DISSERTATION

Master in Electrical and Electronic Engineering

Motion-based Remote Control Device for Interaction with Multimedia Content

Miguel António Cunha Rasteiro

Leiria, September of 2015

DISSERTATION

Master in Electrical and Electronic Engineering

Motion-based Remote Control Device for Interaction with Multimedia Content

Miguel António Cunha Rasteiro

Master dissertation performed under the joint supervision of Doctor Hugo Filipe Costelha de Castro and co-orientation of Master Luís Manuel Conde Bento, Professors at the School of Technology and Management of the Polytechnic Institute of Leiria.

Leiria, September of 2015

This page was intentionally left blank.

Try not to become a man of success, but rather try to become a man of value.

(Albert Einstein)

This page was intentionally left blank.

Acknowledgements

The realization of this work would not be possible without the contribute of a group of people who, besides their friendship, were always there when new challenges arose in work and in life.

To my supervisor Dr. Hugo Costelha, who trusted me throughout the years, I am thankful for his guidance and friendship. To my co-supervisor, Pr. Luís Bento, for all his help, contributions and incentives during the developing of this work and, to Dr. Pedro Assunção, who gave great guidance and quality contributions as main adviser for the project I was involved. To them my sincere gratitude.

To the Tech4Home team, specially to Márcio Barata and Bruno Ribeiro, without whom this project would not have been possible, particularly for their support in the last phase.

To all colleagues from IEEE IPLeiria Student Branch, from CEEE and the Robotics Club of ESTG. I am proud to have had the opportunity to be part of these groups.

To my family, for all dedication, support and faith in me. A special kiss to my grandmother.

Finally but not the least, to my friend and co-worker Ricardo Santos for his patience and good mood. To my colleagues, João Santos and André Guarda, for all the support, and to my dear friends, Tânia Simões, Bruno Carramate and Catarina Santos, thank you for all the fantastic distracting moments even if only for a few hours here and there.

Miguel Rasteiro

This work is co-financed by European Union, COMPETE, QREN and Fundo Europeu de Desenvolvimento Regional (FEDER), Project HERMES, co-promotion n.º 34149.

This page was intentionally left blank.

Resumo

Esta dissertação descreve o desenvolvimento e implementação de técnicas para melhorar a precisão de filtros de baixa complexidade, adequados para dispositivos de controlo remoto utlizados na eletrónica de consumo. A evolução verificada nos últimos anos em conteúdos multimédia disponíveis para os consumidores de *smart TVs* e *set-top-boxes*, não tem despertado o interesse esperado por parte dos utilizadores, e uma das razões apontadas para esta constatação é o interface de utilização. Embora a maioria dos dispositivos apontadores atuais utilizem navegação relativa, a navegação absoluta pode permitir uma utilização mais intuitiva e interativa. Esta é uma possibilidade explorada neste trabalho, bem como a interação com conteúdos multimédia através de gestos.

Os algoritmos clássicos de fusão são por norma computacionalmente intensivos, limitando a sua aplicação em dispositivos de baixo consumo de energia. Para resolver este problema, foi desenvolvido um estudo comparitivo de um conjunto relevante de unidades de alto desempenho para uso profissional, com os filtros de baixa complexidade desenvolvidos, utilizando sensores Magnéticos, Gravíticos e de Velocidade Angular (MARG). Foram realizados testes de avaliação de desempenho em um *setup* com condições adversas, de modo a observar a resposta dos algoritmos num ambiente não-trivial. Os resultados demonstram que a implementação de filtros de baixa complexidade, utilizando sensores de baixo custo, podem fornecer uma precisão aceitável em comparação com as unidades profissionais mais complexas. Estes resultados abrem caminho para a adoção mais rápida de dispositivos apontadores absolutos à base de orientação, em aplicações multimédia interactivas.

Palavras-chave: Controlos Remoto, Filtros Complementares, Filtros de Kalman, Eletrónica de Consumo, Sensores MARG, Unidades de Medição Inercial

This page was intentionally left blank.

Abstract

This dissertation describes the development and implementation of techniques to enhance the accuracy of low-complexity filters, making them suitable for remote control devices in consumer electronics. The evolution verified in the last years, on multimedia contents, available for consumers in Smart TVs and set-top-boxes, is not raising the expected interest from users, and one of the pointed reasons for this finding is the user interface. Although most current pointing devices rely on relative rotation increments, absolute orientation allows for a more intuitive use and interaction. This possibility is explored in this work as well as the interaction with multimedia contents through gestures.

Classical accurate fusion algorithms are computationally intensive, therefore their implementation in low-energy consumption devices is a challenging task. To tackle this problem, a performance study was carried, comparing a relevant set of professional commercial of-the-shelf units, with the developed low-complexity filters in state-of-the-art Magnetic, Angular Rate, Gravity (MARG) sensors. Part of the performance evaluation tests are carried out under harsh conditions to observe the algorithms response in a non-trivial environment. The results demonstrate that the implementation of low-complexity filters using low-cost sensors, can provide an acceptable accuracy in comparison with the more complex units/filters. These results pave the way for faster adoption of absolute orientation-based pointing devices in interactive multimedia applications, which includes hand-held, battery-operated devices.

Keywords: Remote Control Devices, Complementary Filters, Kalman Filters, Consumer Electronics, MARG sensors, Inertial Measurement Units

This page was intentionally left blank.

List of Figures

List of Tables

This page was intentionally left blank.

List of Acronyms

2D	2 Dimensions
3D	3 Dimensions
AGMdF	Adaptive-Gain Madgwick Filter
AGMhF	Adaptive-Gain Mahony Filter
AHRS	Attitude Heading Reference System
AMhF	Adapted Mahony Filter
API	Application Programming Interface
ASIC	Application Specific Integrated Circuit
BPF	Band Pass Filter
CF	Complementary Filters
DC	Direct Current
DCM	Direction Cosine Matrix
DMP	Digital Motion Processor
DoF	Degrees of Freedom
DSP	Digital Signal Processor
DZ	Dead Zone
EKF	Extended Kalman Filter
EvB	Evaluation Board
FOAM	Fast Optimal Attitude Matrix
FQA	Factored Quaternion Algorithm

GDA Gradient Descendent Algorithm

GNA Gauss Newton Algorithm

GPS Global Positioning System

HID Human Interface Device

HU Hardware Units

I2C Inter-Integrated Circuit

IC Integrated Circuit

IMU Inertial Measurement Unit

KF Kalman Filters

LMS Least Mean Square

LPF Low Pass Filter

MARG Magnetic, Angular Rate, Gravity

MEMS Microelectromechanical System

NED North, East, Down

NVM Non-Volatile Memory

OMdF Original Madgwick Filter

PCB Printed Circuit Board

PI Proportional-Integral

QoE Quality of Experience

QUEST Quaternion Estimator

RCD Remote Control Devices

RF Radio Frequency

RF4CE Radio Frequency for Consumer Electronics

RMSE Root-Mean-Square Error

RPY Roll Pitch Yaw

SME	Small Medium Enterprise
SPI	Serial Peripheral Interface
STB	Set-Top-Box
Std	Standard deviation
SoC	System on Chip
TRIAD	Tri-axial Attitude Determination
TV	Television
UIA	User Interface Application
UKF	Unscented Kalman Filter
USB	Universal Serial Bus
YDCF	Yaw Drift Compensation Filter

This page was intentionally left blank.

List of Symbols

Cartesian Coordinate Frames

^{E}	Earth Frame (Reference Frame)
^{S}	Sensor Frame (Body Frame)

Scalars

k_p	Mahony filter cutoff frequency gain (Proportional gain)
k_i	Mahony gyroscope bias drift control gain (Integral gain)
n	An integer positive number
q_s	Quaternion scalar component
q_x, q_y, q_z	Quaternion vector components
$_t$	Step
α, α_2, γ	Algorithm gains
β	Cutoff frequency gain for the Madgwick filter
δt	Sampling time
ϵ_φ	Yaw error for the YDCF
ϵ_ψ	Yaw error
ζ	Gyroscope bias drift control gain for the Madgwick filter
κ	Threshold value for the observation sensors reliability
μ	Convergence rate gain
ϕ	Roll angle
φ	Heading angle for the YDCF
$\psi_{\boldsymbol{m}}$	Heading angle
$\psi_{\boldsymbol{q}}$	Yaw angle from the estimated quaternion

Vectors

$\boldsymbol{a}$	Measured acceleration
$\boldsymbol{a}_{true}$	True linear acceleration
$\boldsymbol{a}_{linear}$	Linear acceleration
$\boldsymbol{b}$	Ideal Earth's magnetic field
$\boldsymbol{d}$	Earth's generic field
$\boldsymbol{f}()$	Cost function
$\boldsymbol{g}$	Gravity acceleration
$\boldsymbol{h}$	Earth's magnetic field from orientation estimate ($^{E}\boldsymbol{m}$)
$\boldsymbol{l}$	A generic Cartesian vector
$\boldsymbol{m}$	Measured magnetic field
$\boldsymbol{m}_{true}$	Earth's magnetic field
$\boldsymbol{n}$	Uncorrelated white Gaussian measurement noise
$\boldsymbol{p}$	Position from accelerometer integration
$\boldsymbol{q}$	Orientation quaternion
$\boldsymbol{q}^{*}$	Quaternion conjugate
$\boldsymbol{q}'$	Drift corrected quaternion
$\boldsymbol{r}$	Generic sensor measurement for the orientation filters
$\boldsymbol{s}$	Velocity from the position filter
$\boldsymbol{u}$	Control vector for Kalman filter
$\boldsymbol{v}$	Measurement noise for Kalman filter
$\boldsymbol{x}$	State vector for Kalman filter
$\boldsymbol{w}$	Process noise for Kalman filter
$\boldsymbol{z}$	Measurement vector for Kalman filter
$\boldsymbol{\rho}$	Constant offset
$\boldsymbol{\sigma_s}$	Variance of the observation sensors
$\boldsymbol{\varphi}$	Innovation vector for the Mahony filter
$\boldsymbol{\omega}$	Measured angular velocity
$\boldsymbol{\omega}_{true}$	True angular velocity
$\boldsymbol{\omega}_{\epsilon}$	Angular error

Matrices

$\boldsymbol{A}$	State transition model for the Kalman filter
$\boldsymbol{B}$	Control-input model for the Kalman filter
$\boldsymbol{F}$	A 3x3 calibration matrix
$\boldsymbol{H}$	Observation model matrix for the Kalman filter
$\boldsymbol{J}()$	Jacobian
$\boldsymbol{K}$	Kalman gain matrix
$\boldsymbol{M}$	A 3x3 axes misalignment and scale factors matrix
$\boldsymbol{P}$	Error covariance matrix
$\boldsymbol{Q}$	Process noise covariance matrix for the Kalman filter
$\boldsymbol{R}$	Measurement noise covariance matrix for the Kalman filter
$\boldsymbol{S}$	*Soft iron* distortions

Other symbols

$\otimes$	Quaternion product
$\times$	Cross product
$\hat{}$	Normalized quantity
$\tilde{}$	Estimated quantity
$\dot{}$	Derivative
∇	Gradient
$\lvert\ \rvert$	Modulus
$\lVert\ \rVert$	Norm
^{T}	Transpose
$_{t\lvert t-1}$	The *a priori* estimate
$\boldsymbol{T}$	A generic coordinate frame transformation operation

This page was intentionally left blank.

Contents

This page was intentionally left blank.

Chapter 1

Introduction

In recent years there has been a huge evolution in user interaction with media content, and an increasing convergence between interactive Television (TV) and multimedia personal devices such as computers, tablets, and smartphones. However, content navigation devices for Smart TVs, Set-Top-Box (STB) or Media Centers did not follow this evolution. While the existing remote controls with air-mouse functionalities are becoming quite common [1], their use is not always intuitive or comfortable, leading to poor Quality of Experience (QoE) [2]. Usually, these devices are based on inertial movements, which are prone to drift over time and less intuitive, because the navigation is based on the remote control local coordinate frame, therefore requiring frequent position resetting (calibration). It is expected that an absolute navigation functionality, is able to increase the QoE in interactive multimedia. For that, the Remote Control Devices (RCD) should be capable of autonomously compute their absolute orientation in a global coordinate frame with good accuracy and low calibration frequency.

The aim of this work is to develop an RCD with 6 Degrees of Freedom (DoF) (position and orientation) to enable new forms of interaction with 3 Dimensions (3D) content, and to improve the user experience. Specifically, to develop a remote control with 3D tracking motion capabilities beyond the current air-mouse remote controls. One of the challenges is to balance the requirements of novel remote control functionalities against the need for it to be a low cost device with low energy consumption. To achieve absolute navigation, the remote control needs to incorporate a set of sensors commonly known as Magnetic, Angular Rate, Gravity (MARG), comprising a magnetometer, a gyroscope and an accelerometer with 3 orthogonal axes each. Also, optical flow sensors can be combined to aid in the pose determination and this topic is explored in this dissertation.

Recently, MARG units were made available, encapsulating all three sensors types into

a single Integrated Circuit (IC), often referred by the industry as 9 DoF sensor units[1,2]. This integration allows reduced sensor noise, lower probability of axis misalignment, lower cost and lower size, which is particularly relevant in the design of an RCD. Optical flow sensors are being used in optical mouses for decades, they are a very mature, low cost and low power consumption technology, which, similarly to the MARG units, makes them suitable to use in this work.

The major challenge using these sensors lies in the many application-specific disturbances they are subjected to: gyroscopes are prone to drift over time; accelerometers do not exclusively measure the gravity acceleration during regular operation of the RCD; magnetometers are subjected to unpredictable disturbances in these application scenario, i.e. indoors; optical flow sensors are quite sensitive to illumination conditions. These disturbances induce error in the system and bias the orientation estimate. Therefore, high computational filters are usually used to compute the orientation and position estimates. That may prevent the use of MARG units in low cost wireless battery operated devices, due to power consumption, cost and latency. As such, determining if low complexity filters can achieve an acceptable level of accuracy for interactive multimedia applications is one of the tasks addressed in this dissertation.

1.1 Goals and contributions

This work shall initially present a study on the state-of-the-art for current motion-based remote controls and for fusion filters. Afterwards the following topics will be developed:

- Characterization of the application-specific sensor disturbances and solutions to mitigate their effects;
- Study and implementation of existent relevant low complexity filters for orientation and position estimate and their bottlenecks;
- Improvements to the filters by adapting them to the specific problem;
- Objective evaluation of the performance of the implemented filters;
- Subjective evaluation of the performance and implementation of algorithms in a RCD, based on the QoE.

[1] www.invensense.com

[2] www.st.com

1.1.1 Publications

The following publications were produced during the development of this work:

- M. Rasteiro, H. Costelha, L. Bento, and P. Assunção, "Low-complexity MARG Algorithms for Increased Accuracy in Space Pointing Devices", in proceedings of the *IEEE CE Workshop, 2015*, Novi Sad, Serbia, March 2015;
- M. Rasteiro, H. Costelha, L. Bento, and P. Assunção, "Accuracy versus Complexity of MARG-based Filters for Remote Control Pointing Devices", in proceedings of the *Consumer Electronics-Taiwan (ICCE-TW), 2015 IEEE International Conference on*, Taipei, Taiwan, June 2015, pp 51-52;
- R. Santos, M. Rasteiro, H. Costelha, L. Bento, and P. Assuncao, "Motion-based Remote Control Device for Enhanced Interaction with 3D Multimedia Content", in proceedings of the *Conference on Telecommunications (Conftele 2015)*, Aveiro, Portugal, September 2015

1.2 Dissertation structure

This dissertation is organized in six chapters. This first chapter addresses an overall description of the dissertation and objectives. The second chapter describes an overview of the most relevant state-of-the-art RCD available on the market for consoles and TV/STB sets, and reviews the state-of-the-art algorithms used for pose estimation in related scenarios. The third chapter includes a characterization of the sensors used, as well as a characterization of the physical quantities they measure. It also introduces the essential related background, namely, quaternions representation, low complexity filters for sensor fusing, and position estimate from accelerometers. The fourth chapter describes the compensation techniques implemented in the filters in order to detect and compensate disturbances in real scenarios. The fifth chapter presents the result of the integration of the proposed methods in an RCD and experimental results. Finally, in chapter six, some conclusions are drawn and future work is discussed.

This page was intentionally left blank.

Chapter 2

Related State-of-the-art

Motion capture and pose tracking is a process of great interest in the current days. It can be used in military, sports, medical care, industrial applications, robotics, entertainment and so on [3, 4, 5]. The objective of this Chapter is to present a review of the state-of-the-art of the existing RCD, available on the market for consoles and TV/STB sets, together with some of the implemented algorithms for pose estimation. It starts with a review of the most recent advances in the technology applied in consumer electronics goods in the first Section, then, in the second Section, the state-of-the-art of the filters used to determine the pose of such devices is detailed.

2.1 Remote control devices for consumer electronics

Motion tracking devices became popular in the consumer world with the Nintendo Wii Remote in 2006. It constituted a revolution in the gaming industry, offering the possibility of gestures identification and motion tracking to gamers, starting an all new kind of interaction with consoles. This evolution sparked the interest of consumers, in contrast to the evolution in TVs and STBs, where new multimedia contents and possibilities did not cause the same effect, with the majority of people ignoring it. According to [2], the main reason for low market penetration is the user interface, therefore efforts have been made to improve the QoE for users.

The game controllers here reviewed rely on external beacons or sensors in order to allow the position tracking of the RCD. For multimedia contents navigation, the absolute position is not as relevant as the absolute orientation. Nevertheless, in order to increase the functionalities of the remote control, that feature should be available, even if not within the same levels of precision as in game controllers. In the last years, there has been an huge growth in Android-based STB, therefore, an RCD that allows gaming and the use of remote sensors on these equipments has an advantage. Some of the technology

used in the game controllers can been applied in the RCD to do so. Some examples of game controllers and TV remote controls are described in the next two Sections.

2.1.1 Game controllers

The three game controllers more relevant in the context of this research are:

Nintendo Wii Remote

Although Nintendo have not officially released the specifications for its remote, the global hacking community[1] has taken great interest in this product and reverse-engineered much of it [6]. Originally it is equipped with a 3 axes accelerometer, the ADXL330 from Analog Devices, which allowed to get the tilt of the remote and the amount of force applied to it in respect to gravity. This way, it could capture and describe some of the human motion when using the remote. It also includes an infra-red camera sensor that mapps a maximum of four infra-red light sources (*sensor bar*). This made it possible to find the Wii Remote position by triangulation, with the *sensor bar* as a reference point, and a full orientation estimate of the remote. On 2009 the Wii Remote Plus was released as an extension for the Wii Remote, adding two gyroscopes: a 2 axes IDG-600 and a 1 axis gyroscope from EPSON TOYOCOM, allowing the measure of the rate of rotation of the remote control, further increasing the precision of the orientation tracking.

Play Station Move

After Nintendo, many game developers expanded their user interface to include motion tracking, including Sony with its launch of the PlayStation Move in 2009. These controllers have also taken interest from the hacking community[2]: they possess a 3 axes accelerometer (Kionix KXSC4 10227 2410), a two axes gyroscope (STM LPR425AL) and a single axis one (Y5250H 2029 K8QEZ), a temperature sensor and a 3 axes magnetometer (AKM AK8974). For the PlayStation Move system to become fully functional, it is needed to be used in combination with the PlayStation 3 Eye, a 2.0 USB camera that allows to track the position of the controller. The camera had been released previously in 2007, and could be bought independently from the remote controller.

Razer Hydra

The Razer Hydra[3] is not a popular game controller as the ones mentioned previously, but it is very interesting product from an engineer point of view. The motion

[1] Available on: www.wiibrew.org

[2] *Repourposing the PS3 Move* by K. Sebesta; Available on: www.eissq.com/ps3_move

[3] Product details available on: www.razerzone.com

tracking is executed by reading a weak magnetic field, generated by its base station, which allows to detect the absolute position and orientation of the controllers with a precision of 1 millimetre and 1 degree, respectively, when closer than 0.5 m from the base station. However, the tracking capabilities become unstable as the remote device moves away from the base, limiting the available working space [7].

2.1.2 STB and TV remote controls

A relevant set of STB and TV remote controls are presented here:

RC11 2.4G Wireless Keyboard + Air Mouse

RC11[4] is a generic input device specially developed for Android-based systems, it was developed by MEASY and it includes air mouse functionalities and a keyboard. It operates over a 2.4 GHz Radio Frequency (RF) technology with a 120 Hz update rate. The air mouse functionality is based on a 3 axes gyroscope, which, by our test, leads to a non intuitive experience since it does not compensate for the device orientation while using it. Hand movements tend to change the orientation of the remote in wider movements leading to the cursor behaves in a non intuitive way.

Samsung Smart Control

In 2014 Samsung released Smart Control for its SmartTVs and Hubs. The navigation using this device can be done either by directional keys, a small touchpad or through motion. It possesses a 3 axes gyroscope for motion navigation[5], so, similaly to the RC11 it is a 3 DoF solution.

LG Magic Motion Remote Control

This LG remote control provides simple gestures to control the volume and channel selection. It incorporates Hillcrest Labs' Freespace technology for motion navigation[6]. This solution compensates for the remote device orientation, it has a 3 axes gyroscope and a 3 axes accelerometer - a 6 DoF solution. It also compensates for natural hand tremors with an adaptive tremor removal algorithm.

Movea Air-mouse

Movea was acquired by Invensense in 2014, merging with one of the fastest growing Microelectromechanical System (MEMS) and System on Chip (SoC) company of the world[7]. Movea, much like Hillcrest, is specialized the developing processing

[4]RC11 MEASY Product. Available on: www.measy.cn

[5]Samsung E-MANUAL (pp.28-37). Available on: www.appliancesonline.com.au

[6]Article "*Hillcrest labs tapped by LG to bring motion control to first 3DTV internet connected LED LCD HDTV*". Available on: www.hillcrestlabs.com

[7]From: www.uk.reuters.com

algorithms and data fusion models that make available accurate data for end product developers. The company has developed an air-mouse compatible with Android, with the capacity of identifying various gestures, with tilt compensated navigation, which can be used for Android games[8]. This solution is one of the best reviewed.

2.2 Pose determination from inertial sensing

All the solutions presented in the last Section have a set of sensors that allow a certain level of determination of the pose of the device and, in some cases, gestures identification. To do so, the sensors data has to be fused through an algorithm, usually referred as a filter, in order to obtain the expected output.

The pose of a rigid body can be defined as a set of parameters that establish the position and orientation of the body frame relatively to a fixed reference frame. There are numerous systems used to accurately estimate the pose of a rigid body using a combination of sensors or markers in the body frame (B-frame) and in the fixed frame (R-frame), such as a Global Positioning System (GPS) plus an Inertial Measurement Unit (IMU) [8], infrared sensors in the R-frame with inertial and infra-red camera sensors in the B-frame (has is the case with Wii Remote), cameras in the R-frame and MARG sensors in the B-frame (has in the PS3 Move), magnetic field beacons in the R-frame with sensing in the B-frame (has in the Razer Hydra), infrared cameras in the R-frame with infrared markers in the B-frame (such as in Vicon Systems [9]). However, these kind of approaches are to be avoided, because the goal of this study is to confine the processing of the orientation/positioning in the RCD as much as possible. In the context of this dissertation, the remote control should not have any external auxiliary system for the determination of its pose. There are approaches that only use image sensors in the B-frame to achieve the same output [10] or a combination with inertial units [11, 12]. These are self-contained solutions, but the drawback is the computational needs and large consumption they usually exhibit. The logical solution is to use inertial and magnetic MEMS sensors, since these combine low cost, are lightweight, low power consumption and output signals that can be processed with less computational efforts. The use of only these sensors really difficult the task of construct a full pose determination system because of the huge errors accumulated in the integration steps of the accelerometer data. Nevertheless for small periods of time, relative positioning is achievable [13]. Other techniques can be employed when there is a cyclic movement [14] or points in time in which a known velocity is present, as in steps counting in fitness applications [15, 16], with recent advances being made in order to reduce the accelerometer errors for that purpose [17].

[8]Movea TV Solutions. Available on: www.movea.com

The miniaturization of inertial and magnetic sensors have opened a new world of applications and consequently has attracted many researchers in the last years to develop filters able to accurately estimate the orientation of a body from them. Three major approaches are used to mitigate sensor errors and fuse sensor data in order to estimate the correct orientation: Stochastic Filters (commonly Kalman Filters (KF)), Deterministic Filters, and Complementary Filters (CF).

Kalman filters

KFs are the most well known stochastic approach that accounts for white noise. Its original description is for linear systems [18], but quickly its popularity (due to its applicability to a wide range of fields) allowed the development of versions for non-linear problems, which revealed very useful in the pose estimation problem. Some works that use these filters can be easily found in the literature. for instance, [19, 20] use the Extended Kalman Filter (EKF) and [21] the Unscented Kalman Filter (UKF). These filters (specially the ones that account for non-linearities) require relatively high computational costs due to intensive matrix operations and Jacobian matrices computation. Its complexity also increases according to the state model implemented. Some KFs include in their state model a sensors bias estimate and sensor readings deviations, further increasing the computational needs, but allowing more accurate estimates even in the presence of disturbances. Some of these solutions are explored in [22, 23]. These solutions are usually the choice for professional use in motion tracking applications, since few filters can achieve the same levels of accuracy. In the context of this particular study, the following two motion tracking solutions, from market leading companies, are presented:

- XSens[9] fuses data from its inertial sensors through a proprietary Xsens Kalman Filter that runs in an on-board Digital Signal Processor (DSP). The MTi-30 [24], was used in this work as reference for results validation. It has a stated dynamic accuracy up to 0.5 degrees in roll/pitch rotation and up to 1 degree in the yaw rotation, with a samplling rate of 100 Hz;

- PNICorp[10] is specialized in Attitude Heading Reference System (AHRS) modules, developing products for several applications. Some of its products have implemented a proprietary KF for sensor fusing, performing automatic continuous *hard* and *soft* iron distortions calibration and magnetic anomalies compensation, with low power consumption. The SENtral M&M Module Blue integrates a PNI Application Specific Integrated Circuit (ASIC) that implements the described filter. This module

[9]More details on: www.xsens.com

[10]More details on: www.pnicorp.com

was used to assess some of the results in this dissertation. Another product worth mentioning is the SpacePoint Scout, a development board that aids in the prototyping of video game, TV and STB controllers, virtual reality and body tracking. According to PNI it allows "pinpoint accuracy" for realistic motion tracking. It is specifically designed to use in highly dynamic conditions, such as high-speed gaming and in remote controls.

The cost of the aforementioned technologies is too high for them to be implemented in a generic RCD without custom made hardware for sensor fusion. As such, several alternatives will be presented next, which, albeit having lower accuracy, have a much lower cost, inasmuch as keeping computational requirements low.

Deterministic filters

The deterministic filters arise from derivations of the "Wahba's problem" [25] which is a variation of a least square minimization problem for finding the rotation matrix from vector measurements taken at a single time. They are based on the solution of optimization problems and normally used for orientation determination. Algorithms such as Tri-axial Attitude Determination (TRIAD) [26], one of the simplest, handles two vectors (gravity and magnetic north direction) which through a system of algebraic equations retrieves the orientation estimate. However the algorithm is sensitive to the order in which it receives the vectors and only accommodates two observation vectors [27]. For two or more observation vectors, and with the possibility of assigning different weights to each measure, there are algorithms such as the Quaternion Estimator (QUEST)[28], the Fast Optimal Attitude Matrix (FOAM)[29] and the Factored Quaternion Algorithm (FQA) [30].

In the case of only two vectors, which is of the interest in this work, there are alternatives that do not suffer from issues like the occurrence of singularities at the exchange of some increase on the computational burden. The Gauss Newton Algorithm (GNA) and the Gradient Descendent Algorithm (GDA) are the most used ones. The main advantage of the GNA is its robust estimation capability, however, it needs to iteratively find the optimal solution with some intensive calculations. Fortunately, after the filter convergence, when errors are smaller, it typically finds the solution with sufficient accuracy in only one or two steps, a reduced order GNA proposed in works such as [31, 19, 32] can provide an ever faster convergence. The GDA is both simple to implement and compute, with an analytically derived and optimised algorithm being described in [33], which enables performing with low sampling rates in real time.

Complementary filters

The CF works in the frequency domain, where some signals pass through a low-pass filter and others signals through its complementary high-pass filter, combining both signals in the end. These are particularly suited to be used in sensor fusing algorithms when the sensors output complementary signals. For this reason, and because they have very low computational needs in terms of implementation, their usage and study have been growing in the last past years. The drawback is that, contrary to KF, CF do not contemplate adaptability and the information about sensors errors is lost during the filter process. Nevertheless, some studies implement ways to dynamically adapt the filter, to estimate sensors bias, and to respond to readings deviations [34, 35, 36, 37], allowing their usage in a wider selection of environmental conditions.

This page was intentionally left blank.

Chapter 3

Background and Supporting Technologies

This Chapter details the more relevant background and supporting technologies used in this work. In the first Section the inertial and magnetic sensors technologies currently available, and used in the work, are presented, as well as their characterization and the characterization of the specific environment in which they will operate. In the second Section, a quick review of the orientation representation using quaternions is presented, in order to allow a better understanding of the topics ahead. The third Section presents the low-complexity filters studied and implemented in this work for estimating the orientation. Finally, the fourth Section presents the study for position determination from accelerometer readings.

3.1 Motion sensors

As already referred, the miniaturization of inertial sensors allowed an increased use in personal devices. Nowadays there are 3x3x1 *mm* chips that incorporate a 3 axes gyroscope, a 3 axes accelerometer and a 3 axes magnetometer in the same encapsulation. These MEMS are low-cost, lightweight and compact. On the other hand, the measurements they give are relatively noisy, can have offset bias, different scale factors and non-orthogonality between axes, even though this single encapsulation and miniaturization has reduced these problems. Due to the fabrication process, the characteristics of individual sensors cannot be guaranteed and can be altered when soldered [38]. In this Section these kind of sensors are analysed. Optical flow sensors can also be incorporated in motion tracking systems, therefore they are also part of this study.

3.1.1 Gyroscope

MEMS gyroscopes are sensors capable of measuring the angular velocity of a body. During rotation movements, the generated Coriolis force can be measured from the capacity difference on a vibrating coupled dual-mass proof-masses in the mechanical structure of the sensor, as shown in Figure 3.1. Deviations on the vibration pattern are captured by electrodes located underneath the plates on the IC, and the angular velocity can be extrapolated from this measure [39].

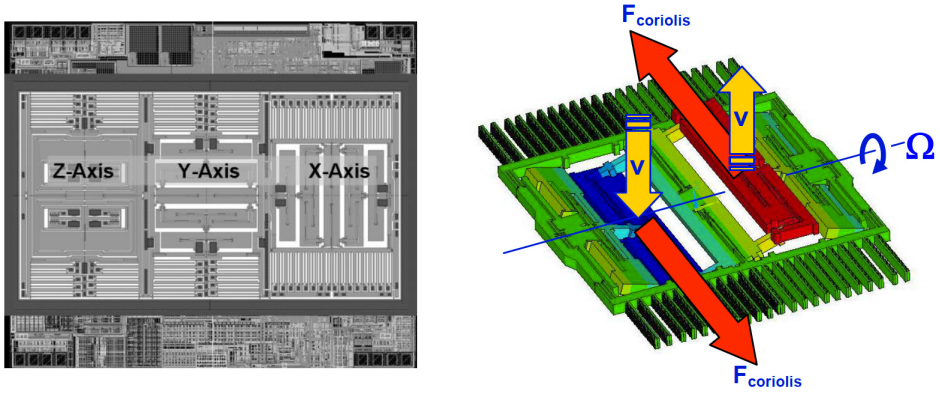

Figure 3.1: MEMS gyroscope micro-picture and operation [39].

The main concern handling these kind of sensors is to remove their typical non-zero offset at rest, and offset drift over time and with temperature, which has a major effect leading to integration errors when estimating the orientation. Typically, when combined with other sensors, fusion algorithms should estimate the offset drift in real-time, leaving the initial constant offset bias to be removed, which can be performed by taking a certain number of measurements at rest, as long as the gyroscope is warmed up, averaging them for posterior removal. For high precision applications the calibration can be extended to correct scale factors between axes, which requires the knowledge of the precise angular velocity applied to the sensor [40]. This last calibration procedure would considerably increase the price and production time for mass production, but is not required for most of the applications. Nevertheless, it is explained in Appendix B.

