MEKF for trajectory 2.

That error at the beginning of the attitude estimation is the one that can be solved with an initial alignment of the aircraft for the sensors to be calibrated. This manoeuvre only takes 5 minutes and can increase considerably the performance of the used INA.

To finalize, it is important to take a closer examination to the bias estimations:

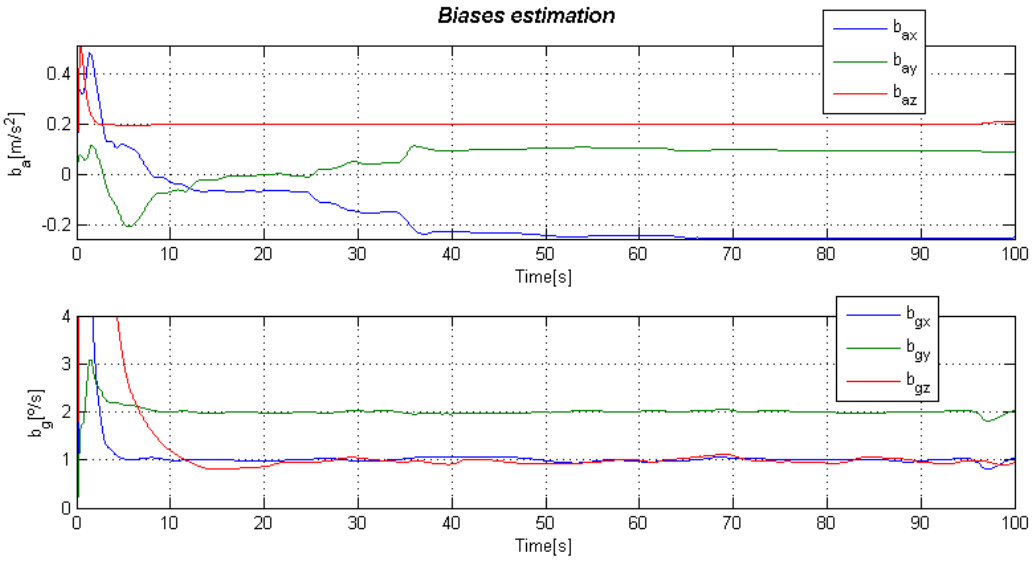

Figure 6-7: Accelerometer and gyros biases estimation with the MEKF for trajectory 2.

Probably it cannot be easily seen in the plots but all the biases estimations are initialized at $0\,m/s^2$ for the accelerometers biases and $0\,°/s$ for the gyroscopes biases. Compared with the previous INAs, the biases are quite quickly estimated. Using such an initial errors and taking into account that the previous cases there were no initial errors for the gyro biases, it can be concluded that the biases estimation in this case is really good.

Moreover, it cannot be directly compared with the latter cases since the definitions of the system and the INAs are very different and lots of other variables influence the problem.

7 INAs Comparison

This chapter will summarize the properties mentioned before for each of the INAs discussed in this work and the comparison between them in different situations, i.e. with different trajectories, will be performed.

As introduced in the section 2.4, there are two cases of study considered for this thesis. The first one is true data, which means perfect data without any kind of noise. In this case the sensor models explained are used in each perfect trajectory to test the algorithms with the complete knowledge of the type of perturbations introduced. The sensor models are very similar to the real case but the main difference is that the situation is a more under-control experience, like in a laboratory, where all the variables are considered.

The second case is a real test where real-time data is extracted from the UAV sensors for the algorithms to be tested and to see how they perform in a real case, where unconsidered variables can influence the measurements.

The comparison of both cases will be stated and then used to see how the algorithms responses develop during the simulation.

7.1 True data comparison

A more in depth comparison of the algorithms with perfect data will be displayed in this section. As a purpose, it is desirable to compare the algorithms in the most similar situation. Though, the problem is that the symmetry-preserving observer and the RIEKF are designed in a completely different manner than the MEKF and the exact comparison will be difficult.

Although the comparability is not straightforward, we will try to reach the most suitable case for the algorithms to be much easier contrasted.

Common estimated variables of each one of the INAs will be plotted and discussed. These variables are the attitude, the velocity and the gyro biases. The scaling factor is not estimated by the MEKF and the accelerometer biases are not considered in the sensor error model of the symmetry-preserving observer and the RIEKF. The tuneable parameters of each algorithm will be displayed in the following tables, but the gain configuration for the observer is the one described in Table 4-7.

Matrix	Initial values
$\boldsymbol{P_0}$	$Diag\left(\begin{pmatrix} rad(30)^2 \\ rad(30)^2 \\ rad(100)^2 \end{pmatrix}^T \quad 1^2 \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}^T \quad rad(3)^2 \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}^T \quad 1^2\right)$
$\boldsymbol{M}$	$Diag\left(0.5 \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}^T \quad 0.01 \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}^T \quad 0.001 \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}^T \quad 0.1\right)$

$\boldsymbol{N}$	$Diag\left(0.1 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 0.1 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T\right)$

Table 7-1: Tuneable parameters configuration for the RIEKF.

And the values for the MEKF are:

Matrix	Initial values
$\boldsymbol{P_0}$	$Diag\left(\begin{pmatrix}rad(30)^2\\rad(30)^2\\rad(100)^2\end{pmatrix}^T \quad 1^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 0.4^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad rad(3)^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T\right)$
$\boldsymbol{Q}$	$Diag\left(rad(3)^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 0.5^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 0.01^2 \cdot \begin{pmatrix}0\\0\\1\end{pmatrix}^T \quad rad(1)^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T\right)$
$\boldsymbol{R_v}$	$0.1 \cdot I_3$
$\boldsymbol{R_B}$	$0.1 \cdot I_3$

Table 7-2: Tuneable parameters configuration for the MEKF.

Regarding the initial values for the estimations are the ones shown after that:

Variable to estimate	Real initial values	Estimated initial values
$\boldsymbol{\phi}[°]$	0	20
$\boldsymbol{\theta}[°]$	0	30
$\boldsymbol{\psi}[°]$	0	60
$\boldsymbol{u}\ [\boldsymbol{m/s}]$	0	0.2
$\boldsymbol{v}\ [\boldsymbol{m/s}]$	0	0.4
$\boldsymbol{w}\ [\boldsymbol{m/s}]$	0	0.1
$\boldsymbol{b_{g_x}}\ [°/\boldsymbol{s}]$	-1	-1
$\boldsymbol{b_{g_y}}\ [°/\boldsymbol{s}]$	-2	-2
$\boldsymbol{b_{g_z}}\ [°/\boldsymbol{s}]$	2	2

Table 7-3: Initial values for the 3 algorithms - true data comparison.

Firstly, it is interesting to evaluate the attitude data displayed in the following figure:

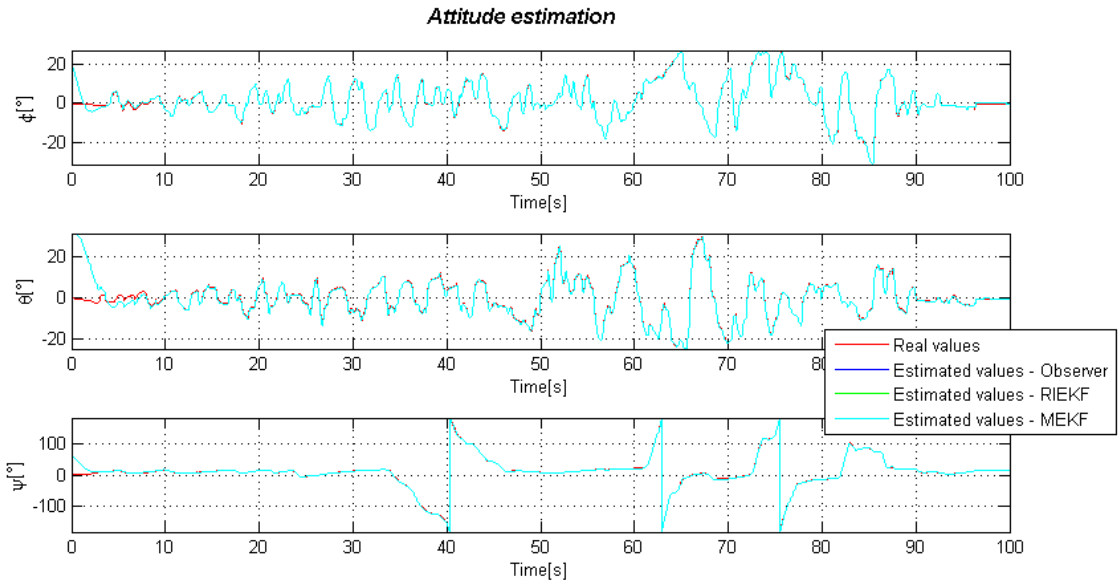

Figure 7-1: Attitude estimation – true data comparison.

Examining the overall plot, it can only be concluded that each three algorithms are really efficient, all start with the same error values and converge to the real values to estimate them really precisely. To have a better idea of what is exactly happening, an amplified view of the initial period (until 10 seconds) and a selected amplified zone at the end will be explored more in detail.

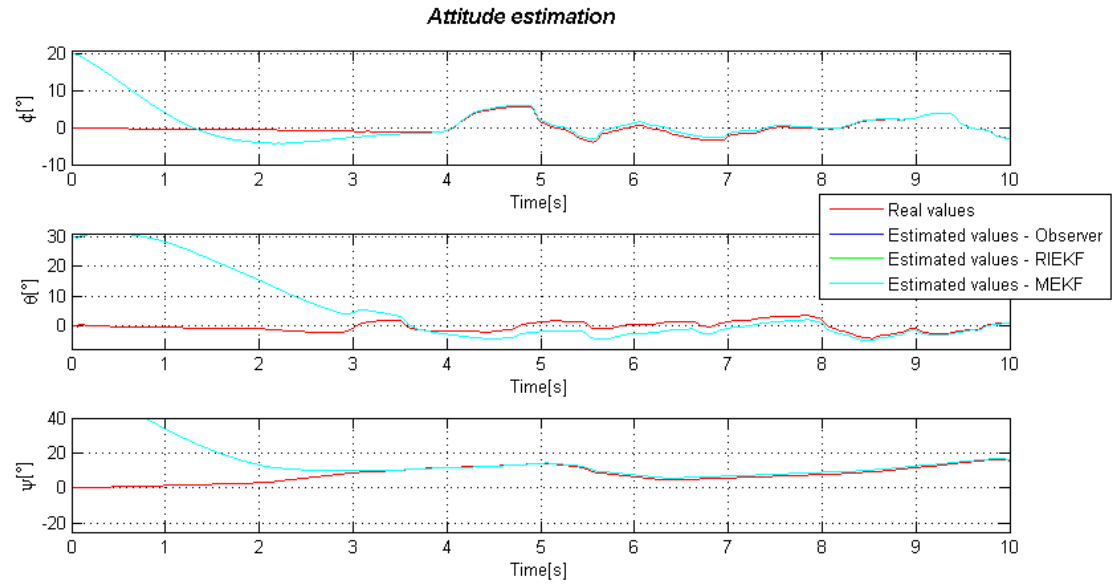

Figure 7-2: Attitude estimation - Amplified initial period from 0s to 10s.

Only two lines are visible, the red line corresponds to the real values curve and the light blue corresponds to the three algorithms because the blue and green lines are underneath the MEKF curve. This means that even with this axis adjustment, no difference between INAs can be observed. It can only be outlined that in approximately 8 seconds, the algorithm estimations converge to the real values, which is a quite fast response.

Another caption was taken to take a closer examination to the estimates but no difference can be easily seen between them.

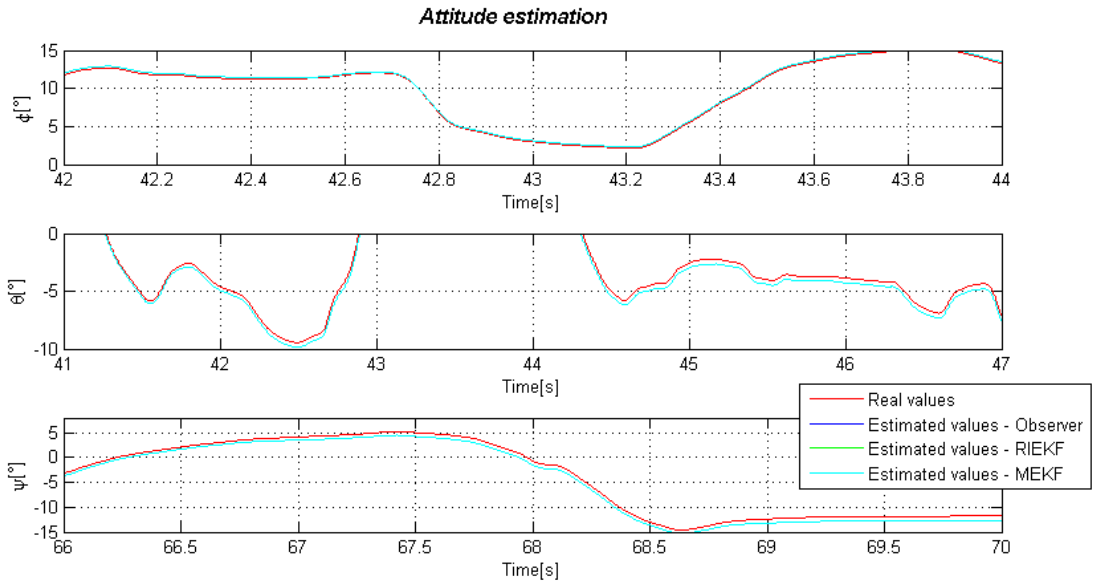

Figure 7-3: Attitude estimation - Amplified middle period.

With these 3 plots it can be concluded, that each of the algorithms is really accurate with the appropriate parameter initial configuration for the attitude estimation.

