POLITECNICO
MILANO 1863

School of industrial and information engineering
Department of Aerospace Science and Technology - DAER

Master of Science in Space Engineering

Avionics and Software Development for DANCE Platform

Facility design and setup

Riccardo De Gasperin
898039

Under the supervision of Prof. M. Lavagna

A.Y. 2019/2020

Abstract

Nowadays research and development in the context of the space industry is showing more and more interest in studying and designing large constellations of satellites characterized by a high level of cooperation among the vehicles. New distributed space systems designs are in fact evolving towards the concept of formation flying, i.e. clusters of two or more satellites capable of tracking and/or maintaining a relative separation, orientation or position with autonomous coordination and control. Motivations are reasonably found in the possibility of exploiting several advantages, in first place related to the reduced cost of developing multiple identical systems, that translates into a overall increased robustness of the mission with respect to the usage of a single satellite, because of the presence of numerous redundancies. Moreover, assets for the amount of information that can be collected spreading the vehicles on a large area can easily be understood. Innovation in this direction poses new serious and hard challenges, confirmed by the actual low technology readiness level (TRL).

From the necessity of more testing and research in this field the project of developing a Device for Autonomous guidance Navigation & Control (DANCE) has been launched and carried out at the Department of Aerospace Science and Technology (DAER) of Politecnico of Milano; the idea is to build a frictionless facility able to simulate the microgravity condition by the means of air bearings which effect is to suspend the vehicle on a thin film of fluid, creating a torque-free environment and granting the motion on five degrees of freedom. In the final concept, two vehicles (DANCERs) are present, allowing to test new formation flight algorithms and complex relative orientation procedures on a testbed ground facility. The present master thesis work is inserted into the overall project progression, with the objective of reducing the gap to the final operative tests. The focus is put on the avionics: the software for the integration of the set of sensors used with the main processing unit is developed, defining the routines for real-time reading and telemetry data collection; sensor calibration prerequisite is satisfied delineating rigorous procedures. The problem of attitude estimation is then faced considering different suitable algorithms that are implemented, tested and optimized for the specific hardware used. Experiments with the actuators are also performed, perfecting mathematical models and design. In this way major improvements have been introduced in multiple subsystems, laying a solid groundwork for future developing.

Keywords: Formation flying, Attitude estimation, Experimental facility, Ground navigation and control, Autonomous relative motion.

Sommario

Al giorno d'oggi il settore di ricerca e sviluppo nell'ambito dell'industria spaziale sta dimostrando un interesse crescente nello studio e nella progettazione di ampie costellazioni di satelliti caratterizzate da un alto livello di cooperazione tra i veicoli. Il design di innovativi sistemi distribuiti sta infatti evolvendo verso la concezione del volo in formazione, i.e. l'insieme di due o più satelliti capaci di tracciare e/o mantenere una relativa distanza, orientamento o posizione in modo completamente autonomo. Le motivazioni sono ragionevolmente ricondotte alla possibilità di sfruttare diversi vantaggi, in primis correlati al costo ridotto nella produzione di molteplici sistemi identici, che si traduce nell'aumento di robustezza dell'intera missione rispetto all'uso di un singolo satellite grazie alla presenza di numerose ridondanze. Inoltre, l'opportunità che si presenta nella raccolta di una maggiore quantità di dati distribuendo più veicoli su un'area estesa può essere facilmente compresa. Portare innovazione in questa direzione predispone di affrontare nuove complesse ed ardue sfide, come è confermato dal basso livello di prontezza tecnologica (TRL) attuale.

Dalla necessità di maggiore sperimentazione in questo campo è nato nel Dipartimento di Scienze e Tecnologie Aerospaziali (DAER) del Politecnico di Milano il progetto per lo sviluppo di DANCE, una piattaforma sperimentale per condurre nuovi test di navigazione e controllo (GNC). Il concept consiste nella costruzione di una facility a basso attrito capace di simulare la condizione di microgravità attraverso l'utilizzo di cuscinetti ad aria il cui effetto è quello di sospendere il veicolo su un sottile strato di fluido, creando un ambiente privo di forze e momenti torcenti esterni e garantendo il movimento su cinque gradi di libertà. Nella versione finale del progetto sono presenti due veicoli (chiamati DANCERs) che consentono di collaudare in laboratorio nuovi algoritmi per il volo in formazione e complesse procedure di orientazione relativa. Il presente lavoro di tesi magistrale si inserisce all'interno del progresso generale del progetto, con l'obiettivo primario di ridurre il gap per eseguire i test operativi finali. L'interesse è posto sull'avionica: il software per l'integrazione del set di sensori con l'unità di elaborazione principale è sviluppato, definendo le routines per la lettura in tempo reale e la raccolta dei dati di telemetria; il prerequisito della calibrazione è soddisfatto delineando delle rigorose procedure. Il problema della stima d'assetto è poi affrontato considerando diversi algoritmi adatti alla sua soluzione, i quali sono implementati, testati e ottimizzati per lo specifico hardware utilizzato. Attività sperimentali sono inoltre condotte sugli attuatori, perfezionandone il design e i modelli matematici. In questo modo è stato possibile apportare rilevanti miglioramenti in diversi sottosistemi, gettando delle solide basi per esperimenti futuri.