The characterization of MEMS gyroscopes leads to the model of this sensor in Equation (3.1) [20]. There, the measured angular velocity in the sensor frame, ${}^{S}\boldsymbol{\omega}$, is the sum of the real angular velocity, $\boldsymbol{\omega}_{true}$, in the global frame (E), transformed to the sensor frame (S) trough a certain transformation representation ${}^{S}_{E}\boldsymbol{T}$, plus the gyroscope offset bias ${}^{S}\boldsymbol{\rho}$ and uncorrelated white Gaussian measurement noise ${}^{S}\boldsymbol{n}$. Matrix ${}^{S}\boldsymbol{M}$ accounts for scale factors and axes misalignment, assumed to be a 3x3 identity matrix in this work.

$$ {}^{S}\boldsymbol{\omega} = {}^{S}\boldsymbol{M} \cdot \left[{}^{S}_{E}\boldsymbol{T} \cdot {}^{E}\boldsymbol{\omega}_{true} + {}^{S}\boldsymbol{\rho} \right] + {}^{S}\boldsymbol{n} \tag{3.1} $$

3.1.2 Accelerometer

MEMS accelerometers are transducers capable of measuring accelerations, typically composed of a movable proof mass with plates that are attached to a mechanical spring on a reference frame (anchor). There are parallel fixed plates that form a structure, designated as sensing fingers [41] which creates several capacitors with the movable plates. The deflection of proof masses due to the accelerations is measured in the form of a capacitance difference (see Figure 3.2).

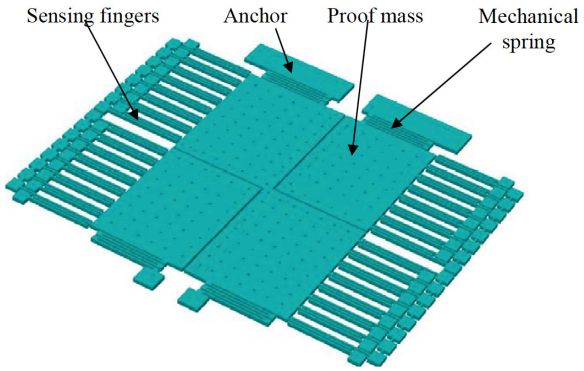

Figure 3.2: MEMS accelerometer scheme [41].

At rest, accelerometers only measure the gravitational acceleration. Although they are characterized by very accurate measurements, calibration is needed for certain applications in order to eliminate any offset bias. The calibration requires minimal human intervention, as long as one ensures that the platform is in a rest position, and keeping in mind that this calibration will define the direction of gravity relatively to the sensor. There are also more extensive calibrations that can correct axes misalignment and scale factors [42]. The procedure involves positioning the sensor at six stationary positions in all orthogonal directions allowing to calculate a calibration matrix that is applied to the raw readings (see Appendix B for more details). As such, a MEMS accelerometer can be modulated as in Equation (3.2) [20]. The model is similar to the gyroscope one, presented in Equation (3.1), but now there is an additional factor present in all moments, i.e., the gravity acceleration, ${}^{E}\boldsymbol{g}$, which adds to the linear real acceleration, ${}^{E}\boldsymbol{a}_{true}$, caused by motion in the world frame coordinates (E).

$$ {}^{S}\boldsymbol{a} = {}^{S}\boldsymbol{M} \cdot \left[{}^{S}_{E}\boldsymbol{T} \cdot ({}^{E}\boldsymbol{g} + {}^{E}\boldsymbol{a}_{true}) + {}^{S}\boldsymbol{\rho} \right] + {}^{S}\boldsymbol{n} \tag{3.2} $$

3.1.3 Magnetometer

For the application in this work, one needs an accuracy higher than one degree. This value is necessary, for instance, for an user to use the remote as a pointer and not noticing any

deviations on the pointer position on the screen during its usage time. Assuming that the magnetometer only measures the Earth's magnetic field, the measurement error should not exceed 750 nT (in Portugal). So, the sensor must fulfil a set of characteristics, such as a proper A/D converter resolution, compensation for temperature effects and for other error sources, in order to achieve that minimum accuracy [43]. There are many technologies able to sense the Earth's magnetic field with the desired accuracy, two types of which are here addressed: Hall effect magnetometers and magneto-inductive magnetometers.

MEMS magnetometers are the most common Hall effect transducers with a magnetic concentrator. Inertial sensors manufacturers are now building them in the same encapsulation with gyroscopes and accelerometers, forming a MARG unit. They are small in size and in cost, but provide much lower measuring accuracy than other sensor types, and are unable to keep a stable reading due to sensor noise. Hall effect sensors also drift significantly and require temperature compensation, which is usually taken into to account by the manufacturer [44].

An interesting technology explored by PNI Corp.[1] is magneto-inductive sensing. It is a method based on the frequency of an oscillating wave in a coil that, when exposed to an external magnetic field, suffers a change in its oscillation period [45, 46]. This is driven by a proprietary ASIC that performs the required operations. It allows high resolution with extremely low-noise, no hysteresis, a sensitivity up to 13 nT and low power consumption (600 μW). On the other hand it is a much more expensive and bigger sensor. A comparison between the two sensors readings can be seen in Figure 3.3.

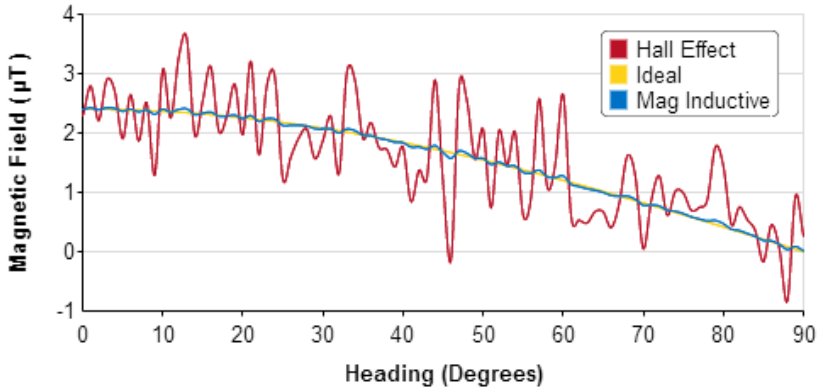

Figure 3.3: Comparison between Hall effect and magneto-inductive sensor [From: PNI site].

A magnetometer would in perfect conditions provide information about the vector pointing North, however, this is not necessarily the case. The Earth magnetic field is not constant over time (it changes over the years), nor it points to North. The difference between the magnetic North and the geographic North is designated by declination angle

[1] Site: www.pnicorp.com

and it depends of where you are on the surface of the planet [47]. Also, the vector contains components in the downward direction. The difference between the magnitude of horizontal and vertical readings is designated by inclination angle, which varies up to 90° near the poles and 0° near the equator. Along with these characteristics, the magnitude of the field over the surface of the Earth has values between 25 and 65 μT. For instance, in Portugal[2] the Earth's magnetic field has about 44 μT and an inclination angle of about 54° [47].

These characteristics of the Earth's magnetic field are not problematic for indoor navigation unless, the inclination angle of the magnetic field in the global frame (for instance, in the living room) is near 90°. However, other issues are encountered in indoor environments [48], as the sensor measures the local magnetic field, usually composed by the sum of: magnetic fields that exist on the local coordinate frame of the sensor; the indoor fields that exist in the global frame; the Earth's magnetic field (also in the global frame). These undesirable magnetic fields that sum to the Earth's one can be divided into two types of distortions: *soft iron* and *hard iron* [33, 49].

The *hard iron* distortions are usually easy to compensate, if the working environment does not change, they cause a constant offset in the measurements of the magnetic field. These distortions arise from permanent magnets, ferrous materials, batteries, Printed Circuit Board (PCB) lines or any material that as a permanent field on the magnetometer platform, and always keep their pose relative to the sensor. *Hard iron* distortions remain constant and only add a certain value of magnitude to each axis of the sensor output. This is valid for a specific location, since variations on the amplitude and on the incident angle of external magnetic fields will change *hard iron* distortions. This means that *hard iron* distortions depend on the *soft iron* distortions. The *soft iron* bias appears when there exists a magnetically soft material in the surroundings of the magnetometer. Most indoor environments contain appliances, building materials and furniture that create/distort magnetic fields. The distortion, in this case, depends on the sensor orientation and position on the 3-dimensional space and they may be not constant over time [50, 51, 43]. In summary, although these disturbances are not completely decoupled, it is possible to somewhat relate the *hard iron* distortions with the device and the *soft iron* distortions with the environment where the device is located. Figure 3.4 shows a 2 Dimensions (2D) representation of the effects in the magnetic field measurements, for these types of distortions, when the magnetometer is rotated 360 degrees in one plane. The combination of both distortions in 3D cause the expected perfect and origin-located sphere to be distorted into a displaced ellipsoid (observable in Figure B.4 - Appendix B).

There are calibrations that have to be performed in order to obtain usable readings

[2]From Portuguese Institute of the Ocean and Atmosphere (IPMA): www.ipma.pt

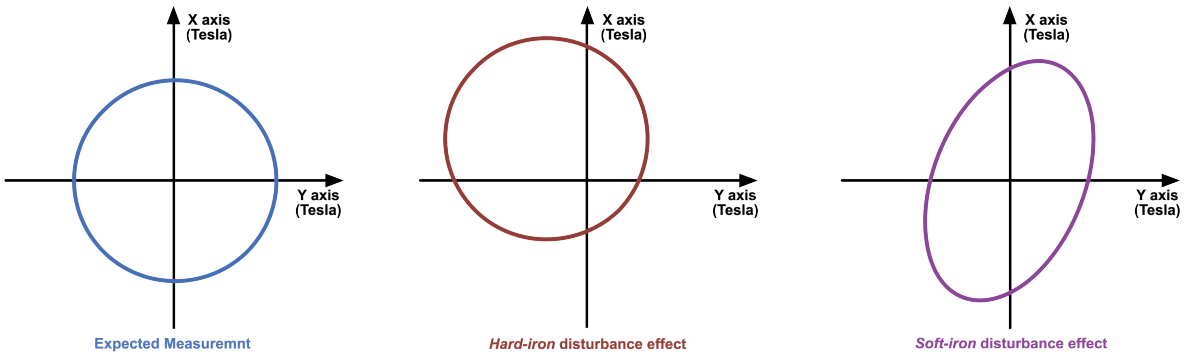

Figure 3.4: Disturbances to Earth's magnetic field readings.

from the magnetometer (these are explained in Appendix B). The calibration methods imply the rotation of the sensor platform in complete circles around it self, in a way that allows to gather many points of an ellipsoid. *Hard* and *soft iron* distortions are different from location to location for the same device and so a particular calibration is only valid for a specific location.

These are very unfavourable points for the use of magnetometers in indoor localization systems. On the other hand, there are no cheaper sensors with such low computational requirements in the market, that can retrieve absolute orientation so, the use of magnetometers is also evaluated as part of this work.

Given the description of the sensor and fields, the magnetometer can be modelled as in Equation (3.3) [20]. The *hard iron* distortions are modelled as a constant offset bias (${}^{S}\boldsymbol{\rho}$) and the *soft iron* distortions as a 3x3 matrix (${}^{E}\boldsymbol{S}_{\rho}$).

$$ {}^{S}\boldsymbol{m} = {}^{S}\boldsymbol{M} \cdot \left[{}^{S}_{E}\boldsymbol{T} \cdot \left({}^{E}\boldsymbol{S}_{\rho} \cdot {}^{E}\boldsymbol{m}_{true} \right) + {}^{S}\boldsymbol{\rho} \right] + {}^{S}\boldsymbol{n} \tag{3.3} $$

3.1.4 Optical flow sensor

Optical flow is the process of estimating motion between two inertial frames using patterns, surfaces or edges displacement in an optical sensor [52]. There are many methods to determine optical flow, such as: Phase correlation, Block-based methods, Lucas–Kanade method [53] and the Black–Jepson method[54]. These methods, together with some other state-of-the-art algorithms, are evaluated on the Middlebury Benchmark Dataset[3].

Optical flow sensors are very common in optical computer mouses. They estimate the relative displacement of the surface under the mouse by tracking reference points on it [55]. In computer mouses, their usage is done within a very controlled environment and their accuracy is extremely high [56]. However, when a non-fixed focal point is not part of the

[3]Available on: www.vision.middlebury.edu/flow/

equation new challenges arise for the use of these sensors [57]. Because of the data volume and data rate that visual systems require, and the computational on-board constrains on a remote control, picking an optical flow sensor is not a trivial task. Many SoC already retrieve displacements estimates, and although they are usually specifically designed for computer mice, they can be modified for other applications.This approach allows removing the computational burden of optical flow algorithms from the main processor unit, while maintaining a low power consumption level.

For this work it was selected the ADNS-3080 [58], manufactured by Avago Technologies. It has a low-resolution camera of 30x30 grey pixels achieving a maximum of 6400 fps. For bad lighting conditions it can be set a lower shutter speed, achieving a framerate of 2000 fps. The chip includes a preprogrammed DSP that performs a comparative analysis of the sequence of images acquired and calculates the quantity of motion, as described in [59]. The sensor algorithm is not open source, and therefore it is not adaptable or extendible. The displacement output is given in counts per inches (cpi), as such if the goal is to measure another physical magnitude, one needs to develop a calibration method to calculate the conversion of cpi to the desired units. The sensor is equipped with Serial Peripheral Interface (SPI), which allows fast readings of the sensor, with a burst mode being available for higher efficiency.

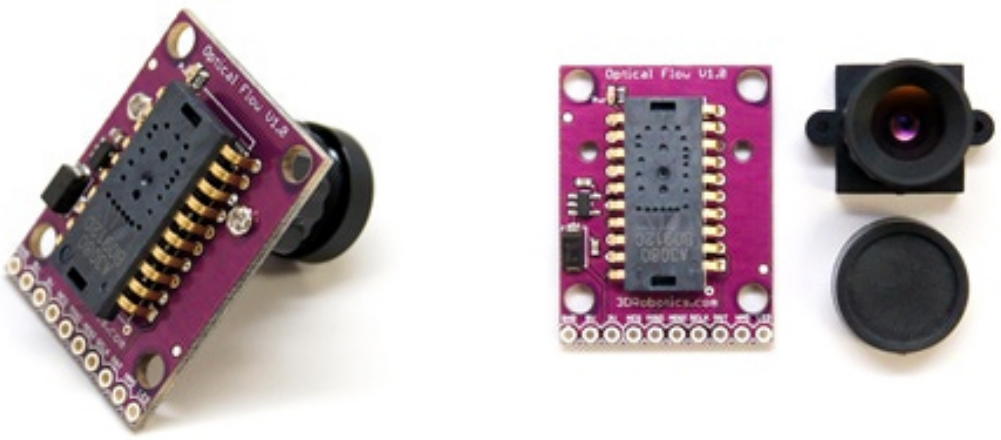

Figure 3.5: CJMCU-110 board incorporates a ADNS-3080.

The ADNS-3080 is prepared to be mounted into a computer mouse support that focuses the sensor at 2.4 mm. For the application scenario, the focal point needs to be about 2 meter of distance from the sensor, so some adaptation has to be made. In fact, there are adaptations available on the market, like the CJMCU-110 board visible in Figure 3.5, that already incorporate a lens that can focus up to 2 m. This is the solution chosen to test optical flow sensors in pose estimation.

3.2 Quaternions

Quaternions, generally represented by $\boldsymbol{q}$, are an extension of complex numbers to the 3D world. When normalized, $\| \boldsymbol{q} \| = 1$, they represent a rotation and can be used to represent the orientation of a rigid body in a given coordinate frame (B) relatively to any other generic coordinate frame (A). They are written as a four-dimensional vector space: the first value represents a scalar component, s, and the last three values represent three vectorial components from a rotation axis $\boldsymbol{l} = [\, l_x,\ l_y,\ l_z \,]$, as in Equation (3.4).

$$\boldsymbol{q} = \begin{bmatrix} s & l_x & l_y & l_z \end{bmatrix} = \begin{bmatrix} q_s & q_x & q_y & q_z \end{bmatrix} \tag{3.4}$$

According to the Euler theorem, any orientation from B to A can be achieved through a single rotation of an angle θ around a vector (in this case $\boldsymbol{l}$) defined in a given frame A. Observing Figure 3.6, the quaternion obtained from this description is defined in Equation (3.5), where $\hat{\boldsymbol{l}}$ represents vector $\boldsymbol{l}$, normalized.

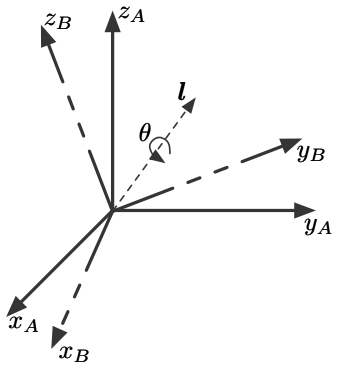

Figure 3.6: Rotation of frame B in relation to frame A around $\boldsymbol{l}$ by θ.

$$ {}^{B}_{A}\boldsymbol{q} = \begin{bmatrix} cos(\frac{\theta}{2}) & \hat{l}_x \cdot sin(\frac{\theta}{2}) & \hat{l}_y \cdot sin(\frac{\theta}{2}) & \hat{l}_z \cdot sin(\frac{\theta}{2}) \end{bmatrix} \tag{3.5}$$

A quaternion conjugate $\boldsymbol{q}^*$ is equal to the inverse quaternion $\boldsymbol{q}^{-1}$ as long as $\| \boldsymbol{q} \| = 1$, which can be used to reverse rotations or swap between frames. This important property is described in Equation (3.6) and (3.7).

$$\boldsymbol{q}^* = \begin{bmatrix} q_s & -q_x & -q_y & -q_z \end{bmatrix} \tag{3.6}$$

$$ {}^{B}_{A}\hat{\boldsymbol{q}}^{-1} = {}^{B}_{A}\hat{\boldsymbol{q}}^{*} = {}^{A}_{B}\hat{\boldsymbol{q}} \tag{3.7}$$

Successive rotations can be computed using Equation (3.8) where $\otimes$ denotes a quaternion multiplication. Also, a vector can be rotated using Equation (3.9), where ${}^{A}\boldsymbol{l}$ and ${}^{B}\boldsymbol{l}$

are the same vector represented in two different frames [60].

$$ {}^{C}_{A}\hat{\boldsymbol{q}} = {}^{B}_{A}\hat{\boldsymbol{q}} \otimes {}^{C}_{B}\hat{\boldsymbol{q}} \tag{3.8} $$

$$ {}^{A}\boldsymbol{l} = {}^{B}_{A}\hat{\boldsymbol{q}} \otimes {}^{B}\boldsymbol{l} \otimes {}^{B}_{A}\hat{\boldsymbol{q}}^{*} \tag{3.9} $$

According to the Hamilton rule, a quaternion multiplication is given by Equation (3.10) and is a non-commutative operation: $\boldsymbol{q} \otimes \mathbf{p} \neq \mathbf{p} \otimes \boldsymbol{q}$.

$$ \boldsymbol{q} \otimes \mathbf{p} = \begin{bmatrix} q_s p_s - q_x p_x - q_y p_y - q_z p_z \\ q_s p_x + q_x p_s + q_y p_z - q_z p_y \\ q_s p_y - q_x p_z + q_y p_s + q_z p_x \\ q_s p_z + q_x p_y - q_y p_x + q_z p_s \end{bmatrix} \tag{3.10} $$

Further information regarding orientation representations, the advantages and disadvantages of each one and reasons for the choice of quaternions over other representations can be found in Appendix A.

3.3 Sensor fusion filters

The fusion algorithms used to treat MARG sensors data, combine the best attributes of each sensor type, and try to mitigate the characteristics that introduce error in the estimates. In this Section, the filters used in the development of this work are introduced and described followed by an explanation of how the orientation can be computed from the sensors using quaternions representation [61, 62, 14].

3.3.1 Orientation from angular rate

Angular rate measurements retrieved from the 3-axis gyroscope can be represented in the form of a vector, ${}^{S}\boldsymbol{\omega}$, as shown in Equation (3.11), with the measured values per axis in $rad \cdot s^{-1}$.

$$ {}^{S}\boldsymbol{\omega} = \begin{bmatrix} 0 & \omega_x & \omega_y & \omega_z \end{bmatrix} \tag{3.11} $$

The quaternion derivative, at step t, that describes the orientation change rate of the sensor frame (S) relative to the Earth frame (E), ${}^{S}_{E}\dot{\boldsymbol{q}}_{\omega,t}$, can be determined from Equation (3.12). Here ${}^{S}_{E}\tilde{\boldsymbol{q}}_{t-1}$ is the quaternion that represents the previous estimate of the orientation.

$$ {}^{S}_{E}\dot{\boldsymbol{q}}_{\omega,t} = \frac{1}{2}\, {}^{S}_{E}\tilde{\boldsymbol{q}}_{t-1} \otimes {}^{S}\boldsymbol{\omega}_{t} \tag{3.12} $$

Therefore, the orientation estimate, ${}^{S}_{E}\tilde{\boldsymbol{q}}_{\omega,t}$, is obtained by integrating the quaternion derivative, as can be observed in Equation (3.13), where δt is the sampling time.

$$ {}^{S}_{E}\tilde{\boldsymbol{q}}_{\omega,t} = {}^{S}_{E}\tilde{\boldsymbol{q}}_{t-1} + {}^{S}_{E}\dot{\boldsymbol{q}}_{\omega,t} \cdot \delta t \tag{3.13} $$

This orientation estimate is always relative to the starting orientation. If the initial pose of the sensor frame is unknown, a fixed frame has to be defined.

3.3.2 Orientation from observation sensors

In order to get a fixed frame, in this case the Earth frame, the accelerometer is not enough, because it can only inform the system about one direction (gravity field direction), it is necessary, at least, another orthogonal direction. This can be obtained by measuring the Earth's magnetic field with a magnetometer. Both sensors are prone to measure undesirable disturbances, but, in this Section, it will be assumed perfect sensing, where only the gravity acceleration and the Earth's magnetic field are measured.

A GDA approach is often used [31, 28], to compute the observation sensors (S) orientation relatively to the Earth frame (E), as shown in Equation (3.14). This is generically a first order optimization problem that converges to a local minimum, after an initial value is attributed for the first iteration. The μ_t gain controls the convergence rate of the algorithm and needs to have a value that ensures a faster convergence of the GDA compared to the actual physical rate of orientation change, but not too high in order to avoid overshooting the optimal point. For computational efficiency it can be used as a constant value. If computing this value in all steps is deemed necessary, for accuracy improvements, it can be calculated from Equation (3.15), where α is only a constant that accounts for noise in measurements and can be adjusted according to the sensors characteristics.

$$ {}^{S}_{E}\tilde{\boldsymbol{q}}_{\nabla,t} = {}^{S}_{E}\tilde{\boldsymbol{q}}_{t-1} - \mu_t \cdot \frac{\nabla \boldsymbol{f}}{\|\nabla \boldsymbol{f}\|} \tag{3.14} $$

$$ \mu_t = \alpha \cdot \|{}^{S}_{E}\dot{\boldsymbol{q}}_{\omega,t}\| \cdot \delta t, \quad \alpha > 1 \tag{3.15} $$

The gradient vector is determined with Equation (3.16).

$$ \nabla \boldsymbol{f}({}^{S}_{E}\tilde{\boldsymbol{q}},\ {}^{E}\hat{\boldsymbol{d}},\ {}^{S}\hat{\boldsymbol{r}}) = \mathbf{J}^{T}({}^{S}_{E}\tilde{\boldsymbol{q}},\ {}^{E}\hat{\boldsymbol{d}}) \cdot \boldsymbol{f}({}^{S}_{E}\tilde{\boldsymbol{q}},\ {}^{E}\hat{\boldsymbol{d}},\ {}^{S}\hat{\boldsymbol{r}}) \tag{3.16} $$

$\boldsymbol{f}({}^{S}_{E}\tilde{\boldsymbol{q}},\ {}^{E}\hat{\boldsymbol{d}},\ {}^{S}\hat{\boldsymbol{r}})$ is the cost function and $\mathbf{J}({}^{S}_{E}\tilde{\boldsymbol{q}},\ {}^{E}\hat{\boldsymbol{d}})$ its Jacobian, which are described in Equations (3.17) and (3.18), respectively.

$$\boldsymbol{f}({}^{S}_{E}\tilde{\boldsymbol{q}},\ {}^{E}\hat{\boldsymbol{d}},\ {}^{S}\hat{\boldsymbol{r}}) = \begin{bmatrix} 2d_x(\frac{1}{2} - q_y^2 - q_z^2) + 2d_y(q_s q_z + q_x q_y) + 2d_z(q_x q_z - q_s q_y) - r_x \\ 2d_x(q_x q_y - q_s q_z) + 2d_y(\frac{1}{2} - q_x^2 - q_z^2) + 2d_z(q_s q_x + q_y q_z) - r_y \\ 2d_x(q_s q_y + q_x q_z) + 2d_y(q_y q_z - q_s q_x) + 2d_z(\frac{1}{2} - q_x^2 - q_y^2) - r_z \end{bmatrix} \tag{3.17}$$

$$\mathbf{J}({}^{S}_{E}\tilde{\boldsymbol{q}},\ {}^{E}\hat{\boldsymbol{d}}) = \begin{bmatrix} 2d_y q_z - 2d_z q_y & 2d_y q_y + 2d_z q_z & 2d_y q_x - 2d_z q_s - 4d_x q_y & 2d_y q_s + 2d_z q_x - 4d_x q_z \\ 2d_z q_x - 2d_x q_z & 2d_x q_y + 2d_z q_s - 4d_y q_x & 2d_x q_x + 2d_z q_z & 2d_z q_y - 2d_x q_s - 4d_y q_z \\ 2d_x q_y - 2d_y q_x & 2d_x q_z - 2d_y q_s - 4d_z q_x & 2d_x q_s + 2d_y q_z - 4d_z q_y & 2d_x q_x + 2d_y q_y \end{bmatrix} \tag{3.18}$$

Superscript symbol ($\hat{}$) denote normalized vectors (in order to represent rotations, the quaternion is normalized by definition and so this superscript is omitted in their representation). The orientation estimate, ${}^{S}_{E}\tilde{\boldsymbol{q}}$, results from the Earth's reference frame, ${}^{E}\hat{\boldsymbol{d}}$, with the frame obtained from the measured fields, ${}^{S}\hat{\boldsymbol{r}}$, as given by Equations (3.19-3.21).

$${}^{S}_{E}\tilde{\boldsymbol{q}} = \begin{bmatrix} q_s & q_x & q_y & q_z \end{bmatrix} \tag{3.19}$$

$${}^{E}\hat{\boldsymbol{d}} = \begin{bmatrix} 0 & d_x & d_y & d_z \end{bmatrix} \tag{3.20}$$

$${}^{S}\hat{\boldsymbol{r}} = \begin{bmatrix} 0 & r_x & r_y & r_z \end{bmatrix} \tag{3.21}$$

In practice, the GDA searches for the point where the transformed frame of the sensors ${}^{S}\hat{\boldsymbol{r}}$ meets the earth frame ${}^{E}\hat{\boldsymbol{d}}$, ergo, the solution of Equation (3.22).

$$min\ [\ \boldsymbol{f}({}^{S}_{E}\tilde{\boldsymbol{q}},\ {}^{E}\hat{\boldsymbol{d}},\ {}^{S}\hat{\boldsymbol{r}}) = {}^{S}_{E}\tilde{\boldsymbol{q}}^{*} \otimes {}^{E}\hat{\boldsymbol{d}} \otimes {}^{S}_{E}\tilde{\boldsymbol{q}} - {}^{S}\hat{\boldsymbol{r}}\ ] \tag{3.22}$$

The number of operations necessary for the generic Equation (3.16) can be reduced, since the gravitational field, ${}^{E}\hat{\boldsymbol{g}}$, contains merely one component along the global frame axis and the magnetic field, ${}^{E}\hat{\boldsymbol{b}}$, two components.

Using a North, East, Down (NED) convention [63], assuming:

North: the x axis

East: the y axis

Down: the z axis

The fields are described by the vectors in Equations (3.23) and (3.24).

$${}^{E}\hat{\boldsymbol{g}} = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.23}$$

$${}^{E}\hat{\boldsymbol{b}} = \begin{bmatrix} 0 & b_x & 0 & b_z \end{bmatrix} \tag{3.24}$$

Equations (3.25) and (3.26) represent the measurements of the accelerometer and the magnetometer, respectively.

$$ {}^{S}\hat{\boldsymbol{a}} = \begin{bmatrix} 0 & a_x & a_y & a_z \end{bmatrix} \tag{3.25} $$

$$ {}^{S}\hat{\boldsymbol{m}} = \begin{bmatrix} 0 & m_x & m_y & m_z \end{bmatrix} \tag{3.26} $$

This way, Equations (3.17) and (3.18) are equivalent to Equations (3.27) and (3.28) for the accelerometer and Equations (3.29) and (3.30) for the magnetometer.

$$ \boldsymbol{f}_g({}^{S}_{E}\tilde{\boldsymbol{q}},\ {}^{S}\hat{\boldsymbol{a}}) = \begin{bmatrix} 2(q_x q_z - q_s q_y) - a_x \\ 2(q_s q_x + q_y q_z) - a_y \\ 2(\frac{1}{2} - q_x^2 - q_y^2) - a_z \end{bmatrix} \tag{3.27} $$

$$ \mathbf{J}_g({}^{S}_{E}\tilde{\boldsymbol{q}}) = \begin{bmatrix} -2q_y & 2q_z & -2q_s & 2q_x \\ 2q_x & 2q_s & 2q_z & 2q_y \\ 0 & -4q_x & -4q_y & 0 \end{bmatrix} \tag{3.28} $$

$$ \boldsymbol{f}_b({}^{S}_{E}\tilde{\boldsymbol{q}},\ {}^{E}\hat{\boldsymbol{b}},\ {}^{S}\hat{\boldsymbol{m}}) = \begin{bmatrix} 2b_x(\frac{1}{2} - q_y^2 - q_z^2) + 2b_z(q_x q_z - q_s q_y) - m_x \\ 2b_x(q_x q_y - q_s q_z) + 2b_z(q_s q_x + q_y q_z) - m_y \\ 2b_x(q_s q_y + q_x q_z) + 2b_z(\frac{1}{2} - q_x^2 - q_y^2) - m_z \end{bmatrix} \tag{3.29} $$

$$ \mathbf{J}_b({}^{S}_{E}\tilde{\boldsymbol{q}},\ {}^{E}\hat{\boldsymbol{b}}) = \begin{bmatrix} 2b_z q_y & 2b_z q_z & -2b_z q_s - 4b_x q_y & 2b_z q_x - 4b_x q_z \\ 2b_z q_x - 2b_x q_z & 2b_x q_y + 2b_z q_s & 2b_x q_x + 2b_z q_z & 2b_z q_y - 2b_x q_s \\ 2b_x q_y & 2b_x q_z - 4b_z q_x & 2b_x q_s - 4b_z q_y & 2b_x q_x \end{bmatrix} \tag{3.30} $$

Both fields are combined, in order to obtain a single possible orientation of body frame holding the sensors. Provided that $b_x \neq 0$, Equation (3.31) has a minimum in a single point.

$$ \boldsymbol{f}_{g,b}({}^{S}_{E}\tilde{\boldsymbol{q}},\ {}^{S}\hat{\boldsymbol{a}},\ {}^{E}\hat{\boldsymbol{b}},\ {}^{S}\hat{\boldsymbol{m}}) = \begin{bmatrix} \boldsymbol{f}_g({}^{S}_{E}\tilde{\boldsymbol{q}},\ {}^{S}\hat{\boldsymbol{a}}) \\ \boldsymbol{f}_b({}^{S}_{E}\tilde{\boldsymbol{q}},\ {}^{E}\hat{\boldsymbol{b}},\ {}^{S}\hat{\boldsymbol{m}}) \end{bmatrix} \tag{3.31} $$

$$ \mathbf{J}_{g,b}({}^{S}_{E}\tilde{\boldsymbol{q}},\ {}^{E}\hat{\boldsymbol{b}}) = \begin{bmatrix} \mathbf{J}_g^T({}^{S}_{E}\tilde{\boldsymbol{q}}) \\ \mathbf{J}_b^T({}^{S}_{E}\tilde{\boldsymbol{q}},\ {}^{E}\hat{\boldsymbol{b}}) \end{bmatrix} \tag{3.32} $$

The cost function gradient, $\nabla \boldsymbol{f}$, is finally given by Equation (3.33).

$$ \nabla \boldsymbol{f} = \begin{cases} \mathbf{J}_g^T({}^{S}_{E}\tilde{\boldsymbol{q}}_{t-1}) \cdot \boldsymbol{f}_g({}^{S}_{E}\tilde{\boldsymbol{q}}_{t-1},\ {}^{S}\hat{\boldsymbol{a}}) \\ \mathbf{J}_{g,b}^T({}^{S}_{E}\tilde{\boldsymbol{q}}_{t-1}) \cdot \boldsymbol{f}_{g,b}({}^{S}_{E}\tilde{\boldsymbol{q}}_{t-1},\ {}^{E}\hat{\boldsymbol{b}},\ {}^{S}\hat{\boldsymbol{m}}) \end{cases} \tag{3.33} $$

3.3.3 Complementary filters

The main goal of a CF is to use the signals from sensors through a low-pass filter and through his complementary high-pass filter. When applied to MARG units, it means combining the low frequency signals from accelerometer and magnetometer with the high frequency signals from de gyroscope.