Although, we have been searching for unfound differences the velocity estimation will illustrate us with more interesting results:

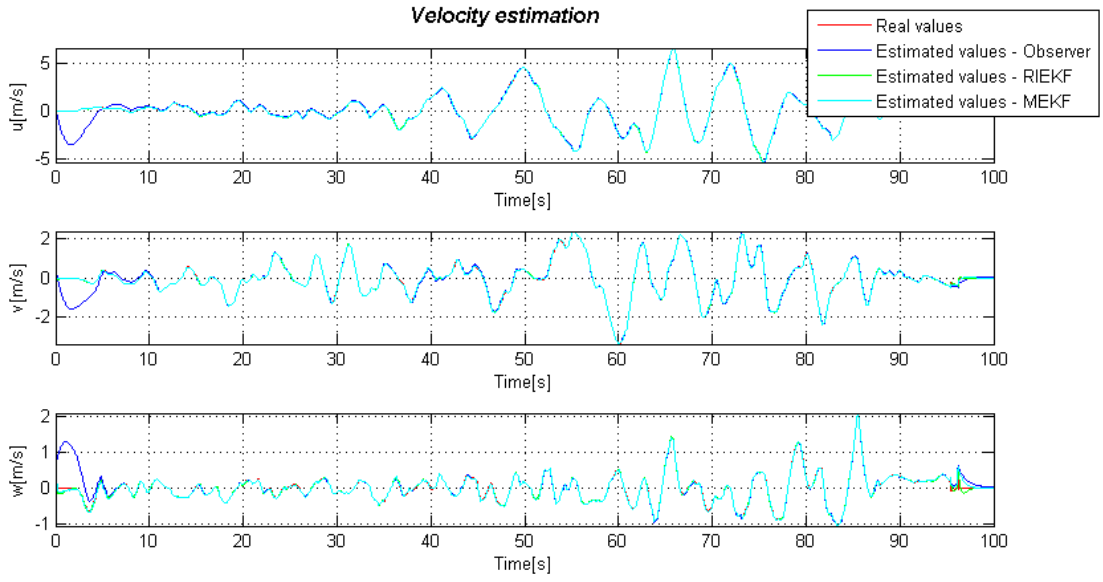

Figure 7-4: Velocity estimation – true data comparison.

As it can be seen in the previous plot and with the information of the exact initial error velocity values of Table 7-3, even the error initial values for the velocity are really small, the observer is still performing big difference estimations with the real values at the beginning. Meanwhile, both filters are converging really quickly. The following figure will show this observation:

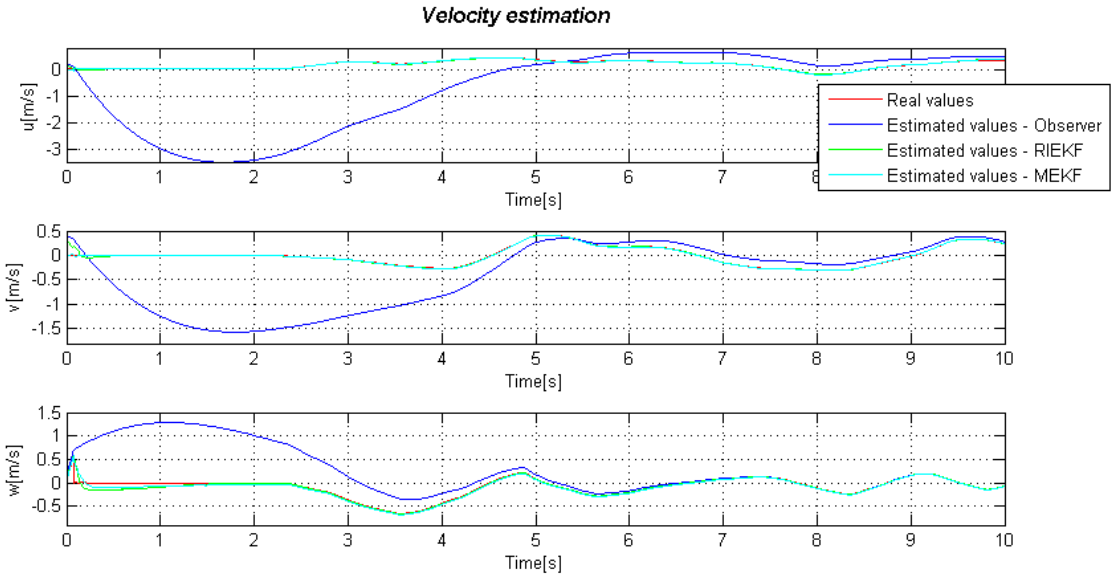

Figure 7-5: Velocity estimation - Amplified initial period from 0s to 10s.

The filters improve the initial situation and need less than 1 second to converge and reach the real values, while the observer needs around 4 – 5 seconds to converge. Adding another caption to our evaluation will clarify what is exactly going on:

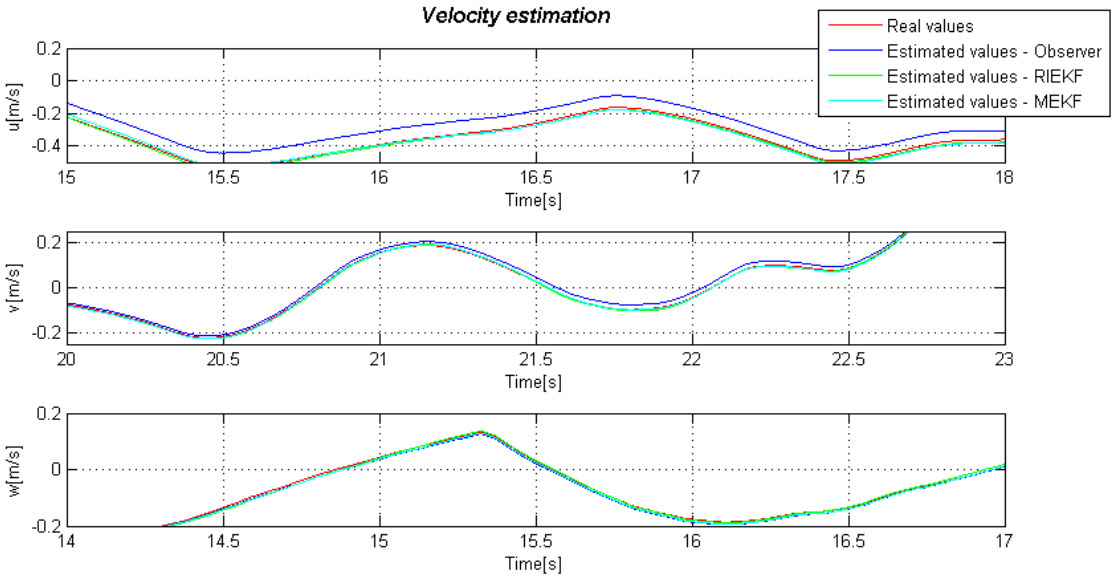

Figure 7-6: Velocity estimation - Amplified middle period.

Even the observer has converged is still not as good as the filters to estimate in each of the three axis. This can be seen more precisely in the x-axis of the previous plot. With the y-axis approximately adjusted between -0.2 (-0.4 for the x-axis) and 0.2 m/s, there is a small scaled

difference between the real values and the observer estimations, where the filters show no evidence of such distinctness.

To provide support for the latter discussion, we finally display the estimation of the gyro biases:

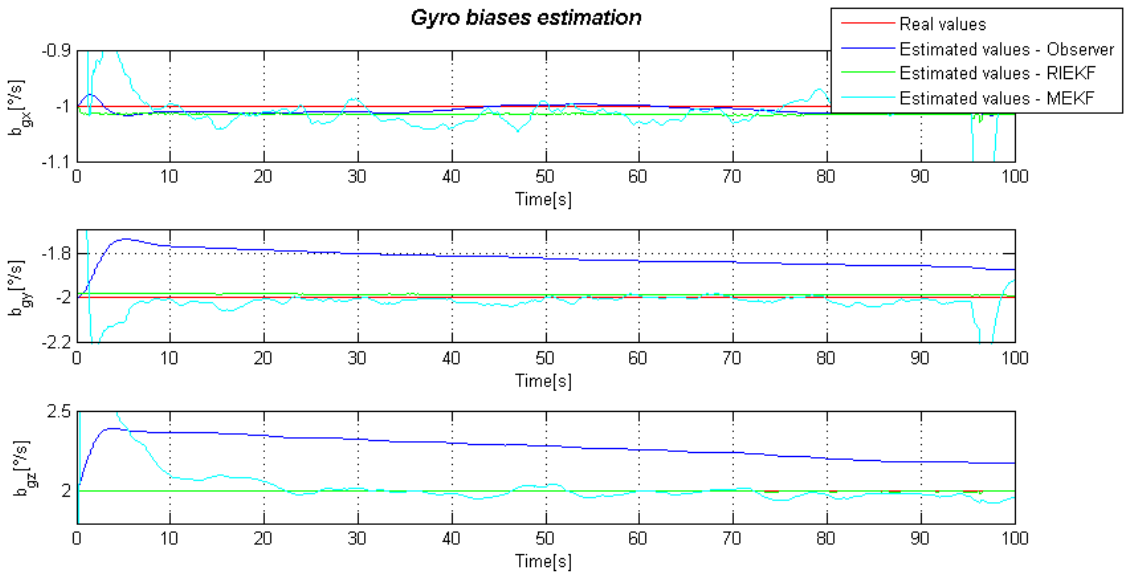

Figure 7-7: Gyro biases estimation.

Probably, these are the most interesting plots of this section. The most straightforward conclusion in here is that both filters behave better regarding the gyro biases estimation, because the observer, even being smoother, needs more than the considered time (100 s) to converge. Although this is currently not an important problem as it is known that the initial calibration procedures are considered of about 5 minutes, the estimations of the filters seem to be much faster.

Taking a part the observe estimation, which is quite good for the x-axis but not for the other two axes, the filters behave a little bit differently.

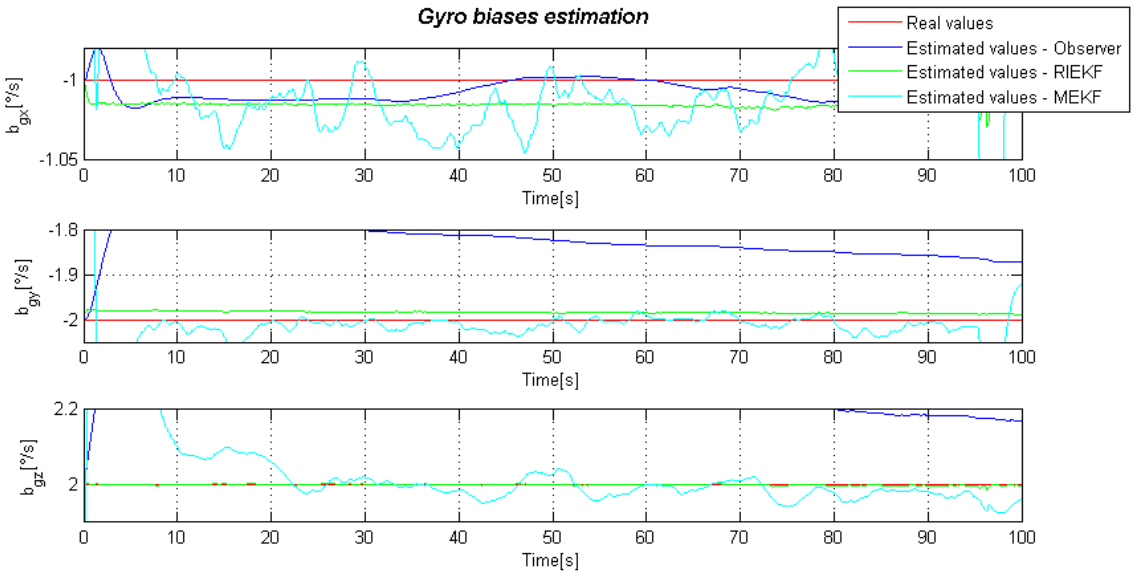

Figure 7-8: Gyro biases estimation – adjusted y-axis.

Figure 7-8 clearly reflects the differences. The MEKF estimates the biases really accurately for the three axes but the estimation curve is more turbulent, probably because of the value of matrix Q, which is more relevant for the MEKF than matrix $\boldsymbol{M}$ for the RIEKF.

Matrix Q influences on the MEKF adding more noise, which in consequence force the filter to pay more attention or to give more importance to the correction terms changing the value of the Kalman gains. Even representing the same matrix, $\boldsymbol{M}$ is not the same as Q. As explained previously, the design of the algorithms that preserve the invariance is different and the symmetries of the designed filter make the estimations be smoother and not turbulent. This is exactly what can be seen in the previous plot, the RIEKF estimations are not turbulent at all and the characteristic of smoothness is preserved during all the simulation.

Even only comprehending only trajectory 2, it can be concluded that the RIEKF is the algorithm that has a better performance in isolated cases with true data simulations. The results for the other perfect trajectories were similar with an appropriate selection of the tunable parameters.

7.2 Real-Time data comparison

For a more realistic scenario, the algorithms must be tested in a real situation with real-time data collected directly in a real flight with the appropriate sensors.

The case of study will be performed with 3 different types of data sequences collected in real situations. The first case shows flight data during a normal flight with the platform AscTec Hummingbird 1, which is a small quadrotor equipped with inertial sensors, GPS receiver, etc. A picture of the UAV platform is extracted from [26]:

Figure 7-9: AscTec hummingbird 1 quadrotor.

For a technical description of the quadrotor, it is recommended to see [26].

The other two cases consist on data sequences recorded with the same platform but in a special situation. To obtain these sequences, the quadrotor was used without propellers to record data on the rollercoasters of the Oktoberfest of this year 2014. The selected rollercoasters were Wilde Maus, which is appropriate to record big lateral accelerations but is mainly a rollercoaster for children, and the Alpina Bahn that is a funnier rollercoaster from my point of view. Both of them were selected because they do not perform any loop that is important for the GPS receiver not to lose reception.