Parole chiave: Volo in formazione, Controllo e stima d'assetto, Piattaforma sperimentale, Moto relativo autonomo.

Contents

List of Figures

List of Tables

Variables

Symbol	Definition	Unit
$\boldsymbol{\alpha} = [\theta,\ \phi,\ \psi]$	Euler angles vector with rotations around $x,\ y,\ z$ axes	$[°]$
$\beta,\ \zeta$	Tuning gains for Madgwick filter	[/]
ε_{cal}	Error on calibrated values	[~]
μ	Friction coefficient	[/]
ν	Poisson's ratio	[/]
σ_n	Noise standard deviation	[~]
σ_{cal}	Equivalent standard deviation for calibrated values	[~]
A	Attitude matrix	[/]
$A_R,\ A_n,\ A_s$	Accelerometer calibration auxiliary matrices	[/]
$\boldsymbol{a}$	Acceleration field	$[m/s^2]$
$\boldsymbol{b}$	Angular velocity bias	$[°/s]$
E	Young's modulus	$[Pa]$
$\bar{E}_{\%}$	Mean percentage error	[%]
$f_{sampling}$	Sensors sampling frequency	$[Hz]$
$\boldsymbol{g}$	Gravity acceleration	$[m/s^2]$
I	Impulse	$[N \cdot s]$
$K_p,\ K_{int}$	Proportional, integral tuning gains	[/]
$K_a,\ K_m$	Accelerometer, magnetometer tuning gains	[/]
$\boldsymbol{m}$	Magnetic field	$[T]$
N_{meas}	Number of measurements	[/]
n_{nD}	Noise density	$[\sim/\sqrt{Hz}]$
$\boldsymbol{O}$	Offset vector for calibration procedures	[~]
$\boldsymbol{q}$	Quaternions	[/]
P	Pressure	$[Pa]$
$pWidth$	Pulse-width	$[s]$
$pwmPrd$	Pulse-width modulation period	$[s]$
$Q_k,\ R_K$	Noise covariance matrices	$[\sim^2]$
$R_G,\ R_A,\ R_M$	Sensors measurements range	[~]
RMS_{error}	Root Mean Square error	[/]
Sf	Raw sensors measurements scaling factor	[LSB/~]
T	Thrust	$[N]$
t_{sample}	Sample time	$[s]$
t_{ref}	Time interval for initial reference computation	$[s]$
tol	Tolerance	[/]
V	Supply voltage	$[V]$
$\boldsymbol{\omega}$	Angular rate	$[°/s]$
$W,\ X,\ Y$	Accelerometer calibration auxiliary matrices	[/]

A vectorial quantity is indicated with bold notation $\boldsymbol{x}$, a scalar one with x [~].

Acronyms & Abbreviations

Symbol	**Definition**
ABS	Acrylonitrile Butadiene Styrene
Acc.	Accelerometer
AHRS	Attitude and Heading Reference Systems
AP	Attitude Platform of DANCER vehicle
DAER	Department of Aerospace Science and Technology of PoliMi
DANCE	Device for Autonomous guidance Navigation and Control
DC	Continuous current
DOF	Degree of Freedom
EEPROM	Electrically Erasable Programmable Read Only Memory
FOC	Fast Offset Compensation
GNC	Ground Navigation and Control
Gyro.	Gyroscope
IDE	Integrated Development Environment
IMU	Inertial Measurement Unit
I^2C	Inter-Integrated Circuit
Mag.	Magnetometer
MARG	Magnetic, Angular Rate and Gravity
MEKF	Multiplicative Extended Kalman Filter
MCU	Microcontroller Unit
MEMS	Micro Electro-Mechanical System
NVM	Non-Volatile Memory
OBC	On Board Computer
ODR	Output Data Rate
OS	Operative System
PET	Polyethylene Terephthalate
PWM	Pulse-Width Modulation
PLA	Polylactic Acid
PMU	Power Management Unit
RF	Radio Frequency
RMS	Root Mean Square
RW	Reaction Wheel
SBC	Single Board Computer
SCL	Serial Clock pin
SDA	Serial Data pin
SPI	Serial Peripheral Interface
ODR	Output Data Rate
TI	Texas Instruments
TP	Translational Platform of DANCER vehicle
UAV	Unmanned Aerial Vehicle