Madgwick filter

In 2010, Madgwick *et al.* [33] described the implementation of a CF with low computational needs. The author includes some features that allow compensation of the gyroscope offset drift over time, and compensation of the declination distortion of the magnetic field. Data from the accelerometer and magnetometer are fused through a GDA, which estimates the absolute orientation of the Earth's frame, and is also used for the correction of the gyroscope offset drift.

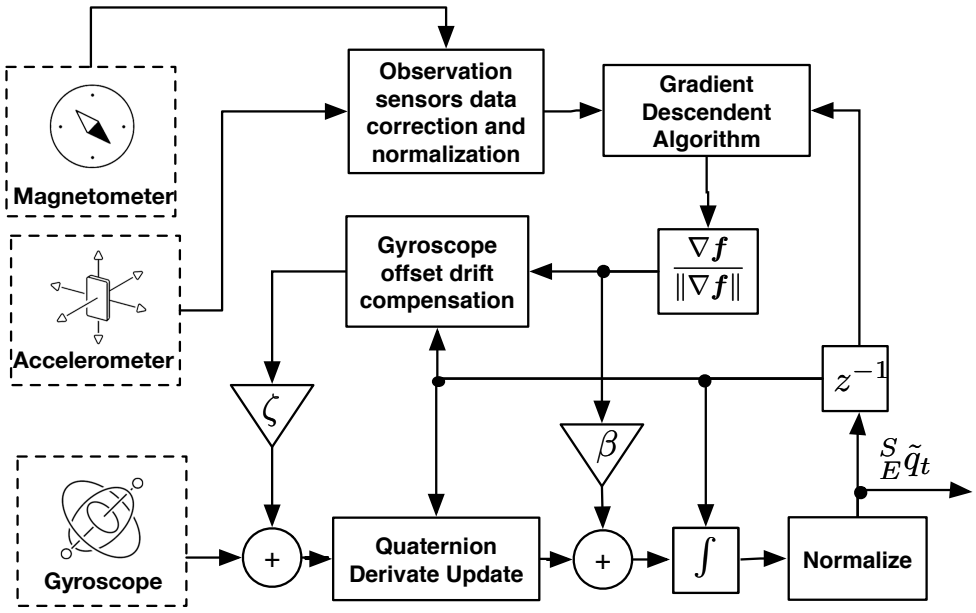

Figure 3.7: Madgwick filter block diagram.

The block diagram for this filter is depicted in Figure 3.7. In the *Observation sensors data correction and normalization* block, normalization of the sensor measurements and magnetic declination compensation are performed; then, the orientation error is estimated from the observation sensors on the *Gradient Descendent Algorithm* block, and normalized thereupon. This estimate is used to correct the readings from the gyroscope in two ways: through the β gain and through the ζ gain, after the gyroscope offset drift update on the corresponding block. Finally the quaternion derivative is integrated and the quaternion normalized. This filter is referred in this dissertation as Original Madgwick Filter (OMdF).

The estimated quaternion, ${}^{S}_{E}\tilde{\boldsymbol{q}}_t$, is obtained by merging high frequency signals from the orientation computed using the gyroscope, ${}^{S}_{E}\tilde{\boldsymbol{q}}_{\omega,t}$, with the low frequency signals from the orientation computed using the GDA, ${}^{S}_{E}\tilde{\boldsymbol{q}}_{\nabla,t}$. Equation (3.34) sets the generic approach of implementation to CFs. The trust of each estimative (can also be interpreted as the normalized frequency that delimits each information), is controlled by γ_t.

$$ {}^{S}_{E}\tilde{\boldsymbol{q}}_t = \gamma_t \cdot {}^{S}_{E}\tilde{\boldsymbol{q}}_{\nabla,t} + (1-\gamma_t) \cdot {}^{S}_{E}\tilde{\boldsymbol{q}}_{\omega,t} \qquad 0 \leq \gamma_t \leq 1 \tag{3.34} $$

There are a few developments explored in [33] having the main Equation (3.34) as reference. Valid approximations are made, allowing some simplifications of the algorithm and a smaller number of constants for the filter tuning.

An optimal value is defined for γ_t as the value that ensures an equal weighted divergence of ${}^{S}_{E}\tilde{\boldsymbol{q}}_{\omega,t}$ and convergence of ${}^{S}_{E}\tilde{\boldsymbol{q}}_{\nabla,t}$. This is shown in Equation (3.35), where $\frac{\mu}{\delta t}$ is the convergence rate of ${}^{S}_{E}\tilde{\boldsymbol{q}}_{\nabla,t}$ and β the divergence rate of ${}^{S}_{E}\tilde{\boldsymbol{q}}_{\omega,t}$.

$$ (1-\gamma_t)\beta = \gamma_t \frac{\mu}{\delta t} \;\Leftrightarrow\; \gamma_t = \frac{\beta}{\frac{\mu}{\delta t}+\beta} \tag{3.35} $$

Often μ, calculated in Equation (3.15), is a large value because of the (already referred) need for a faster converge rate of ${}^{S}_{E}\tilde{\boldsymbol{q}}_{\nabla,t}$ than the physical change rate of the body frame. This way, ${}^{S}_{E}\tilde{\boldsymbol{q}}_{t-1}$ in Equation (3.14) becomes negligible and the same equation can be approximated to Equation (3.36). The same argument applied to Equation (3.35) reduces it to Equation (3.37).

$$ {}^{S}_{E}\tilde{\boldsymbol{q}}_{\nabla,t} \approx -\mu \cdot \frac{\nabla \boldsymbol{f}}{\|\nabla \boldsymbol{f}\|} \tag{3.36} $$

$$ \gamma_t \approx \frac{\beta \delta t}{\mu} \tag{3.37} $$

When Equations (3.13), (3.36) and (3.37) are joined together into Equation (3.34), Equation (3.38) is obtained, where γ_t is approximated to Equation (3.37).

$$ {}^{S}_{E}\tilde{\boldsymbol{q}}_t = \frac{\beta \delta t}{\mu} \cdot (-\mu \cdot \frac{\nabla \boldsymbol{f}}{\|\nabla \boldsymbol{f}\|}) + (1-0) \cdot ({}^{S}_{E}\tilde{\boldsymbol{q}}_{t-1} + {}^{S}_{E}\dot{\boldsymbol{q}}_{\omega,t} \cdot \delta t) \tag{3.38} $$

Finally, Equation (3.39) is derived after the simplification of Equation (3.38). This development means that only one gain in the filter needs adjustment (β value), controlling the filter response to the sensors and/or applications.

$$ {}^{S}_{E}\tilde{\boldsymbol{q}}_t = {}^{S}_{E}\tilde{\boldsymbol{q}}_{t-1} + ({}^{S}_{E}\dot{\boldsymbol{q}}_{\omega,t} - \beta \cdot \frac{\nabla \boldsymbol{f}}{\|\nabla \boldsymbol{f}\|}) \cdot \delta t \tag{3.39} $$

Further improvements are made to the filter. As detailed previously, the gyroscope is a sensor prone to have some offset in its output, which is usually removed with calibration. However, this offset drifts over time and with temperature variations. Such characteristic is a major concern and all filters should take it into account if an accurate estimate is needed. This issue assumes great importance once it is known that the orientation is essentially obtained from the integration of gyroscope measurements, therefore, any offset will cause a significant impact on the orientation error over time.

The vector obtained by the gradient in Equation (3.39) is, in fact, the estimated error in the orientation change rate and may be expressed as the angular rate error, ${}^{S}\boldsymbol{\omega}_{\epsilon,t}$, using Equation (3.40). As the gyroscope offset is treated as the Direct Current (DC) component of the signal, it can be removed from the raw readings (${}^{S}\boldsymbol{\omega}_{r,t}$) as the integral of ${}^{S}\boldsymbol{\omega}_{\epsilon,t}$ over time, as shown in Equation (3.41). This correction is controlled through gain ζ.

$$ {}^{S}\boldsymbol{\omega}_{\epsilon,t} = 2 \cdot {}^{S}_{E}\tilde{\boldsymbol{q}}^{*}_{t-1} \otimes \frac{\nabla \boldsymbol{f}}{\|\nabla \boldsymbol{f}\|} \tag{3.40} $$

$$ {}^{S}\boldsymbol{\omega}_{t} = {}^{S}\boldsymbol{\omega}_{r,t} - \zeta \cdot \sum {}^{S}\boldsymbol{\omega}_{\epsilon,t} \cdot \delta t \tag{3.41} $$

The output values of the compensated gyroscope ${}^{S}\boldsymbol{\omega}_{t}$ are to be used in Equation (3.12).

Regarding the observation sensors, the magnetometer is the main source of errors due to numerous external factors and characteristics of the Earth's magnetic field, as described in Section 3.1.3. However, the effect of erroneous information provided by the magnetometer can be limited to the yaw angle, assuming that any y component of the measured direction of the Earth's magnetic field in the Earth frame, ${}^{E}\hat{\mathbf{h}}_{t}$, is due to the declination in the measured magnetic field. Thus, it is ensured that the filter's reference direction of the Earth's magnetic field, ${}^{E}\hat{\boldsymbol{b}}_{t}$, has only x and z components, as can be seen in Equation (3.42). ${}^{E}\hat{\mathbf{h}}_{t}$ is calculated as the normalised magnetometer measurement, ${}^{S}\hat{\boldsymbol{m}}_{t}$, rotated by the estimated orientation of the body frame, as can be seen in Equation (3.43).

$$ {}^{E}\hat{\boldsymbol{b}}_{t} = \begin{bmatrix} 0 & \sqrt{h_x^2 + h_y^2} & 0 & h_z \end{bmatrix} \tag{3.42} $$

$$ {}^{E}\hat{\mathbf{h}}_{t} = \begin{bmatrix} 0 & h_x & h_y & h_z \end{bmatrix} = {}^{S}_{E}\tilde{\boldsymbol{q}}_{t-1} \otimes {}^{S}\hat{\boldsymbol{m}}_{t} \otimes {}^{S}_{E}\tilde{\boldsymbol{q}}^{*}_{t-1} \tag{3.43} $$

Furthermore, the initial values used for the two gains of this filter (β and ζ) are explored in [33]. They can be computed from the estimated gyroscope measurement errors. Calculating β from the estimated frequency response and zero mean errors, $\tilde{\boldsymbol{\omega}}_{\beta}$, and ζ from the estimated rate of gyroscope bias drift for the three axes, $\tilde{\boldsymbol{\omega}}_{\zeta}$. Those estimates are shown in Equations(3.44) and (3.45), where $\hat{\boldsymbol{q}}$ is a generic unity quaternion.

$$\beta = \left\| \frac{1}{2}\hat{\boldsymbol{q}} \otimes [0 \quad \tilde{\omega}_{\beta_x} \quad \tilde{\omega}_{\beta_y} \quad \tilde{\omega}_{\beta_z}] \right\| = \sqrt{\frac{3}{4}} \cdot \|\tilde{\boldsymbol{\omega}}_{\beta}\| \tag{3.44}$$

$$\zeta = \sqrt{\frac{3}{4}} \cdot \|\tilde{\boldsymbol{\omega}}_{\zeta}\| \tag{3.45}$$

Mahony filter

Another CF relevant implementation was developed by Mahony *et al.* [64, 62, 61], which has low computational needs and uses a Proportional-Integral (PI) feedback of the angular error ${}^{S}\boldsymbol{\omega}_{\epsilon,t}$. The implementation of this filter can be easily understood following the high-level block diagram in Figure 3.8. As can be seen, the error ${}^{S}\boldsymbol{\omega}_{\epsilon,t}$ is obtained from the sum of the cross product between sensors normalized vectors, and the previous orientation estimate. This error can be used similarly as in the OMdF, in order to correct the offset drift of the gyroscope through the integral gain in the *PI* block. The error is fed through a PI gain controller to correct orientation errors, then integrated and normalized to obtain the orientation quaternion estimate. In this work, this filter is designated as the Adapted Mahony Filter (AMhF), because the original version was described using rotation matrices.

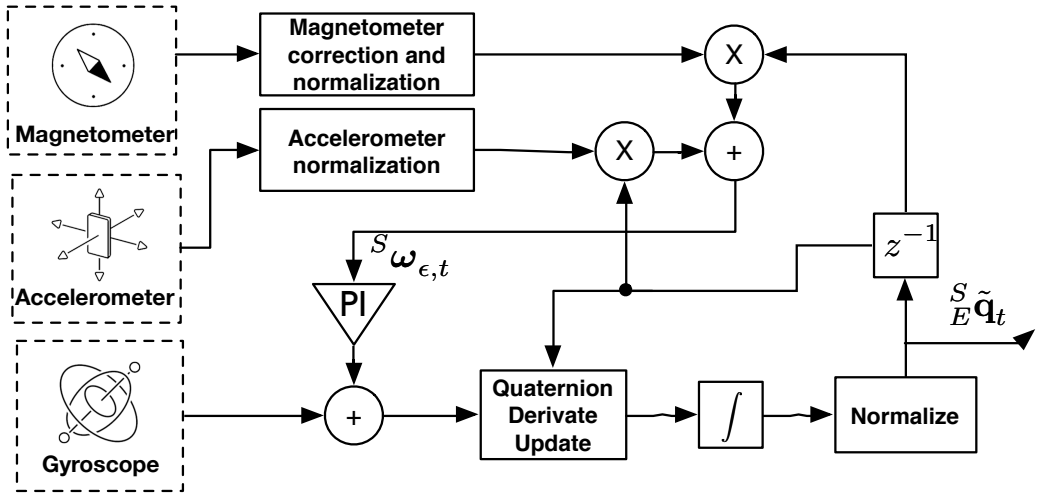

Figure 3.8: Mahony filter block diagram.

The angular error, ${}^{S}\boldsymbol{\omega}_{\epsilon,t}$, is relative to the difference between the measured orientation and the predicted one. In this case no GDA is used to calculate it, instead it is obtained with the cross product between the orientation given by magnetometer measures, ${}^{S}\hat{\boldsymbol{m}}_{t}$ and the predicted one, ${}^{S}\hat{\boldsymbol{b}}_{t}$, plus the gravitational field direction measured by the accelerometer, ${}^{S}\hat{\boldsymbol{a}}_{t}$, and the predicted one, ${}^{S}\hat{\boldsymbol{g}}_{t}$, as given by Equations (3.46-3.48).

$$ {}^{S}\boldsymbol{\omega}_{\epsilon,t} = {}^{S}\hat{\boldsymbol{a}}_t \times {}^{S}\hat{\boldsymbol{g}}_t + {}^{S}\hat{\boldsymbol{m}}_t \times {}^{S}\hat{\boldsymbol{b}}_t \tag{3.46} $$

$$ {}^{S}\hat{\boldsymbol{b}}_t = {}^{S}_{E}\tilde{\boldsymbol{q}}^{*}_{t-1} \otimes {}^{E}\hat{\boldsymbol{b}} \otimes {}^{S}_{E}\tilde{\boldsymbol{q}}_{t-1} \tag{3.47} $$

$$ {}^{S}\hat{\boldsymbol{g}}_t = {}^{S}_{E}\tilde{\boldsymbol{q}}^{*}_{t-1} \otimes {}^{E}\hat{\boldsymbol{g}} \otimes {}^{S}_{E}\tilde{\boldsymbol{q}}_{t-1} \tag{3.48} $$

The cinematic equation for the orientation of the sensor (3.49), regarding an innovation vector $\boldsymbol{\varphi}_t$, is obtained through a PI gain of the angular error described in Equation (3.50).

$$ {}^{S}_{E}\dot{\boldsymbol{q}}_{\omega,t} = \frac{1}{2} {}^{S}_{E}\tilde{\boldsymbol{q}}_{t-1} \otimes \left({}^{S}\boldsymbol{\omega}_t + \boldsymbol{\varphi}_t \right) \tag{3.49} $$

$$ \boldsymbol{\varphi}_t = k_p \cdot {}^{S}\boldsymbol{\omega}_{\epsilon,t} + k_i \cdot \int {}^{S}\boldsymbol{\omega}_{\epsilon,t} \tag{3.50} $$

The gains k_p and k_i correspond to the proportional and integral gains, respectively. The proportional gain controls the frequency value that delimits the importance of the gyroscope sensor information versus the accelerometer and the magnetometer sensors data. The integral gain is adjusted to correct the gyroscope offset drift.

The error is proportional to the sine of the angle between the measured directions and the expected ones (cross product). The feedback of the error through the PI controller into the gyroscope measurements, forces the estimated orientation to follow the reference vectors and, this way, the gyroscope offset and gyroscope offset drift is minimized.

3.3.4 Kalman filters

The Kalman Filter is an optimal estimator for linear zero mean Gaussian noise processes. It iteratively estimates a prediction and correction of the state, considering the expected values, the previous values of the state and the information from the sensors. For the kind of sensors and application used at this work, the EKF is often used, with good results, because the system is highly non-linear. It is possible to estimate the gyroscope offset drift and magnetic distortions, although that usually means increasing the number of state variables. In [22], Roetenberg *et al.* purposed a complementary EKF that could estimate the magnetic distortion error, the gyroscope bias error and the orientation error. Also A. M. Sabatini [20] implemented a quaternion-based EKF for determining the orientation of a body frame. However, these approaches have computational requirements that are incompatible with the goal of this study, due to calculations needed to compute the mean square errors, inverse matrices, multiple multiplications of matrices and the determination of the matrices Jacobians. For the assigned goals, the solution was to use a regular KF, based on the work of *Comotti et al.* [65], even though simplifications had to be done.

For a space-state model, the real state $\boldsymbol{x} \in \Re^n$ of the system can be given by the linear

stochastic difference Equation (3.51) [66].

$$\boldsymbol{x}_t = \boldsymbol{A}\boldsymbol{x}_{t-1} + \boldsymbol{B}\boldsymbol{u}_t + \boldsymbol{w}_t \tag{3.51}$$

Where, $\boldsymbol{x}_t$ represents the state vector at step t, $\boldsymbol{A}$ is the state transition model which is applied to the previous state $\boldsymbol{x}_{t-1}$, $\boldsymbol{B}$ is the control-input model which is applied to the control vector $\boldsymbol{u}_t$, and $\boldsymbol{w}_t$ describes the process noise, assumed to have zero mean.

The measurements, $\boldsymbol{z} \in \Re^m$, are represented in Equation (3.52), where $\boldsymbol{H}$ is the observation model matrix, $\boldsymbol{v}_t$, the measurement noise and $\boldsymbol{z}_t$ is the measurements vector at step t.

$$\boldsymbol{z}_t = \boldsymbol{H}\boldsymbol{x}_t + \boldsymbol{v}_t \tag{3.52}$$

The process and measurement noise, $\boldsymbol{w}_t$ and $\boldsymbol{v}_t$, respectively, are assumed to be white noise independent variables, with a gaussian probabilistic distribution. So, matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ are defined as the process noise covariance and the measurement noise covariance (equations (3.53-3.55)).

$$\boldsymbol{Q} = \left[\boldsymbol{w}_i\ \boldsymbol{w}_j^T\right] = \begin{cases} var(\boldsymbol{w}), & i = j \\ 0, & i \neq j \end{cases} \tag{3.53}$$

$$\boldsymbol{R} = \left[\boldsymbol{v}_i\ \boldsymbol{v}_j^T\right] = \begin{cases} var(\boldsymbol{v}), & i = j \\ 0, & i \neq j \end{cases} \tag{3.54}$$

$$\left[\boldsymbol{w}_i\ \boldsymbol{v}_j^T\right] = 0, \quad \forall\, i, j \tag{3.55}$$

The filter estimates the state of a process using a form of feedback control. Kalman equations are divided in two groups: prediction and correction. The prediction equations obtain the *a priori* state estimation, the correction equations, feedback from measurements into the *a priori* estimate, resulting in an improved *a posteriori* estimate. The algorithm continually executes a prediction and correction of the process estimates. Let $(_{t|t-1})$ define a multi-dimensional variable as a *a priori* estimate; then, $\tilde{\boldsymbol{x}}_{t|t-1}$ is the *a priori* state estimation based on the last *a posteriori* estimate; $\tilde{\boldsymbol{x}}_t$ is the *a posteriori* state estimate, being the actual estimated state regarding the current observations.

The *a posteriori* state estimation must be calculated as a linear combination of the *a priori* estimate $\tilde{\boldsymbol{x}}_{t|t-1}$, and the weighted difference between an actual measurement $\boldsymbol{z}_t$ and a predicted one, $\boldsymbol{H}\tilde{\boldsymbol{x}}_{t|t-1}$, Equation (3.56).

$$\tilde{\boldsymbol{x}}_t = \tilde{\boldsymbol{x}}_{t|t-1} + \boldsymbol{K}_t(\boldsymbol{z}_t - \boldsymbol{H}\tilde{\boldsymbol{x}}_{t|t-1}) \tag{3.56}$$

The difference $\boldsymbol{K}(\boldsymbol{z}_t - \boldsymbol{H}\tilde{\boldsymbol{x}}_{t|t-1})$ is called the measurement innovation, or the residual,

which reflects the discrepancy between the real measurement and the predicted one. $\boldsymbol{K}_t$, referred to as Kalman gain, is used to minimize the *a posteriori* covariance error, and it can be calculated with Equation (3.57), where $\boldsymbol{P}_{t|t-1}$ is the *a priori* estimate error covariance matrix.

$$\boldsymbol{K}_t = \boldsymbol{P}_{t|t-1}\boldsymbol{H}^T(\boldsymbol{H}\boldsymbol{P}_{t|t-1}\boldsymbol{H}^T + \boldsymbol{R})^{-1} \tag{3.57}$$

$\boldsymbol{P}$, the error covariance matrix is updated both in the prediction and update state, providing a sense of the state estimate accuracy.

Kalman filter works in a prediction and correction loop, through Equations (3.58-3.59).

Prediction:

$$\begin{aligned} \tilde{\boldsymbol{x}}_{t|t-1} &= \boldsymbol{A}\tilde{\boldsymbol{x}}_{t-1} + \boldsymbol{B}\boldsymbol{u}_{t-1} \\ \boldsymbol{P}_{t|t-1} &= \boldsymbol{A}\boldsymbol{P}_{t-1}\boldsymbol{A}^T + \boldsymbol{Q} \end{aligned} \tag{3.58}$$

Correction:

$$\begin{aligned} \boldsymbol{K}_t &= \boldsymbol{P}_{t|t-1}\boldsymbol{H}^T(\boldsymbol{H}\boldsymbol{P}_{t|t-1}\boldsymbol{H}^T + \boldsymbol{R})^{-1} \\ \tilde{\boldsymbol{x}}_t &= \tilde{\boldsymbol{x}}_{t|t-1} + \boldsymbol{K}(\boldsymbol{z}_{t|t-1} - \boldsymbol{H}\tilde{\boldsymbol{x}}_{t|t-1}) \\ \boldsymbol{P}_t &= (\boldsymbol{I} - \boldsymbol{K}_t\boldsymbol{H})\boldsymbol{P}_{t|t-1} \end{aligned} \tag{3.59}$$

The block diagram in Figure 3.9 describes the filter operation, in this work. The dashed arrow to the **Q** matrix means that only an initial estimate of the covariance of this sensor is made, with no updates after this. The process of the implementation of this filter is explained in more in the remainder of this Section.

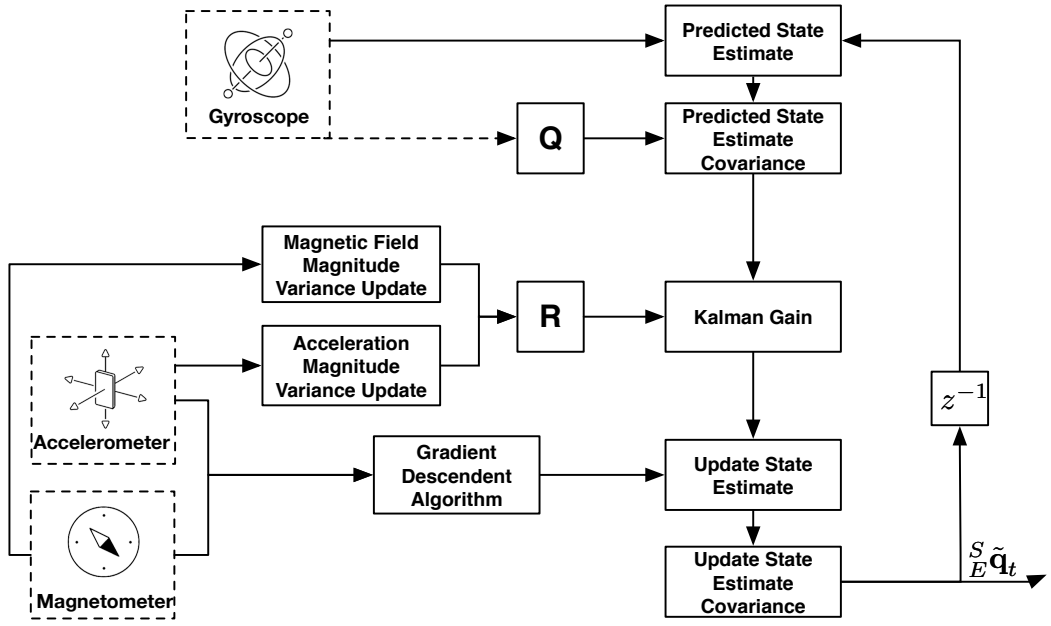

Figure 3.9: Kalman filter block diagram

To minimize the computation complexity of the filter, the number of state variables has to be small. The simplest state of the system are the values of the quaternion components, corresponding to only four state variables. The state of the system, chosen for the implementation of the KF, is represented in Equation (3.60).

$$\boldsymbol{x} = [\ q_s \quad q_x \quad q_y \quad q_z\ ]^T \tag{3.60}$$

Considering Equations (3.12) and (3.13), the prediction of the state is given by Equation (3.61). Thus, the state transition model is as in Equation (3.62).

$$\boldsymbol{x}_t = \boldsymbol{x}_{t-1} + {}^{S}_{E}\dot{\boldsymbol{q}}_t \cdot \delta t = \boldsymbol{A} \cdot \boldsymbol{x}_{t-1} \tag{3.61}$$

$$\boldsymbol{A}_t = \begin{bmatrix} 1 & -\frac{1}{2}\cdot\omega_{x,t}\cdot\delta t & -\frac{1}{2}\cdot\omega_{y,t}\cdot\delta t & -\frac{1}{2}\cdot\omega_{z,t}\cdot\delta t \\ \frac{1}{2}\cdot\omega_{x,t}\cdot\delta t & 1 & \frac{1}{2}\cdot\omega_{z,t}\cdot\delta t & -\frac{1}{2}\cdot\omega_{y,t}\cdot\delta t \\ \frac{1}{2}\cdot\omega_{y,t}\cdot\delta t & -\frac{1}{2}\cdot\omega_{z,t}\cdot\delta t & 1 & \frac{1}{2}\cdot\omega_{x,t}\cdot\delta t \\ \frac{1}{2}\cdot\omega_{z,t}\cdot\delta t & \frac{1}{2}\cdot\omega_{y,t}\cdot\delta t & -\frac{1}{2}\cdot\omega_{x,t}\cdot\delta t & 1 \end{bmatrix} \tag{3.62}$$

Assuming a constant variance $\boldsymbol{\sigma}^2_{\omega}$ per axis on the gyroscope measurements, the process covariance matrix $\boldsymbol{Q}$ can be computed with the variance of the measurements of the gyroscope at rest, as is shown in Equation (3.63).

$$\boldsymbol{Q} = \begin{bmatrix} \sigma^2_{\omega_x}+\sigma^2_{\omega_y}+\sigma^2_{\omega_z} & -\sigma^2_{\omega_x}+\sigma^2_{\omega_y}-\sigma^2_{\omega_z} & -\sigma^2_{\omega_x}-\sigma^2_{\omega_y}+\sigma^2_{\omega_z} & \sigma^2_{\omega_x}-\sigma^2_{\omega_y}-\sigma^2_{\omega_z} \\ -\sigma^2_{\omega_x}+\sigma^2_{\omega_y}-\sigma^2_{\omega_z} & \sigma^2_{\omega_x}+\sigma^2_{\omega_y}+\sigma^2_{\omega_z} & \sigma^2_{\omega_x}-\sigma^2_{\omega_y}-\sigma^2_{\omega_z} & -\sigma^2_{\omega_x}-\sigma^2_{\omega_y}+\sigma^2_{\omega_z} \\ -\sigma^2_{\omega_x}-\sigma^2_{\omega_y}+\sigma^2_{\omega_z} & \sigma^2_{\omega_x}-\sigma^2_{\omega_y}-\sigma^2_{\omega_z} & \sigma^2_{\omega_x}+\sigma^2_{\omega_y}+\sigma^2_{\omega_z} & -\sigma^2_{\omega_x}+\sigma^2_{\omega_y}-\sigma^2_{\omega_z} \\ \sigma^2_{\omega_x}-\sigma^2_{\omega_y}-\sigma^2_{\omega_z} & -\sigma^2_{\omega_x}-\sigma^2_{\omega_y}+\sigma^2_{\omega_z} & -\sigma^2_{\omega_x}+\sigma^2_{\omega_y}-\sigma^2_{\omega_z} & \sigma^2_{\omega_x}+\sigma^2_{\omega_y}+\sigma^2_{\omega_z} \end{bmatrix} \tag{3.63}$$

The measurements $\boldsymbol{z}_t$ are calculated using the GDA described in Section 3.3.2, which retrieves the quaternion estimate for at step t. $\boldsymbol{H}$ is set as the identity matrix because the state space corresponds to the observations space.

In the update step, the covariance matrix $\boldsymbol{R}$ is updated based on the reliability of the measurements from the accelerometer and the magnetometer. The confidence of the observations should fall in the case of quick movements, where the measurements are corrupted by other accelerations, and in moments where there are disturbances on the magnetic field. To do this, the measured observation sensors magnitude is verified to determine deviations from the expected gravity or expected magnetic field. Therefore, the standard deviation of the observation sensors is determined using formula in Equation (3.64). This approach has the disadvantage of losing the information about which one of

the observation sensors is providing noisy data [22].

$$\boldsymbol{\sigma}_s = \boldsymbol{\sigma}_m \cdot \| \| \boldsymbol{m}_t \| - \| \boldsymbol{d}_t \| \| + \boldsymbol{\sigma}_a \cdot \| \| \boldsymbol{a}_t \| - \| \boldsymbol{g}_t \| \| \tag{3.64}$$

3.4 Position from acceleration

To accurately estimate position, a set of inertial, magnetic, optical and GPS sensors is usually used, with different levels of accuracy, depending on the technology used. However, for indoor low computational devices, the 3D position estimate is still a challenge, since GPS sensors cannot be used indoors and estimates from optical sensors have usually high computational needs and problems with lightning conditions. The accelerometers can be used, with a major drawback, given the exponential growth in error due to the double integration of its data. There are two major sources of error: the orientation error bias and the double integration of noise/bias. Given these issues, the position estimate from double integration can only obtain acceptable results if performed on small periods of time, enabling its use to estimate gestures and small hand movements displacements [13, 67].