To give an overview of the characteristics of the rollercoasters, the following figures display their shape:

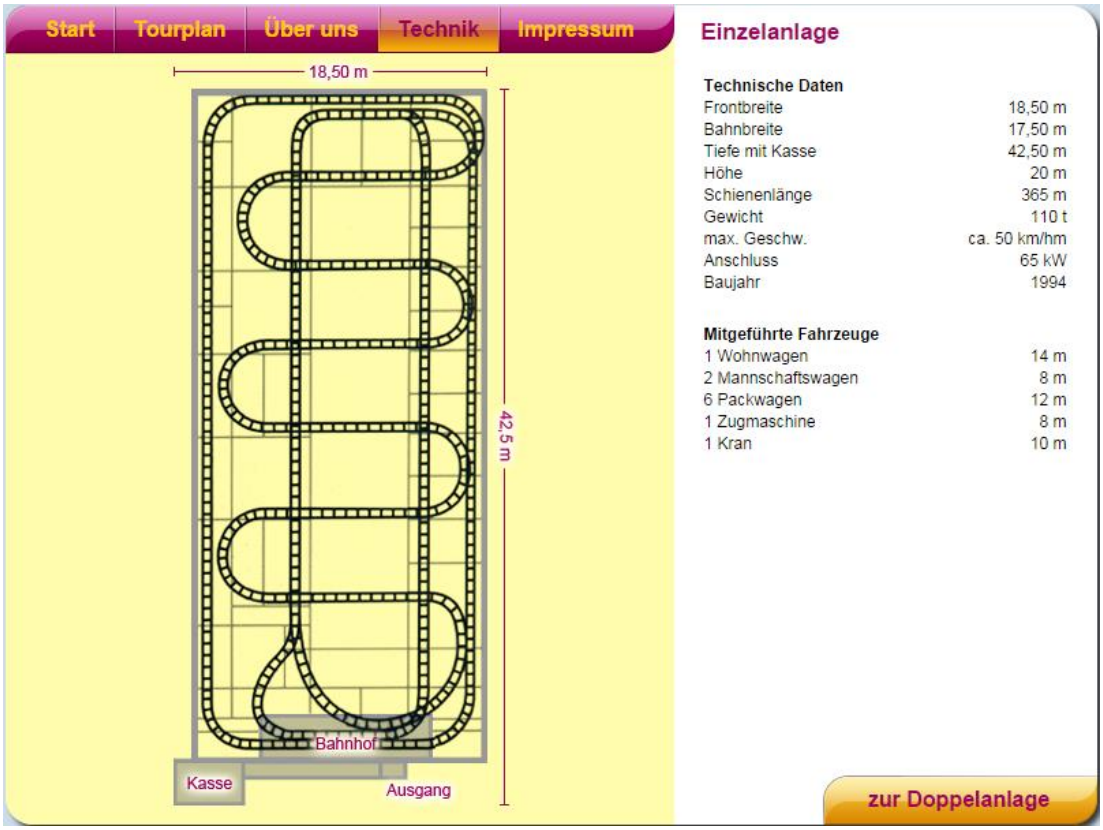

Figure 7-10: Wilde Maus Rollercoaster plan, extracted from [27].

As it can be seen, the size of the rollercoaster, maximum velocity and other interesting data is available to be read. None of the slope's steeps were really pronounced, but the lateral accelerations were quite sudden and big to obtain quite interesting data. This is because the train has no bank inclination; it was always completely horizontal all the time, which increases the quickness of the turn.

The Alpina Bahn website has no technic map but for the reader to make themself an idea, the following figure is added:

Figure 7-11: Alpina Bahn Rollercoaster overview, extracted from [28].

For the technical specifications it is interesting to check [28], but the overall main idea is that the maximum velocity is about 80 km/h, which is bigger in this case. The slopes are also steeper (specially the first one) that enables the train to reach a vertical acceleration of about 4.8 g. To appreciate the difference a closer picture of a high-speed turn is displayed:

Figure 7-12: High-speed turn of Alpina Bahn rollercoaster, extracted from [28].

In this case, it can be seen that the turn have a bank inclination which helps the train not to adhere to the railway. This is required because if the higher velocities and accelerations of this rollercoaster.

It is important to notice that the propellers of the quadrotor are probably not enough powerful to reconstruct and perform the same trajectory as the train with the recorded data, but it is interesting to evaluate these data sequences to see how the sensors react and how the INAs manage the sensor data, which is quite extreme, in these cases.

Regarding the real-time data of the sensors, it is fundamental to make some differences on which of the sensors are used. The main variables that we are interested in to take as inputs of the INAs are: the rotational rate, the acceleration, the Earth magnetic field and the velocity.

The former two first variables are obtained from the IMU. Also the Earth magnetic field is recorded with the IMU. But the GPS velocity is not fully obtained from the GPS sensor because the component in the z-axis is not accurate enough. Thus, the third component of the velocity is measured with the IMU as well and used as an input for the algorithms. The differences between the data used in section 7.1 and the real-time data is updating rate of the GPS measurement. Meanwhile, the IMU data is saved approximately every 0.001s, the GPS change every 0.01s. Due to the measuring frequency of the different sensors, a delay is applied to the estimation of the velocity.

Contemplating all the other variables, the unique extra important difference is that the Earth magnetic field measurement was considered a normalized rotation of the nominal Earth magnetic field to the b-frame as displayed in (3-11) and now is a real measurement recorded in Tesla [T].

On the previous section the comparison of all the different INAs was in a unique plot, thus the differences were difficult to be differed. In this actual section the analogy between algorithm estimations will be treated in different plots for a better evaluation.

It is also important to highlight that only the estimations of attitude and velocity will be discussed in this section as the previous one is representative enough of how each algorithm handles with the gyro biases estimation.

This case is even more difficult to contrast as the perfect trajectory does not exist. The comparison will be performed with reference to the sensor measurements of the variables, which means the reference is far from being true. For this reason, the ability to read which trend follows the curves is needed.

To start the identification of each of the parameters to select, first of all, I want to define the correct nominal Earth magnetic field $\left(\vec{\boldsymbol{B}}\right)_{\boldsymbol{W,nom}}$ for the case of the real-time measurements. In this case, the magnetic field measurement is also normalized but is not understood as a rotation of the nominal value with the attitude estimation in quaternion form as displayed in equation (4-4). Instead of this, the measurement is directly extracted from the magnetometers in the b-Frame and then the computations regarding the nominal value in the w-Frame are performed in the correction step as before displayed in equation (6-56) replacing the previous value for: $\left(\vec{\boldsymbol{B}}\right)_{\boldsymbol{W,nom}} = (\boldsymbol{20973.8e^{-9}} \quad \boldsymbol{807.8e^{-9}} \quad \boldsymbol{43313.9e^{-9}})^{\boldsymbol{T}}\ \boldsymbol{T}$.

Due to the different value, equation (6-56) can be changed to:

$$\frac{\partial \vec{y}_B}{\partial \psi} = \frac{\partial}{\partial \psi}\left(\left(\vec{B}\right)_{B,meas} - R_{B\tilde{W}} \cdot \left(\vec{B}\right)_{W,nom} - R_{B\tilde{W}} \cdot \begin{pmatrix} -\delta\theta \\ \delta\phi - \delta\psi \\ \delta\theta \end{pmatrix}\right) = R_{B\tilde{W}} \cdot \begin{pmatrix} 807.8e^{-9} \\ 20973.8e^{-9} \\ 0 \end{pmatrix} \quad \textbf{(7-1)}$$

This modification is only applied to the MEKF, the design of the symmetry-preserving observer and the RIEKF is defined in a kind of way that the magnetometer nominal value has to use the value of $\left(\vec{\boldsymbol{B}}\right)_{\boldsymbol{W,nom}} = (\boldsymbol{1} \quad \boldsymbol{0} \quad \boldsymbol{1})^{\boldsymbol{T}}$**,** which has no units. It has something to deal with the manner the authors designed the INAs.

After the necessary details recently explained, we can go on with the results of the simulations. It is fundamental to understand that the algorithms are brought into the most extreme cases for the cases of the rollercoaster data, because a quadrotor of this size can never reach such high velocity, acceleration, etc. values. It has to be taken into account that due to the assumption of flat earth and the neglecting of higher-order terms, better accuracies for more complicated cases like the ones which rule the system models of bigger UAVs cannot be reached.

That is the reason why is important to understand that the rollercoaster cases are included only for testing the algorithms under extreme conditions but for more reliable results is better to trust the ones concluded in the previous section or the data of real flights extracted from the quadrotor, which are going to be analyzed subsequently.

Firstly, the tunable parameters must be selected. It is noticeable that the parameters will be different from the ones of the previous section and that is because the algorithms could be adapted for really specific cases and for more general situations. This general parameter selection is going to be used in this section as it is the best option for a real-time flight data. For each of the plots the tunable parameters will be displayed in a table but for all the other cases, the initial error values will be:

Variable to estimate	Real initial values	Estimated initial values
$\boldsymbol{\phi}[°]$	0	0
$\boldsymbol{\theta}[°]$	0	0
$\boldsymbol{\psi}[°]$	0	0
$\boldsymbol{u}\ [\boldsymbol{m/s}]$	0	0
$\boldsymbol{v}\ [\boldsymbol{m/s}]$	0	0
$\boldsymbol{w}\ [\boldsymbol{m/s}]$	0	0

Table 7-4: Initial values for the algorithms.

Normally the data sequences start at a value of zero, but sometimes this value is change in one sample to a further value (mostly in the attitude case) depending for instance of the orientation of the quadrotor at the moment of start. Moreover, the algorithms had proved to be robust and no approximation to the sudden initial changes is necessary as they are able to converge really accurately and quickly to the corresponding real values. There are only stated the values of attitude and velocity as they are the variables to be evaluated in this section.

For the symmetry-preserving observer, the gain configuration used in all the following sections is previously displayed in Table 4-7, named experimental gain configuration. This has been proved to be the best parameter selection for the observer in a general point of view, which means that best adapts the estimations for a bigger set of different trajectories. Probably, there is the possibility to find a better gain configuration for each specific trajectory but it is not necessary because the estimations will not be significantly enhanced, as seen in section 4.3.

The parameters for the RIEKF are shown subsequently:

Matrix	Initial values
$\boldsymbol{P_0}$	$Diag\left(rad(20)^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 2^2 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad \begin{pmatrix}0\\0\\0\end{pmatrix}^T \quad 1^2\right)$
$\boldsymbol{M}$	$Diag\left(0.5 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 0.1 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 0.001 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 0.1\right)$
$\boldsymbol{N}$	$Diag\left(0.1 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T \quad 0.1 \cdot \begin{pmatrix}1\\1\\1\end{pmatrix}^T\right)$

Table 7-5: RIEKF tuneable parameters.

And finally the parameter pattern for the MEKF is also outlined:

Matrix	Initial values
$\boldsymbol{P_0}$	$Diag\left(\begin{pmatrix} rad(30)^2 \\ rad(30)^2 \\ rad(60)^2 \end{pmatrix}^T \quad \begin{pmatrix} 2^2 \\ 2^2 \\ 20^2 \end{pmatrix}^T \quad \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}^T \quad \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}^T\right)$
$\boldsymbol{Q}$	$Diag\left(0.5^2 \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}^T \quad 0.1^2 \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}^T \quad 0.001^2 \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}^T \quad 0.001^2 \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}^T\right)$
$\boldsymbol{R_v}$	$1^2 \cdot I_3$
$\boldsymbol{R_B}$	$0.2e^{-3} \cdot I_3$

Table 7-6: MEKF tuneable parameters.

As it can be seen, the parameters are a little bit different from the selected ones on the previous section. In here we do not pay attention to the biases estimation and the scaling factor for the symmetry-preserving algorithms and the values of the covariance matrices regarding the attitude and velocity are adapted for simulation.

The measurement covariance matrices, $\boldsymbol{R_v}\ and\ \boldsymbol{R_B}$ for the MEKF and $\boldsymbol{N}$ for the RIEKF, are also correctly adjusted for the following trajectories' estimation.

For each of the trajectories there are estimations of attitude and velocity for each of the 3 INAs and plots of the whole trajectory and estimations as well as amplified parts of these paths are taken. For space and fluency issues, not all the plots are added but the important figures.

7.2.1 Quadrotor flight data

The quadrotor flight data has an approximately duration of 830 seconds, which is a period of about 13 – 14 minutes of flight. Here it is remarkable to see how the INAs' estimations behave at the beginning. At the two other trajectories, this issue is not as notable because some minutes until the ride on the rollercoasters are left for calibration and alignment processes. But returning to the subject, the overall trajectory is shown subsequently:

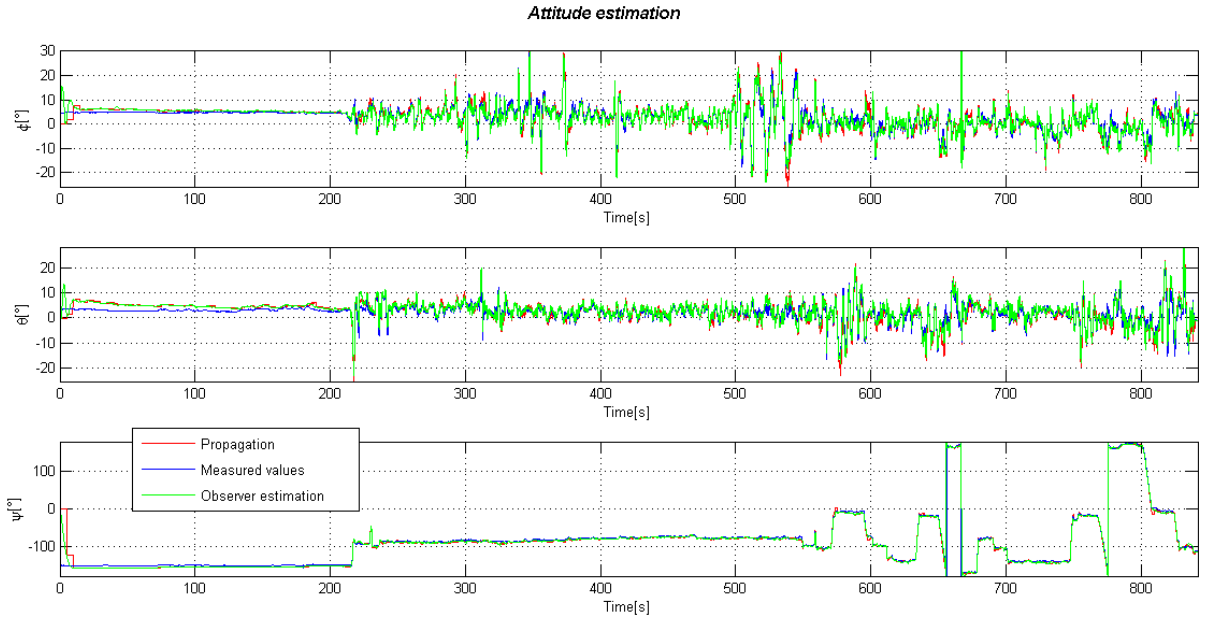

Figure 7-13: Propagation and observer attitude estimation – Quadrotor flight.