1 Introduction

1.1 Project Description

The project of designing a Device for Autonomous guidance Navigation and Control, named DANCE, was initiated in 2016 at the laboratory of the Department of Science and Technologies (DAER) of Politecnico of Milano. The main objective is to contribute to the need of more experimental testing in the field of formation flying, i.e. new space mission concepts in which multiple satellites are cooperating. The difference with respect to traditional distributed systems is that the vehicles must be capable of tracking and/or maintaining a relative separation, orientation or position in complete autonomy. The technology readiness level (TRL) in this field is still low, mainly because of the severe complexity of the problem, especially from the attitude estimation and control point of view; from here arises the purpose of developing a testbed facility for ground navigation and control (GNC) experimenting, in order to contribute to the overall scientific progress and innovation. A more detailed description of the project and its state at the beginning of this thesis work are hereafter exposed, explaining the main final objectives to be reached together with the present work motivations. A brief overview on how the present activity is organized is also given.

1.2 DANCE Facility Overview

In its final version, the DANCE facility will be constituted by two identical vehicles, called DANCERs, free to move inside a testbed arena appositely built; formation flight algorithms for relative tracking can in this way be tested. Two main features have to be granted in order to simulate the motion of a satellite in the space environment, reached in different ways:

- Complete absence of friction: it is possible to operate in a frictionless condition by suspending the vehicle on a thin film of air which physically separates its components from the ground, in order to avoid the sliding contact condition. This is achieved by the means of several air bearings that expel a pressurized fluid, exploiting the action-reaction principle.
- Microgravity: the lack of gravity that characterizes orbital motion translates in controlling a vehicle in a torque-free environment; the way by which this condition can be accomplished is by ensuring that the DANCER center of mass coincides with the center of rotation, as further explained hereafter.

At the state of the art the first DANCER structure and the configuration of its component have already been entirely designed with some iterations, ensuring manufacturing feasibility by realizing a complete CAD technical drawing; assembling of the main components has already been done as well, inspiring the overall arrangement to other existing facilities [1, 2, 3]. The purpose of this section is to give only a general overview on the actual state of development of the vehicle, to avoid repeatability with respect to the exhaustive explanation of all the different subsystems done in previous works [4, 5]. No substantial modifications have been in fact made to the overall design, with the exception of the reaction wheels support and flywheel (described later in Sec. 5.2).

Each DANCER vehicle (Fig. 1.1), in its final version, is characterized by a total mass of about 44 kg and is constituted by two parts: an attitude platform (AP) and a translational platform (TP). The AP is kept on the top of the TP by the means of a supporting stem, on the summit of which a hemispherical air bearing is placed with the purpose of eliminating the dynamic friction between the two parts; in this way the AP is able to freely rotate around the z vertical body axis, while a maximum rotation of $\pm 35°$ to $45°$ is possible around the x and y body axes. The translational movements of the whole vehicle on the testbed plane are instead granted by the TP, equipped with three linear flat air bearings that prevent it from touching the ground while air is being expelled. The frictionless motion is thus granted on 5 degrees of freedom (DOFs), only one less with respect to a satellite moving on the space environment, since for a DANCER the vertical translations are not possible. As mentioned, to simulate the microgravity condition is instead fundamental that the AP center of mass coincides with its center of rotation on the spherical air bearing and for this reason the design has been conceived to be as symmetrical as possible; currently a static balancing system is present and a dynamic balancing system is still in a conceptual phase.

The attitude control is performed by the means of a set of 3 to 4 reaction wheels, made by an inertial mass (flywheel) assembled with an electric motor (Fig. 5.2), and 12 air nozzles organized in 4 multi-directional components (Fig. 5.15) that work as thrusters. A pneumatic system constituted by rigid and flexible tubes, a pressure regulator, electrovalves and highly pressurized air vessels is needed to feed both the thrusters and the air bearings. Finally, a feeding system composed by battery packs, DC-DC converters and shunt regulator is needed to supply the correct voltage for the whole avionic system and any other electric component (like the motors and the electrovalves). More details are also given in Chapter 5.