The position can be generically estimate from an accelerometer, using Equations (3.65 - 3.67). The corrected acceleration vector in the global frame, ${}^{E}\tilde{\boldsymbol{a}}_{c,t}$, is drawn from the measurements of the accelerometer, ${}^{S}\boldsymbol{a}_t$, without the gravity, which allows only the accelerations due to motion to be taken into account. Then, an integration of this value (equation (3.66)) obtains the velocity of the sensor in the global frame, ${}^{E}\tilde{\boldsymbol{s}}_t$, assuming that the previous velocity, ${}^{E}\tilde{\boldsymbol{s}}_{t-1}$, is known. The integration (equation (3.67)) of the velocity obtains the position, ${}^{E}\tilde{\boldsymbol{p}}_t$, using the same method. The δt represents the accelerometer sampling time between steps t and $t-1$.

$${}^{E}\tilde{\boldsymbol{a}}_{c,t} = {}^{S}_{E}\tilde{\boldsymbol{q}}_t \otimes {}^{S}\boldsymbol{a}_t \otimes {}^{S}_{E}\tilde{\boldsymbol{q}}_t^{*} - {}^{E}\boldsymbol{g} \tag{3.65}$$

$${}^{E}\tilde{\boldsymbol{s}}_t = {}^{E}\tilde{\boldsymbol{s}}_{t-1} + {}^{E}\tilde{\boldsymbol{a}}_{c,t} \cdot \delta t \tag{3.66}$$

$${}^{E}\tilde{\boldsymbol{p}}_t = {}^{E}\tilde{\boldsymbol{p}}_{t-1} + {}^{E}\tilde{\boldsymbol{s}}_t \cdot \delta t \tag{3.67}$$

A constant bias, ${}^{E}\boldsymbol{a}_\epsilon$, on the accelerometer output will turn into an error in position (${}^{E}\boldsymbol{p}_\epsilon$) that grows quadratically with time, as shown in Equation (3.68), where n is the step number. The bias has to be removed by calibration and preferentially estimated over time to correct any bias drift.

$${}^{E}\boldsymbol{p}_{\epsilon,\, t+n} = \frac{1}{2}\, {}^{E}\boldsymbol{a}_\epsilon \cdot (n \cdot \delta t)^2 \tag{3.68}$$

The integration of white noise causes what is know as "random walk", which is an error on the position estimated with zero mean, but with a growing variance over time, proportional to $(n \cdot \delta t)^{\frac{3}{2}}$ [68].

Orientation errors cause incorrect projections of the acceleration signals onto the global coordinates frame, leading to the integration of the acceleration in the wrong direction, and to the gravity acceleration not being correctly removed, which reports back to the bias problem.

All these errors propagate and end up accumulating in the position estimate. For instance, let a sensor remain still in the world frame but with an error of 0.5 degrees in the tilt orientation estimate. This error will cause a component of acceleration projected on the horizontal plane, due to a wrong gravity removal, of about 0.086 $\mathrm{m{\cdot}s^{-2}}$. With only this orientation error the position estimate can end up to be wrong by 8.6 meters after 10 seconds. When the other error factors are brought together, it is easily understandable that the position estimate is very difficult, especially for long periods of time. There are some techniques that can be implemented to minimize the error, which can be used for gesture recognition involving specific movements during very small periods of time. These techniques are explained in more detail in Chapter 4.

Chapter 4

Motion-based Remote Control Device

Both fusion filters and position estimate methods, studied in the previous chapter, constitute generic approaches that allow to achieve an approximated orientation and position of a rigid body, using the described sensors. The filters consider stable measurements of the fields and account only for the major and more common error causes. Improvements can be made to the filters to further adapt them to the specific environment of operation where other error causes may be present and, in contrast, use application specificities to reduce the overall error. Some of the points that can be improved were already referred, but in this chapter the modifications performed to the filters that allow the detection and reaction to undesirable factors are detailed. The first section of this chapter presents the modifications implemented in the original CF (described in Section 3.3) and a description of the proper signal conditioning that should be applied to the accelerometer data, in order to achieve a reasonable level of accuracy for the pose estimate. The aim of these modifications is to improve the orientation estimate in indoor environments. The second section proceeds with a description of the implementation of such improved filters into a working prototype of an RCD with air-mouse capabilities.

4.1 Low-complexity fusion filters

Adapting the filters to the specific environment of this work, allows to achieve better results. Some adjustments to the original CF are performed in this section. The method in Section 4.1.1 was implemented and its performance quantitatively analysed in Chapter 5. The method in Section 4.1.2 was implemented later and, due to time constrains, its performance was not quantitatively analysed, with subjective results being presented in Chapter 5.

4.1.1 Sensors reliability and adaptive gains

The original CFs described by Mahony and Madgwick [61, 33] in Section 3.3 have a significant disadvantage: when compared with KFs, they do not estimate the reliability of the measurements. However, one can add this capability, obtaining improved and more robust results while maintaining the typical low-computational needs of CFs [34]. This is done with time variant parameters, that set weights to the measurements, according to the reliability of the sensor in that moment.

Several methods were implemented and analysed to adjust the level of contribution of each sensor:

1. **Gyroscope *versus* Observation sensors - according to dynamic conditions.**

   The cutoff frequency of the CF is a threshold value which controls the fusion between the reliable gyroscope signals with the reliable observation sensors signals. This frequency can be adjusted according to the dynamic conditions of the system. The cutoff frequency is mediated through the β gain in the case of the Madgwick filter, and through the proportional gain, k_p, in the case of the Mahony filter. As the gyroscope and accelerometer are opposites (in frequency) in terms of retrieving accurate readings in dynamic conditions, this adaptation further extends the complementarity of the CF, improving orientation estimate both by increasing the quality of the information of the signals and by decreasing the convergence time of the filter.

   To evaluate the dynamic conditions the rate of change of orientation was used, measured by the gyroscope, ${}^{S}_{E}\dot{\boldsymbol{q}}_{\omega,t}$ (Equation (3.12)). The norm of that vector is directly related with the dynamic conditions of the system and inversely related to the cutoff frequency of the CFs. So, Equation (4.1) can be used to adjust the β_t gain, having for base the β_{ref} gain, assuring that β_t is never negative or 0.

$$\beta_t = (\ 1 - \|{}^{S}_{E}\dot{\boldsymbol{q}}_{\omega,t}\|\ ) \cdot \beta_{ref}\ , \qquad \|{}^{S}_{E}\dot{\boldsymbol{q}}_{\omega,t}\| < 1 \tag{4.1}$$

2. **Observation sensors reliability - comparing the magnitude of the measured vector against the expected one.**

   Usually, misleading measurements of the observation sensors constitute the major error source. The observation sensors measure physical quantities that can be the sum of many contributions. For this specific application, only specific contributions of the measurements are targeted in the case of the accelerometer and the magnetometer, namely, the gravity acceleration and the Earth's magnetic field, respectively. These fields are very well characterized and the filters can be adapted

to respond to deviations on the measurements.

Regardless the dynamic conditions and the magnetic field of the environment, the gyroscope readings are essentially dominated by angular velocity, so, its measurements are more trustful than the measurements of an observation sensor when disturbances occur.

Regarding the aforementioned considerations, if one, or both, of the input vectors used in the GDA, ${}^{S}\hat{\boldsymbol{a}}$ and ${}^{S}\hat{\boldsymbol{m}}_t$, are considerably affected by a disturbance, that vector is discarded and it is used his expected value instead, therefore, the integration of the gyroscope data prevails [34, 31]. The vectors ${}^{S}\hat{\boldsymbol{a}}'$ and ${}^{S}\hat{\boldsymbol{m}}'$ can be calculated through the expressions in Equations (4.2-4.4). Equation (4.3) allows an adaptive reference vector for more robustness. The κ_a and κ_m values represent threshold factors from where the measurements of the accelerometer and magnetometer, respectively, should be disregarded.

$$
{}^{S}\hat{\boldsymbol{a}}'_t = \begin{cases} \dfrac{{}^{S}\boldsymbol{a}_t}{\|{}^{S}\boldsymbol{a}_t\|}, & if \quad |\,\|{}^{S}\boldsymbol{a}_t\| - \|{}^{E}\boldsymbol{g}\|\,| \leq \kappa_a \\ {}^{S}_{E}\tilde{\boldsymbol{q}}^{*}_{t-1} \otimes {}^{E}\boldsymbol{g} \otimes {}^{S}_{E}\tilde{\boldsymbol{q}}_{t-1}, & otherwise \end{cases} \tag{4.2}
$$

$$
{}^{E}\hat{\mathbf{h}}_t = \begin{cases} \dfrac{{}^{S}\boldsymbol{m}_{t=0}}{\|{}^{S}\boldsymbol{m}_{t=0}\|}, & if \quad |\,\|{}^{S}\boldsymbol{m}_t\| - \|{}^{E}\boldsymbol{m}_{t=0}\|\,| \leq \kappa_m \\ {}^{S}_{E}\tilde{\boldsymbol{q}}_{t-1} \otimes {}^{S}\hat{\boldsymbol{m}}'_{t-1} \otimes {}^{S}_{E}\tilde{\boldsymbol{q}}^{*}_{t-1}, & otherwise \end{cases} \tag{4.3}
$$

$$
{}^{S}\hat{\boldsymbol{m}}'_t = \begin{cases} \dfrac{{}^{S}\boldsymbol{m}_t}{\|{}^{S}\boldsymbol{m}_t\|}, & if \quad |\,\|{}^{S}\boldsymbol{m}_t\| - \|{}^{E}\boldsymbol{m}_{t=0}\|\,| \leq \kappa_m \\ {}^{S}_{E}\tilde{\boldsymbol{q}}^{*}_{t-1} \otimes {}^{E}\hat{\mathbf{h}}_t \otimes {}^{S}_{E}\tilde{\boldsymbol{q}}_{t-1}, & otherwise \end{cases} \tag{4.4}
$$

In the current work, the implementation of these improvements into the regular CF does not include the magnetic declination compensation (used in [33] and described in Section 3.3.3), since the disturbances are accounted in a different way. Therefore, for the GDA, the complete solution of Equations (3.17-3.18) has to be used, instead of Equations (3.29-3.30).

Both OMdF and AMhF were adapted using the method described in this Section, with the altered filters here described being hereinafter denoted as Adaptive-Gain Madgwick Filter (AGMdF) and Adaptive-Gain Mahony Filter (AGMhF), respectively.

4.1.2 Yaw drift compensation

The accelerometer and the gyroscope achieve accurate results on the roll/pitch estimate, while distorted measurements of Earth's magnetic field may introduce errors to these estimates. The magnetometer can be constrained to correct only the yaw drift and not

interfere with the estimates from the inertial sensors, or, alternatively, an optical flow sensor can be used to perform the same correction.

Recently, Invensense has included in their IMU and MARG units a Digital Motion Processor (DMP) that can compute the orientation based on the accelerometer and gyroscope data, relieving the main processor of applications of this task, while decreasing the energy consumption. Although this filter is proprietary, this new capacity can provide some advantages. The algorithms explained in this Section, integrate magnetometer data after the orientation estimate using the accelerometer and the gyroscope data fusion. This method is used to minimize drift from the orientation retrieved by the DMP. The computational needs to performed this task are smaller than having to perform the complete fusion algorithm in the microcontroller, reducing power consumption (Invensense states a DMP consumption of 300 μA [69]).

Magnetometer-based yaw drift compensation

In order to constrain the magnetometer observation correction to the yaw angle of the orientation estimate, the heading angle, $\psi_{\boldsymbol{m}_t}$, can be computed from the current magnetic field readings using Equation (4.5) and Equation (4.6)[1], where ${}^{S}_{E}\tilde{\boldsymbol{q}}_t$ is the current orientation estimate retrieved by the gyroscope and accelerometer data fusion.

$$ {}^{E}\tilde{\boldsymbol{m}}_t = {}^{S}_{E}\tilde{\boldsymbol{q}}_t \otimes {}^{S}\boldsymbol{m}_t \otimes {}^{S}_{E}\tilde{\boldsymbol{q}}_t^{*} \tag{4.5} $$

$$ \psi_{\boldsymbol{m}_t} = atan2(\ {}^{E}\tilde{m}_{y,t},\ {}^{E}\tilde{m}_{x,t}) \tag{4.6} $$

Then, comparing the yaw angle from the inertial sensors orientation estimate, computed as in Equation (4.7), with the heading angle, it is possible to calculate the yaw error, ϵ_ψ, through Equation (4.8).

$$ \psi_{\boldsymbol{q}_t} = atan2\left(\ 2\ (q_y q_z + q_s q_x),\ q_s^2 - q_x^2 - q_y^2 + q_z^2\ \right) \tag{4.7} $$

$$ \epsilon_\psi = \psi_{\boldsymbol{q}_t} - \psi_{\boldsymbol{m}_t} \tag{4.8} $$

Having calculated the yaw error, the drifted quaternion can be corrected through Equation (4.9). Here, ${}^{S}_{E}\tilde{\boldsymbol{q}}'_t$ is the corrected, or the compensated, orientation estimate that has integrated the information given by the magnetometer.

$$ {}^{S}_{E}\tilde{\boldsymbol{q}}'_t = \left[\ \cos\tfrac{\epsilon_\psi}{2} \quad 0 \quad 0 \quad \sin\tfrac{\epsilon_\psi}{2}\ \right] \otimes {}^{S}_{E}\tilde{\boldsymbol{q}}_t \tag{4.9} $$

This approach, during tests using an air mouse (see Section 4.2), did not reveal the

[1] $atan2$ denotes the arctangent function, a standard C library function.

necessary accuracy to maintain a satisfactory QoE. Although the Hall effect magnetic sensor in the MPU9250 has a sensibility of 600 μT, the noise level demonstrated by this sensor (very similar to the one in Figure 3.3) is high and has a considerable effect on the target application. Even low-pass filtering the magnetometer signal and using an adaptive gain, conditioning the yaw correction convergence, was not enough to obtain good results. Using the PNI magneto-inductive sensor, improved the results considerably, but as this sensor could not be integrated in the final prototype due to its cost, a different technique was explored in order to verify if the desired performance could be achieved. This technique is detailed below.

Differential approach

In order to cope with the described disturbances in the magnetic field of the indoor global frame presented in Section 3.1.3, some assumptions are made:

1. Only a fixed direction is needed in the working environment, not necessarily the true North;
2. The magnetic field magnitude is not relevant, it only has to be measurable;
3. Changing environment will result in different measurements from the calibrated magnetometer (this can be something as simple as putting down the measurement object at the top of a table or wandering around in the room).

With these assumptions a method for the yaw drift correction can be implemented by saving magnetic reference points as the platform changes orientation [49]. The reference points are stored as a φ_{ref} angle obtained from Equation (4.11) and (4.10), ${}^{E}\tilde{\boldsymbol{m}}_{t_{ref}}$, which corresponds to a reference measurement of the magnetic field in the world frame. The ${}^{S}_{E}\tilde{\boldsymbol{q}}_{t_{ref}}$ is the current orientation estimate retrieved by the gyroscope and accelerometer data fusion.

$$ {}^{E}\tilde{\boldsymbol{m}}_{t_{ref}} = {}^{S}_{E}\tilde{\boldsymbol{q}}_{t_{ref}} \otimes {}^{S}\boldsymbol{m}_{t_{ref}} \otimes {}^{S}_{E}\tilde{\boldsymbol{q}}^{*}_{t_{ref}} \tag{4.10} $$

$$ \varphi_{ref} = atan2({}^{E}\tilde{m}_{y,t_{ref}}, {}^{E}\tilde{m}_{x,t_{ref}}) \tag{4.11} $$

A measurement of the magnetic field (in the world frame) made after the reference as been defined, ${}^{E}\tilde{\boldsymbol{m}}_{t'}$, providing $t' > t_{ref}$ in Equation (4.12), in the case of no distortions in the magnetic field, would satisfy the relation $\varphi_{ref} - \varphi' = 0$. Here, symbol φ' stands for the angle of the projected magnetic field vector in the horizontal plane (same calculation as φ_{ref} but calculated in posterior steps), this is explicit in Equation (4.13).

$$ {}^{E}\tilde{\boldsymbol{m}}_{t'} = {}^{S}_{E}\tilde{\boldsymbol{q}}_{t'} \otimes {}^{S}\boldsymbol{m}_{t'} \otimes {}^{S}_{E}\tilde{\boldsymbol{q}}^{*}_{t'} \tag{4.12} $$

$$ \varphi' = atan2({}^{E}\tilde{m}_{y,t'}, {}^{E}\tilde{m}_{x,t'}) \tag{4.13} $$

The previous argument leads to Equation (4.14), where is possible to compute an angle error, ϵ_φ, proportional to the error in the quaternion for the yaw rotation.

$$\epsilon_\varphi = \varphi_{ref} - \varphi' \tag{4.14}$$

The angle error can be used to correct yaw rotation error, following the expression in Equation (4.15), where ${}^S_E\tilde{\boldsymbol{q}}'_t$ is the estimated quaternion after correction. The α_2 gain is a constant that should provide an adequate convergence to the error correction. It can also be a variable gain that changes according to the dynamic conditions.

$${}^S_E\tilde{\boldsymbol{q}}'_t = \left[\begin{array}{cccc} \sqrt{1-(\alpha_2 \cdot \epsilon_\varphi)^2} & 0 & 0 & (\alpha_2 \cdot \epsilon_\varphi) \end{array} \right] \otimes {}^S_E\tilde{\boldsymbol{q}}_t \tag{4.15}$$

In real world applications the distortions, specifically the *soft iron* distortions, will cause a growing error as the new sampled points are further away from the reference point. To counteract this problem, new reference points can be added. This point differentiates this approach from the non-differential explained before. Typically, when the orientation exceeds 10°, a new reference point is saved. The nearest reference point is used to calculate the yaw error. This maximum of 10° also allows to linearise the correction, as can be observed in Equation (4.15), avoiding the use of trigonometric functions. To improve robustness, the reference points are analysed and swapped for new ones in case magnetic conditions suffers a substantial change, or if it is considered a bad reference point. The drawback is that, by admitting new reference points to be added, substituting the older ones, the orientation system is no longer absolute. Nevertheless, the local errors on orientation are reduced and yaw drift is reduced.

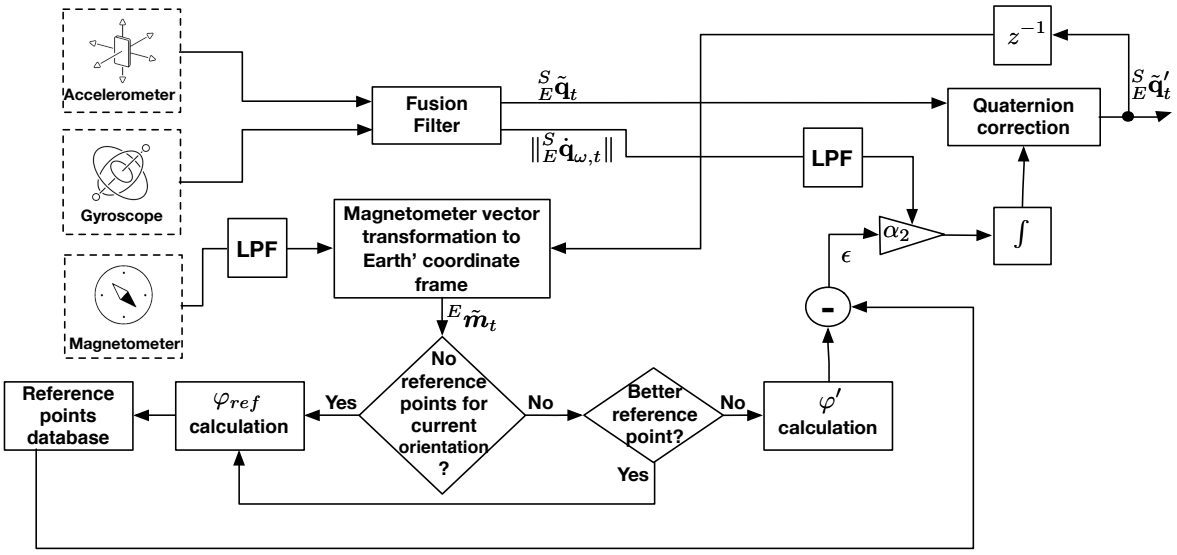

Figure 4.1: Correcting yaw drift using magnetometer block diagram.

The block diagram in Figure 4.1 describes the operation of the implemented filter to eliminate the yaw drift. Note that this correction is limited to the rotation around z axis and is controlled through α_2, an adaptive gain according to the dynamic conditions, for better results. The dynamic conditions signal passes through a Low Pass Filter (LPF) because the delay introduced is necessary for main filter to converge. This implementation is denominated in this dissertations as Yaw Drift Compensation Filter (YDCF).

Optical flow based yaw drift compensation

In spite all the methods described earlier, the magnetometer data can still be unreliable, making it hard to obtain correct estimates of the orientation. Replacing the magnetometer by an optical flow sensor will not achieved absolute orientation relatively to the world frame, but the yaw drift can be significantly reduced. The drawback is that the drift can only be estimated in very specific circumstances, namely: only images with a sufficient number of features (the more high frequency transitions the image has, the more features it presents) captured by the optical sensor are eligible to compute optical flow, otherwise the optical flow is not correctly estimate; it is not a trivial task to distinguish motion due to rotation or due to translation without a proper estimate of the distance of the sensor to the focus surface; in addition, motion displayed in a TV screen will also be detected by the sensor. Nevertheless, at least at rest, the optical flow can be used to prevent drift.

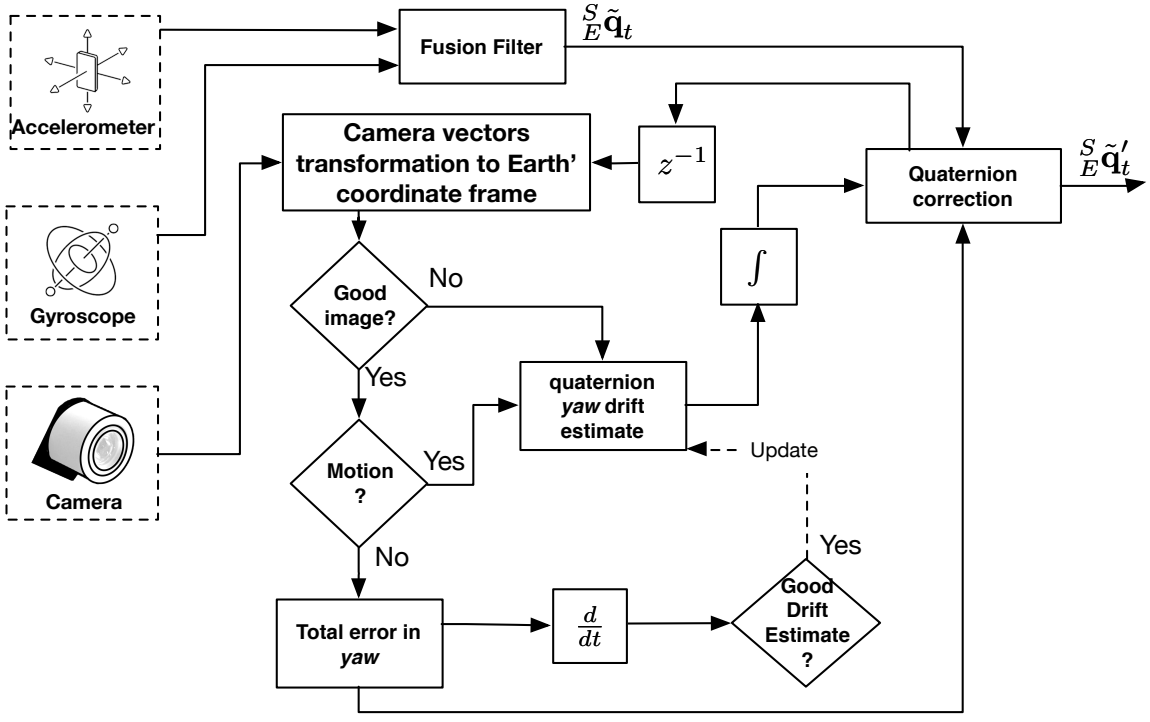

Figure 4.2: Correcting yaw drift using an optical flow sensor block diagram.

The ADNS-3080 provides a "surface quality" component, variable in a scale from 0 to 169, which can be used to assign levels of confidence on the sensor readings. When good images are captured and the sensor does not indicate motion, any change in the orientation estimate is due to drift. The drift can be removed while the body frame of the sensors is not moving, using the last drift estimate when the device is in motion. The correction can be executed using Equation (4.9).

Figure 4.2 presents the process used to remove drift using an optical flow sensor. If the sensor is acquiring good images and no motion is detected, the drift is removed from the estimate and the drift estimates are also updated. When there is motion or no eligible images for optical flow processing, the correction is performed using the last drift estimate.

4.1.3 Minimizing errors for position estimate

As the orientation errors are being mitigated with the proposed solutions from the previous sections, the other error sources described in Section 3.4 have to be treated in order to achieve an acceptable level of accuracy in the position/displacement estimate [67]. In the current work, some important steps of the process where identified and improvements performed, in order to achieve better results. These steps are described in the following points:

Accelerometer calibration

For the position estimate it is very important to calibrate the accelerometer in order to remove bias, axes misalignments and compensate scale factors. This extended calibration needs the active participation of an user and it is explained in more detail in Appendix B.

Bias estimate

Each axis acceleration bias must be estimated in real time in order to remove an eventual bias drift after calibration. The drift is a very slow signal, so a low-pass filter with a large time constant (τ) can be used to perform the bias drift estimate. A moving average filter of 10 seconds can be chosen to do this, for instance. There is no need to save n, the number of measurement samples, if the moving average is calculated with the addition of a new sample while removing an weighted part of the bias $\boldsymbol{\rho}$, as in Equation (4.16).

$$ {}^{E}\tilde{\boldsymbol{\rho}}_t = {}^{E}\tilde{\boldsymbol{\rho}}_{t-1} - \frac{{}^{E}\tilde{\boldsymbol{\rho}}_{t-1}}{n} + \frac{{}^{E}\boldsymbol{a_t}}{n}, \qquad n = \frac{\tau}{\delta t} \tag{4.16} $$

Filtering

Implementing a low-pass filter reduces the noise magnitude from the accelerometer

signal, reducing the "random walk" errors. An high pass filter increases the effectiveness of the bias removal by eliminating the DC components of the signal. The suggestion is to use a first order Butterworth filter because of its simplicity and low computational needs.

Dead-zone (DZ)

Due to noise, the accelerometer provides acceleration values different from zero even when no movements are occurring. These readings, although with small amplitude, can be interpreted as a constant velocity if the acceleration sum does not add to zero. A common way to eliminate this problem is by adding a discrimination window to the accelerometer readings, establishing a threshold value, where the reading is set to zero if smaller. This filtering method is referred as a dead-zone window.

Trapezoidal integration

The normal integration steps add sampling losses due to rectangular integration. The best way to perform the integration is with the trapezoidal rule that approximates the upper end of an integral to a triangle, minimizing the sampling losses. This way, Equations (3.66 , 3.67) have to be substitute by Equations (4.17 , 4.18).

$$ {}^{E}\tilde{\boldsymbol{s}}_t = {}^{E}\tilde{\boldsymbol{s}}_{t-1} + \left({}^{E}\tilde{\boldsymbol{a}}_{c,t-1} + \frac{{}^{E}\tilde{\boldsymbol{a}}_{c,t} - {}^{E}\tilde{\boldsymbol{a}}_{c,t-1}}{2} \right) \cdot \delta t \tag{4.17} $$

$$ {}^{E}\tilde{\boldsymbol{p}}_t = {}^{E}\tilde{\boldsymbol{p}}_{t-1} + \left({}^{E}\tilde{\boldsymbol{s}}_{t-1} + \frac{{}^{E}\tilde{\boldsymbol{s}}_{t} - {}^{E}\tilde{\boldsymbol{s}}_{t-1}}{2} \right) \cdot \delta t \tag{4.18} $$

End of motion detection

In theory, when a movement is performed starting and finishing in a rest position, the integration of all accelerations is equal to zero. However this is not the case when using measuring devices, which results in a velocity different of zero after the sensor body is still. Because of this, it is crucial to force the velocity to zero after no motion is detected. Readings of the acceleration can be compared to zero and, if after a certain amount of time the condition is still valid, the velocity is set to zero.

Applying the described signal conditioning steps should be enough to obtain acceptable results in the position/displacements estimates for small periods of time.

The block diagram in Figure 4.3 describes the sequence of operations implemented to obtain the position from the MARG unit. After the orientation estimate update, the acceleration vector is transformed into world coordinates. The bias estimate is updated and removed from the accelerometer signal as explained in this Section. The gravity is also removed. Hence, the acceleration in the world frame generated only due to motion is obtained. This signal is treated in a Band Pass Filter (BPF) , with a pass band between

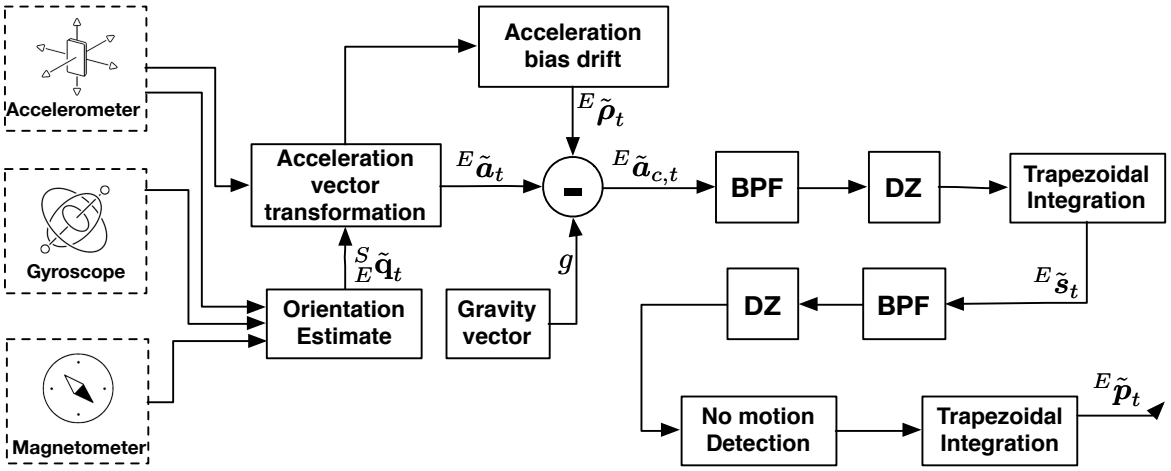

Figure 4.3: Position estimation block diagram.

0.1 Hz and 20 Hz. A Dead Zone (DZ) is applied to eliminate noise and small vibrations. The signal is then integrated with the trapezoidal method to obtain the velocity. The signal passes through another set of filters and DZ, and a *No Motion Detection* block forces velocity to zero if the accelerometer indicates that there is no motion. Finally, another integration step, and the position of the MARG unit is obtained.

4.2 Remote control device prototype

Not all the work developed in the first Section was applied to the prototype due to processing and memory limitations of the main processor unit of the RCD, however the work can be used and included in future optimizations of the code or in a different processing unit. The RCD prototype used for this work was provided by the Tech4Home[2] company, a Small Medium Enterprise (SME) expert in the field of RCDs. It includes eleven keys, directed to the Android market devices. A picture of the remote control can be seen in Figure 4.4. The software and harware needed for the integration of the Android STB was developed by R. Santos [70] in the same scope of this research.

The prototype includes a capacitive proximity sensor under the **OK** key, in order to the user activate the motion-based navigation system (the user only has to place its finger over that key without pressing it). The MARG sensor selected was the MPU-9250 from Invensense, that includes an accelerometer, a gyroscope and a magnetometer. It communicates via an Inter-Integrated Circuit (I2C) with a CC2533F96 8-bit microcontroller from Texas Instruments, which has a 96 KB Flash memory, 6 KB of RAM, operates at 24 MHz and is a SoC solution for wireless systems. It possesses a Radio Frequency for

[2]Tech4Home: www.tech4home.pt; Address: Rua de Fundões n.151, 3700121 S. J. da Madeira, Portugal

Figure 4.4: Remote Control Device prototype.

Consumer Electronics (RF4CE) stack from ZigBee to communicate with a base system (STB).

The RCD implements the flowchart shown in Figure 4.5 and, for demonstration purposes, has six modes of operation. The mode of operation is switched by pressing the combination of keys **Home** + **Menu**.

Idle Mode (IM)

In this mode of operation, only the keys are sent to the STB. The MPU-9250 remains in sleep mode and no motion events are performed nor interrupts from the proximity sensor are taken into account. Besides the information identifying the key, also information about key events are sent, including: key pressed, key released or key repeated. A radio transmission in this mode is performed for each key event.

Relative Mouse Mode (RMM)

When a user activates the proximity sensor, the MARG unit is brought back from sleep to normal operation, activating the accelerometer, the gyroscope and the DMP. Then the relative navigation is calculated on-board with tilt compensation, sending the mouse displacement information to the STB. This information is identified by the operating system through a relative mouse Human Interface Device (HID) profile over Universal Serial Bus (USB), developed in [70].In this mode of operation keys can be send. Data is only transmitted in this mode when there is relative motion or a key is pressed.