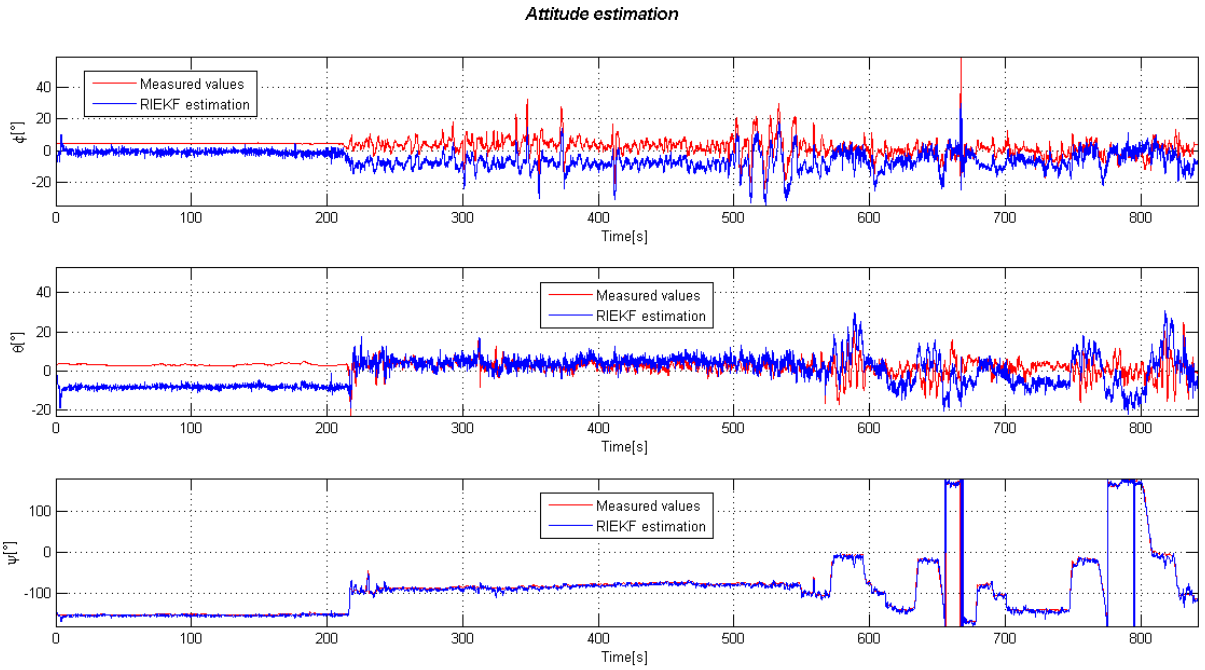

Figure 7-14: RIEKF attitude estimation – Quadrotor flight.

As it is easily seen, the RIEKF estimates $\phi \, and \, \theta$ quite badly. Moreover, thanks to the Earth magnetic field measurement ψ is perfectly estimated. This problem is not as pronounced for the observer and for the MEKF. It has to be remarked that it was not straightforward to select the covariance matrices for the KFs to estimate correctly as the trajectories have really sudden and abrupt changes difficult to predict.

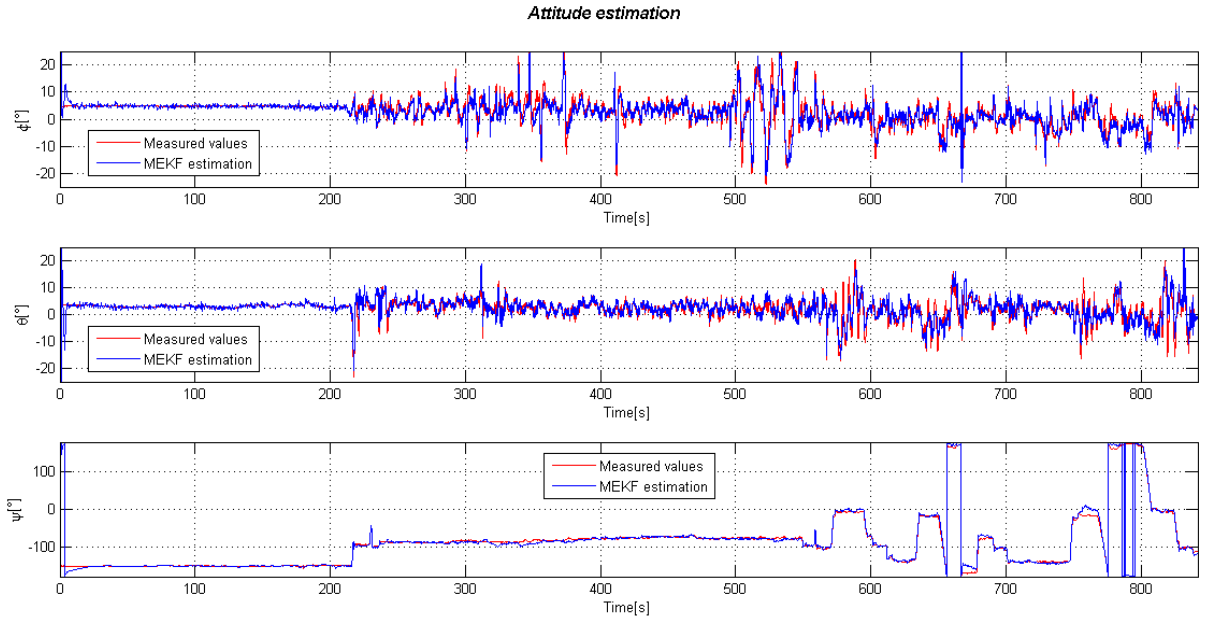

Figure 7-15: MEKF attitude estimation – Quadrotor flight.

The MEKF is really better than the MEKF in this case; when the initial error is corrected and the algorithm has converged it always remains close to the measured values. It is highly important to remind that we are not comparing which estimation is closer to the measured attitude angles because they might not be perfect either. Then the question is: to what information should we compare the estimations? To solve this issue, amplified plots of the situation will be analysed in detail to try to conclude which one is the best estimation or the most suitable for our interests. Another aid added to the plots is the propagation of the attitude itself, which is a quite reliable source to compare the estimations with.

A closer examination can be performed with amplified plots. For instance, on the subsequent figure, there is a clearer example of how this should be interpreted to conclude on which one of the curves is closer to the real signal without any perturbation.

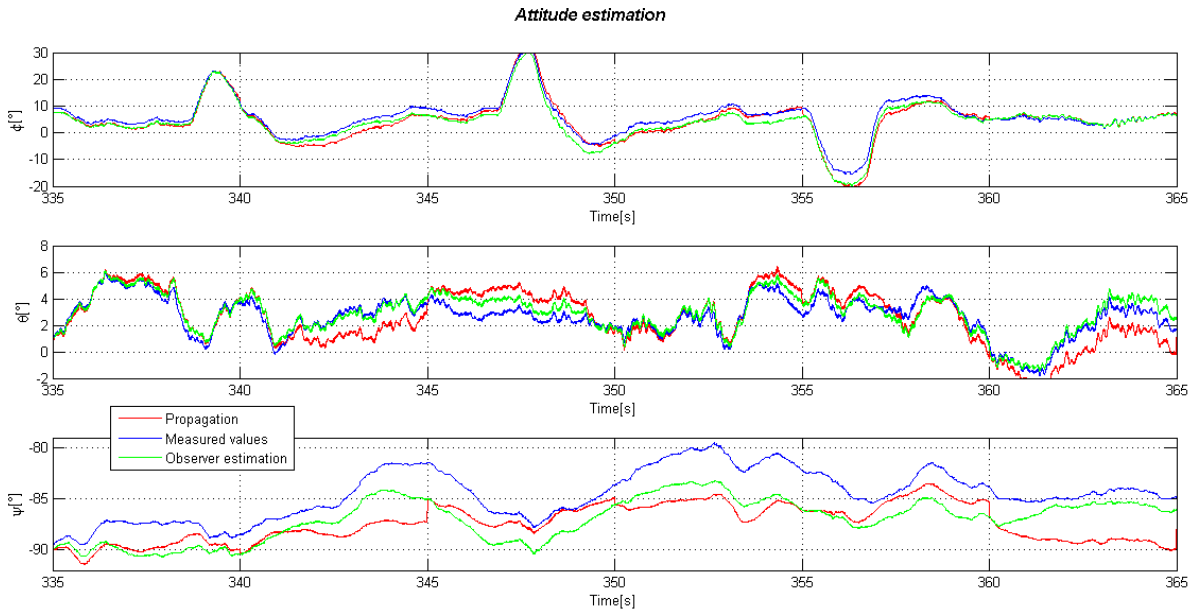

Figure 7-16: Amplified observer attitude estimation – Quadrotor flight.

As it is seen, all three curves are accurate but paying attention to the second curve (pitch angle estimation) it is difficult to conclude which curve is better. The common sense leads us to the conclusion that the true/real attitude should be somewhere in between the three lines, and if we focus on the time period between 345 – 350 seconds, it is clear that the best estimation curve is the green line (or the observer estimation line). Moreover, even seeing that the green line is most of the time in between the other two lines, there are some parts of the plot where this is not accomplished.

To finish with this deduction, it is interesting to fix to the third plot of the previous figure. Here, a more evident conclusion can be extracted. As it is shown, the blue line (measured values line) is far from being the best of three, but there is a correlation between the green and red lines which seem to be following the same behaviour. The red line is updated every 5 seconds with the reference of the green line, but most of the 5 sec-period is not diverging that much. Thus, I am able to conclude that the observer estimation in this period is closer to the real values more time than the other evaluated curves.

After that, two amplified versions of the same plot in a different time period for the RIEKF and MEKF estimations are displayed:

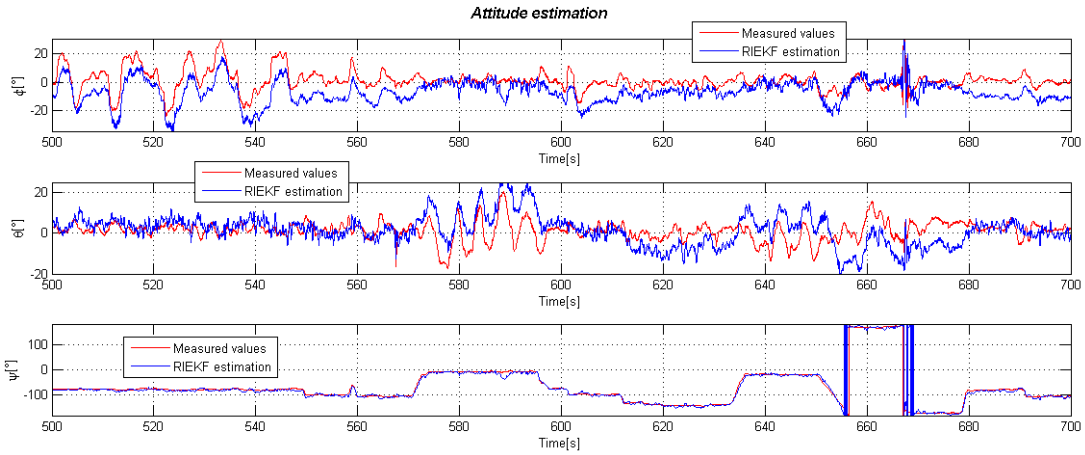

Figure 7-17: Amplified RIEKF attitude estimation – Quadrotor flight.

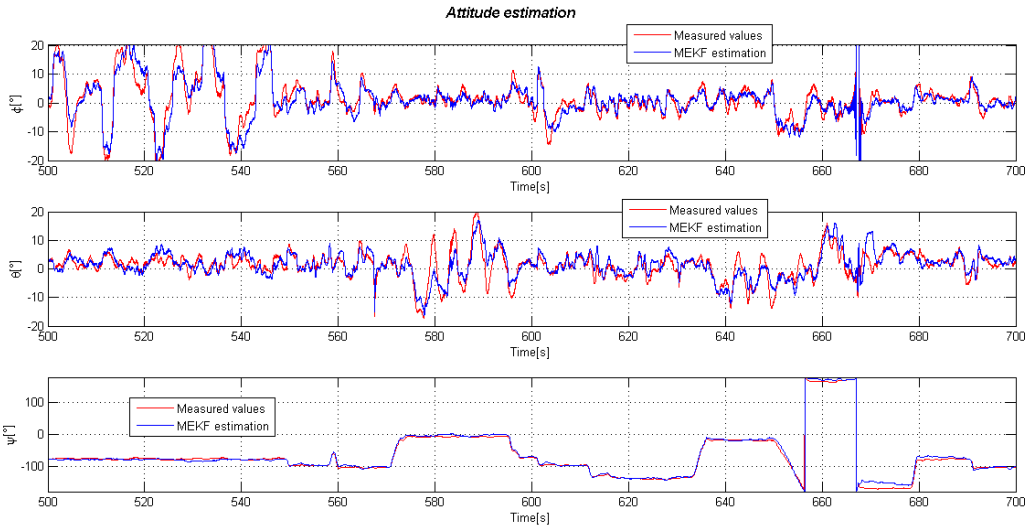

Figure 7-18: Amplified MEKF attitude estimation – Quadrotor flight.

From my point of view, it can be said that the MEKF estimation is better than the RIEKF, which seems to be more turbulent and not that accurate. Meanwhile, the MEKF is more similar to the observer estimation.