1.3 Objectives & Goals

The final goal of the DANCE project is to have two identical fully operative vehicles, able to move autonomously inside the dedicated testbed arena. The avionics together with a suitable set of sensors consent each DANCER to orient itself inside the environment and compute the relative position with reference to the other vehicle. Complex formation flight algorithms for relative position tracking, correcting and maintaining can be downloaded to the main on board computer in order to test their functionality; an external optical system of cameras working with reference visual markers is needed to validate the correctness of attitude estimation and control results, as depicted in Fig. (1.2). Being the systems identical it is obviously possible to develop only the first one, which is then duplicated in manufacturing. The final objective is reached step by step, starting from a general concept of the whole structure, modelling each component according to the imposed requirements and assembling the parts, up to the first operative tests. It is in this context that the present thesis work is inserted: beginning from the state of the art of previous works ([4, 5]) the general goal is to shorten the gap to the first experimental tests with

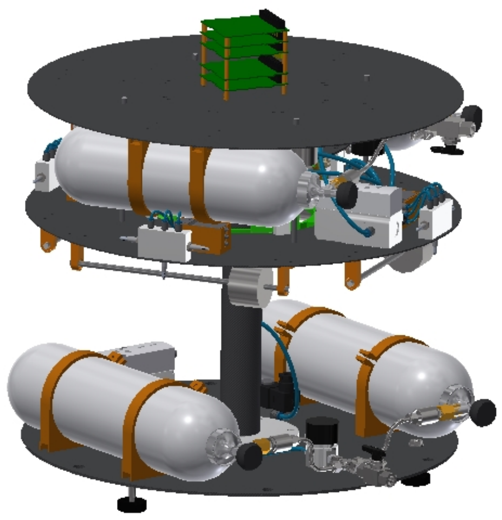


Figure 1.1: Technical concept representation of a DANCER.

the AP. Focusing mainly on the avionics, the present work poses its specific objectives in developing suitable calibration procedures for the sensors adopted, proceeding with a hardware-software integration for attitude estimation and increasing the TRL for the actuators, favouring their realization and connection to finally close the loop.

1.4 Work Organization

This master thesis work is characterized both by numerical simulations and experimental activities, being DANCE a wide project with a lot of aspects to focus on. Particular attention is given to the avionics: at first, a detailed description of the components of the electronic system is reported, explaining the type of controller adopted and the sensors that will be used by the vehicle to interact with the environment. Integration of the hardware parts together with the developed software for correct activation and reading procedures are exposed in details, indicating the modality by which it is possible to retrieve telemetry data on a personal computer for real-time analysis and post-processing computations. In Chapter 3 the approaches by which the gyroscope, accelerometer and magnetometer used can be calibrated are instead presented, as result from a trade-off evaluation between the simplicity of application and the expected precision. The problem of attitude reconstruction is then faced in Chapter 4, indicating three among the most common estimation algorithms used for unmanned aerial vehicles, accompanied by the results obtained after their implementation, optimization and the performing of different kind of simulations. Finally, Chapter 5 deals with the attitude control aspects, in particular with the activities related to the actuators: new iterations on the previously realized design for the reaction wheels and their structural support are done, and development of a preciser mathematical model built for the thrusters after some experimental tests, repeated with more different conditions with reference to the past works, is exposed. At the end of each chapter, a brief recap is present, indicating technical advice and suggestions