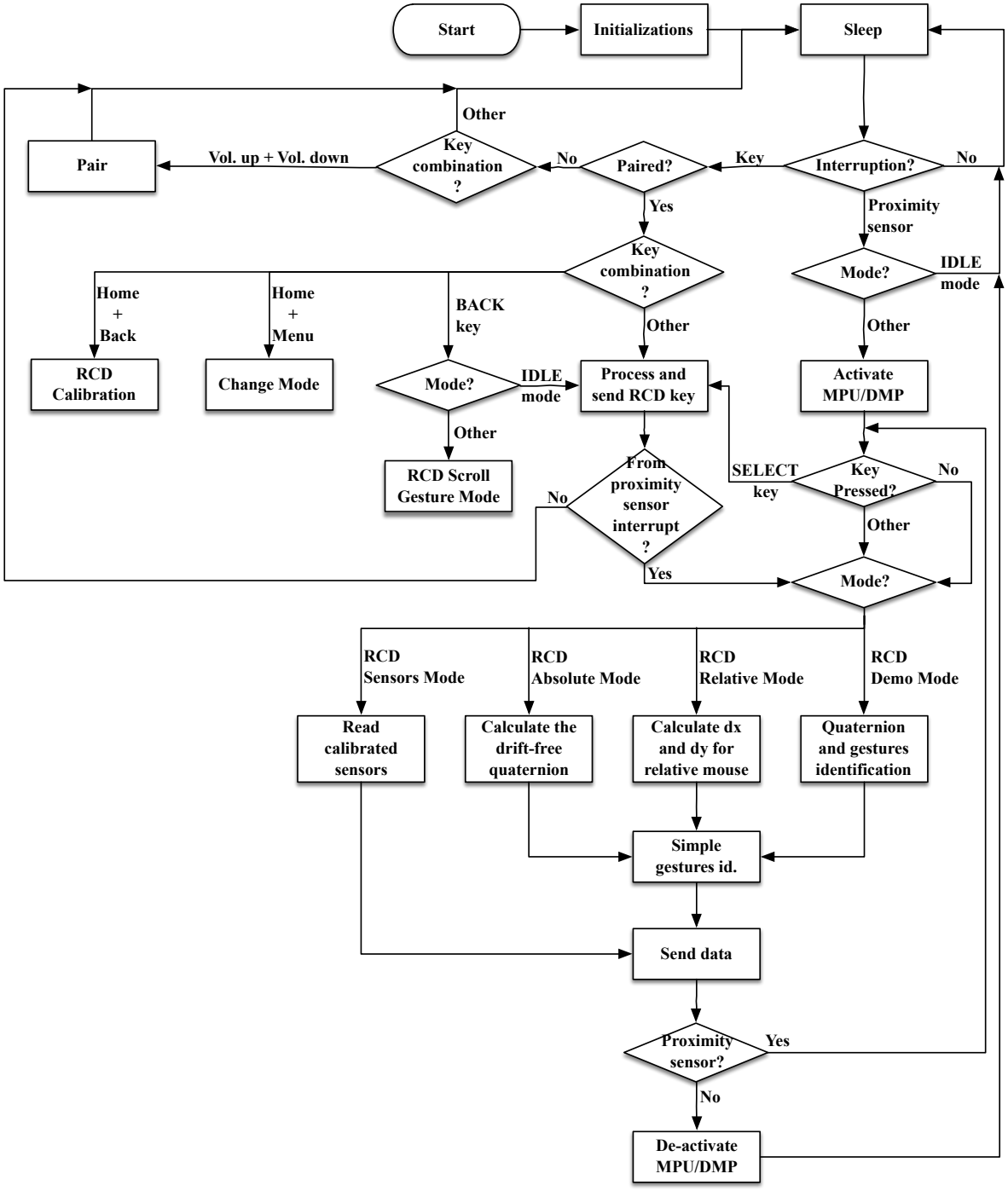

Figure 4.5: Remote Control Device operation flow chart.

Absolute Mouse Mode (AMM)

The difference of this mode to the RMM one is the type of navigation. The orientation estimate is converted to absolute coordinates in the screen. In this work, the YDCF was implemented in the RCD, with the option of not using this filter, if the mouse is not behaving has expected due to magnetic disturbances. The commutation is performed by pressing **Volume Up** to use the magnetometer or **Volume Down** not to use it. Either the corrected drift, or the DMP quaternion is sent to a custom Android Application Programming Interface (API). The pointer coordinates are calculated in the STB because the information about the screen resolution is needed and the communication was unidirectional. The transmission of the quaternion from the RCD to the STB is performed at 50 Hz with a resolution of 2 bytes per quaternion component.

Demo Mode (DM)

During the developing of this work an Android application was made in order to qualitatively access the orientation retrieved by the implemented filters and the DMP. The Android application is described more in detail in the Annex C. This application simulates a multimedia content application for a STB and has some 3D geometric figures (cube, pyramid, etc.) that change orientation with the RCD. This mode was created to allow demonstrations of the accuracy of the remote orientation in a subjective way and to give an example of navigation in multimedia contents using gestures. The data transmitted in this mode has a maximum payload of 10 bytes at 50 Hz, which includes the value of the quaternion and the identified gestures.

Scroll Gestures Mode (SGM)

This is a mode for temporary use that can be activated when the remote is in any mode other than IM and the **Back** key is pressed. It allows scroll and zoom using rotations of the RCD. Similarly to the RMM, the information is sent only when there is a rotation wide enough to generate a scroll event on the screen. Two bytes are sent, identifying the type of scroll and its quantity.

Sensors Mode (SM)

In this mode all the sensor readings are sent to the Android system for custom made applications and debug purposes. The data rate transmission for this mode is 50 Hz. The API records a text file with the data received in order to allow the magnetometer calibration. At the moment, the calibration values have to be *hardcoded* in the RCD. The computation needed to determine the calibration matrix is way to intensive to be performed on board the device, so it would have to be determined in the STB and then transferred to the RCD.

Calibration routine (CR)

Pressing **Home** + **Back** key combination enables the calibration routine of the RCD. After releasing this key combination, the user has 8 seconds to place the RCD in a flat surface with the keys facing up. The calibration process removes the gyroscope and accelerometer biases allowing a more accurate estimate with reduced drift over time. The calibration is saved in the Non-Volatile Memory (NVM) of the prototype and it only needs to be performed again if navigation is not behaving as expected.

When in normal operation, the DMP reads the sensors and computes the quaternion at 200 Hz. The microcontroller operates at 50 Hz for the remaining operations. The navigation modes, simple gestures and scroll gestures are explained in the next Sections. For more details regarding the modes payload information, consult Annex D.

4.2.1 Air-mouse

Two types of air-mouse were implemented in the RCD: one with a relative navigation system, with tilt compensation, based on the gyroscope readings, and another with an absolute navigation system, based on the orientation of the RCD. It is important to define the frames involved and the transformations needed, so that the the mouse can be operated in an intuitive way.

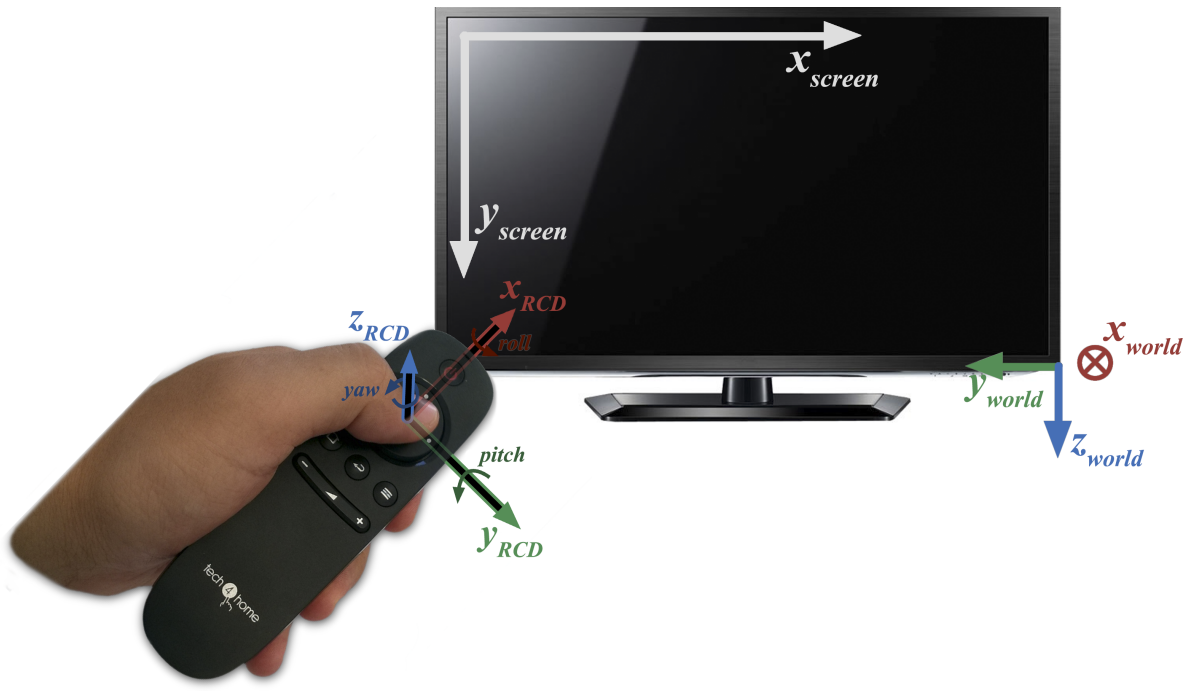

Figure 4.6: Frames involved in multimedia navigation.

In Figure 4.6 the conventions used in this work are defined. The world frame used follows a NED convention to compute the orientation of the RCD. TV sets and most devices use a coordinate system based on the pixels, where the pixel in the upper left corner of the screen has coordinate (0,0), with the x coordinate growing in value along the horizontal direction, and y, in the vertical direction.

Two-dimensional relative pointing device

In order to compensate tilt rotations of the hand while operating the RCD, the gyroscope readings have to be transformed to the world coordinates through Equation (4.19). The components y and z of ${}^{E}\tilde{\boldsymbol{\omega}}_t$ can be weighted to transmit relative motion to the mouse in the screen.

$$ {}^{E}\tilde{\boldsymbol{\omega}}_t = {}^{S}_{E}\tilde{\boldsymbol{q}}_t \otimes^{S} \boldsymbol{\omega}_t \otimes^{S}_{E} \tilde{\boldsymbol{q}}^{*}_t \tag{4.19} $$

Most devices only use the gyroscope and accelerometer to compute the orientation. In this case, the drift on the XY plane over time will degrade the tilt compensation. As drift accumulates, and the XY plane rotates, it will reach a point where the x axis is swapped with the y axis, therefore not using an horizontal reference can create a problem over time. There are a few techniques usually applied to mitigate this problem. Frequent calibrations (resetting the quaternion) are normally executed to ensure that drift does not accumulates. This can be done using a gesture or a press of a button when the user wants to navigate. In the case of this prototype the quaternion is resettled everytime the proximity sensor is activated.

As only tilt along the x axis, i.e., the roll (ϕ) rotation, has to be compensated to allow a more intuitive experience, the set of Equations (4.20-4.21) should be used to perform this tilt compensation.

$$ \phi = \tan^{-1}\left(\frac{2(q_s q_x + q_y q_z)}{1 - 2(q_x^2 + q_y^2)}\right) \tag{4.20} $$

$$ {}^{S}_{E}\boldsymbol{q}_{\phi} = \begin{bmatrix} \cos\frac{\phi}{2} & \sin\frac{\phi}{2} & 0 & 0 \end{bmatrix} \tag{4.21} $$

This compensation involves the use of trigonometric functions, which should be avoided in low-end devices. Considering small angles for screen navigation due to the high rate of orientation computation and to frequent calibrations, it is possible to avoid the use of trigonometric functions by using Equation (4.22), where components q_y and q_z are normalize into q_s. This technique is not mathematically accurate, but does not have a significant impact in the air-mouse usability.

$$ {}^{S}_{E}\boldsymbol{q}_{\tilde{\phi}} = \begin{bmatrix} \sqrt{q_s^2 + q_y^2 + q_z^2} & q_x & 0 & 0 \end{bmatrix} \tag{4.22} $$

Both methods simplify the number of operations needed to perform the vector rotation on Equation (4.19), since two of the components of the quaternion are equal to zero.

Two-dimensional absolute pointing device

For the absolute navigation, the RCD computes the coordinates to set for the mouse in the screen by directly convert quaternions into Roll Pitch Yaw (RPY) angles, then use a gain and an offset to convert the yaw angle into an x coordinate on the screen, and the pitch angle into an y coordinate. There are twelve forms of establishing the sequence of RPY angles. For the case of this RCD, as a NED convention was in use, the axis sequence *"zyx"* was selected as a valid sequence to the RPY angles calculation, because it avoids axis dependencies that, in normal usage would affect the navigation (normal usage means that the RCD is somewhat pointing, facing up, to the TV).

Most operating systems are prepared for the use of relative mouses and have filters and acceleration ramps to allow a good user experience. When using the absolute coordinate system on a mouse, that is not the case. As such, an adaptive filter was implemented in order to increase the QoE for the user when using this type of navigation. When small movements are made, precision is usually required, so a low-pass filter with a low cutoff frequency should be used. As the movement speed increases, the cutoff frequency should move to higher frequencies allowing quicker movements of the pointer with no perceptible delay. The implementation of such a filter can be computational expensive as it involves complex calculations of the filter coefficients, or it may use too much memory if data is saved in the form of a table of coefficients *versus* frequency. It was found that the coefficients of a first order Butterworth filter can be calculated in a linear relation in respect to each other, which simplifies the implementation of an adaptive filter.

A first order digital filter is generically implemented using Equation (4.23), where y is the output value and x the input. Coefficients a_1, a_2, b_1 and b_2 in an first order Butterwoth low-pass filter have the properties show in Equations (4.24) [71].

$$y_t = \frac{1}{a_1}\big(b_1 x_t + b_2 x_{t-1} - a_2 y_{t-1}\big) \tag{4.23}$$

$$\begin{aligned} a_1 &= 1 \\ b_1 &= b_2 \\ \frac{b_1 + b_2 + a_2}{a_1} &= 1 \end{aligned} \tag{4.24}$$

Plotting coefficients a_2 and b_1, in Figure 4.7, it shows that there exists a linear relation between them. Given that $a_2 \propto b_2$, it is only necessary to find a relation of the orientation change rate of the RCD to the desired cutoff frequency of the filter, which is also almost

linear in respect to the filter coefficients.

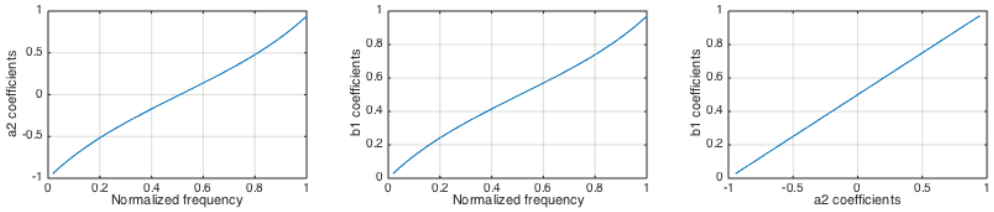

Figure 4.7: Butterworth filter coefficients.

The orientation change rate of the RCD, in this case, is calculated using the derivative of the yaw and pitch angles, independently. Passing both signals through a 5 Hz first-order Butterworth low-pass filter improves the estimative of the cutoff frequency for the adaptive filter.

4.2.2 Gestures

Some gestures were implemented in the RCD to demonstrate the system capabilities. Gestures can allow a quicker and more intuitive navigation, both between and in the menus. Two types of gestures were implemented: quick gestures are performed in quick hand movements when the navigation is activated, which send an action (key) when detected; scroll gestures allow the control of scroll menus and zoom/scale events. Figure 4.6 presents the axes on which the actions implemented to date are identified. A rotation around the x_{world} axis is ${}^{E}roll$, around y_{world} is ${}^{E}pitch$ and around the z_{world} axis, ${}^{E}yaw$. Following the right hand rule, these rotations can be positive or negative depending if they go clockwise (+) or counter-clockwise (-), respectively.

Quick gestures

The RCD in Relative and Absolute Mode constantly checks hand movements in order to detect 6 possible gestures. Table 4.1 matches the gestures to the action they execute. The gesture is identified when the rotation speed around one axis exceeds a given threshold value, and only if the RCD is not tilted.

Table 4.1: Quick gestures (according to the convention in Figure 4.6) and the corresponding action.

Gesture	${}^{E}roll^{+}$	${}^{E}roll^{-}$	${}^{E}pitch^{+}$	${}^{E}pitch^{-}$	${}^{E}yaw^{+}$	${}^{E}yaw^{-}$
Action	Enter	Back	Down	Up	Right	Left

Scroll gestures

The scroll gestures allow scrolling horizontally, vertically, zoom in and zoom out. Pressing the **Back** key while in a navigation mode, enables the scroll gestures. Performing a rotation around one axis will lock the type of scroll/zoom. The gesture is differential, so as the rotation is performed, a menu or a screen is scrolled. To perform a different scroll gesturem the **Back** key has to be release and then pressed again. Table 4.2 identifies the scroll gestures that can be executed with the RCD.

Table 4.2: Scroll gestures (according to convention in Figure 4.6) and correspondent action.

Gesture	${}^{E}roll^{+}$	${}^{E}roll^{-}$	${}^{E}pitch^{+}$	${}^{E}pitch^{-}$	${}^{E}yaw^{+}$	${}^{E}yaw^{-}$
Action	Zoom out	Zoom in	Scroll vertical down	Scroll vertical up	Scroll horizontal right	Scroll horizontal left

Chapter 5

Tests and Results

In order to access the improvements implemented in the original fusion filters, quantitative tests were performed to validate the algorithms design assumptions. As the project was being developed and as decisions were paving the path of the project, the tests became more specific to the application in sight and culminated with the implementation of a balanced setup in terms of accuracy/computational load on the RCD. This Chapter presents the results obtained during this process, being divided in four sections. The first section presents the results of the implementation of the proposed filters in a testing scenario for a quantitative analysis. In the second section, a comparative analysis of the filters computational burden is evaluated. The third section describes a qualitative analysis of the implemented air-mouses compared with a reference air-mouse. Finally, in the fourth section, subjective results for the yaw drift compensation using the magnetometer and the optical flow sensor are presented, as well as the result of a magnetometer calibration for the RCD prototype.

5.1 Quantitative comparative tests

In order to compare the implemented filters in a harsh dynamic environment with no prior knowledge of the magnetic field disturbances or linear accelerations caused by the imposed motion, tests were carried out using an industrial robot that could retrieve the MARG unit orientation at 15 Hz with a resolution of 0.01° and a pose repeatability of 0.06 mm, establishing the ground-truth used afterwards to compute the accuracy of the implemented and reference filters/sensors. The experimental setup is presented in Figure 5.1. The error values are calculated using RPY angles computed from the quaternions of the tested filters/sensors, in relation to the ground-truth given by industrial robot. The results are presented in degrees.

Five low-complexity MARG fusion filters were evaluated: the OMdF, the AMhF, the

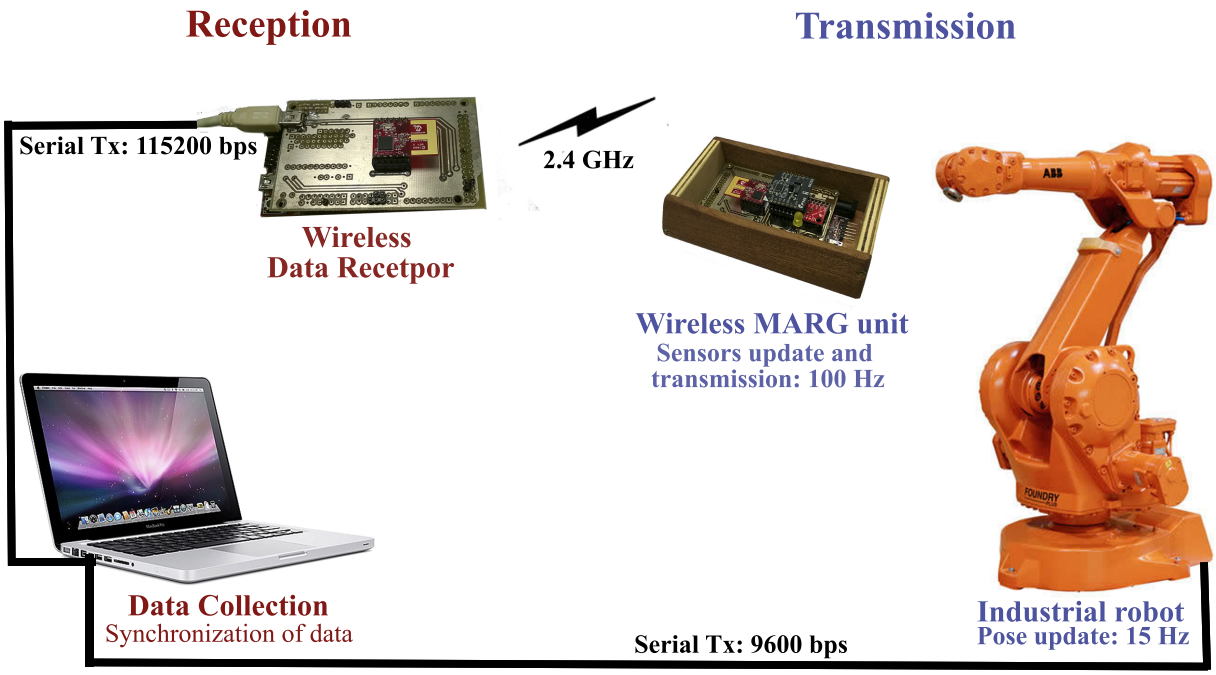

Figure 5.1: Quantitative tests setup.

AGMdF, the AGMhF and the KF described in Section 3.3.4. They were compared with two reference MARG units/filters available on the market: the PNI SENtral M&M Blue Module with a state-of-the-art patented KF performing continuous *hard* and *soft-iron* disturbance calibration and magnetic anomalies compensation; and the XSENS MTi-30 series device with a proprietary XSens Kalman Filter.

The tests were performed with four different sets of sensors: the 9-axes integrated MPU9150 from Invensense; the 9-axes integrated LSM9DS0 from STMicroeletronics; a 9-axes integration from the 6-axes MPU6050 (Gyroscope+Accelerometer module of MPU9150) with a 3-axes magnetic-inductive RM3100 Evaluation Board (EvB) from PNI Corp.; and a 9-axes integration from the 6-axes MPU6500 (Gyroscope+Accelerometer) with a 3-axes RM3100 EvB. All sensor signals were sampled at 100 Hz and interfaced to a PC via a custom wireless protocol between two MRF24J40 Microchip modules. The serial communication between the computer and the industrial robot did not have the necessary bandwidth for synchronizing the data in real time. The solution for this problem was to have the robot sending synchronization tokens during the test, while storing its pose data, then, when the test was finished, the robot transmitted all the stored data and the computer synchronized this data with the previous received data from the MARG units.

The tests included multiple accelerations and quick rotations along the three sensor axes frame with a total duration of about 30 seconds. The NED convention (North - x axis, East - y axis, Down - z axis) was used for the absolute coordinate frame. The

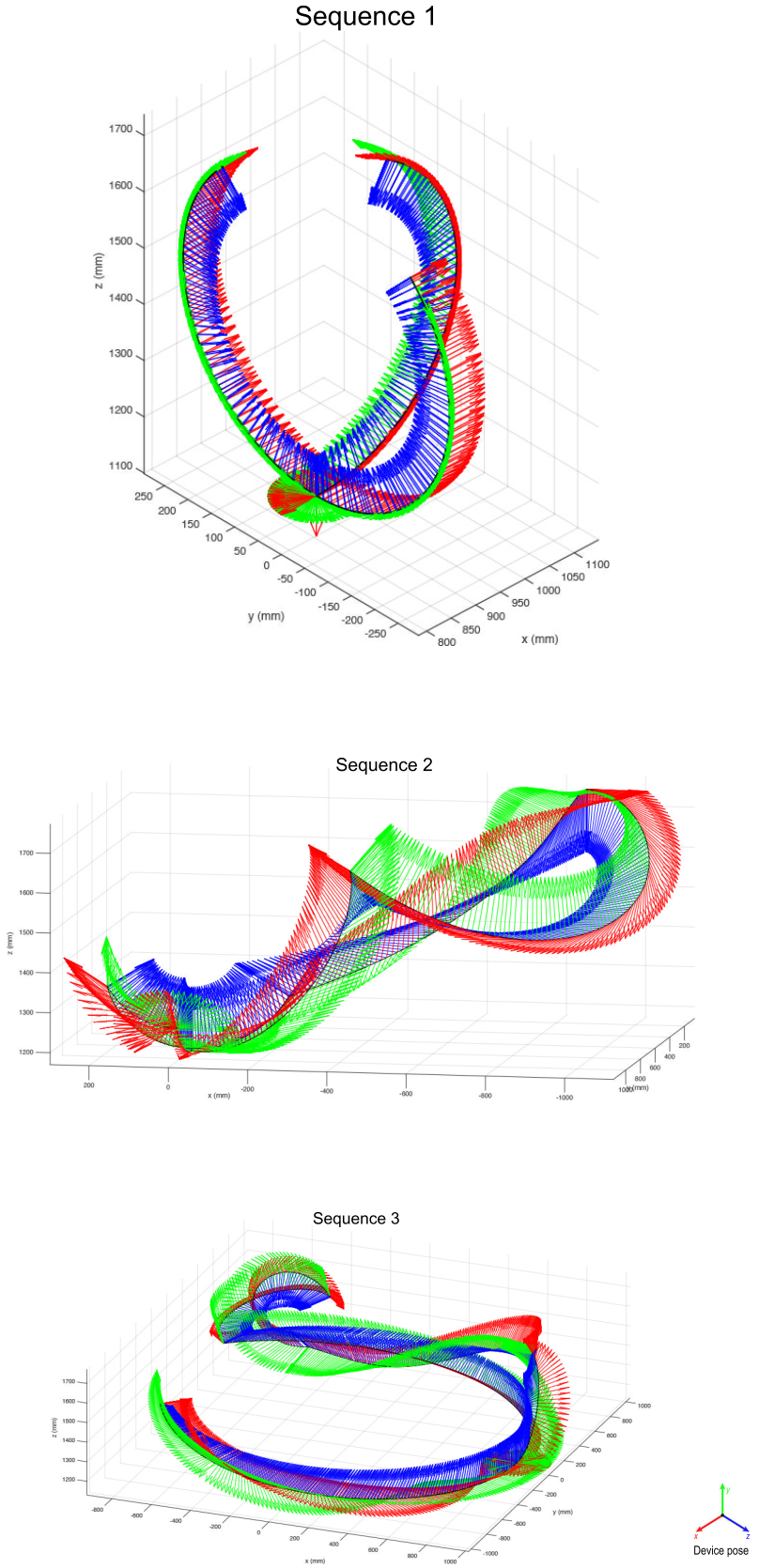

Figure 5.2: Path and platform orientation for the three sequences tested.

environment found during these tests was characterized by: in approximately 52% of the time, the magnitude of the measured earth's magnetic field had a deviation by more than 5% due to magnetic disturbances; 10% of the time there were accelerations deviating the estimated magnitude of the earth gravitational acceleration vector by more than 5%.

As said, tests were performed using five low-complexity fusion filters in four different sets of MARG units. Four different use cases were tested regarding the magnetometer, namely: no magnetometer usage (only 6-axes integration); the integration of the magnetometer without any calibration; the usage of the magnetometer calibrated for the device; and the usage of the magnetometer calibrated in the industrial robot. An ellipsoid fit method was used to calibrate the magnetometer, this way compensating the *hard* and *soft iron* distortions. Furthermore the tests were executed over three different trajectories (sequence 1, 2 and 3) with ascending complexity in rotations and path. An illustration of the path and orientation of the MARG unit for each sequence can be observed in Figure 5.2, which show plots of the data retrieved from the industrial robot. In summary, there was total of 240 different tests plus 6 tests for the reference sensors: PNI SENtral M&M Blue Module and XSENS MTi-30. The diagram in Figure 5.3 shows the performed tests in the form of a branched tree. In each branch the model is replicated, which sums to the 240 tests.

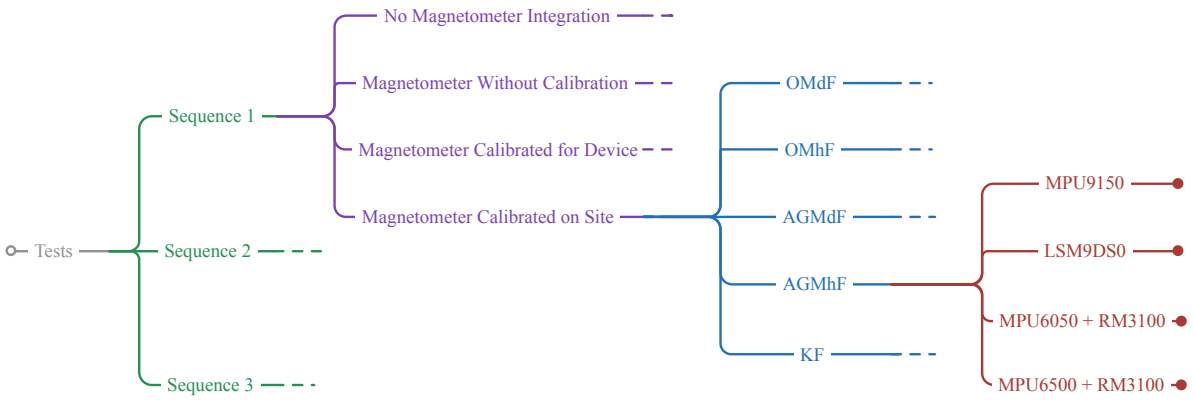

Figure 5.3: Quantitative tests summary diagram.

In order to allow an easier comparison between each test, the first step was to average the Root-Mean-Square Error (RMSE) obtained for all the filters and for all the sensors, in order to assess which filter and sensor gave the smallest error, and so focus future development on that combination.

The results shown in Figure 5.4 were based in 80 tests for sequence 1 using four sets of MARG units, discarding the worst 5% results, for a 95% confidence interval. It is possible to observe that from all implemented filters, the adaptive-gain versions of the CFs present the best performance followed by the implemented Kalman Filter. An additional useful

information is that the 5% discarded tests were all from the OMdF and AMhF. The coloured bars represent the average RMSE in degrees for each corresponding axis, and the line bars correspond to the Standard deviation (Std) for the same average.

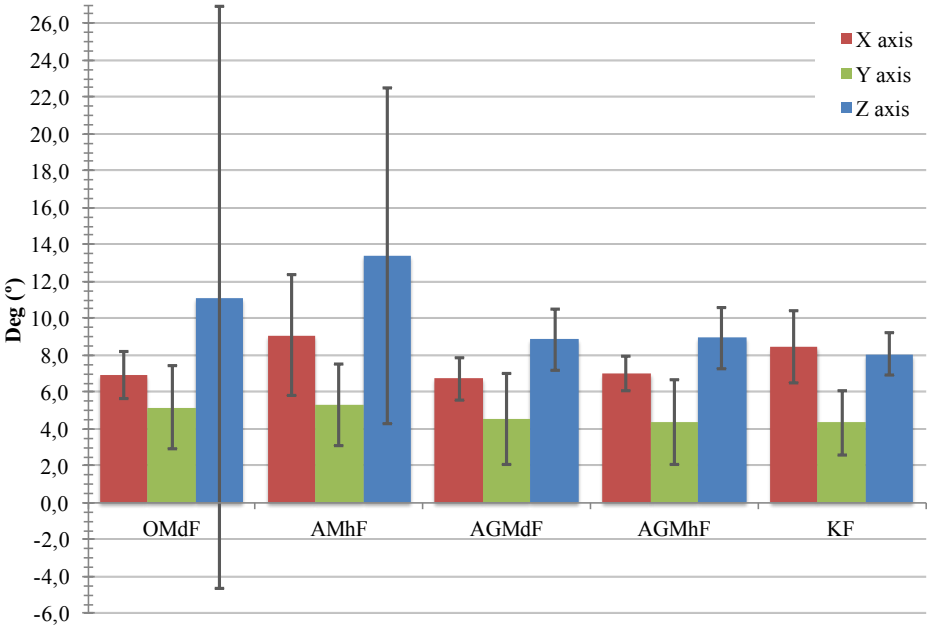

Figure 5.4: Average RMSE and respective Std per filter for all the sensors.

The results for the sensors are in Figure 5.5, with the RMSE average for the same 80 tests described. Clearly, the MPU9150 from Invensense presented the best overall results independently of the used filter for sequence 1.