For comparing both of them, the initial part of the previous plots will be displayed to be able to evaluate the behaviour of the beginning in a better way:

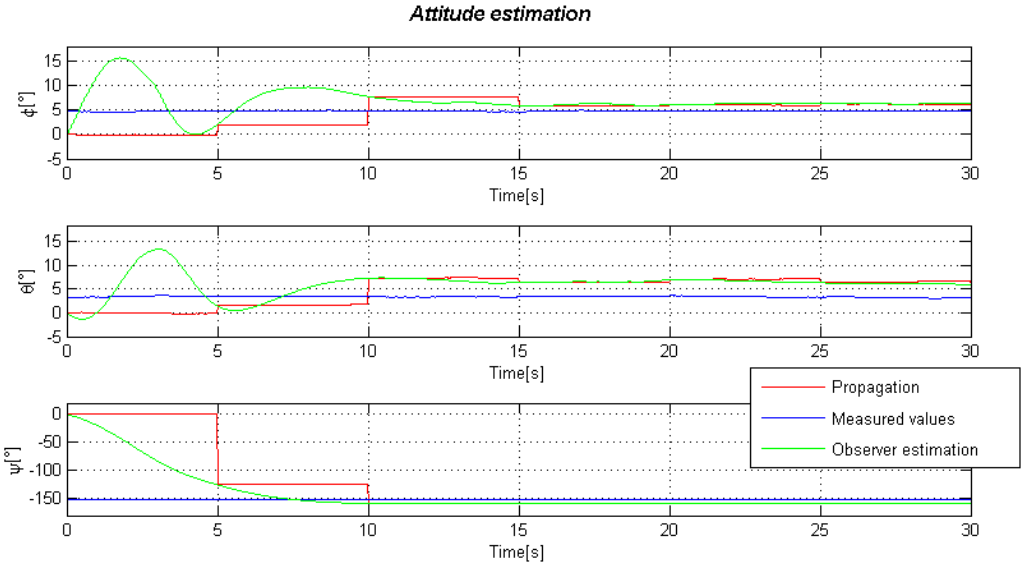

Figure 7-19: Amplified initial observer attitude estimation – Quadrotor flight.

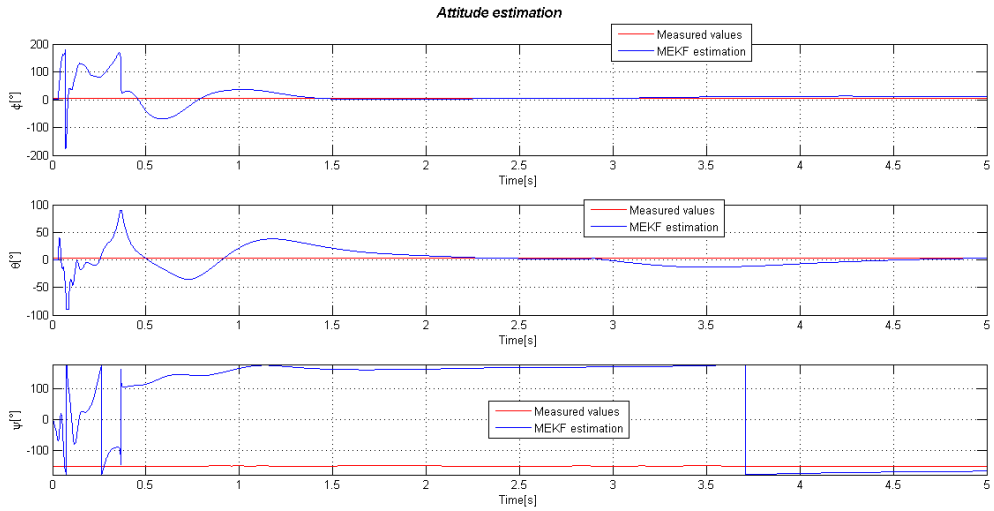

Figure 7-20: Amplified initial MEKF attitude estimation – Quadrotor flight.

The time scales are different but even being more turbulent at the beginning the MEKF is faster in converging to the desired measured values than the observer which lasts about 10 seconds to converge, while the MEKF only lasts 4 seconds. From another point of view, sometimes this is not an advantage as it is probably that the faster the convergence the more turbulent of noisy is the estimation. In this case, the observer overall estimation seems to be smoother than the MEKF one. This characteristic seems to be the result of the symmetry-preserving theory.

Evaluating the last sentence, our thoughts can lead us to the conclusion that the RIEKF should be, then, better than the observer. But probably what has happened is that the RIEKF parameters should be tuned in a different manner to adjust better the estimation. The issue is

that parameter configuration is far from being straightforward and no better configuration has been achieved regarding that the configuration should work in a more general case as the observer and the MEKF configurations do.

To finalize with this section it is also shown the velocity estimations without going more deep into them:

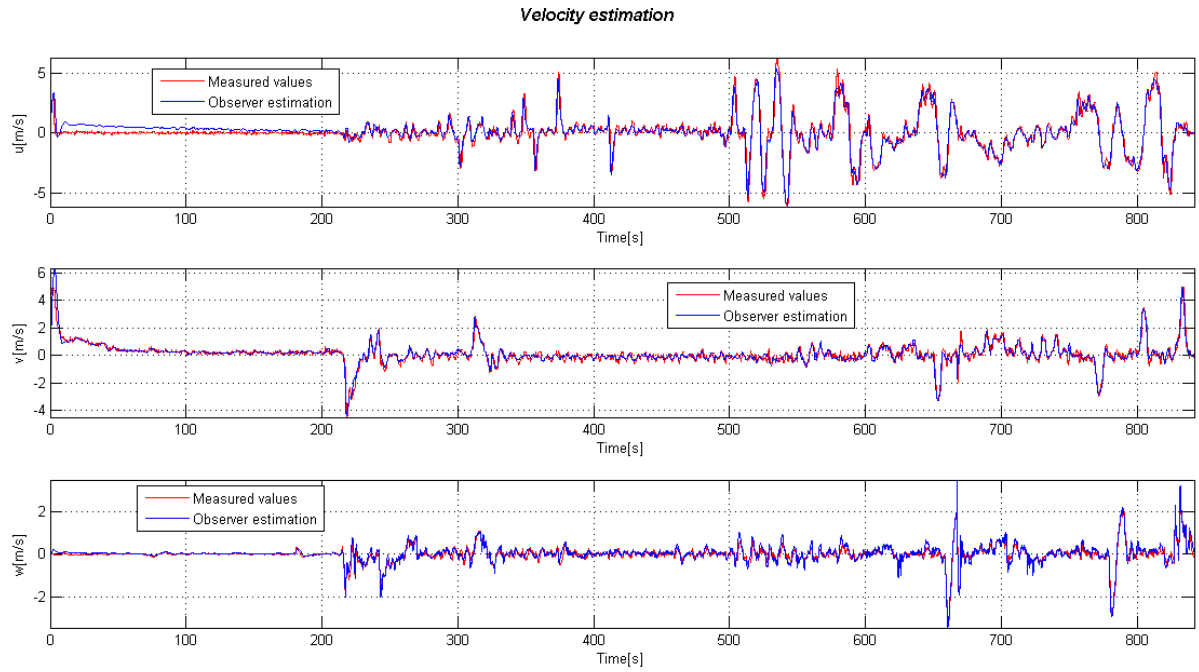

Figure 7-21: Observer velocity estimation – Quadrotor flight.

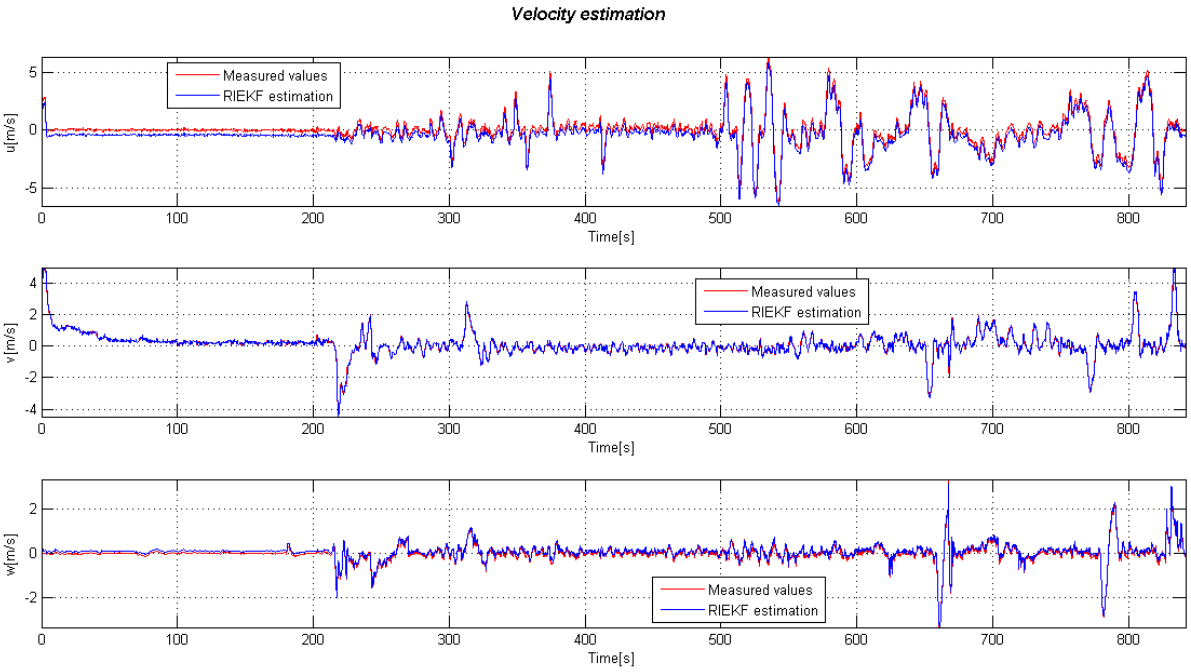

Figure 7-22: RIEKF velocity estimation – Quadrotor flight.

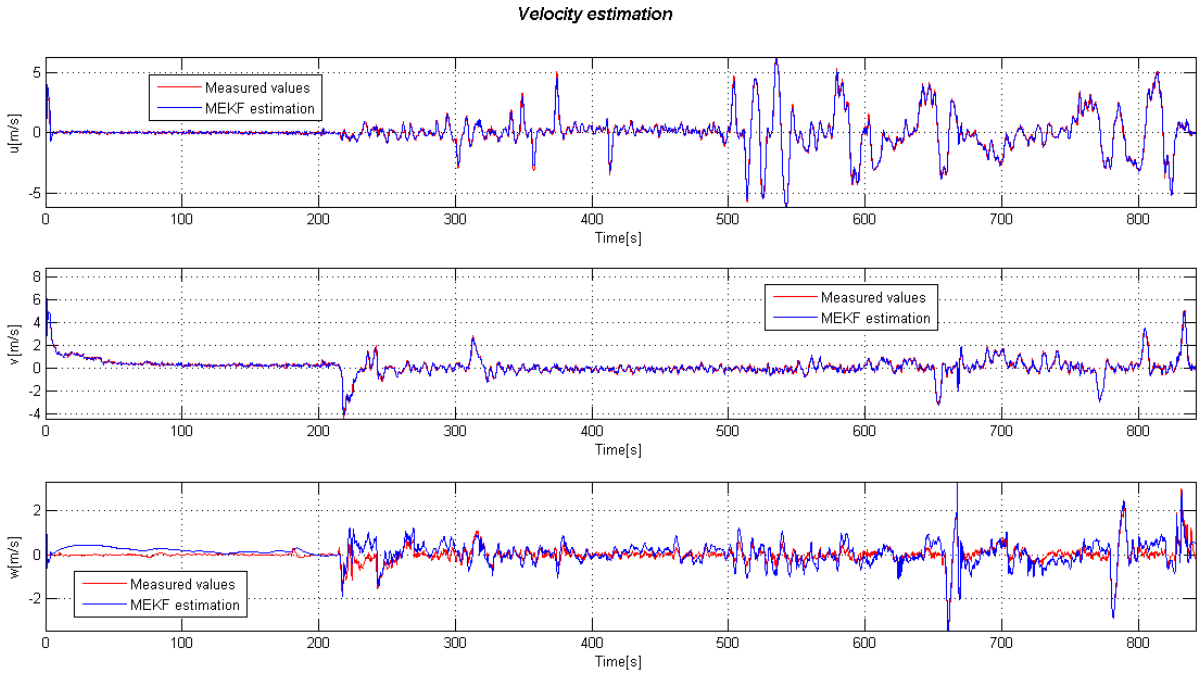

Figure 7-23: MEKF velocity estimation – Quadrotor flight.

For the velocity estimation is hard to conclude which of the algorithms is better. The MEKF seems to be better but for the z-axis. Then the observer is not really good for the x-axis, neither the RIEKF. But during the overall flight the RIEKF is more accurate from my point of view.

The reason why I was saying at the beginning of this chapter that the absolute truth of which of the algorithms is more accurate is difficult to know, can be observed in here where for the attitude estimation is better the observer or the MEKF but for the velocity is better the RIEKF. With real-time data is not straightforward to contrast the algorithms. It seems to be more reliable the evidences extracted from the previous section 7.1. More specific conclusions will be stated at section 8.

7.2.2 Wilde Maus data

The data recorded in the rollercoaster shown in Figure 7-10 is used. As we have two different runs for this case I selected the second run, which seems to be better in terms of the estimated variables.

Only the sequence of data between 520 – 630 seconds is selected as it is the time riding the rollercoaster, the previous data is just calibration time. So the ride is about 1.50 min. This calibration period is used for the initial alignment and for the calibration of the sensors of the quadrotor. As explained previously, this procedure brings the possibility of improving the performance of the INAs, as they need to be properly initialized to the real values to be completely sure that the algorithm converges.

To have a better idea of what is exactly the trajectory followed; a 3D plot is displayed to clarify the 2D map of Figure 7-10.