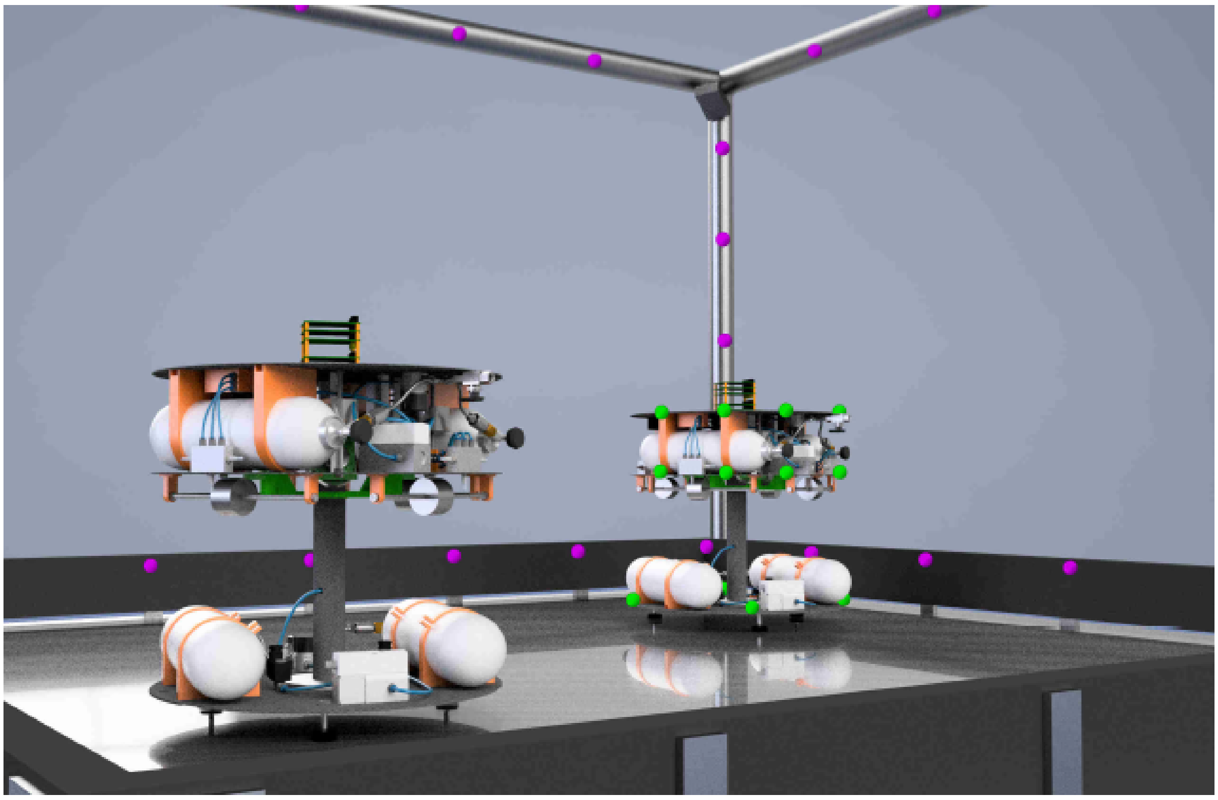


Figure 1.2: Concept of the complete facility with two DANCERS and a set visual markers for the optical system of cameras. *Credits: [5]*

that in the author's opinion could improve the results obtained, favouring in this way the continuity of the project. A final summary of the future goals to be achieved is reported in Chapter 6, hoping that will serve as a great help for the orientation of future efforts.

2 Avionics Description and Integration

2.1 Chapter Overview

The electronic subsystem of DANCE is a mandatory part to be developed, otherwise without it the whole vehicle would be useless: it is required in order to control in real-time the motion of the system, interact with the surrounding environment through suitable sensors and record the telemetry data to be analyzed after the experimental tests, making possible to study the correct working of formation flight algorithms when more than one DANCER vehicle will be present. In this chapter the electronic components adopted are exposed, explaining their general characteristics. The strategy used to program and read the sensors is then mentioned, with particular attention to the procedure by which they can be read in real-time by the main processing unit and the modality by which collected data can be transmitted to a personal computer; type of communication protocols used are also briefly described.

2.2 Necessity & Motivations

As explained in detail at the beginning of development and conceptual definition of the DANCE facility project [4] and here described in Chapter 1, in final operative conditions each DANCER vehicle must be able to move autonomously: the TP is required to perform movements on the plane and rotations around the z vertical axis (yaw) of the body-fixed frame while the attitude platform must also rotate around x and y horizontal axes (roll and pitch), controlling the dynamics and counteracting external disturbances (such as gravity) without relying on external additional support systems. These requirements turn into the necessity of equipping the vehicle with a suitable set of sensors that gives it the capability of interacting with the environment and to determine its position and orientation with respect to an absolute inertial fixed reference frame; once this information is available, the system can control the needed actuators response to follow a trajectory imposed by the user. The strategies that can be adopted to achieve this goal are essentially two, one is based on the employment of devices that rely on external references, like star trackers or sun sensors that are typical of real space systems, but sensors of this kind can be hardly used for ground facilities and are expensive. The other solution consists in operating with inertial measurement units (IMU) that provide measurements of the angular velocity of the body $\boldsymbol{\omega}$ by the means of gyroscopes and measurements of the acceleration field $\boldsymbol{a}$ using accelerometers. Since IMUs are prone to noise and error accumulation also a magnetometer can be employed to correct measurements; systems of this kind are often referred to as magnetic, angular rate and gravity systems (MARG) or Attitude and Heading Reference Systems (AHRS) [6, 7]. Moreover, an IMU alone can only estimate the orientation of the body relative to the gravity direction but is not reliable in estimating the yaw angle, so additional measurements of another field (in this case the magnetic field, $\boldsymbol{m}$) is required for a complete attitude reconstruction, needed in the case of DANCE. In fact this last solution is adopted for the present work, using a set of tri-axis sensors composed by a gyroscope, an accelerometer and a magnetometer, giving the possibility to update each time instant measurements of $\boldsymbol{\omega}$, $\boldsymbol{a}$, $\boldsymbol{m}$ in the body-fixed frame which are then used by the DANCER vehicle to autonomously compute its position and orientation during real-time experimenting. As explained in [4] and remarked in [5] an external optical system of cameras that tracks the vehicle movements is still required for absolute referencing and for the validation of the formation flight algorithms tested.