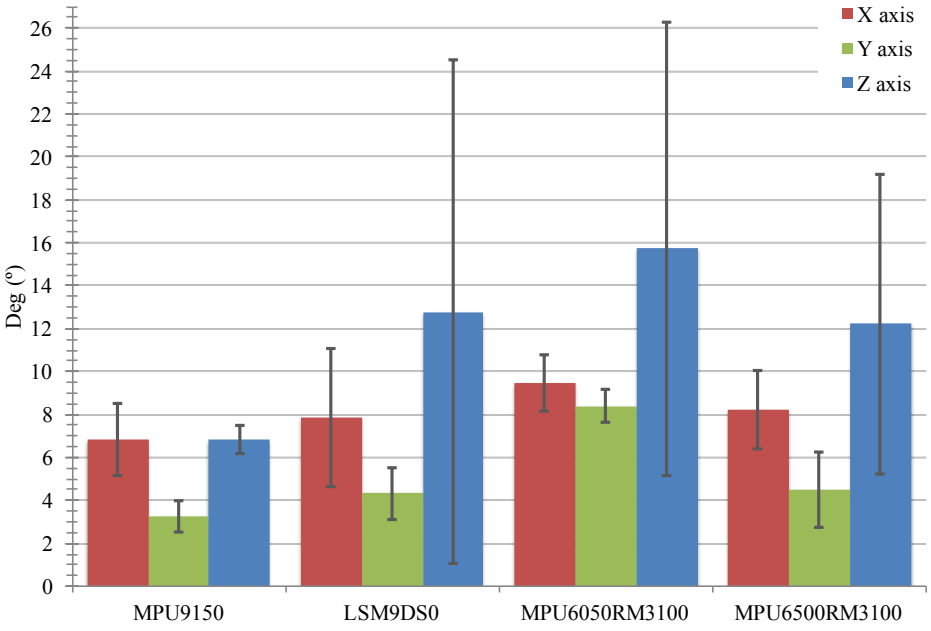

Figure 5.5: Average RMSE and respective Std per sensor for all the filters.

Figure 5.6 presents the results for the average RMSE and its Std, of 12 tests performed with the MPU9150, for the implemented filters, averaging all magnetometer usages and all the three sequences. These were compared with the results obtained with the reference sensors also in the three sequences. It can be seen, that the results for the adaptive-gain versions of the CF are comparable to the results obtained for the reference sensors, being even better than the results from the highly complex filter running in XSENS MTi-30. The performance is consistent, as testified by the observed Std. It was also verified that the performance of the implemented Kalman Filter generally drops when subjected to higher dynamic conditions.

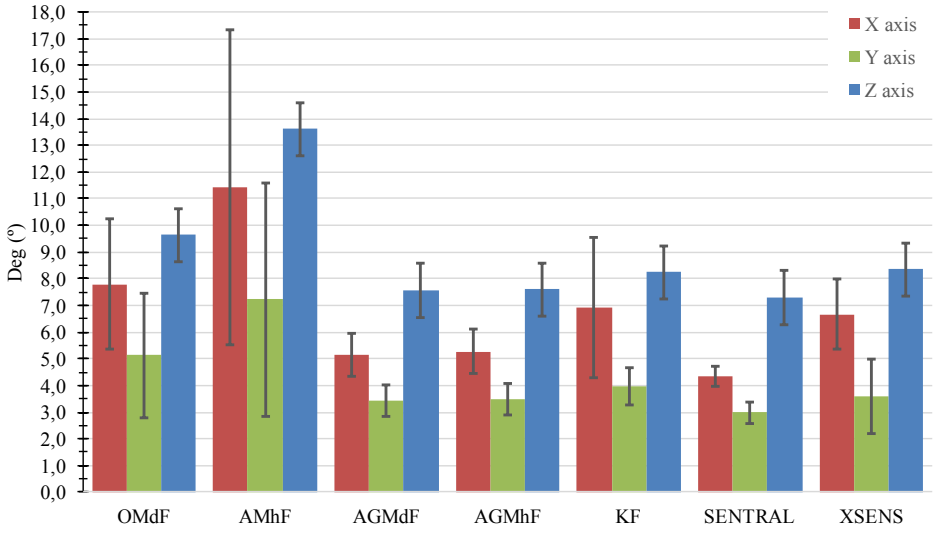

Figure 5.6: Average RMSE and respective Std per filter for the MPU9150 sensor.

Figure 5.7 presents the results according to the use of the magnetometer for the MPU9150 MARG unit, in the three sequences and for all the filters. It was observed that the general use of the magnetometer, in the tested environment, tended to worsen the results for the rotation along the z axis. The test were executed in a small amount of time so, for longer periods, the drift rotation along the z axis would accumulate error, since there is no reference for stabilizing the XY plane when the magnetometer is not used. In this case, the error around the z axis would increase with time, while, when using the magnetometer, it would tend to stabilize.

The magnetometer calibration, using an ellipsoid fitting method, can improve the results, specially if the calibration is performed in the environment where the MARG unit is to be used, because a distorted magnetic field mislead the filters estimates. There is a certain tendency to confirm this affirmation with our results, but the errors difference between tests according to the type of magnetometer calibration is not clear enough to reach a definitive conclusion.

Figure 5.8 shows the result of the integration of the AGMhF in the MPU9150 and the

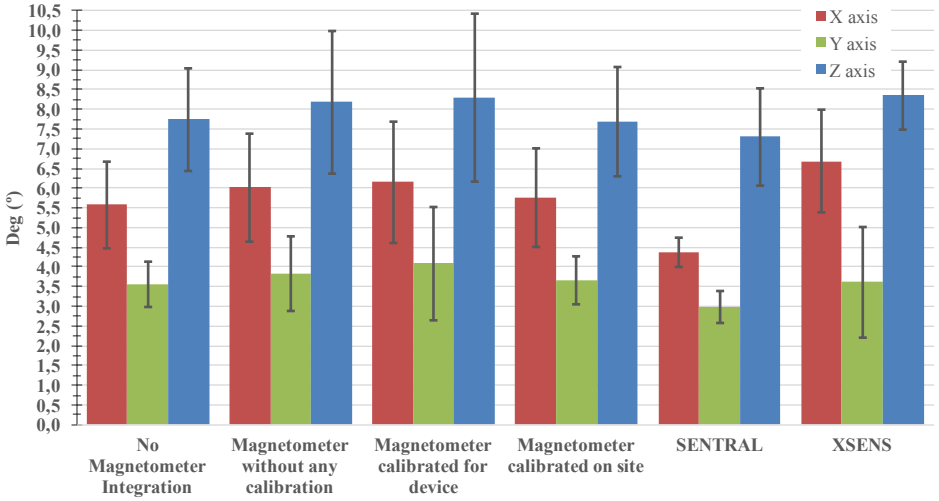

Figure 5.7: Average RMSE and its Std according to magnetometer usage for the MPU9150 sensor.

results retrieved by the XSENS sensor, compared to the ground-truth. These results are relative to the path covered in sequence 3.

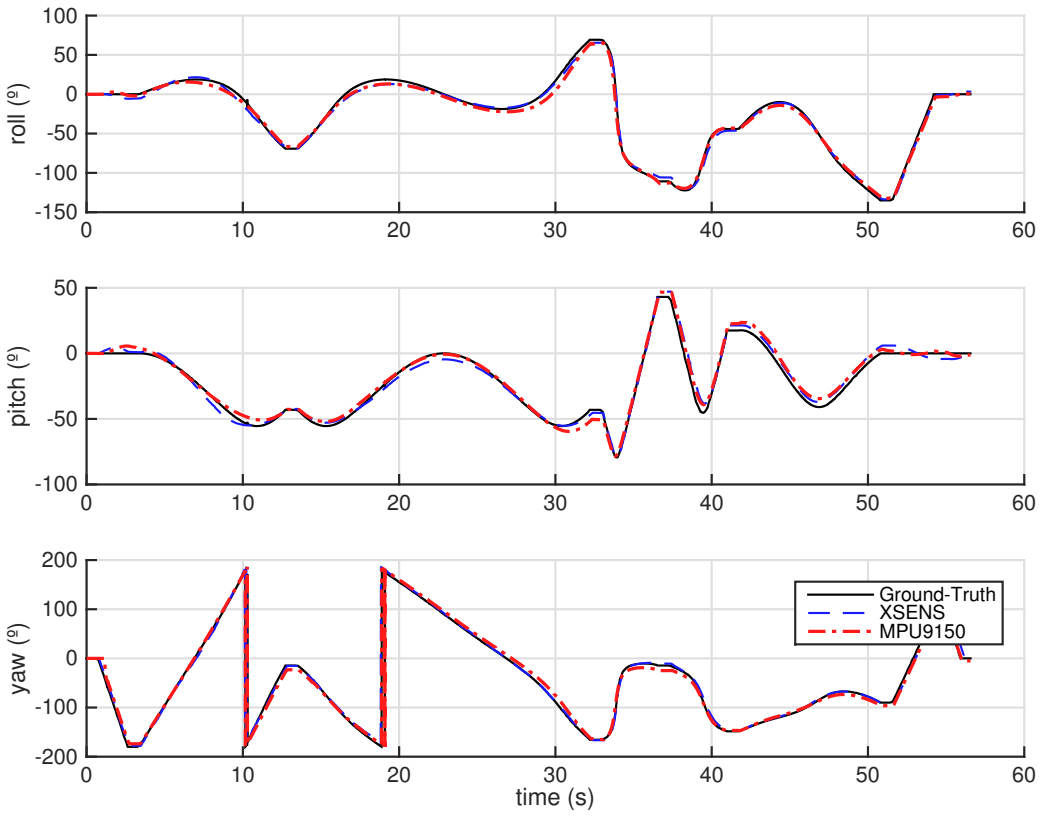

Figure 5.8: RPY angles retrieved with the AGMhF on the MPU9150 and with the XSENS.

As a result of the experimental methodology used, a dataset with the collected data was created and made available on Github[1]. This dataset can be used for any future work that uses magnetic, acceleration and/or angular rate data.

5.2 Computational burden comparative tests

For the computational burden analysis the number of operations per filter was considered. The number of basic operations that a processor can perform gives the most generic assessment for the computational burden analysis, because the processors can implement these operations in different ways, making the algorithm more efficient in some architectures and less in others. Table 5.1 quantifies the computational burden of each filter. The number of elementary operations is for a non-optimized C implementation of the filters, however, it gives a good idea of the computational weight of each filter. As can be seen, the KF is by far, the heaviest algorithm as expected, even though it implements only a state with four variables. Table 5.1 also shows that the adaptive-gain versions of the CF are heavier than the originals, but the results from the last section (Figure 5.1) prove they are the best filters to use in real environments.

The YDCF does not computes orientation on its own, so it should not be compared to the other filters in that sense. Its number of operations was considered because of the new functionalities that Invensense added to their sensors: performing data fusion in a DMP under the same encapsulation as the sensor. As the orientation retrieved from the DMP drifts in the yaw rotation, one only needs to fuse data from magnetometer to correct it. As can be seen, from the microcontroller perspective, this filter looks like the most computational efficient option, it depends on the implementation of the trigonometric functions.

Table 5.1: Number of operations per iteration of the implemented filters.

	Multiplications	Sums	Subtractions	Divisions	Square Roots	Trign. Functions
YDCF	70	35	27	1	4	2
AMhF	121	46	31	10	4	0
OMdF	183	64	40	14	5	0
AGMhF	220	89	68	10	4	0
AGMdF	337	118	94	14	6	0
KF	625	255	115	24	6	0

These results give some insight about the computational burden of the filters and are to be used as a complement of the results presented in Section 5.1.

[1] Available on: www.github.com/miguelrasteiro/IMU_dataset.git

5.3 Qualitative tests

A survey was conducted to access the QoE by a human acting as an end user of tree different air-mouse implementations, henceforward designated by air-mouse 1, 2 and 3. The Movea company provides high performance solutions for TV remote control, designated as MoveaTV[2], hereby presented as air-mouse 1. The MoveaTV solution used in the test implements a relative navigation with tilt compensation, meaning that it does not use magnetometer data. The air-mouse 2 implements the relative navigation described in Section 4.2.1, which is similar to the Movea solution, with tilt compensation provided by the orientation information based on the gyroscope and the accelerometer. Air-mouse 3 is the air-mouse described in Section 4.2.1, it provides absolute navigation based on the orientation of the device, converting RPY angles computed from the quaternion into coordinates on the screen, based on all MARG sensors data. This air-mouse 3 uses the yaw drift compensation method described in Section 4.1.2.

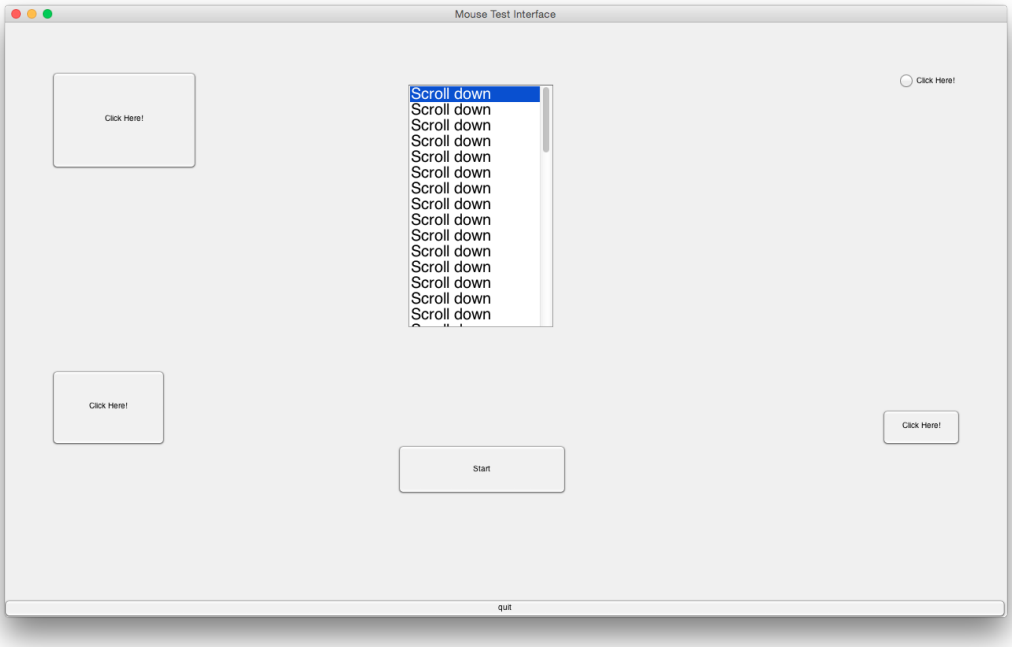

Figure 5.9: Test window for the three different implementations of the air-mouses.

A demonstration application was developed (showed in Figure 5.9) which allowed to subjectively assess the type of favourite navigation by users and also evaluate the performance of each air-mouse. Air-mouses 2 and 3 were implemented using the MPU9250 from Invensense and a PIC32MX795F512L for the filters and mouse data processing, compared to the MoveaTV Solution (air-mouse 1) implemented in a remote control prototype. The survey test involved 60 people, where, for each user, the following task was evaluated:

[2]Available on: www.movea.com

they should navigate in a window and click on five buttons with different levels of difficulty according to the button size; a scroll task had to be performed with the pointer to finalize each mouse test. In the end, a survey was displayed where the user should grade each air-mouse in terms of accuracy, velocity and intuitiveness. The user could attribute a grade in a 1-5 scale (being 5 the best grade) for each evaluated characteristic.

Table 5.2 presents the results for this subjective evaluation test. The results reveal that absolute navigation achieves the best grade in almost all metrics used, having only a modest grade on the velocity parameter. Therefore these results provide evidence about the user preference for navigating with absolute orientation. Most users also found the air-mouse 3 to be the most intuitive and accurate.

Table 5.2: Subjective tests results.

	Accuracy	Velocity	Intuitiveness
Air-mouse 1	3,7	4,0	3,5
Air-mouse 2	3,6	4,4	3,8
Air-mouse 3	4,3	3,5	4,3

5.4 Yaw drift compensation

This section presents the results concerning the yaw drift reduction using the optical-flow sensor, namely the methodologies of section 4.1.2, and also the YDCF. The plot in Figure 5.10 shows the result estimating the yaw orientation when using an uncalibrated gyroscope. The test was executed in front of a screen with mouse navigation. The red line represents the yaw when there is no yaw compensation, as can be seen, the use of a uncalibrated gyroscope has an enormous impact in the yaw rotation, but applying the compensation using the optical flow sensor reduces that drift significantly. As referred, the drift is completely removed when the image quality retrieved by the ADNS3080 (gray line) is bigger than 10 (image with good detectable features) and there is no motion detected. At the same time, a drift estimate is being performed so that, when in motion, the last drift estimate is used for correcting the yaw drift. The plot in Figure 5.11, shows a 30 second test of the orientation retrieved by the DMP after calibration at rest (red, green and purple lines). As can be seen, using the magnetometer and the the YDCF the yaw drift is removed (blue line). The DMP in this case presented a drift of about 0.1 °/s.

The drift reduction verified using the two methods is sufficient for not being noticed by the user while using the air-mouse with absolute navigation, during typical times of usage. Any miscalculation on the yaw drift estimate using the optical flow sensor, causes an error that can not be recovered by the system. By using the magnetometer it is possible

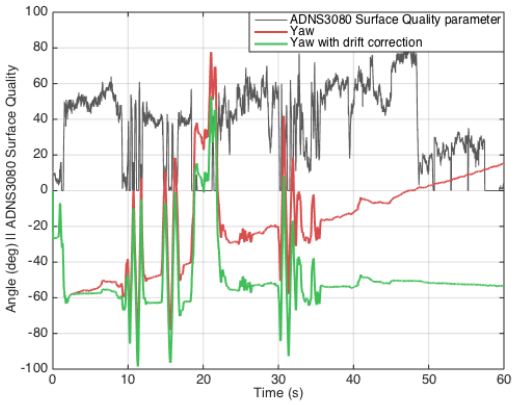

Figure 5.10: Drift compensation using the optical flow sensor.

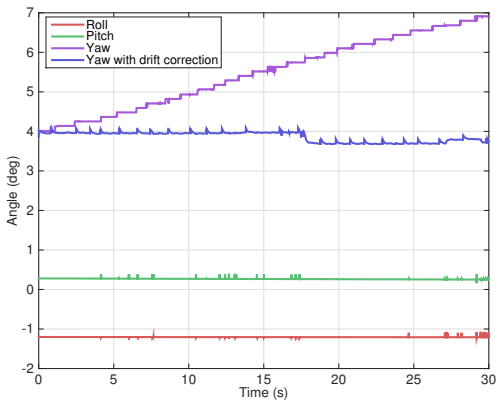

Figure 5.11: Drift compensation using the YDCF.

to recover from previous errors as long as the error do not exceed the difference between reference points.

This section also presents an example regarding the calibration of the magnetometer for the final RCD prototype in a living room, using an ellipsoid fitting method. The result of such calibration can be observed in Figure 5.12.

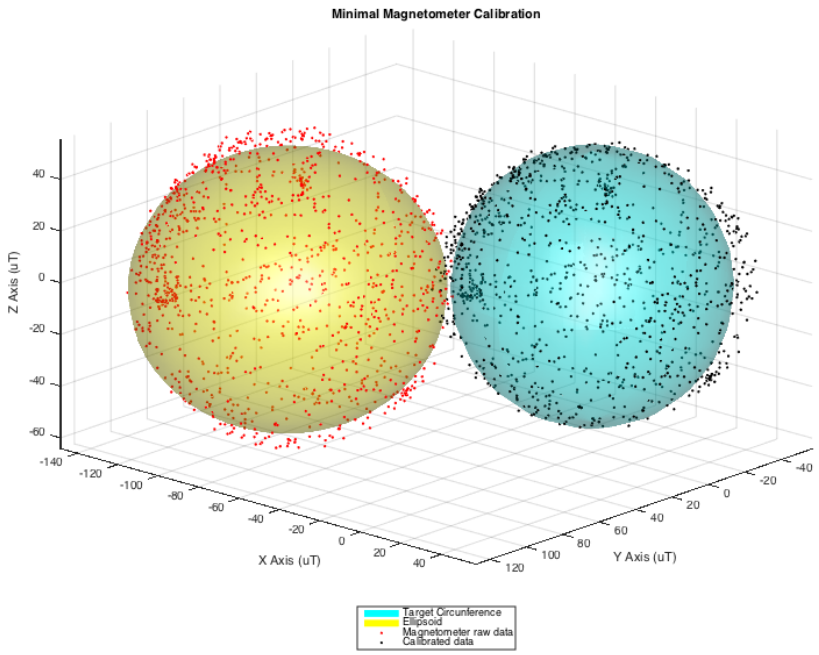

Figure 5.12: Prototype magnetometer raw readings and the result after calibration.

As can be seen, the magnetic distortions present in this experiment would have a great impact in the orientation estimate, if not properly addressed. This is specially true for

the *hard iron* distortion along the x axis of the RCD, since these add an offset so big that never allow negative readings of the magnetic field in that axis. The offset verified on that axis is probably due to the presence of the battery in the RCD. This would cause an heading error which would lead to a very low user QoE.

Chapter 6

Conclusions and Future Work

Current RCDs in the market are not far from delivering a satisfactory QoE for users. However, the existing solutions which allow navigating new multimedia contents can still be frustrating, since users cannot easily explore all the available features. The work developed in this dissertation paves the way to new forms of interaction and new possibilities in low-cost RCDs, for current and future 3D interactive multimedia content. An absolute navigation form was explored, using an RCD as a pointer, to verify if it could be more intuitive and bring increased usability during navigation. The users evaluation of the implemented system indicates that a two dimensional pointing device based on the absolute orientation of the RCD is preferred, when compared to the traditional relative navigation interfaces. This solution was fully implemented in the final prototype, though, due to hardware and communications limitations, the computation of the mouse pointer position on the screen has not been actually implemented in the RCD, but in the developed Android API instead. The RCD sends the reduced drift quaternion to the STB, from where the absolute mouse position is calculated. Future developments can correct this situation using a calibration routine that allows the remote device to infer the TV screen resolution, or by implementing bidirectional communication. As for the typical relative navigation, a tilt compensated navigation is, in the author's opinion, more intuitive than using a 3 DoF solution based only on the gyroscope readings; the pointer on the screen behaves as a human would expected even when the RCD tilts during its typical usage. The developed relative solution is available in the prototype for people who might still prefer to use it instead of the absolute solution.

Typically, RCDs have hardware constrains and an expected long battery life. These characteristics were to be maintained as much as possible, both for usability and to keep the low price of the device. The study performed about fusion filters that could aid in the process of bringing new navigation capabilities to RCDs, lead initially to filters with high computational needs, that hardly would be able to run in real time on such devices. Even

the simple KF implemented and tested in this study, that only used the quaternion as the state estimate, had much higher computational needs when compared to the original CFs, even though they had higher accuracy too. Understanding the key points and differences between the filter approaches, sensors and environment characteristics, made it possible to apply some modifications to the original CFs in order to improve their accuracy, at the cost of an increment on the original CFs computational complexity. The accuracy results showed that the modified adaptive-gain CFs, for this specific application, despite requiring much lower computational power than the KFs, can achieve similar results to the ones obtained with more advanced fusion filters, performing, in our tests, within the same accuracy levels of the ones presented by the tested professional MARG units. Invensense is a MEMS manufacturer making efforts to deliver SoC solutions that already performs onboard sensor fusion to estimate the orientation, freeing main processors from this task. However, the existing solutions cannot retrieve an absolute orientation since the integration is performed with only the gyroscope and the accelerometer data. The products with these characteristics are currently more expensive than MEMS sensors that do not perform filter fusion. Nevertheless, the methods detailed on this document, where the magnetometer or the optical flow sensor are used to correct the orientation provided by the DMP, show that the yaw drift can be corrected or reduced with less computational efforts in the RCD main processor. Not having to perform the complete fusion of all the sensors, which allows the use of cheaper main processing units, keeping the total costs low. The solution with the magnetometer and DMP data integration, using a MPU9250 from Invensense, was the one implemented in the final RCD.

While the gyroscopes and the accelerometers are sensors that present application-specific disturbances that can be identified and corrected, the magnetometers, specifically the MEMS Hall effect magnetometers, present high measurement noise levels, making it hard to successfully integrate its readings into the process. Other magnetic sensing technologies, like the magneto-inductive RM3100, tested in this work, can improve the results, but they are much more expensive than the Hall effect sensors, thus not yet applicable to RCDs. Furthermore, the main issue using the magnetometer are the magnetic field distortions, which exist with even a greater expression in indoor environments and are difficult to cope with. In order to get reliable measurements, it is imperative to use a complex calibration routine, which, nevertheless, is not enough to deal with changes on the magnetic field, subjected to the *hard* and *soft iron* distortions that are spacial and time variant. Using the differential approach (both in time and in space) method presented in this work, one can improve the orientation estimate using magnetic fields by reducing the yaw drift. On the other hand, experiments were performed using an low-end optical sensor instead of the magnetometer, also to compensate the yaw drift. The use of optical sensors, with ever reducing prices, can reduce the orientation error by estimating

the yaw drift present in the gyroscope/accelerometer fusion estimate. In this work, it was chosen to point the optical flow sensor to the TV screen, in order to be able to get better illumination conditions for the sensor, but other setups and optical technologies can be further explored and so, avoid the use of the magnetometer. One can, for instance, use the optical sensor pointed in other direction, since on-screen motion deceives the system, or use another type of optical sensors with non-visible wavelengths.

As for the position estimate, a solution was implemented yielding a good performance using the accelerometer data and the orientation from MARG sensors. Even though the error becomes huge as time goes by, for small periods of time such estimate is accurate enough; it can allow, for instance, the identification of gestures, or a push/pull navigation type. The position estimate was evaluated during the development of this work, but not implemented in the final RCD, with its applications and possibilities being explored in future work. The position can also be estimated using the optical flow sensor with expected lower error levels, since only one integration step is necessary. Other issues would have to be addressed when using this sensor for a correct position estimate: the lightning conditions are crucial for good results; one needs an approximated estimate of the distance of the remote device to the visible surface; and, for a 3D estimate, at least two sensors would have to be used if no additional information would be present. The yaw drift reduction method developed in this work had satisfactory results, allowing to improve the orientation estimate and, consequently, improve the position estimate using a set of inertial and optical flow sensors. However, one should take into account that the use of the output of an optical flow algorithm, similarly to the gyroscope output, does not yield an absolute reference and, as such, it cannot be used to obtain an absolute orientation. For a pose determination system the use of low-resolution optical sensors can, in theory, achieve such task if low-complexity pose algorithms are developed and/or implemented using low-end hardware-based solutions.

The final RCD prototype included: two types of navigation modes using the mouse, one using relative-motion and another one using absolute-motion; navigation through the RCD keys; and navigation through a set of predefined gestures. These features were developed taking into account the implemented fusion filtering techniques. Gestures add new degrees of freedom to the navigation using the RCD, as they can be used to execute actions, namely controlling audio volume, channel selection, push/pull/open menus, and so on; and can be used for user identification as well, by entering personal pass-codes through gestures. As gestures add new dimensions on the navigation process, and as they have so many possibilities, they are a topic that can be further developed in future work.

This page was intentionally left blank.

Bibliography

[1] K. K. Cheong, S. K. Park, and Y. J. Park, "An evaluation of moving target selection for remote pointing in TV input devices," in *International Journal of Digital Content Technology and its Applications,*, vol. 6, no. 5, March 2012, pp. 298–307.

[2] C. Gritton, "Whats wrong with SmartTV: How to improve user experience," in *Consumer Electronics Magazine, IEEE*, vol. 2, no. 4, October 2013, pp. 40–43.

[3] N. Abbate, A. Basile, C. Brigante, and A. Faulisi, "Development of a MEMS based wearable motion capture system," in *in Human System Interactions, 2009. HSI '09. 2nd Conference on*, May 2009, pp. 255–259.

[4] C. Brigante, N. Abbate, A. Basile, A. Faulisi, and S. Sessa, "Towards miniaturization of a MEMS-based wearable motion capture system," in *in Industrial Electronics, IEEE Transactions on*, vol. 58, no. 8, August 2011, pp. .3234–3241.

[5] T. Brown, "Harsh military environments and microelectromechanical (MEMS) devices," in *in Sensors, 2003. Proceedings of IEEE*, vol. 2, October 2003, pp. 753–760.

[6] J. C. Lee, "Hacking the nintendo Wii remote," in *IEEE Pervasive Computing*, vol. 7, no. 3, July 2008, pp. 39–45.

[7] S. Kuntz and J. Cíger, "Low-cost and home-made immersive systems," in *The International Journal of Virtual Reality*, vol. 11, no. 3. VRGeeks, 2012, pp. 9–17.

[8] S. Du, "Integration of precise point positioning and low cost MEMS IMU," Masters Thesis, SCHULICH - School of Engineering - University of Calgary, Department Of Geomatics Engineering, Calgary, Canada, November 2010.

[9] V. M. S. Ltd., *Vicon MX Hardware*, 1st ed., Vicon Motion Systems Limited, 5419 McConnell Avenue, Los Angeles, CA 90066, USA, 2004.

[10] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, May 2014, pp. 15–22.

[11] R. Valenti, I. Dryanovski, and J. Xiao, "A non-inertial acceleration suppressor for low cost inertial measurement unit attitude estimation," in *Robotics and Biomimetics (ROBIO), 2013 IEEE International Conference on*, December 2013, pp. 639–644.

[12] Y. Tian, J. Zhang, and J. Tan, "Adaptive-frame-rate monocular vision and imu fusion for robust indoor positioning fusion for robust indoor positioning," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, May 2013, pp. 2257–2262.

[13] P. Neto, J. N. Pires, and A. Moreira, "3-d position estimation from inertial sensing: Minimizing the error from the process of double integration of accelerations," in *Industrial Electronics Society, IECON 2013-39th Annual Conference of the*, November 2013, pp. 4026–4031.

[14] S. Madgwick, A. Harrison, and R. Vaidyanathan, "Estimation of IMU and MARG orientation using a gradient descent algorithm," in *Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on*, June 2011, pp. 1–7.

[15] X. Yun, E. R. Bachmann, and H. M. I. andJames Calusdian, "Self-contain position tracking of human movement using small inertial/magnetic sensors modules," in *Robotics and Automation, 2007 IEEE International Conference on*, April 2007, pp. 2526–2533.

[16] I. Pappas, M. Popovic, T. Keller, V. Dietz, and M. Morari, "A reliable gait phase detection system," in *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 9, no. 2, June 2001, pp. 113–125.

[17] M. Zhang, J. Hol, L. Slot, and H. Luinge, "Second order nonlinear uncertainty modeling in strapdown integration using MEMS IMUs," in *Information Fusion (FUSION), 2011 Proceedings of the 14th International Conference on*, July 2011, pp. 1–7.

[18] R. E. Kalman, "A new approach to linear filtering and prediction problems," in *Transactions of the ASME - Journal of Basic Engineering*, vol. 82, no. 1, 1960, pp. 35–45.

[19] J. Marins, X. Yun, E. Bachmann, R. McGhee, and M. Zyda, "An extended Kalman filter for quaternion-based orientation estimation using MARG sensors," in *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, vol. 4, November 2001, pp. 2003–2011.

[20] A. M. Sabatini, "Quaternion-based extended Kalman filter for determining orientation by inertial and magnetic sensing," in *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 7, July 2006, pp. 1346–1356.

[21] E. Kraft, "A quaternion-based unscented Kalman filter for orientation tracking," in *Information Fusion, 2003. Proceedings of the Sixth International Conference of*, vol. 1, July 2003, pp. 47–54.

[22] D. Roetenberg, H. J. Luinge, C. T. M. Baten, and P. H. Veltink, "Compensation of magnetic disturbances improves inertial and magnetic sensing of human body segment orientation," in *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 13, no. 3, September 2005, pp. 395–405.

[23] X. Yun, C. Aparicio, E. Bachmann, and R. McGhee, "Implementation and experimental results of a quaternion-based Kalman filter for human body motion tracking," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, April 2005, pp. 317–322.

[24] Xsens, *MTi User Manual*, Xsens, Pantheon 6a, 7521 PR Enschede, The Netherlands, January 2014.

[25] G. Wahba, "Problem 65-1: A least squares estimate of satellite attitude." in *SIAM Rev.*, vol. 7, no. 3, July 1965, p. 409.

[26] I. Y. Bar-Itzhack and R. R. Harman, "Optimized TRIAD algorithm for attitude determination," in *Journal of Guidance, Control, and Dynamics*, vol. 20, no. 1. American Institute of Aeronautics and Astronautics, 2015/07/08 1997, pp. 208–211. [Online]. Available: http://dx.doi.org/10.2514/2.4025

[27] A. M. Sabatini, "Estimating three-dimensional orientation of human body parts by inertial/magnetic sensing," in *Sensors*, vol. 11, January 2011, pp. 1489–1525.

[28] M. D. Shuster and S. D. Oh, "Three-axis attitude determination from vector observations," in *Journal of Guidance, Control, and Dynamics*, vol. 4, no. 1, 1981, pp. 70–77.

[29] F. L. Markley, "Attitude determination from vector observations: A fast optimal matrix algorithm," in *J. Astronaut. Sci.*, vol. 41, no. 2, April-June 1993, pp. 261–280.