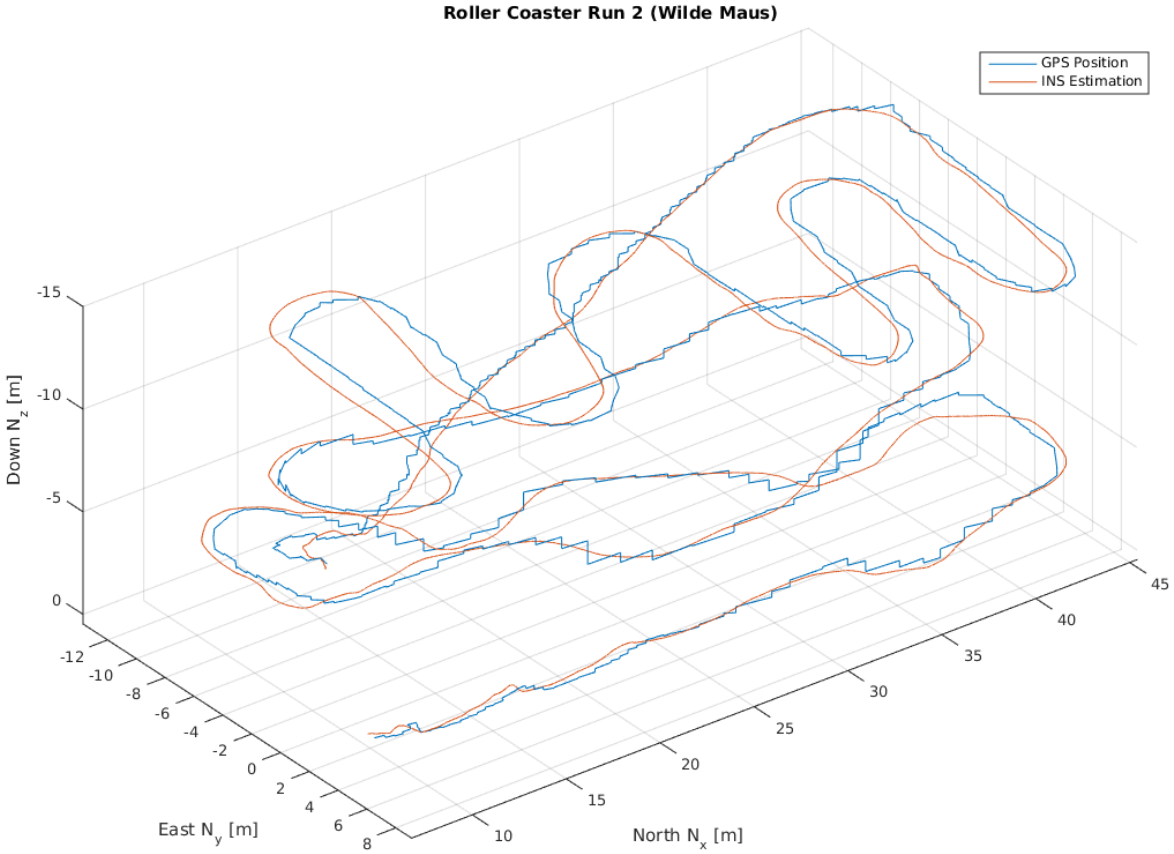

Figure 7-24: Trajectory 3D plot - Wilde Maus.

The plot compares the GPS position, which is not used for the algorithms but is recorded as well, and the INS Estimation of the position using the IMU data.

Regarding the same comparison, it can be interesting to take a closer examination to the norm of the velocity and the measured acceleration:

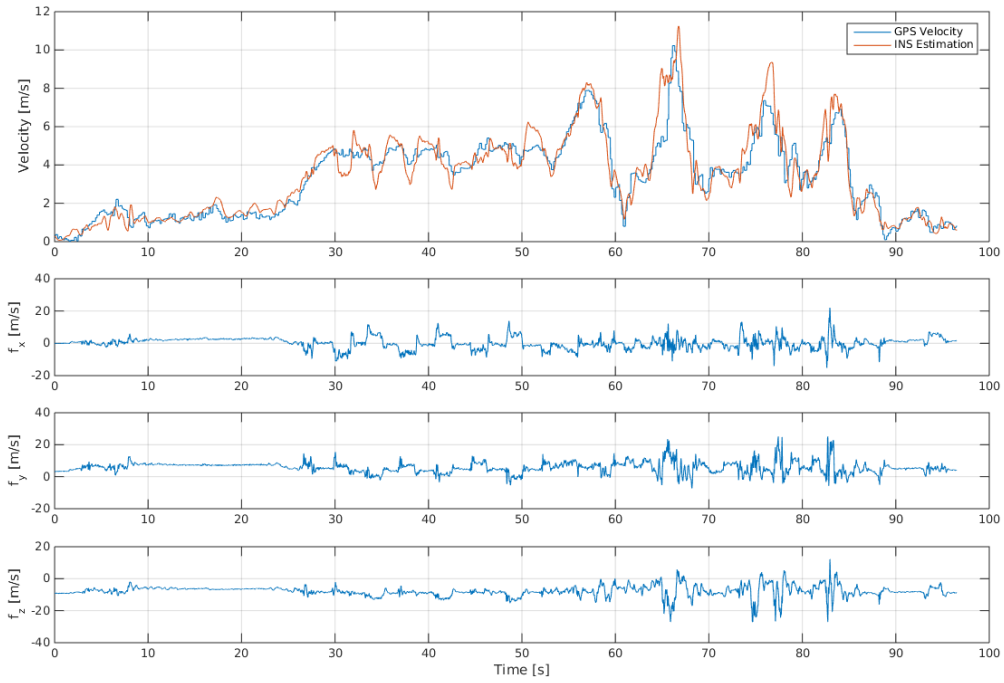

Figure 7-25: Norm of the velocity and measured acceleration components – Wilde Maus.

It is also interesting to observe the estimations of the algorithms for this case:

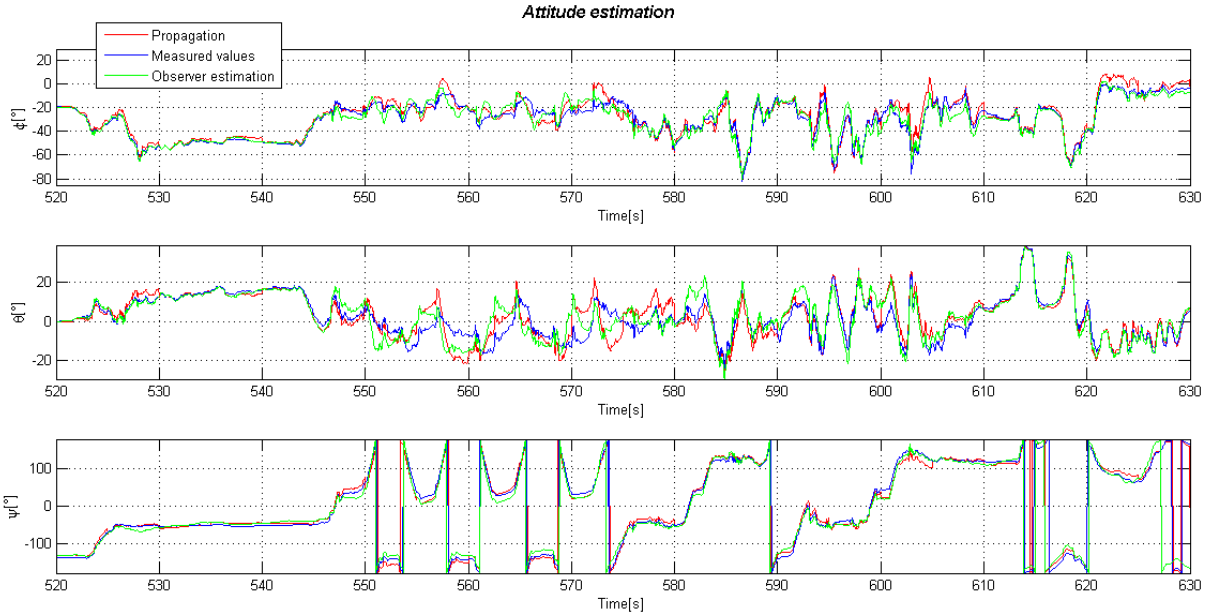

Figure 7-26: Observer attitude estimation – Wilde Maus.

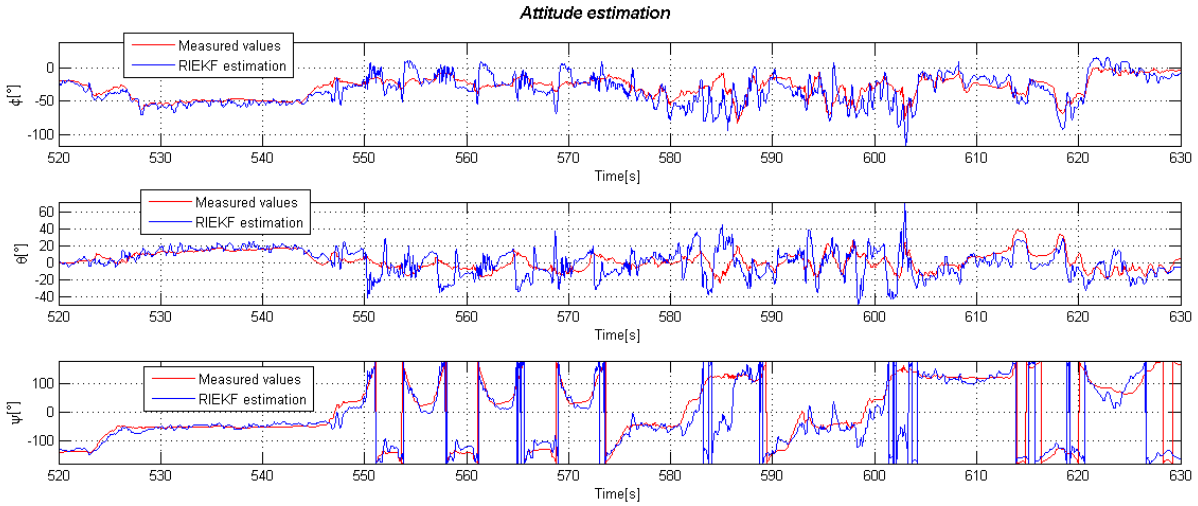

Figure 7-27: RIEKF attitude estimation – Wilde Maus.

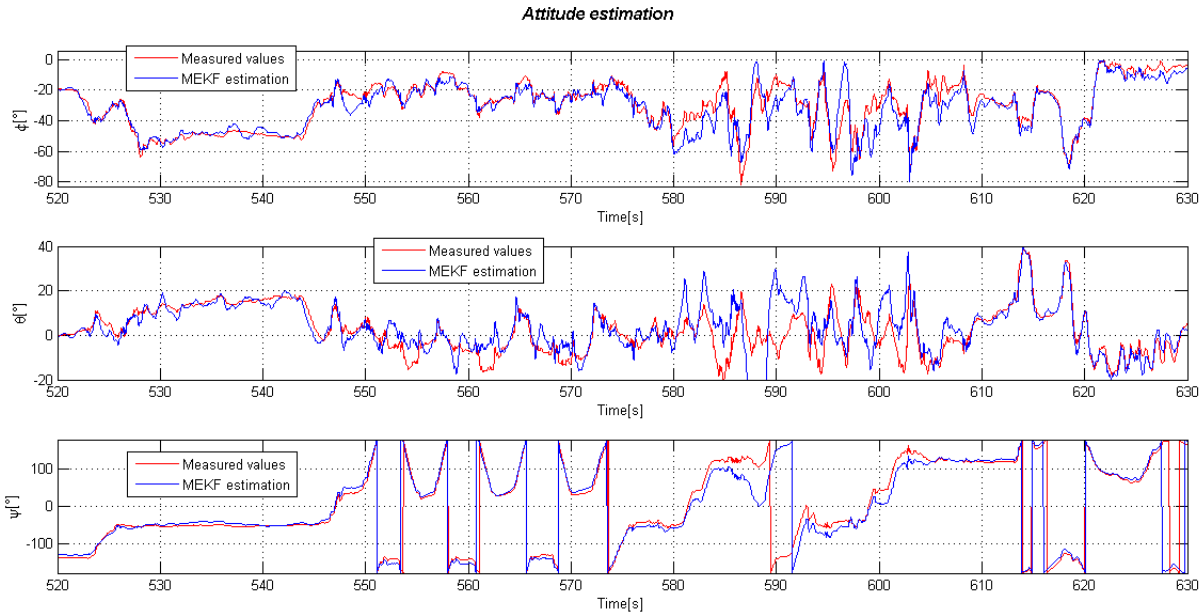

Figure 7-28: MEKF attitude estimation – Wilde Maus.

Contrasting the three previous plots, it is clear that the conclusion is the same as in section 7.2.1. The estimation of the attitude is best achieved with the observer and the MEKF, probably not turbulent with the observer compared with the MEKF, which is always a good feature to take into account.

The important difference is not at the attitude estimation, but the velocity estimation:

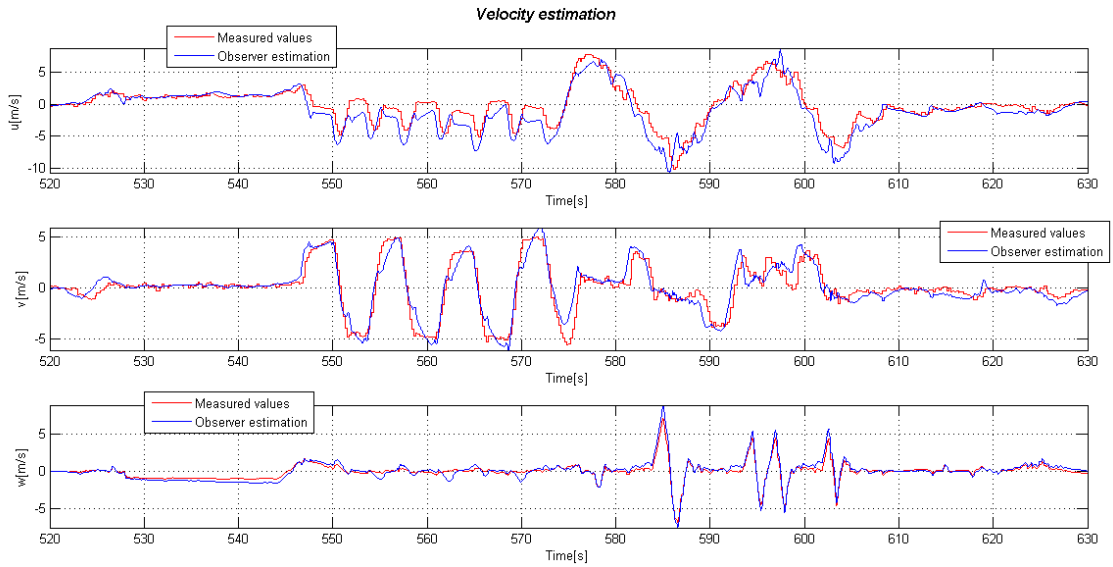

Figure 7-29: Observer velocity estimation – Wilde Maus.