2.3 Microcontroller

The central processing unit for the system electronics, named on-board computer (OBC), is conveniently selected to be a single-board computer (SBC) meaning that a complete operative system is run on a single circuit board; evaluating the alternatives present on the market, the SBC selected in previous evaluations [4] for DANCE application is the BagleBone Black Wireless module [8]. The OBC is used to run the heaviest parts of the software that need to be executed, which in the perspective of attitude and formation flight algorithms testing can regard complex control laws or optical navigation, giving at any case the maximum flexibility for algorithms development. Nevertheless this kind of SBC is not suitable for real-time control of the actuators, mainly because of the fact that the numbers of output signal pins in not sufficient; for this motivation, the use also of microcontrollers is a mandatory requirement. Microcontrollers do not run a complete operative system but can, of course, be programmed to perform the necessary operations; usually, simpler algorithms are deployed to this kind of devices, making them appropriate to generate the output signals for the actuators control. The microcontroller board employed in the current project is hereafter briefly described and used for sensors reading as well as attitude estimation algorithms running.

2.3.1 Texas Instruments LaunchPad

The microcontroller that has to be employed for the real-time control of the actuator was already available at the beginning of the present thesis work. It has been selected considering as the main driver of choice the availability of the highest number of output peripherals with reference to similar devices of the same class present on the market (for additional considerations about the trade-off choice see [4]). The hardware is developed by Texas Instruments (TI), in particular the model is the F28379D LaunchPad Development Kit from C2000 Real-Time Control MCUs family; electronic specifications and wiring are not here reported since have been exhaustively exposed in previous literature of DANCE project and can be found directly on the technical documents [9, 10, 11]. The board can be accessed and programmed directly from a personal computer by using the serial connection of a USB cable and the dedicated interfacing software Code Composer Studio [12], but another great advantage of using this TI board is that a complete support package in present also in the MATLAB & Simulink suite, asset that makes possible to program it using a Simulink model with the dedicated Embedded Coder, which generates the necessary code directly from the model. This approach is in fact used in the present work to develop algorithms for real-time serial read (Sec. 2.6.1) and attitude reconstruction (Chapter 4); details for installation procedures, software requirements and troubleshooting can be found in Appendix (D.1). In future development, a radio frequency (RF) module will be used to transmit data between the board and the computer allowing in this way to download recorded telemetry data without the need of disassembling the microcontroller from the DANCER vehicle, as suggested in [5]. If the USB cable is not plugged in, the LaunchPad needs to be fed directly with a 3.3 V supply voltage that can be retrieved from an external battery.

The TI LaunchPad is used to be directly interfaced with the MARG sensors and to recover their measurements used for attitude reconstruction; algorithms have been tested both in post-processing using recorded measurements and in real-time by deploying the software to the board, with the results reported in Chapter 4. A representation of the hardware is reported in Fig. (2.1). For operative purposes it is worth to mention the presence of the lateral yellow button (visible on the right side) which allows performing a soft reset and it is used to initialize the sensors reading procedure getting the absolute reference for attitude estimation (Sec. 4.2 for details).

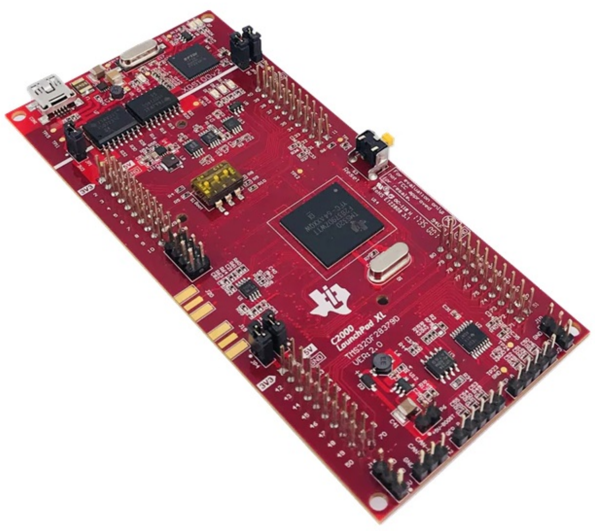


Figure 2.1: Texas Instruments C2000 F28379D LaunchPad microcontroller.

2.4 MEMS Sensors

The sensor class selection must take into consideration a trade-off decision between the wanted accuracy, the expected performance, weight, power consumption and the cost. Evaluating all the aspects a decision has been made [4] to use MARG micro electro-mechanical systems (MEMS) which constitutes a good compromise, especially for the reduced cost and the performance which is usually sufficient for a lot of applications, in particular in the case of unmanned aerial vehicles (UAV) [13] like DANCER could be considered to be. The overall performance in attitude reconstruction is nevertheless determined by multiple factors and not only by the class of sensors used: calibration and attitude estimation algorithms play a fundamental role as well. It will be possible to definitely determine if the MEMS sensor class is suitable for the required precision after experimental tests are performed with the full operative system. In this thesis work calibration procedures that can be easily performed without the need for additional equipment are proposed in detail; some attitude estimation algorithms are tested, so the bases for future experimenting are completely set.