[30] X. Yun, E. Bachmann, and R. McGhee, "A simplified quaternion-based algorithm for orientation estimation from earth gravity and magnetic field measurements," in *Instrumentation and Measurement, IEEE Transactions on*, vol. 57, no. 3, March 2008, pp. 638–650.

[31] J. K. Lee and E. Park, "A fast quaternion-based orientation optimizer via virtual rotation for human motion tracking," in *Biomedical Engineering, IEEE Transactions on*, vol. 56, no. 5, May 2009, pp. 1574–1582.

[32] X. Yun, M. Lizarraga, E. Bachmann, and R. McGhee, "An improved quaternion-based Kalman filter for real-time tracking of rigid body orientation filter for real-time tracking of rigid body orientation," in *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 2, October 2003, pp. 1074–1079.

[33] S. Madgwick, A. Harrison, and R. Vaidyanathan, "An efficient orientation filter for IMU and MARG sensor arrays," Department of Mechanical Engineering, University of Bristol, Tech. Rep., April 2010.

[34] Y. Tian, H. Wei, and J. Tan, "An adaptive-gain complementary filter for real-time human motion tracking with MARG sensors in free-living environments," in *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 21, no. 2, March 2013, pp. 254–264.

[35] J. Calusdian, X. Yun, and E. Bachmann, "Adaptive-gain complementary filter of inertial and magnetic data for orientation estimation," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, May 2011, pp. 1916–1922.

[36] A. El Hadri and A. Benallegue, "Attitude estimation with gyros-bias compensation using low-cost sensors," in *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, December 2009, pp. 8077–8082.

[37] M. Wang, Y. Yang, R. Hatch, and Y. Zhang, "Adaptive filter for a miniature MEMS based attitude and heading reference system based attitude and heading reference system," in *Position Location and Navigation Symposium, 2004. PLANS 2004*, April 2004, pp. 193–200.

[38] "Invensense MEMS handling," Invensense Inc., 1745 Technology Drive, San Jose, CA95110 U.S.A, Application Note AN-IVS-0002A-00, November 2014.

[39] J. Seeger, M. Lim, and S. Nasiri, "Development of high-performance, high-volume consumer MEMS gyroscopes," in *In Proc. Tech. Dig. Solid-State Sensors Actuators Microsystems Workshop*, 1197 Borregas Avenue, Sunnyvale, CA 94089, USA, June 2010, pp. 61–64.

[40] STMicroeletronics, "Everything about STMicroelectronics' 3-axis digital MEMS gyroscopes," ST Microeletronics, Tech. Rep. Doc ID 022032 v01, July 2011.

[41] P. Qu and H. Qu, "Design and characterization of a fully differential MEMS accelerometer fabricated using metalmumps technology," in *Sensors*, no. 13, May 2013, pp. 5720–5736.

[42] "Using LSM303DLH for a tilt compensated electronic compass," ST Microeletronics, Tech. Rep., 2010.

[43] T. Ozyagclicar, "Calibrating an ecompass in the presence of hard and soft-iron interference," Freescale Semiconductor Ltd, Tech. Rep., 2012.

[44] P. Ripka and M. Janosek, "Advances in magnetic field sensors," in *IEEE Sensors Journal*, vol. 10, no. 6, June 2010, pp. 1108–1116.

[45] A. Leuzinger and A. Taylor, "Magneto-inductive technology overview," PNI, White paper, February 2010.

[46] PNI, "RM3100 evaluation board user manual," PNI, Tech. Rep. Doc 1017252 r02.

[47] A. Chulliat, S. Macmillan, P. Alken, C. Beggan, M. Nair, B. Hamilton, A. Woods, V. Ridley, S. Maus, and A. Thomson, "The US/UK world magnetic model for 2015-2020," National Geophysical Data Center, NOAA, Tech. Rep. 10.7289/V5TB14V7, 2015.

[48] W. Storms, J. Shockley, and J. Raquet, "Magnetic field navigation in an indoor environment," in *Ubiquitous Positioning Indoor Navigation and Location Based Service (UPINLBS)*, Wright-Patterson AFB, United States, 2010, pp. 1–10.

[49] S. M. LaValle, A. Yershova, M. Katsev, and M. Antonov, "Head tracking for the oculus rift," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, 2014, pp. 187–194.

[50] M. J. Caruso, "Applications of magnetic sensors for low cost compass systems," in *Position Location and Navigation Symposium*, 2000, pp. 177–184.

[51] T. Ozyagcilar, "Layout recommendations for PCBs using a magnetometer sensor," Freescale, Tech. Rep., 2011.

[52] K. R. Aires, A. M. Santana, and A. A. Medeiros, "Optical flow using color information: preliminary results," in *Proceedings of the 2008 ACM symposium on Applied computing*, 2008, pp. 1607–1611.

[53] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *IJCAI*, vol. 81, 1981.

[54] M. J. Black and A. D. Jepson, "Estimating optical flow in segmented images using variable-order parametric models with local deformations." in *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 18, no. 10, 1996, pp. 972–986.

[55] D. Font, M. Tresanchez, T. Pallejà, M. Teixidó, and J. Palacín, "Characterization of a low-cost optical flow sensor when using an external laser as a direct illumination source," in *Sensors*, vol. 11, no. 12, December 2011, pp. 11 856–11 870.

[56] T. Ng, "The optical mouse as a two-dimensional displacement sensor," in *Sensors and Actuators*, vol. 107, June 2003, pp. 21–25.

[57] N. Gageik, M. Strohmeier, and S. Montenegro, "An autonomous UAV with an optical flow sensor for positioning and navigation with an optical flow sensor for positioning and navigation," in *International Journal of Advanced Robotic Systems*, vol. 10, no. 341, July 2013.

[58] A. Technologies, *ADNS-3080 High-Performance Optical Mouse Sensor*. [Online]. Available: http://people.ece.cornell.edu/land/courses/ece4760/FinalProjects/s2009/ncr6_wjw27/ncr6_wjw27/docs/adns_3080.pdf

[59] T. Siah and H. Kong, "Optical navigation using one-dimensional correlation," Avago Technologies, 2005.

[60] S. Madgwick, "Quaternions," September 2011.

[61] R. Mahony, T. Hamel, and J.-M. Pflimlin, "Nonlinear complementary filters on the special orthogonal group," in *Automatic Control, IEEE Transactions on*, vol. 53, no. 5. IEEE, 2008, pp. 1203–1218.

[62] M. Euston, P. Coote, R. Mahony, J. Kim, and T. Hamel, "A complementary filter for attitude estimation of a fixed-wing UAV," in *Intelligent Robots and Systems, 2008 IEEE/RSJ International Conference on*, 2008, pp. 340–345.

[63] D. D. S. Santana, "Estimação de trajetórias terrestres utilizando unidade de medição inercial de baixo custo e fusão sensorial," Master's thesis, Escola Politecnica da Universidade de Sao Paulo, Sao Paulo, 2005.

[64] R. Mahony, T. Hamel, and S. Cha, "A coupled estimation and control analysis for attitude stabilisation of mini aerial vehicles," in *Australiasian Conference on Robotics and Automation*, vol. 1-10, Auckland, New Zealand, 2006.

[65] D. Comotti and M. Ermidoro, "Sviluppo di algoritmi per la stima dell'orientamento di un sensore inerziale," Master's thesis, University of Bergamo, March 2011.

[66] G. Welch and G. Bishop, "An introduction to the Kalman filter," University of North Carolina at Chapel Hill, Tech. Rep., 2006.

[67] K. Seifert and O. Camacho, "Implementing positioning algorithms using accelerometers," Freescale Semiconductor, Application Note AN3397, February 2007.

[68] O. J. Woodman, "An introduction to inertial navigation," University of Cambridge, 15 JJ Thomson Avenue Cambridge CB3 0FD United Kingdom, Tech. Rep. UCAM-CL-TR-696, August 2007.

[69] I. Inc., *MPU-9250 Product Specification Revision 1.0*, Invensense Inc., 1745 Technology Drive, San Jose, CA 95110 U.S.A., January 2014.

[70] R. Santos, M. Rasteiro, H. Costelha, L. Bento, and P. Assuncao, "Motion-based remote control device for enhanced interaction with 3d multimedia content," in *Conference on Telecommunications (Conftele 2015)*, September 2015.

[71] A. Oppenheim and R. Schafer, *Discrete-time signal processing*, Prentice-hall, Ed. Englewood Cliffs, 1989, vol. 2.

[72] E. B. Dam, M. Koch, and M. Lillholm, "Quaternions, interpolation and animation," University of Copenhagen, Tech. Rep., July 1998.

[73] A. Janota, V. Šimák, D. Nemec, and J. Hrbček, "Improving the precision and speed of euler angles computation from low-cost rotation sensor data," in *Sensors*, vol. 15, no. 3, March 2015, pp. 7016–7039.

[74] D. Eberly, "Rotation representations and performance issues," Geometric Tools, LLC, Tech. Rep., 2002.

[75] J. B. Kuipers, "Quaternions and rotation sequence," Calvin College, Grand Rapids, MI 49546, USA, Tech. Rep., September 1999.

[76] A. Thompson, "Euler sequence and conversions," Advanced Technology Associates, Tech. Rep. [Online]. Available: http://www.atacolorado.com/library.htm

[77] J. R. W. Computer Sciences Corporation. Attitude Systems Operation, *Spacecraft Attitude Determination and Control*, S. S. . B. Media, Ed. Reidel, 1978. [Online]. Available: https://books.google.pt/books?id=GtzzpUN8VEoC

This page was intentionally left blank.

Appendix A

Mathematical Background

In this appendix some mathematical notions and notation are further described for the sake of completion of this document.

A.1 Rotation representations

There are various mathematical formulations to represent the orientation of a rigid body in Euclidean space. In the DCM, RPY angles and quaternion representations are the most commonly used. There is no real answer for which is the best representation for rotations, it depends of the application and there are a few trade-offs to consider.

Quaternions were chosen as the mathematical ground in this work because they deliver a more compact representation than DCM, and need less operations to compute successive rotations. Moreover, during processing operation, numerical errors cumulate, which deteriorates the representations. A quaternion only needs to be normalized in order to represent an orientation; a DCM, besides a more expensive normalization, also needs to be orthogonal to represent the same orientation. Quaternions do not suffer from the singularities commonly know as gimbal lock, which occurs in RPY angles. Also, they allow a smooth interpolation between two points, while in RPY angles there is up to a three step interpolation [72, 73, 74].

Table A.1: Number of operations for quaternions and DCM representations [74].

	Vector frame swap		Successive Rotations	
	DCM	Quaternions	DCM	Quaternions
Multiplications	9	24	27	16
Aditions	6	17	18	12

Table A.1 compares the number of basic operations for DCM and quaternions. RPY angles are not used for tracking the rotation of a rigid body because it implies the use of

trigonometric functions. Keep in mind that it is always possible to optimize representations for the application requirements reducing the number of required operations.

A.2 Direction cosine matrices

A DCM is a transformation matrix obtained with the cosines between all the axes of two frames. In order to determine the relative orientation of a free rotating frame B in relation to a fixed frame A, where A is constituted by $\boldsymbol{x}_a, \boldsymbol{y}_a$ and $\boldsymbol{z}_a$ axes and B by $\boldsymbol{x}_b, \boldsymbol{y}_b$ and $\boldsymbol{z}_b$ axes, then, the DCM that transforms a vector from frame A to frame B is designated by ${}^{A}_{B}\boldsymbol{T}$ and is given by Equation (A.1), where $\lhd$ represents the closest angle between axes.

$$ {}^{A}_{B}\boldsymbol{T} = \begin{bmatrix} \cos(\boldsymbol{x}_a \lhd \boldsymbol{x}_b) & \cos(\boldsymbol{y}_a \lhd \boldsymbol{x}_b) & \cos(\boldsymbol{z}_a \lhd \boldsymbol{x}_b) \\ \cos(\boldsymbol{x}_a \lhd \boldsymbol{y}_b) & \cos(\boldsymbol{y}_a \lhd \boldsymbol{y}_b) & \cos(\boldsymbol{z}_a \lhd \boldsymbol{y}_b) \\ \cos(\boldsymbol{x}_a \lhd \boldsymbol{z}_b) & \cos(\boldsymbol{y}_a \lhd \boldsymbol{z}_b) & \cos(\boldsymbol{z}_a \lhd \boldsymbol{z}_b) \end{bmatrix} \tag{A.1} $$

Or, put in a different way, by the internal product as shown in Equation (A.2).

$$ {}^{A}_{B}\boldsymbol{T} = \begin{bmatrix} \boldsymbol{x}_a \cdot \boldsymbol{x}_b & \boldsymbol{y}_a \cdot \boldsymbol{x}_b & \boldsymbol{z}_a \cdot \boldsymbol{x}_b \\ \boldsymbol{x}_a \cdot \boldsymbol{y}_b & \boldsymbol{y}_a \cdot \boldsymbol{y}_b & \boldsymbol{z}_a \cdot \boldsymbol{y}_b \\ \boldsymbol{x}_a \cdot \boldsymbol{z}_b & \boldsymbol{y}_a \cdot \boldsymbol{z}_b & \boldsymbol{z}_a \cdot \boldsymbol{z}_b \end{bmatrix} \tag{A.2} $$

These are orthogonal matrices so, a DCM transpose is equal to its inverse, which becomes a very important property that can be useful to allow reducing the computational needs, for instance when inverting rotations. In this case, one has:

$$ {}^{A}_{B}\boldsymbol{T}^{T} = {}^{A}_{B}\boldsymbol{T}^{-1} = {}^{B}_{A}\boldsymbol{T} \tag{A.3} $$

and:

$$ det({}^{A}_{B}\boldsymbol{T}) = det({}^{B}_{A}\boldsymbol{T}) \tag{A.4} $$

Elementary transformations of the coordinate system around only one axis of a coordinate fixed frame by a given θ angle results in the following matrices:

$$ \boldsymbol{R}_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix} \tag{A.5} $$

$$ \boldsymbol{R}_y(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{A.6} $$

$$\boldsymbol{R}_z(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.7}$$

A.3 RPY angles

The pose of a coordinate frame relative to another one can be completely described within a maximum set of three sequential rotations around the axes of the fixed frame. This set of angles are designated as RPY angles. These are given by the right hand rule, where usually the roll (ϕ) angle is around the x axis (normally this is the axis that points in the direction of the body, widely use in the aeronautical convention), the pitch (θ) angle is around the y axis (the other axis in the motion plane of the body), and the yaw angle (ψ) around the z axis (See Figure A.1).

The rotations can be performed around any axis, as long as there are no consecutive rotations around the same axis. Altogether, there are 12 different rotation sequences which can fully describe the final orientation of any body. The rotation matrix is obtained by multiplying the sequence of elementary rotation.

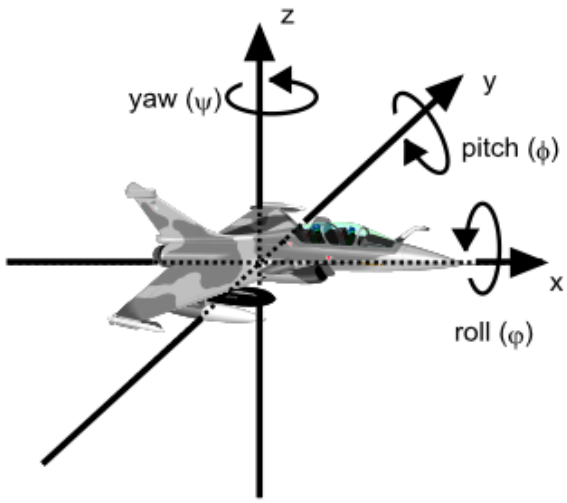

Figure A.1: Roll, pitch and yaw angles used in an aeronautical convention.

As explained, the pose of a body, associated to a frame B, can be described by three successive rotations: $\boldsymbol{R}_z(\theta_z)$, $\boldsymbol{R}_y(\theta_y)$ and $\boldsymbol{R}_x(\theta_x)$; these rotations around the z axis, y and x, respectively, of fixed a frame A, with a magnitude of θ_z, θ_y and θ_x, respectively, results in a rotation sequence given by Equation (A.8), in the base coordinate frame.

$$B = \boldsymbol{R}^x(\theta_x) \cdot \boldsymbol{R}^y(\theta_y) \cdot \boldsymbol{R}^z(\theta_z) \cdot A \tag{A.8}$$

Using matrices (A.5), (A.6) and (A.7) in rotation (A.8), representing $sin(\theta)$ and $cos(\theta)$ as s_θ and c_θ, Equation (A.9) is obtained.

$$ {}^{A}_{B}\boldsymbol{R}_{zyx} = \begin{bmatrix} c_{\theta_y} s_{\theta_x} & c_{\theta_y} s_{\theta_x} & -s_{\theta_y} \\ s_{\theta_z} s_{\theta_y} c_{\theta_x} - c_{\theta_z} s_{\theta_x} & s_{\theta_x} s_{\theta_y} s_{\theta_z} + c_{\theta_x} c_{\theta_z} & s_{\theta_z} c_{\theta_y} \\ c_{\theta_x} s_{\theta_y} c_{\theta_z} + s_{\theta_x} s_{\theta_z} & c_{\theta_z} s_{\theta_y} s_{\theta_x} - s_{\theta_z} c_{\theta_x} & c_{\theta_z} c_{\theta_y} \end{bmatrix} \tag{A.9} $$

All 12 rotation sequences can be represented in matrices like the one in (A.9), but for small angles (values lower than 10°), they can be linearised and congregated into a single, simpler, matrix:

$$ \boldsymbol{R} = \begin{bmatrix} 1 & \theta_z & -\theta_x \\ -\theta_z & 1 & \theta_y \\ \theta_x & -\theta_y & 1 \end{bmatrix} \tag{A.10} $$

A.4 Quaternions

Quaternions are an extension of the complex number into 3 dimensions, first described by Irish mathematician William Rowan Hamilton in 1843, normally used to represent rotations. They have a scalar component s and three vectorial components $x\hat{i}$, $y\hat{j}$ e $z\hat{k}$ [75]. Equation (A.11) represents a generic quaternion, which has the fundamental property described in Equation (A.12).

$$ q = s + x\hat{i} + y\hat{j} + z\hat{k} \tag{A.11} $$

$$ i^2 = j^2 = k^2 = ijk = -1 \tag{A.12} $$

According to the Euler theorem, any rotation from B to A can be achieved through a single rotation of an angle θ around a vector $\boldsymbol{l}$ defined in frame A (see Figure A.2). The vectorial part of a quaternion indicates the axis of rotation and the scalar component relates to the magnitude of this rotation.

If represented in a four dimensional vector, it takes the form of Equation (A.13).

$$ \boldsymbol{q} = \begin{bmatrix} s \\ \boldsymbol{v} \end{bmatrix} = \begin{bmatrix} s \\ l_x \\ l_y \\ l_z \end{bmatrix} = \begin{bmatrix} q_s \\ q_x \\ q_y \\ q_z \end{bmatrix} \tag{A.13} $$

The quaternion that gives the orientation of frame B relative to a reference frame A,

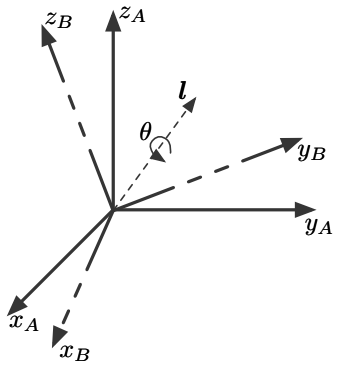

Figure A.2: Rotation of frame B relates to frame A around $\boldsymbol{l}$ by an angle θ.

i.e., rotated by an angle θ around the axis described by vector $\boldsymbol{l}$, represented in frame A, is presented in Equation A.15. Here $\boldsymbol{l}$ has to be normalized according to Equation (A.14).

$$\hat{\boldsymbol{l}} = \frac{\boldsymbol{l}}{\sqrt{l_x^2 + l_y^2 + l_z^2}} \tag{A.14}$$

$$^{A}_{B}\boldsymbol{q} = \begin{bmatrix} cos(\frac{\theta}{2}) \\ \hat{l}_x \cdot sin(\frac{\theta}{2}) \\ \hat{l}_y \cdot sin(\frac{\theta}{2}) \\ \hat{l}_z \cdot sin(\frac{\theta}{2}) \end{bmatrix} \tag{A.15}$$

A quaternion only represents a rotation if it is normalized. The normalization procedure is presented in Equation (A.16) [75].

$$\hat{\boldsymbol{q}} = \frac{\boldsymbol{q}}{\parallel \boldsymbol{q} \parallel} = \frac{\boldsymbol{q}}{\sqrt{q_s^2 + q_x^2 + q_y^2 + q_z^2}} \tag{A.16}$$

Note that, after normalization, it results in: $\parallel \hat{\boldsymbol{q}} \parallel = 1$.

Its conjugate is much like complex numbers, the real part stays the same and the imaginary part is symmetric. This operations is described in Equation (A.17).

$$\boldsymbol{q}^* = \begin{bmatrix} s \\ -\boldsymbol{v} \end{bmatrix} = \begin{bmatrix} q_s \\ -q_x \\ -q_y \\ -q_z \end{bmatrix} \tag{A.17}$$

Its norm is calculated using Equation (A.18).

$$\boldsymbol{q}\boldsymbol{q}^* = \parallel \boldsymbol{q} \parallel^2 \tag{A.18}$$

And its inverse using Equation (A.19).

$$\boldsymbol{q}^{-1} = \frac{\boldsymbol{q}^*}{\| \boldsymbol{q} \|^2} \tag{A.19}$$

Therefore, for normalized quaternions, the Equation (A.20) is valid.

$$\hat{\boldsymbol{q}}^{-1} = \hat{\boldsymbol{q}}^* \tag{A.20}$$

Quaternions are very useful for rotation representations using micro-controllers, because this way, inverse rotations can be done simply by following Equation (A.21) which leads to a very efficient form of processing rotations.

$${}^{A}_{B}\hat{\boldsymbol{q}}^{-1} = {}^{A}_{B}\hat{\boldsymbol{q}}^{*} = {}^{B}_{A}\hat{\boldsymbol{q}} \tag{A.21}$$

To perform the transformation of a vector $\boldsymbol{l}$ (a vector can be seen as a pure quaternion, with its real part equal to 0) from frame A to frame B, one needs to multiply the quaternion that gives the orientation of B relatively to A, by the vector and by the inverse quaternion, as in Equation (A.22) [60].

$${}^{B}\boldsymbol{l} = {}^{A}_{B}\hat{\boldsymbol{q}} \otimes {}^{A}\boldsymbol{l} \otimes {}^{A}_{B}\hat{\boldsymbol{q}}^{-1} \tag{A.22}$$

The result of the quaternion ($\boldsymbol{q}$ and $\mathbf{p}$) multiplication, according to the Hamilton rule, is obtained by Equation (A.23). This is a non-commutative operation, i.e.: $\boldsymbol{q} \otimes \mathbf{p} \neq \mathbf{p} \otimes \boldsymbol{q}$.

$$\boldsymbol{q} \otimes \mathbf{p} = \begin{bmatrix} q_s p_s - q_x p_x - q_y p_y - q_z p_z \\ q_s p_x + q_x p_s + q_y p_z - q_z p_y \\ q_s p_y - q_x p_z + q_y p_s + q_z p_x \\ q_s p_z + q_x p_y - q_y p_x + q_z p_s \end{bmatrix} \tag{A.23}$$

As an example, a 90° rotation of vector $\boldsymbol{l} = [0\ 1\ 1\ 1]^T$, around the x axis of frame A, is given by Equation (A.24).

$$\boldsymbol{l}' = \begin{bmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \\ 0 \\ 0 \end{bmatrix} \otimes \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \otimes \begin{bmatrix} \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{-\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \\ 0 \\ \sqrt{2} \end{bmatrix} \otimes \begin{bmatrix} \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ -1 \\ 1 \end{bmatrix} \tag{A.24}$$

The intermediate step of Equation (A.24) is difficult to perceive since at that step of the operation, the vector assumes dimension 4. Only after being multiplied by the inverse of the quaternion, it is brought back to dimension 3, which is more intuitive. Its easy

to see that the vector will suffer a rotation around x axis, since the quaternion only has values in the scalar component and in the first vectorial component.

A.5 Conversions between representations

The rotation sequence has major importance when establishing the equivalence between quaternion elements and RPY angles. For the "zyx" sequence used in this work, the rotation matrix can be obtained from the quaternions according to Equation (A.25) and the roll, pitch and yaw angles from Equations (A.26-A.28). A full description of all rotation sequences conversions can be found in [76, 77].

$$ {}^{A}_{B}\boldsymbol{R} = \begin{bmatrix} q_s^2 + q_x^2 - q_y^2 - q_z^2 & 2(q_x q_y + q_s q_z) & 2(q_x q_z - q_s q_y) \\ 2(q_x q_y - q_s q_z) & q_s^2 - q_x^2 + q_y^2 - q_z^2 & 2(q_y q_z + q_s q_x) \\ 2(q_x q_z + q_s q_y) & 2(q_y q_z - q_s q_x) & q_s^2 - q_x^2 - q_y^2 + q_z^2 \end{bmatrix} \tag{A.25} $$

$$ \phi = arctan\left(\frac{2(q_x q_y + q_s q_z)}{q_s^2 + q_x^2 - q_y^2 - q_z^2}\right) \tag{A.26} $$

$$ \theta = arcsin\left(-2(q_x q_z - q_s q_y)\right) \tag{A.27} $$

$$ \psi = arctan\left(\frac{2(q_y q_z + q_x q_s)}{q_s^2 - q_x^2 - q_y^2 + q_z^2}\right) \tag{A.28} $$

This page was intentionally left blank.

Appendix B

Sensors Calibration

The reduced cost of MEMS is advantageous for any kind of system, but these sensors are characterized by larger errors and deviations. The calibration of such sensors is always necessary in order to obtain more accurate values. The calibration type is adjusted according to the application and precision needed. Calibrations can only correct systematic errors such as constant bias, scale factors and axes misalignment. This appendix describes how MEMS sensors can be calibrated.

B.1 Gyroscope

Gyroscope readings have several problems as explained in Section 3.1.This sensor is characterized by having a high noise level and bias when stationary, but very precise readings when subjected to angular velocities. Assuming a model for the gyroscope in its own reference frame, the real angular velocity, $\boldsymbol{\omega_{real}}$, one wants to measure, is given by equation (B.1) [40].

$$\boldsymbol{\omega_{real}} = \boldsymbol{F}_g \cdot (\boldsymbol{\omega} - \boldsymbol{\omega}_0) \tag{B.1}$$

- $\boldsymbol{\omega_{real}}$ - real angular velocity [º/s];
- $\boldsymbol{F}_g$ - Scale factor for the gyroscope (converts Hardware Units (HU) into º/s).
- $\boldsymbol{\omega}$ - Raw readings in HU;
- $\boldsymbol{\omega}_0$ - Average raw readings in HU at rest.

Minimal usage calibration

In order to find $\boldsymbol{\omega}_0$, initial readings should be discarded in order to eliminate any initial instability, allowing the sensor to warm-up. After this step, the readings average for each

gyroscope axis gives $\boldsymbol{\omega}_0$. Readings change with time and temperature variations, so a moving average filter can be used to compensate variations in $\boldsymbol{\omega}_0$. However, it has to be assured that these readings are made with the sensor "at rest", meaning that the sensor is not subjected to any angular velocity. An example of a calibration of this type can be seen in Figure B.1 where, as can be seen, the uncalibrated readings present an initial bias and are quite noisy. The calibration corrects the "at rest" bias.

One way to eliminate noise "at rest" (low frequency) is by setting limits to valid readings with a discriminating window. Readings within this window are set to zero. The problem with this approach is that very slow rotations may be ignored.

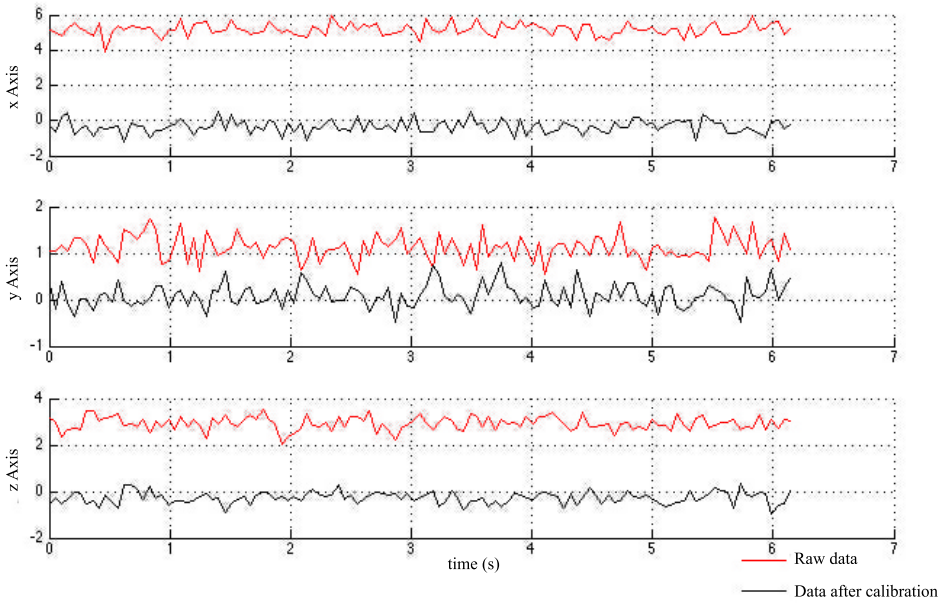

Figure B.1: Gyroscope simple calibration example.

Extended calibration

By expanding Equation (B.1), Equation (B.2) can be written.

$$\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} F_{gx} & 0 & 0 \\ 0 & F_{gy} & 0 \\ 0 & 0 & F_{gz} \end{bmatrix} \cdot \begin{bmatrix} \omega_{rx} - \omega_{0x} \\ \omega_{ry} - \omega_{0y} \\ \omega_{rz} - \omega_{0z} \end{bmatrix} \tag{B.2}$$

The determination of $\boldsymbol{F}_g$ for scale factors and non-orthogonalities correction is more difficult, requiring the use of reference systems. It is only necessary for highly sensitive applications, where the precise angular velocity is needed. The matrices of Equation (B.2) can be expressed as Equation (B.3), and the twelve parameters G_{mn} have to be

determined.

$$\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} G_{11} & G_{12} & G_{13} \\ G_{21} & G_{22} & G_{23} \\ G_{31} & G_{32} & G_{33} \end{bmatrix} \cdot \begin{bmatrix} \omega_{rx} - G_{10} \\ \omega_{ry} - G_{20} \\ \omega_{rz} - G_{30} \end{bmatrix} \tag{B.3}$$

G_{10}, G_{20} and G_{30} are calculated using the previous method for bias removal. The remaining parameters may be determined using a rotary table with an axis only, or any set where a constant known angular velocity can be applied [40]. An example of the set is outlined in Figure B.2.

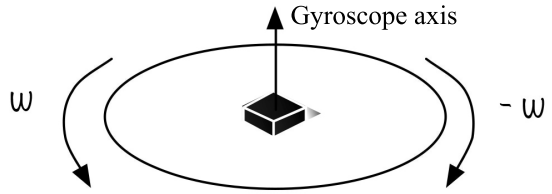

Figure B.2: Generic setup for gyroscope extended calibration .

In order to calibrate the gyroscope, one first need to obtain the angular velocity value read by the gyroscope while knowing the real angular velocity. Measurements should be done for each axis independently at two different speeds and in opposite directions. Table B.1 is an example of the measurements that can be done with the gyroscope. At least 100 samples should be collected after the angular velocity stabilizes [40].

Table B.1: Angular velocity and expected values for gyroscope extended calibration.