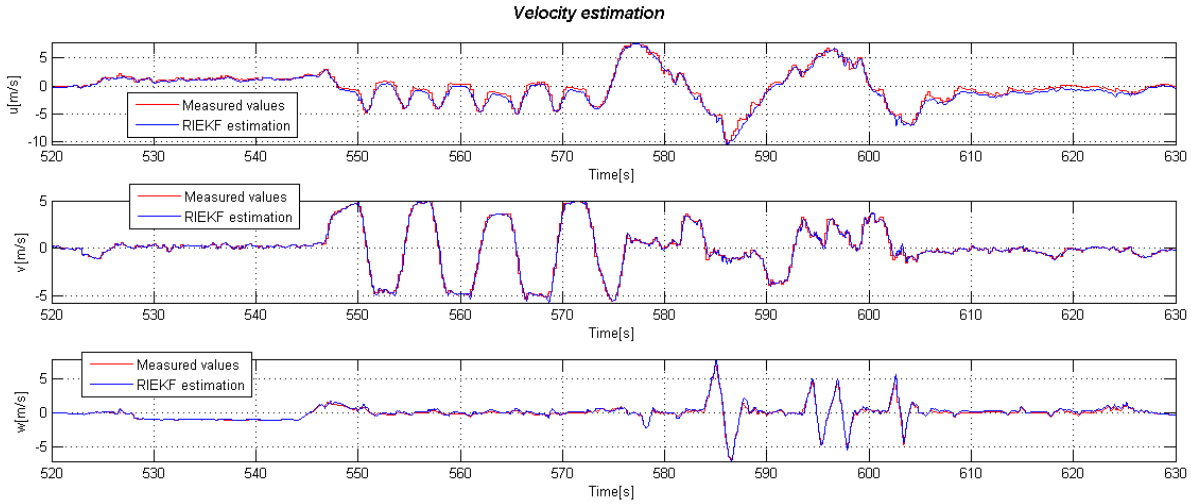

Figure 7-30: RIEKF velocity estimation – Wilde Maus.

As it is concluded in the previous section with the quadrotor flight data, the velocity estimation is more accurate using the RIEKF. The symmetries really enhance the estimations in this variable. These new plots support the same results that we extracted previously.

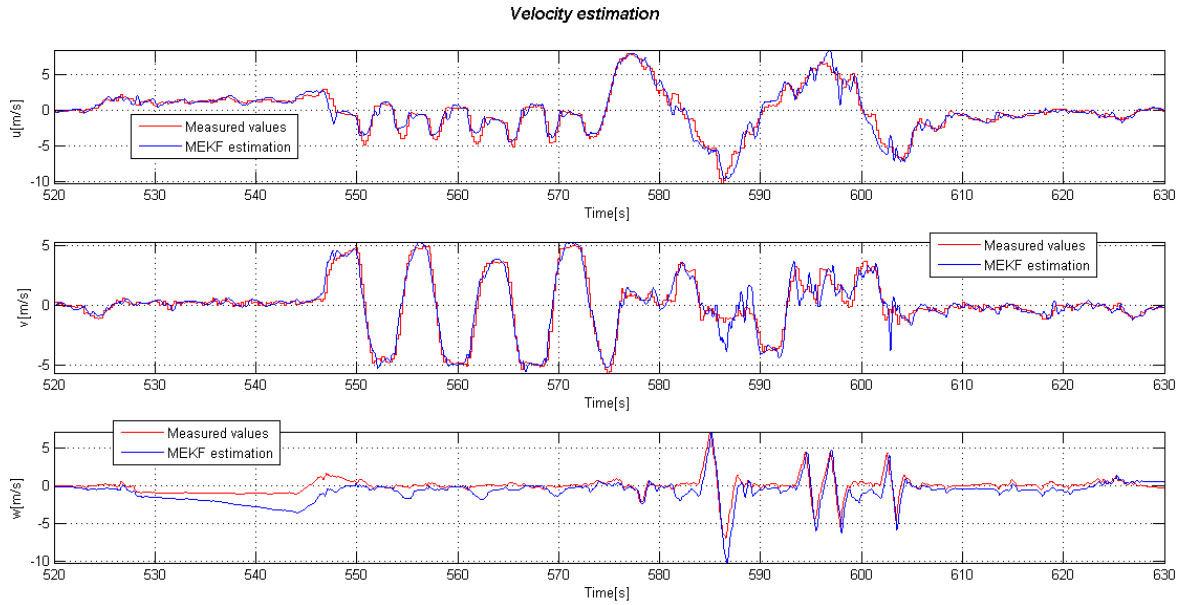

Figure 7-31: MEKF velocity estimation – Wilde Maus.

7.2.3 Alpina Bahn data

As explained on the previous section, the data sequence considered to evaluate is between 310 – 440 seconds, which is an approximately duration of 2.16 min of the rollercoaster ride. A 3D plot outlining the trajectory shown in Figure 7-11 is displayed subsequently:

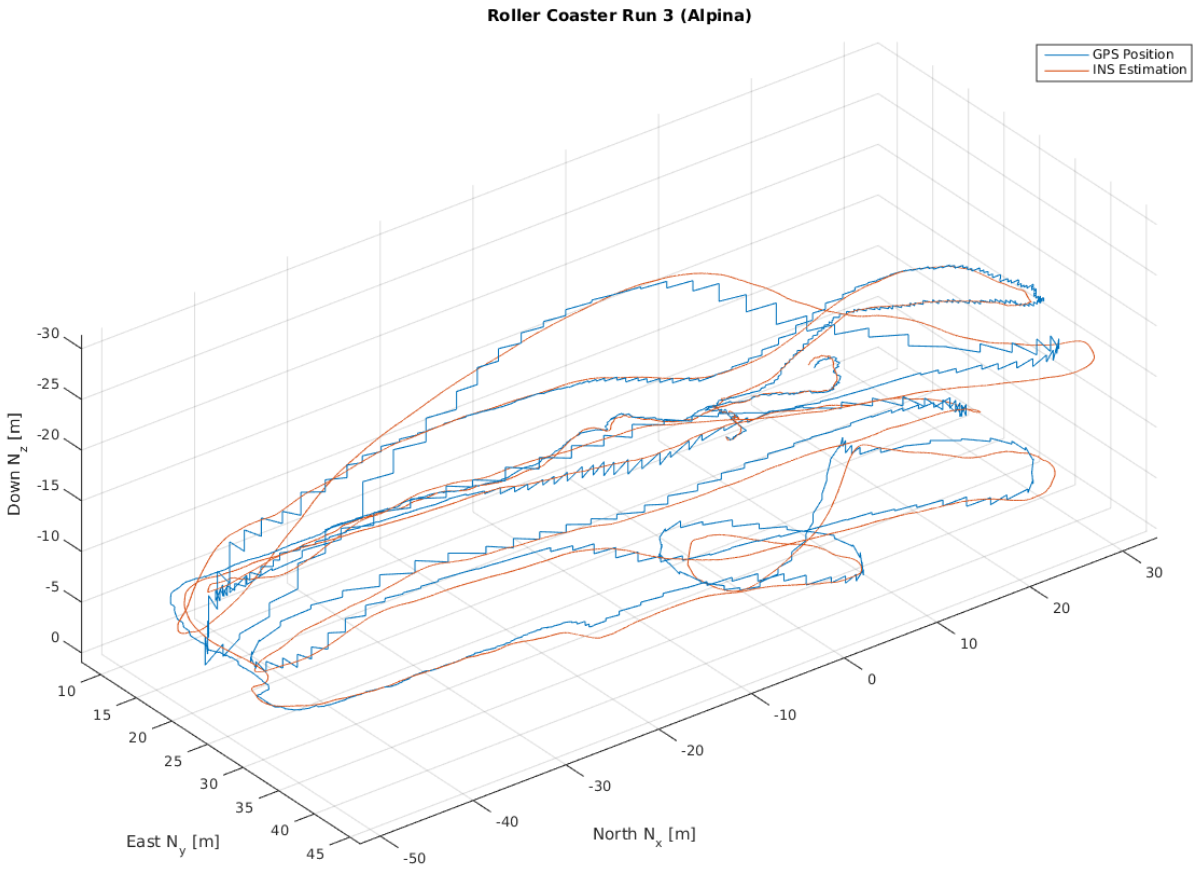

Figure 7-32: Trajectory 3D plot – Alpina Bahn.

As it can be seen in the z-axis, the height of the rollercoaster is bigger and there are more turns. Even there are some points where the reception is lost, but it is still a quite representative 3D plot.

To assess the information shown with the 3D plot, the velocity norm and the measured acceleration is plotted:

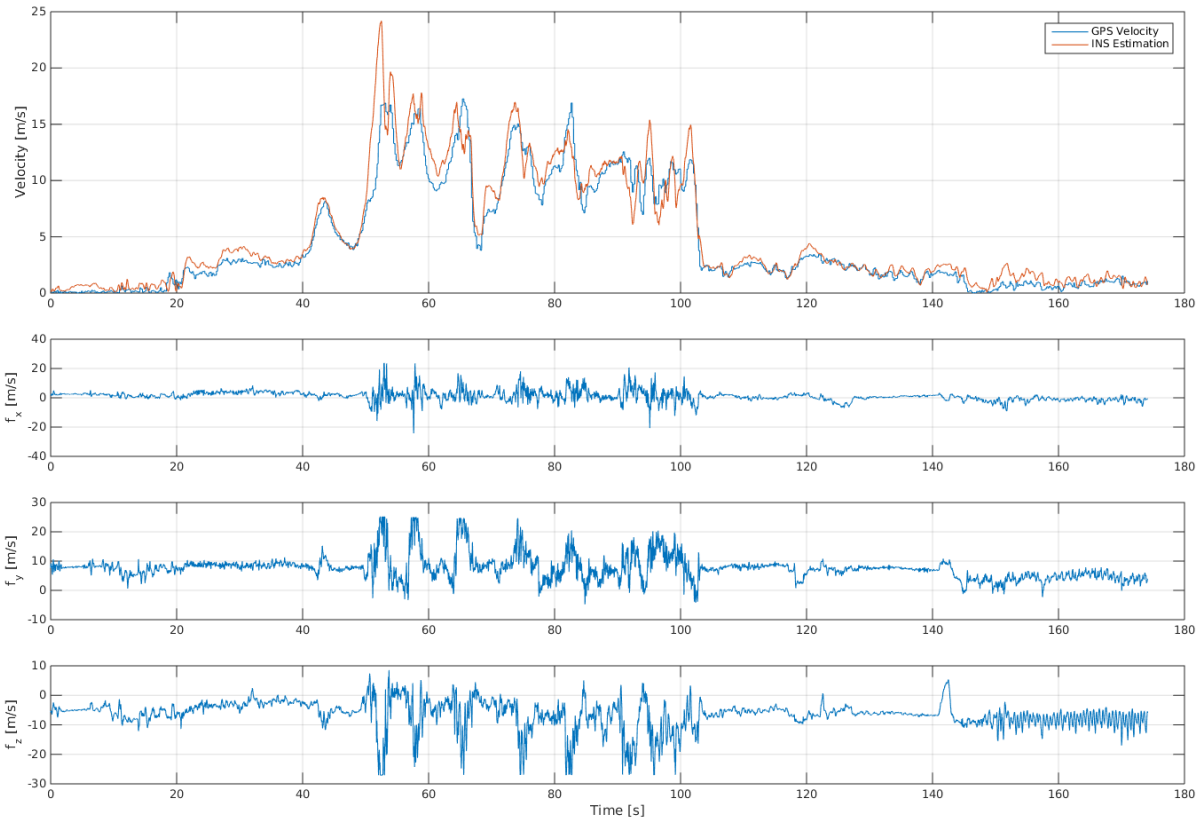

Figure 7-33: Norm of the velocity and measured acceleration components – Alpina Bahn.

As it can be observed, the maximum velocities and the accelerations are bigger in this case than the previous section 7.2.2. The acceleration is really huge in this case and the velocities reach the approximate value of 20 m/s, while the Wilde Maus was poorly reaching the 10 m/s. That is what makes really interesting to test the algorithms with this data that cannot be recorded with the quadrotor propellers.

Normally, the attitude data is straightforward to be brought into extreme cases as the quadrotor can be piloted by means of attitude angles as inputs of the controller algorithm instead of rotational rates, which are commonly limited to certain maximum values of input per time period. But for the velocity data is difficult to achieve extreme data with the propellers of the quadrotor with maximum velocity of 15 m/s and maximum thrust of 20 N, but these are only the theoretical maximum values which are quite impossible to achieve in real flights, because it depends on the take-off weight, for example.

It is really spectacular to see how the velocity until values of 15 m/s separately in each of the axes, milestone that could not have been achieved with the power of the quadrotor propellers.

The display of the overall estimated trajectory is also remarkable, but only the velocity is added as no new deduction is clearly concluded from the attitude plots:

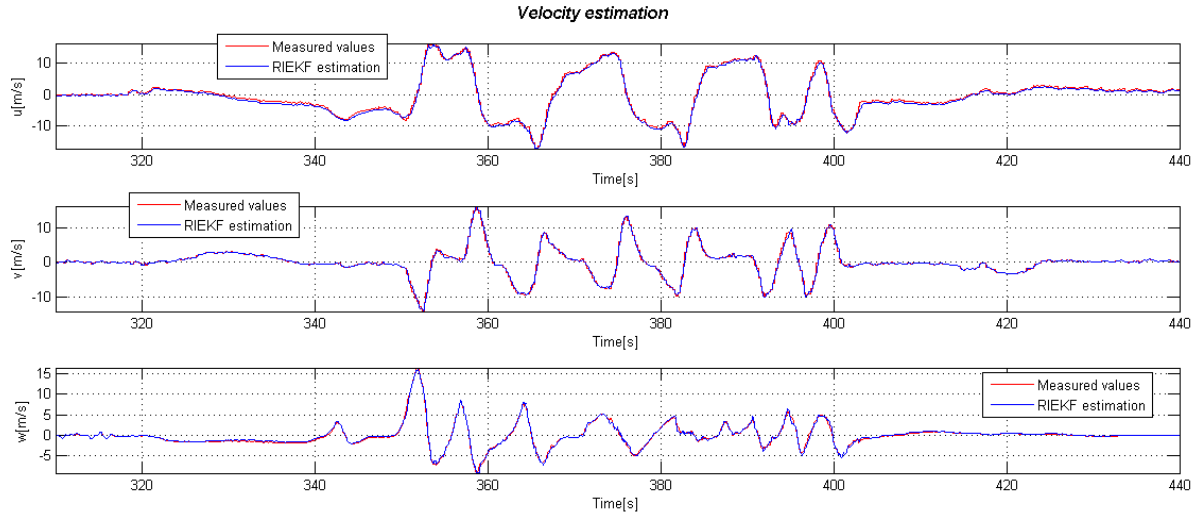

Figure 7-34: RIEKF velocity estimation – Alpina Bahn.