In [5] are indicated different alternatives for the MARG MEMS sensors to be used that are equivalent in terms of performance, in fact the main drivers for the choice are suggested to be only the simplicity of use, cost and availability; technical data comparison between the proposals is not here reported to avoid repeatability. The set adopted for the present work and future DANCE development is described in the next section.

2.4.1 Texas Instruments BOOSTXL Sensors

The Texas Instruments BOOSTXL Sensors BoosterPack plug-in module (Fig. 2.2) has been evaluated as the best alternative between the proposed ones and used in the context of the present thesis work. The main reason of this choice is the possibility of using a compact MARG system already compatible with the LaunchPad microcontroller since both are developed by the same industry, avoiding in this way time-consuming processes for hardware integration and compact fixing. Plug-in procedure is in fact easy and fast (Fig. 2.3). The software developed is optimized for this MARG set, but can be modified and adapted to different devices with little effort, since the main part of the algorithms is used to program the TI LaunchPad to read and process incoming measurements; changing the sensor may result in the need of only modify the addresses of the specific registers to be programmed. Also the calibration procedures are performed with this particular set but can be generalized to any MEMS device.

The TI BOOSTXL expansion pack is equipped with tri-axis sensors, in particular the Bosch Sensortec BMI160 IMU constituted by a gyroscope and an accelerometer (Fig. 2.4a) and the Bosch Sensortec BMM150 magnetometer (Fig. 2.4b), as required for a complete attitude determination. Technical data about complete set usage can be found on the datasheet [14], while specific sensors data included the register map are present on the dedicated datasheets [15, 16]. Register addresses and specific values to be written for the activation procedure are described precisely in Chapter 3.

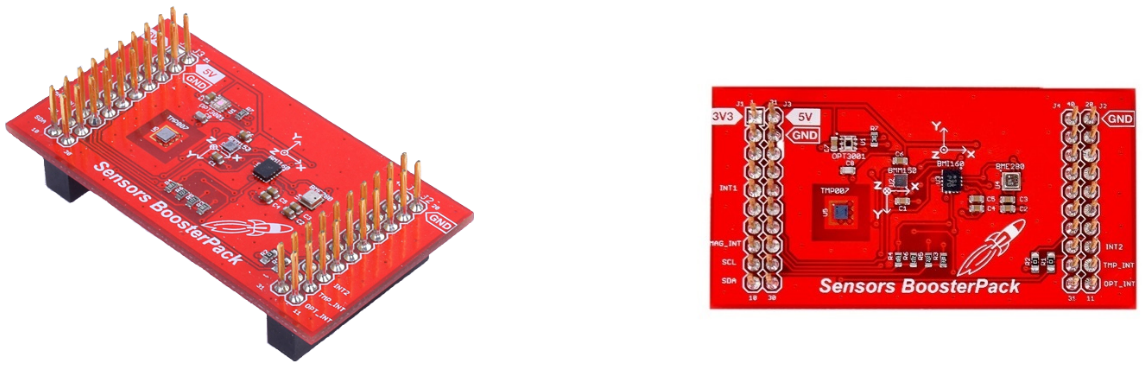


Figure 2.2: BOOSTXL Sensors BoosterPack plug-in module.

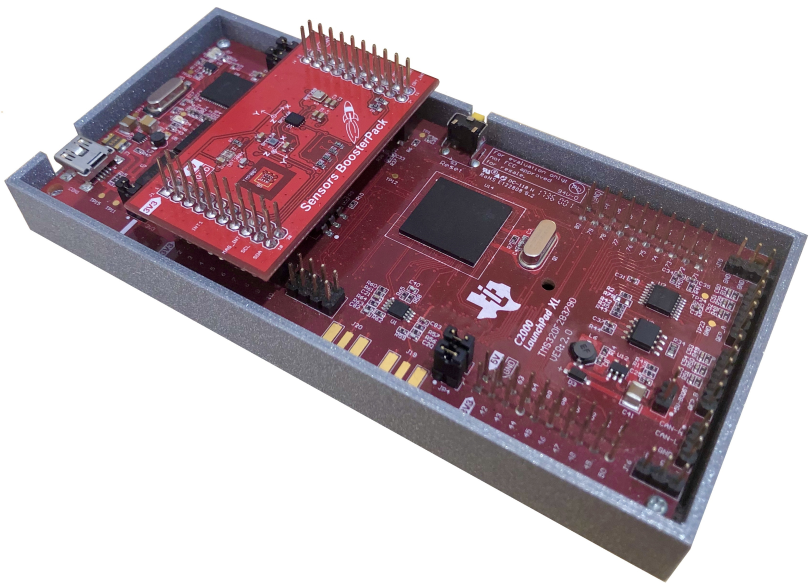


Figure 2.3: BOOSTXL Sensors BoosterPack assembled with the TI LaunchPad inserted in a 3D printed case.