Gyroscope Position	**Angular velocity (º/s)**	ω_x **(º/s)**	ω_y **(º/s)**	ω_z **(º/s)**
x axis up	+50	+50	0	0
	-50	-50	0	0
	+100	+100	0	0
	-100	-100	0	0
y axis up	+50	0	+50	0
	-50	0	-50	0
	+100	0	+100	0
	-100	0	-100	0
z axis up	+50	0	0	+50
	-50	0	0	-50
	+100	0	0	+100
	-100	0	0	-100

After data collection, a matrix with the true angular velocity and a matrix with the average angular velocity are constructed as in Equation (B.4). Then, applying a Least

Mean Square (LMS) method, the remaining nine parameters from Equation (B.3) can be found.

$$\begin{bmatrix} 50 & 0 & 0 \\ -50 & 0 & 0 \\ 100 & 0 & 0 \\ -100 & 0 & 0 \\ 0 & 50 & 0 \\ 0 & -50 & 0 \\ 0 & 100 & 0 \\ 0 & -100 & 0 \\ 0 & 0 & 50 \\ 0 & 0 & -50 \\ 0 & 0 & 100 \\ 0 & 0 & -100 \end{bmatrix} = \begin{bmatrix} \omega_{rx1} & \omega_{ry1} & \omega_{rz1} \\ \omega_{rx2} & \omega_{ry2} & \omega_{rz2} \\ \omega_{rx3} & \omega_{ry3} & \omega_{rz3} \\ \omega_{rx4} & \omega_{ry4} & \omega_{rz4} \\ \omega_{rx5} & \omega_{ry5} & \omega_{rz5} \\ \omega_{rx6} & \omega_{ry6} & \omega_{rz6} \\ \omega_{rx7} & \omega_{ry7} & \omega_{rz7} \\ \omega_{rx8} & \omega_{ry8} & \omega_{rz8} \\ \omega_{rx9} & \omega_{ry9} & \omega_{rz9} \\ \omega_{rx10} & \omega_{ry10} & \omega_{rz10} \\ \omega_{rx11} & \omega_{ry11} & \omega_{rz11} \\ \omega_{rx12} & \omega_{ry12} & \omega_{rz12} \end{bmatrix} \cdot \begin{bmatrix} G_{11} & G_{12} & G_{13} \\ G_{21} & G_{22} & G_{23} \\ G_{31} & G_{32} & G_{33} \end{bmatrix} \tag{B.4}$$

In another format, Equation (B.4) can be generically represented as Equation (B.5).

$$\boldsymbol{Y}_{\omega} = \boldsymbol{W}_{\omega} \cdot \boldsymbol{X}_{\omega} \tag{B.5}$$

Where:

- $\boldsymbol{Y}_{\omega}$ - is a 12×3 matrix with real angular velocity;
- $\boldsymbol{W}_{\omega}$ - is the collected data from the gyroscope for the tests performed (12×3 matrix);
- $\boldsymbol{X}_{\omega}$ - is a 3×3 calibration matrix that is to be computed.

All G_{mn} parameters from the $\boldsymbol{F}_g$ matrix can be determined, using the pseudo-inverse as in Equation (B.6).

$$\boldsymbol{X}_{\omega} = [\boldsymbol{W}_{\omega}^{T} \cdot \boldsymbol{W}_{\omega}]^{-1} \cdot \boldsymbol{W}_{\omega}^{T} \cdot \boldsymbol{Y}_{\omega} \tag{B.6}$$

B.2 Accelerometers

Similarly to the gyroscope sensor there are calibrations for the accelerometer that can be done with minimal human intervention, simply by ensuring a rest position of the sensor (no external acceleration besides gravity). These calibrations are to remove systematic measurement bias, which vary both with temperature and in time, so, if there is a need, they will have to be updated periodically. There are also more complete calibrations to correct scaling factors and correct axes misalignment [42], which are also described in here.

Accelerometer measurements can be modelled as expressed in Equation (B.7), similarly to Equation (B.1).

$$\boldsymbol{a} = \boldsymbol{F}_a \cdot (\boldsymbol{a}_{raw} - \boldsymbol{a}_0) \tag{B.7}$$

Knowing that:

- $\boldsymbol{a}$ - Acceleration in g's (1g = 9.81 $m \cdot s^{-2}$);
- S_a - Scale factor for the accelerometer. Converts HU into g's;
- $\boldsymbol{a}_{raw}$ - Raw readings from the accelerometer in HU;
- $\boldsymbol{a}_0$ - Average raw readings in HU at rest.

At rest only the gravity acceleration is measured, so, for any orientation of the sensor, the readings norm after conversion to g's should satisfy Equation (B.8).

$$\parallel acc \parallel = \sqrt{acc_x^2 + acc_y^2 + acc_z^2} = 1 \tag{B.8}$$

Unfolding Equation (B.7) to Equation (B.9):

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \cdot \begin{bmatrix} a_{rx} - A_{10} \\ a_{ry} - A_{20} \\ a_{rz} - A_{30} \end{bmatrix} \tag{B.9}$$

The values of A_{mn} correspond to the calibration parameters needed to adjust the output data from the accelerometer. With this technique, the major error factors for these sensors are corrected, through the calibration procedure that is explained below. It consists in placing the sensor at six stationary positions, in all three orthogonal directions, while collecting the raw values (see Table (B.2)) in order to calculate the calibration matrix. In order to apply a LMS method, Equation (B.9) can also be written as Equation (B.10).

$$\begin{bmatrix} a_x & a_y & a_z \end{bmatrix} = \begin{bmatrix} a_{rx} & a_{ry} & a_{rz} & -1 \end{bmatrix} \cdot \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \\ A_{10} & A_{20} & A_{30} \end{bmatrix} \tag{B.10}$$

Or yet, as Equation (B.11).

$$\boldsymbol{Y}_a = \boldsymbol{W}_a \cdot \boldsymbol{X}_a \tag{B.11}$$

Where:

- $\boldsymbol{Y}_a$ - is the known normalized gravity vector ([0 0 1]);

- $\boldsymbol{W}_a$ - Collected data from the 6 stationary positions;
- $\boldsymbol{X}_a$ - is the calibration matrix that need to be determined.

Ideally the vector $\boldsymbol{Y}_a$ would output the values shown in table B.2.

Table B.2: Expected output values for the 6 calibration positions.

Position	a_x	a_y	a_z
z axis up	0	0	+1g
z axis down	0	0	-1g
y axis up	0	+1g	0
y axis down	0	-1g	0
x axis up	+1g	0	0
x axis down	-1g	0	0

An average of the output values for each axis from the sensor, for the six positions, is placed in an Equation system like in Equation (B.12).

$$\begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} a_{rx1} & a_{ry1} & a_{rz1} & -1 \\ a_{rx2} & a_{ry2} & a_{rz2} & -1 \\ a_{rx3} & a_{ry3} & a_{rz3} & -1 \\ a_{rx4} & a_{ry4} & a_{rz4} & -1 \\ a_{rx5} & a_{ry5} & a_{rz5} & -1 \\ a_{rx6} & a_{ry6} & a_{rz6} & -1 \end{bmatrix} \cdot \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \\ A_{10} & A_{20} & A_{30} \end{bmatrix} \tag{B.12}$$

Where a_{rxn}, a_{ryn} e a_{rzn}, $(n = 1, 2, 3, 4, 5, 6)$, correspond to the readings average of each axis, in each of the six positions. It is recommended to collect, for about 10 seconds, data with a sampling frequency of at least 100 Hz, for each axis.

Rewriting Equation (B.12) to Equation (B.13), and applying the LMS method is possible to determine the calibration matrix (equation (B.14)).

$$\boldsymbol{Y}_{6\times3} = \boldsymbol{W}_{6\times4} \cdot \boldsymbol{X}_{4\times3} \tag{B.13}$$

$$\boldsymbol{X}_a = [\boldsymbol{W}_a^T \cdot \boldsymbol{W}_a]^{-1} \cdot \boldsymbol{W}_a^T \cdot \boldsymbol{Y}_a \tag{B.14}$$

Figure B.3 shows one example of an accelerometer calibration. In this figure one can observe the effect of the systematic deviation correction that clearly exists on the x axis, and correction of the visible scale factor on the z axis.

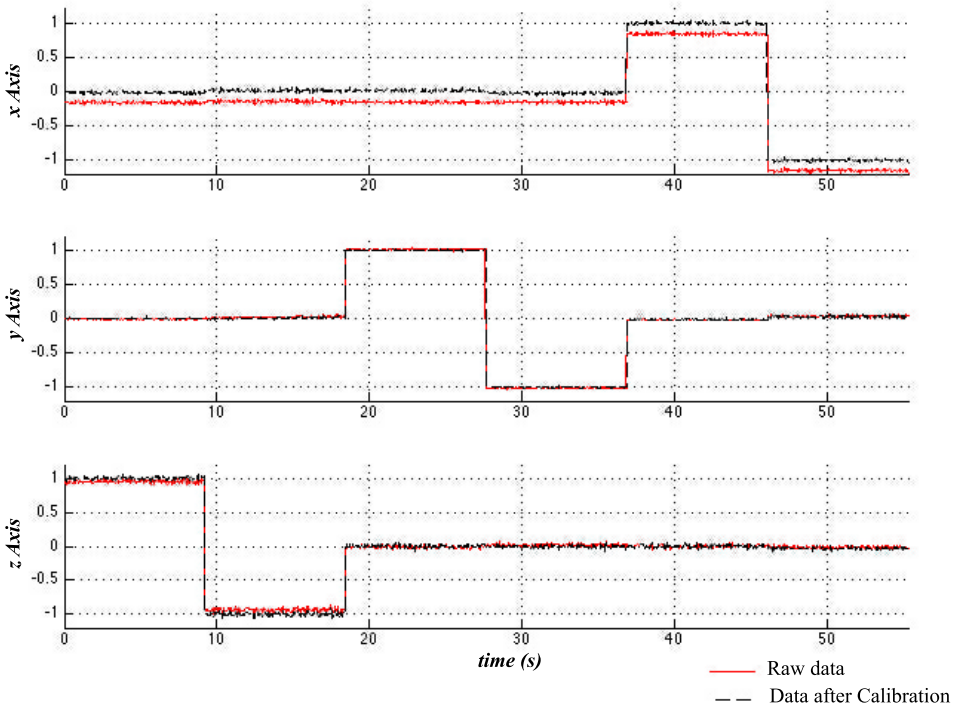

Figure B.3: Example of an accelerometer calibration.

B.3 Magnetometers

Soft and *hard-iron* distortions, discussed in Section 3.1.3, can be compensated using the method presented here. As referred, the calibration is only valid for the location where it is executed.

Similarly to the accelerometer and the gyroscope, 12 parameters can be calculated for a calibration matrix in order to obtain a correct output of the magnetometer [42], by assuming a similar model of the sensor, as expressed in Equation (3.26).

$$\boldsymbol{m} = \boldsymbol{F}_m \cdot (\boldsymbol{m}_{raw} - \boldsymbol{m}_0) \tag{B.15}$$

Where:

- $\boldsymbol{m}$ - Corrected measurement of the magnetic field [μT];
- $\boldsymbol{m}_{raw}$ - Raw data from magnetometer;
- $\boldsymbol{F}_m$ - Parameters that compensate for axis misalignment, scaling factors and *soft-iron* effects;
- $\boldsymbol{m}_0$ - Parameters that compensate *hard-iron* effects.

Equation (B.15) can be seen as Equation (B.16). Once again twelve parameters exist,

which need to be calculated.

$$\begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix} \cdot \begin{bmatrix} m_{rx} - M_{10} \\ m_{ry} - M_{20} \\ m_{rz} - M_{30} \end{bmatrix} \tag{B.16}$$

Calibration is executed by rotating the sensor on its own axis in order to try to cover the largest possible number of points of a sphere. The collection of such data should be made for at least 30 seconds at 100 Hz (typically the magnetometer is used at lower sampling rates during operation, but, to obtain a good calibration, since Hall effect magnetometers have an high level of noise, the sampling rate should be as high as possible). If there are distortions, the readings result in an ellipsoid that can be described following formula in Equation (B.17) [43].

$$h_1x^2 + h_2y^2 + h_3z^2 + 2h_4xy + 2h_5xz + 2h_6yz + 2h_7x + h_8hy + 2h_9z + h_0 = 0 \tag{B.17}$$

Or in a matricial form defined as Equations (B.18).

$$\mathbf{C}^T\mathbf{N}\mathbf{C} = 0 \tag{B.18}$$

With:

$$\mathbf{C} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad \mathbf{N} = \begin{bmatrix} h_1 & h_4 & h_5 & h_7 \\ h_4 & h_2 & h_6 & h_8 \\ h_5 & h_6 & h_3 & h_9 \\ h_7 & h_8 & h_9 & h_0 \end{bmatrix} \tag{B.19}$$

Consider Equation (B.20).

$$\mathbf{L} = \begin{bmatrix} h_1 & h_4 & h_5 \\ h_4 & h_2 & h_6 \\ h_5 & h_6 & h_3 \end{bmatrix} \qquad \mathbf{U} = \begin{bmatrix} h_7 \\ h_8 \\ h_9 \end{bmatrix} \tag{B.20}$$

The coordinates for the centre of the ellipsoid ($\boldsymbol{E}_c$) are given by Equation (B.21),

$$\mathbf{E}_c = -\mathbf{L}^{-1}\mathbf{U} \tag{B.21}$$

And its semi-axes (d_1, d_2 and d_3) by Equations (B.22).

$$
\begin{aligned}
d_1 &= \left(\frac{\mathbf{E}_c{}^T \mathbf{L} \mathbf{E}_c - h_0}{\lambda_1} \right)^{\frac{1}{2}} \\
d_2 &= \left(\frac{\mathbf{E}_c{}^T \mathbf{L} \mathbf{E}_c - h_0}{\lambda_2} \right)^{\frac{1}{2}} \\
d_3 &= \left(\frac{\mathbf{E}_c{}^T \mathbf{L} \mathbf{E}_c - h_0}{\lambda_3} \right)^{\frac{1}{2}}
\end{aligned}
\tag{B.22}
$$

Here, λ_1, λ_2 and λ_3 are the eigen values of matrix Q. Grouping them into the diagonal matrix R, as shown in Equation (B.23), will allow its use in the computation of the twelve parameters of Equation (B.16).

$$
\mathbf{D} = \begin{bmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & d_3 \end{bmatrix} \tag{B.23}
$$

The normalized eigen vectors $\mathbf{v}_1$, $\mathbf{v}_2$ and $\mathbf{v}_3$ of matrix $\mathbf{L}$ describe the direction of the main three axes of the ellipsoid. Matrix $\mathbf{V}$ in Equation (B.24), formed by column vectors $\mathbf{v}_1$, $\mathbf{v}_2$ and $\mathbf{v}_3$, is the rotation matrix that describes the orientation of the ellipsoid relatively to the reference frame.

$$
\mathbf{V}_{3\times3} = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \mathbf{v}_3 \end{bmatrix} \tag{B.24}
$$

The parameters can be calculated using Equations (B.25) and (B.26).

$$
\begin{bmatrix} M_{10} \\ M_{20} \\ M_{30} \end{bmatrix} = \mathbf{E}_c \tag{B.25}
$$

$$
\begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix} = \mathbf{V}\ \mathbf{D}^{-1} \mathbf{V}^T \tag{B.26}
$$

Figure B.4 shows an example of the calibration of the magnetometer. As explained previously, the data is collected with rotational movements of the magnetometer in such a way, that it allows covering a large number of points on a sphere. The values of the raw sensor are 3D plotted and approximated to an ellipsoid (yellow in the figure). Following the procedure described herein, the calibrated data readings will tend to the sphere shown in blue.

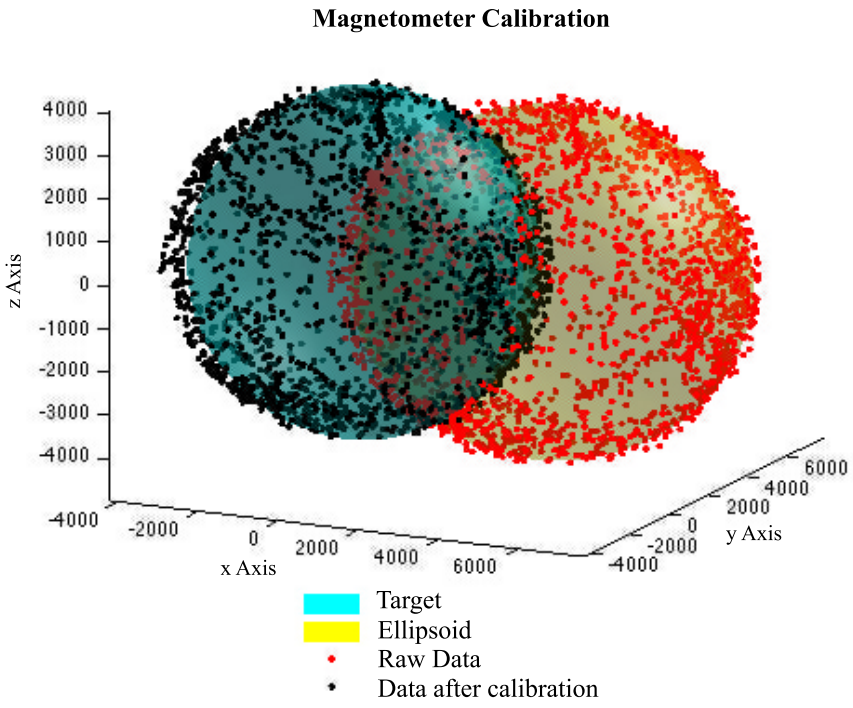

Figure B.4: Magnetometer calibration example.

Appendix C

User Interface Application

During the development of this work, there was a need for an application that could demonstrate the potentialities of the system in hands. This User Interface Application (UIA) was developed in a joint effort with R. Santos [70]. It was developed for Android systems, currently tested in an OUYA[1] STB with Android 4.2, and in the Android 5.1 on Nexus 7 and Nexus 10. The UIA was used to evaluate and test the result of the sensor fusion system in the RCD from the point of view of using it in applications. It can constitute the basis for a future API to develop for Android. It has a simple user interface with only one main menu from which it is possible to access new screens that allow different interactions. The main menu was created with a style popularized by Android, a drawer menu, which is hidden on the left side of the screen. The access to the menu can be done by simply clicking on the application icon at the top left of screen or dragging from the left to right from the border of the screen. In this menu (Figure C.1) there are four different options: Home, 3DVisualizer, TVSimulation and Logger.

Home

Is the default screen when the application starts and only has an image that describes the project;

3D Visualizer

Has a text box at the top left of the screen that displays, in real time, data received from the transmission of the RCD. The first four data fields represent the quaternion, the fifth corresponds to the identified gesture and the last three fields are the RPY angles (roll, pitch and yaw, respectively). The center of the screen has a figure, that follows the pose of the remote control;

TV Simulation

Aims to simulate a menu where the user can change the channels and choose the one

[1]For more information consult: www.ouya.tv

that he wants to see. The navigation in this menu can be performed by pressing the arrow in the screen with the mouse or through gestures, such as the ones described in Table 4.1;

Logger

Allows viewing and recording the last received data. This screen is divided into three parts, on the left it has a sequence of the last 20 lines of data received, updated in real time. On the right side there are three buttons, which allow the users to save, read or delete a file with the data displayed on the left side of the screen.

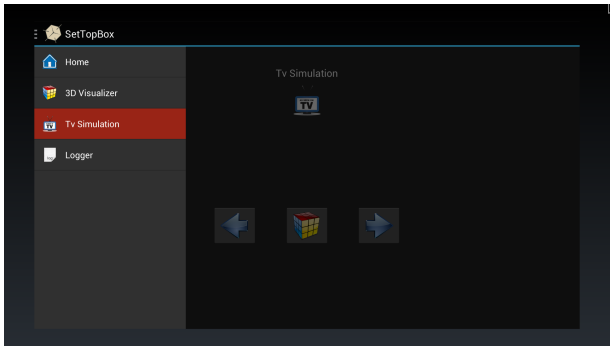

Figure C.1: Main Menu.

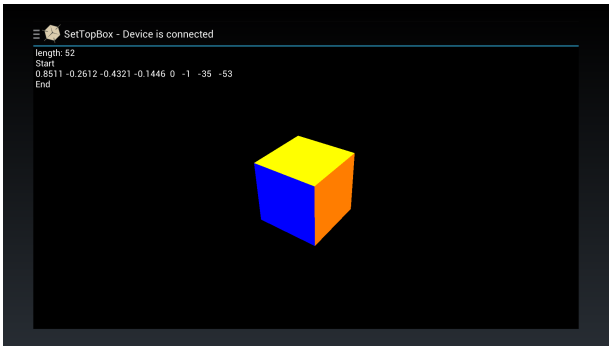

Figure C.2: 3D Visualizer.

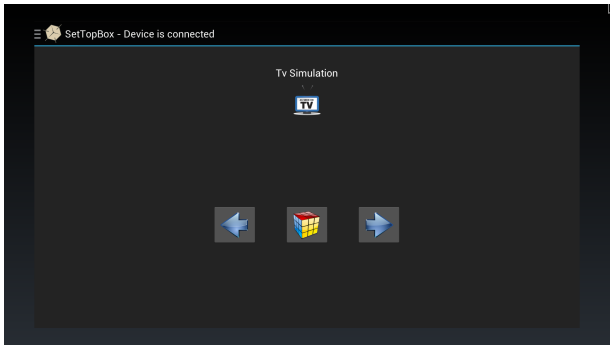

Figure C.3: TV Simulation.

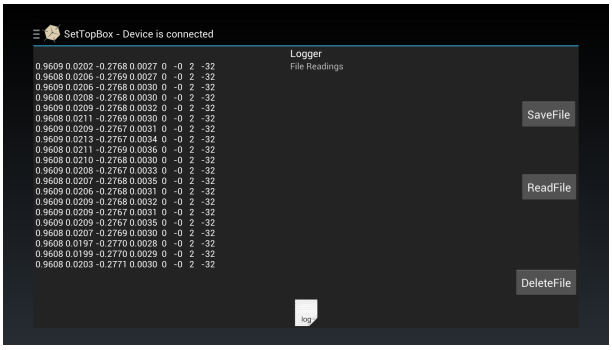

Figure C.4: Logger.

This UIA, as referred, was created during the development phase and besides being demonstrable, it allowed to test the algorithms subjectively, as well as gesture recognition.

Appendix D

RF Payload Frames

This appendix pretends to document the payload frames send from the RCD to the STB.

The RCD have six modes of operation, where the button frame (Figure D.1) can be send at any time, as well as the gestures (Figure D.2) and User Id frame (Figure D.3). All the other frames (Figure D.5-D.7) correspond directly to the mode with same name. The frames do not have any specific order to be sent by the Remote Control Device.

As referred in Section 4.2, there are 6 available modes, namely:

- Idle;
- Relative Air Mouse;
- Absolute Air Mouse;
- Demo;
- Scroll Gestures;
- Sensors.

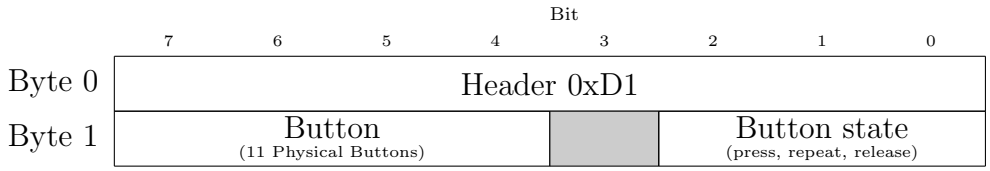

Figure D.1: Button frame – 2 Bytes

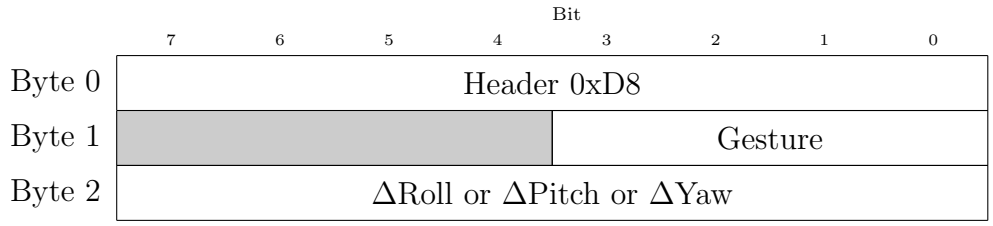

Figure D.2: Gestures frame – 3 Bytes

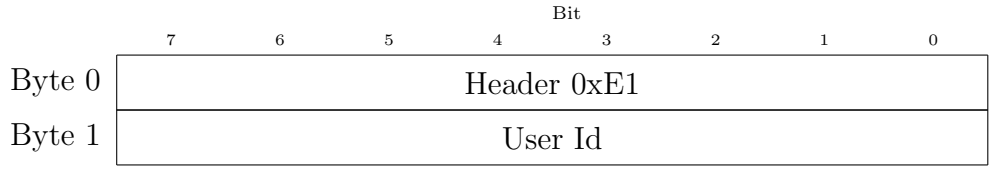

Figure D.3: User Id frame – 2 Bytes

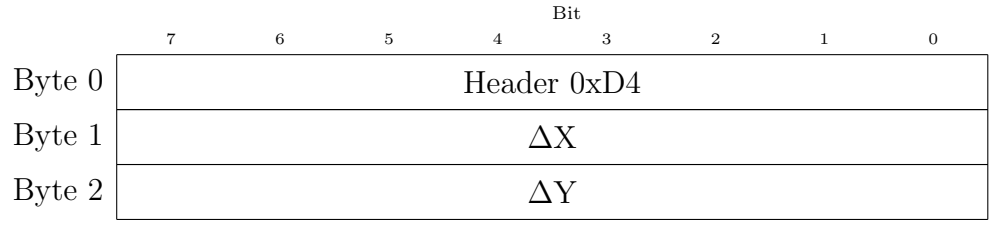

Figure D.4: Relative Air Mouse frame – 3 Bytes

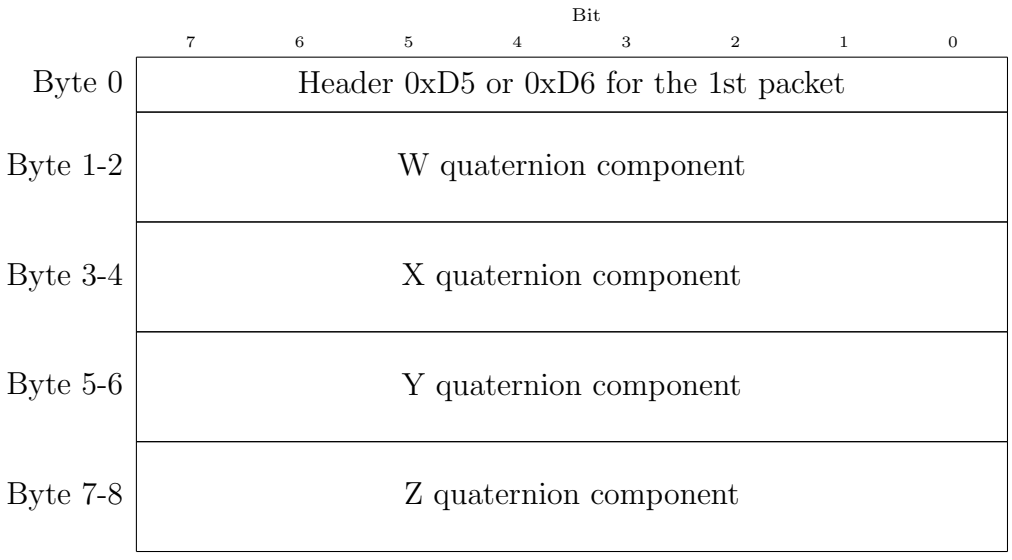

Figure D.5: Absolute Air Mouse frame – 5 Bytes

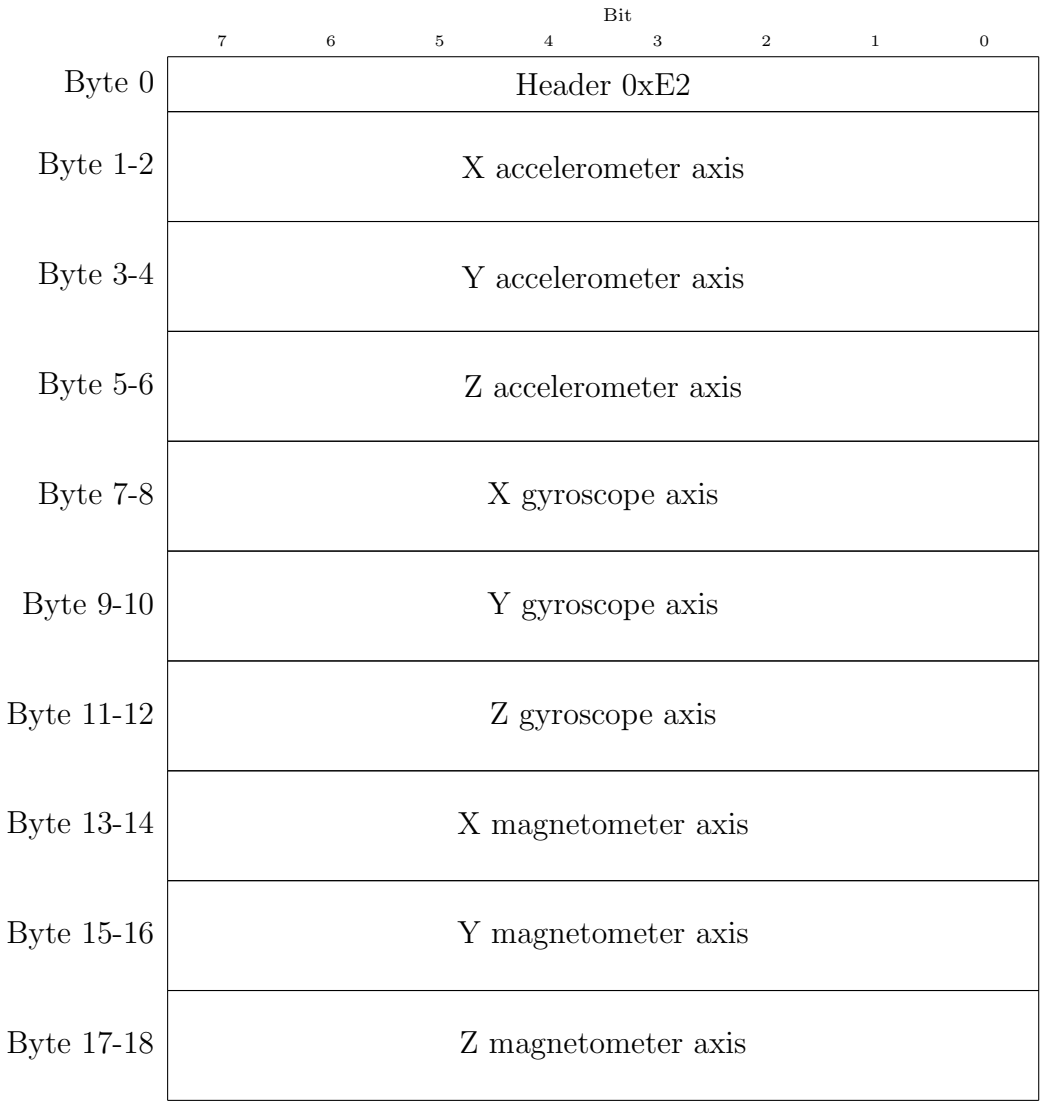

Figure D.6: Sensors frame – 19 Bytes

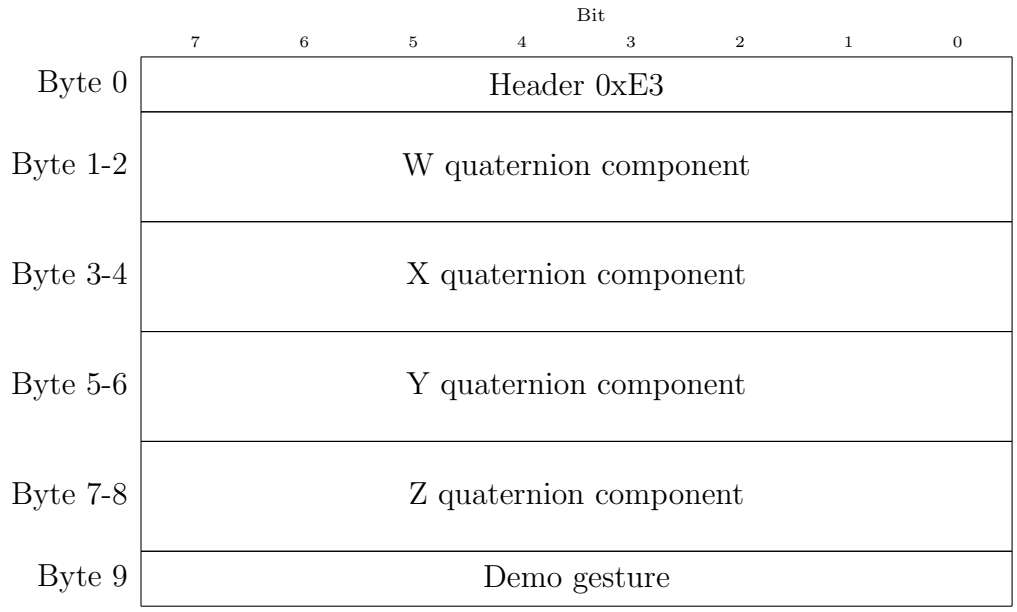

Figure D.7: Demo frame - 10 Bytes