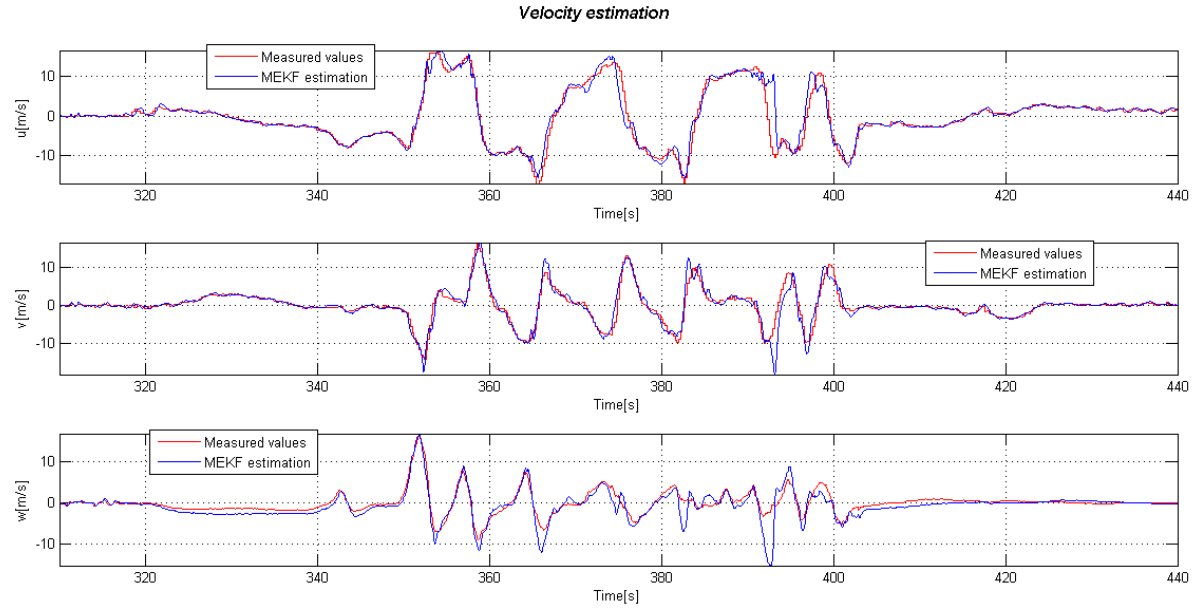

Figure 7-35: MEKF velocity estimation – Alpina Bahn.

The comparison only between the KFs is really interesting. The RIEKF estimates the velocity really close to the measure values. The MEKF is not that perfect but seems to be noisier. The reason why the MEKF seems more affected by the perturbations is that the matrix $\boldsymbol{Q}$ is used to pay more attention to the input values but it also adds more noise to the estimates, while the advantage of the RIEKF is that the symmetries of the filter preserves the estimations of being affected by the noise, which brings the possibility of increasing the matrix $\boldsymbol{M}$ (similar repercussion of $\boldsymbol{Q}$) without adding more noise but improving the accuracy of the estimations.

8 Results and conclusions

First of all, it is important to highlight that the considerations developed in here are based on the results extracted in the previous chapter 7 and in each of the previous chapters analysing each of the INAs separately, chapters 4, 5 and 6.

The first remarkable conclusion is that the navigation filter initialization has to be brought to fruition in a proper manner for the INAs to be efficient. That means that a step of calibration at the beginning must be performed. For bigger UAVs as the Predator, the assumption of Flat Earth is not applied and special and more important initializations of the filter have to be performed, see [2] for further details.

As it is concluded in section 7.2.1, the results bring us to the conclusion that there is no evidence of any of the algorithms being more accurate than the others. It can be inferred that the symmetry-preserving feature of the evaluated observer and RIEKF really improves the estimations in some situations, but it is not that clear the improvement when the perturbations introduced in our system are unknown.

Moreover, the basics of the symmetry-preserving theory are that the system and the way the inputs are introduced to the system preserve the invariance property of both the system and the algorithm, which is designed based on the system. So the system is the part of the procedure that should be treated more carefully during the design of the observer. My deduction is that in the real-time data evaluation, the variables' noise and perturbations that are added to the sensors are totally unknown and this brings us to the point that the system consideration for preserving the invariance property of the system can be broken, as only Gaussian noise is supposed when designing the algorithms. Even though, this conclusion is just my point of view contrasting the results of the true data and the real-time data previously displayed. Moreover, the estimations with real-time data of the symmetry-preserving INAs are really accurate and do not seem to be incorrect at all. This is just a conclusion extracted from the comparison with the MEKF, which is our reference of algorithm without symmetry-preserving properties.

As it can be observed at the end of section 7.2.3, the symmetry-preserving RIEKF seems to be really accurate and improve the estimations of the velocity compared with the MEKF estimations. This can be absolutely applied to the invariance property, but this same property is not enhancing the attitude estimation in the real-time data cases.

Regarding the design and creation of these algorithms, it is clear that the usage of the MEKF is the most straightforward case. Its design is easy and the tune of its parameters is clear as well. This should be similar with the RIEKF, but the usage of the time-continuous implementation and the special design preserving the invariance of the system of this algorithm makes the tuneable values to impact on the results in a different manner. Finally the observer is clearly the most difficult algorithm to design but the easiest one to tune, and taking a look at all the evaluated trajectories seems to be the most accurate algorithm in a general case, i.e. the most accurate one for a wider range of trajectories but not the best for specific cases.

Other interesting further developments and research with this topic must be the implementation of the RIEKF algorithm in a time-discrete manner, which seems to be more efficient in terms of computation time. Also can be interesting to design a desired INA

following the pattern to design symmetry-preserving algorithms and reach the same properties as the authors with different types of selected INA, i.e. symmetry-preserving algorithms estimating more variables. The paramount point in these algorithms is the way how they are designed and for this reason, it is fundamental to follow the design steps to be sure the property of invariance is maintained. This is the base of the symmetry-preserving theory and it is the property which gives the improvements on the estimations. If this property is not accounted the advantages of this not straightforward design are useless.

9 Acknowledgements

I hereby convey my gratefulness to the Institute of Flight System Dynamics of the Technische Universität München for giving me the opportunity of developing my bachelor thesis with them; it was an unforgettable experience.

I would like to express my gratitude to everyone who contributed to my work and supported me throughout the time I spent working at the Institute of Flight System Dynamics. I am really thankful for their invaluable advice and inspiring guidance during the project. I am sincerely grateful to all of them for spending a little bit of their time to support me on certain parts of this project.

Firstly, I would like to express my deepest thanks to Dipl.-Ing. Thomas Raffler, who has been my supervisor in this Bachelor thesis and has provided me with a really interesting topic to work on, valuable advice, guidance and materials throughout all the work, without him this would not have been possible. It is also important for me to thank him for giving me the opportunity to test the UAV in a real rollercoaster, which was an incredible experience and a really valuable and interesting opportunity to record new amazing data.

Secondly, I express my warm thanks to M. Sc. Venkata Sravan Akkinapalli for helping me and discussing with me concepts of paramount importance on this topic and Mr. Yashvin Beni for his motivation, help and contribution with his French language skills, which was a key point for me to be able to continue. I would also like to thank Mr. Sanchito Banerjee for his unconditional support.

I am also using the opportunity to thank Dr.-Ing. Silvère Bonnabel for answering related questions on his papers, which were the base of my topic, and for giving me some additional fundamental related papers.

Finally, this project would not have been possible without the support of my parents and family. For this, I specially thank them.

10 References

[1] O. J. Woodman, „An introduction to inertial navigation," University of Cambridge, Cambridge, United Kingdom, August 2007.

[2] J. Dambeck, *Navigation und Datenfusion Seminar,* München: Technische Universität München, 2009.

[3] E. Salaün, *Algorithmes de Filtrage et systemes avioniques pour vehicules aeriens autonomes,* Paris, Januray 2009.

[4] „Reference Frame Definitions (GPS)," [Online]. Available: http://www.google.de/imgres?imgurl=http%3A%2F%2Fwhat-when-how.com%2Fwp-content%2Fuploads%2F2012%2F02%2Ftmp2046_thumb.jpg&imgrefurl=http%3A%2F%2Fwhat-when-how.com%2Fgps-with-high-rate-sensors%2Freference-frame-definitions-gps%2F&h=415&w=557&tbnid=wpp8EoGDv. [Zugriff am 15 September 2014].

[5] T. Raffler, *Entwicklung eines integrierten Navigations-systems für ein Multirotor UAV,* München: Technische Universität München, August 2010.

[6] „Understanding Euler Angles," CH Robotics, [Online]. Available: http://www.google.de/imgres?imgurl=http%3A%2F%2Fwww.chrobotics.com%2Fwp-content%2Fuploads%2F2012%2F11%2FInertial-Frame.png&imgrefurl=http%3A%2F%2Fwww.chrobotics.com%2Flibrary%2Funderstanding-euler-angles&h=759&w=1186&tbnid=AwxFZY9kMLb4uM%3A&zoom=1&docid=g. [Zugriff am 15 September 2014].

[7] „Moving coordinate system," Wikipedia, [Online]. Available: http://www.google.de/imgres?imgurl=http%3A%2F%2Fupload.wikimedia.org%2Fwikipedia%2Fcommons%2F4%2F49%2FMoving_coordinate_system.PNG&imgrefurl=http%3A%2F%2Fen.wikipedia.org%2Fwiki%2FFictitious_force&h=668&w=864&tbnid=YzneCcn5Kzhe8M%3A&zoom=1&docid=ghqIkDMdq. [Zugriff am 15 September 2014].

[8] J. Diebel, „Representing Attitude: Euler Angles, Unit Quaternions and Rotation Vectors," pp. 1-35, 20 October 2006.

[9] E. W. Weisstein, „Mapping," Wolfram MathWorld, [Online]. Available: http://mathworld.wolfram.com/Map.html. [Zugriff am 13 September 2014].

[10] T. Rowland und E. W. Weisstein, „Group," Wolfram MathWorld, [Online]. Available: http://mathworld.wolfram.com/Group.html. [Zugriff am 14 September 2014].

[11] T. Rowland, „Lie Group," Wolfram MathWorld, [Online]. Available: http://mathworld.wolfram.com/LieGroup.html. [Zugriff am 14 September 2014].

[12] E. W. Weisstein, „Category,“ Wolfram MathWorld, [Online]. Available: http://mathworld.wolfram.com/Category.html. [Zugriff am 14 September 2014].

[13] E. W. Weisstein, „Morphism,“ Wolfram MathWorld, [Online]. Available: http://mathworld.wolfram.com/Morphism.html. [Zugriff am 2013 September 2014].

[14] J. Lipp und E. W. Weisstein, „Topological Space,“ Wolfram MathWorld, [Online]. Available: http://mathworld.wolfram.com/TopologicalSpace.html. [Zugriff am 14 September 2014].

[15] T. Rowland, „Manifold,“ Wolfram Mathworld, [Online]. Available: http://mathworld.wolfram.com/Manifold.html. [Zugriff am 14 September 2014].

[16] J. W. Robbin und D. A. Salamon, 5 September 2013. [Online]. Available: http://www.math.ethz.ch/~salamon/PREPRINTS/diffgeo.pdf. [Zugriff am 13 September 2014].

[17] E. Meinrenken, Fall 2010. [Online]. Available: http://www.math.toronto.edu/mein/teaching/lie.pdf. [Zugriff am 13 September 2014].

[18] S. Bonnabel, P. Martin und P. Rouchon, „Symmetry- Preserving Observers,“ in *IEEE TRANSACTIONS ON AUTOMATIC CONTROL, VOL. 53, NO. 11*, Paris, December 2008.

[19] „State estimators with Kalman Filter,“ [Online]. Available: http://home.hit.no/~hansha/documents/control/theory/stateestimation_with_kalmanfilter.pdf. [Zugriff am 18 September 2014].

[20] „State estimation with observers,“ [Online]. Available: http://home.hit.no/~hansha/documents/control/theory/stateestimation_with_observers.pdf. [Zugriff am 18 September 2014].

[21] G. A. Terejanu, „Department of Computer Science and Engineering, University of Buffalo, Buffalo, NY,“ [Online]. Available: http://www.cse.sc.edu/~terejanu/files/tutorialKF.pdf. [Zugriff am 29 August 2014].

[22] A. Barrau und S. Bonnabel, „Intrinsic filtering on Lie groups with applications to attitude estimation,“ pp. 1-21, 15 September 2014.

[23] S. Bonnabel, "Left-invariant extended Kalman filter and attitude estimation," in *Proceedings of the 46th IEEE Conference on Decision and Control*, New Orleans, LA, USA, Dec. 12-14, 2007.

[24] S. Bonnabel, P. Martin und E. Salaün, „Invariant Extended Kalman Filter: theory and application to a velocity-aided attitude estimation problem,“ in *Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference#*, Shangai, P.R. China, December 16-18, 2009.

[25] M. S. Grewal und A. P. Andrews, Kalman Filtering, theory and pratcie using MATLAB,

New Jersey: John Wiley & Sons, Inc., 2008.

[26] „Flight System Dynamics (FSD),“ Technische Universität München (TUM) - Institute of Flight System Dynamics (FSD), [Online]. Available: http://www.fsd.mw.tum.de/infrastructure/unmanned-systems/. [Zugriff am 14 10 2014].

[27] „Wilde Maus,“ [Online]. Available: http://www.muenchs-wildemaus.de/html/technik-einzelanlage.html. [Zugriff am 14 10 2014].

[28] „Alpina Bahn,“ [Online]. Available: http://www.alpina-bahn.de/technische_daten.html. [Zugriff am 14 10 2014].