2.5 Inter-Integrated Circuit Communication

By default the BOOSTXL sensors use the inter-integrated circuit (I^2C) serial computer bus as the main channel for the data transfer between the sensors and the LaunchPad controller; other peripherals can be used (like serial peripheral interface, SPI) but are not convenient for this purpose and would need additional configuration procedures. I^2C has in fact revealed to be efficient, relatively simple to be used and satisfying the actual needs for sensors and microcontroller intercommunication, permitting the real-time data streaming with a user imposed output data rate (ODR) from the slave devices (sensors) to the master device (LaunchPad) which controls them through signals and commands described hereafter. The ODR coincides with the sample time of the Simulink model deployed to the LaunchPad, currently set to 0.01 (100 Hz); a bigger data rate is practically useless since the maximum ODR of the sensors is set to 25 Hz (see Chapter 3) to have the highest possible precision in measurements, thus it is already widely sufficient to transmit all collected data; essential parameters can be found in Tab. (2.3) at the end of the present chapter.

The essential signal lines to be used with this communication protocol are reported in Tab. (2.1) with a brief explanation; using the BOOSTXL plug-in module permits to avoid prototype wiring and possible connection troubles that may be encountered, being a device ready-to-use with the TI LaunchPad[1]. Additional pins may be connected for

[1]It is worth to mention that if other external MEMS sensors are used instead of BOOSTXL, besides the default pin wiring for the intercommunication, pull-up resistors are also needed; for detail see the wiring electric scheme at the end of [14].

(a) BMI160 **(b)** BMM150

Figure 2.4: Bosch Sensortec BMI160 IMU with gyroscope and accelerometer and BMM150 magnetometer equipped on the Texas Instruments BOOSTXL Sensors BoosterPack plug-in module.

other control purposes but are not here reported since not strictly necessary for closing the communication.

Signal Line	Name	Description
SCL	Serial Clock	synchronization line
SDA	Serial Data	data streaming line
Vcc	Voltage	supply voltage line
GND	Ground	ground reference of operating voltage

Table 2.1: I2C communication fundamental signal lines.

Each connected device is characterized by a slave I^{2}C address that must be forcibly specified during the configuration procedures; addresses for the BMI160 IMU and BMM150 magnetometer are reported in Tab. (2.2). All the main operations performed to configure, activate and read the BOOSTXL module are done communicating directly with the BMI160 IMU I^{2}C slave address, while the BMM150 requires the specification of a different address since it is connected to the IMU as a secondary slave device. Detailed procedures are further explained in Chapter 3.

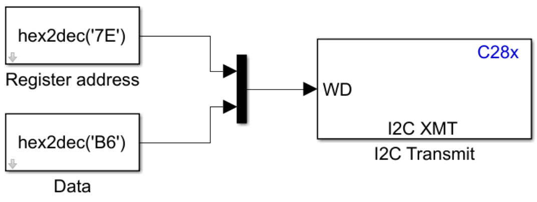
Slave device	Name	I^{2}C slave address
BMI160	IMU	0x69
BMM150	Magnetometer	0x26

Table 2.2: Default I^{2}C slave addresses for the BOOSTXL sensors.

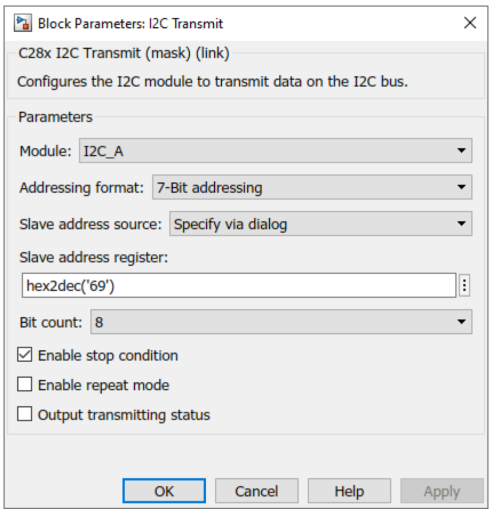
2.5.1 Read & Write Operations

From a software development point of view, the Simulink model to be deployed to the TI LaunchPad makes possible the inter-communication between the microcontroller and the sensors through basics write and read operations, implemented with the dedicated transmit (I^{2}C XMT) and receive (I^{2}C RCV) blocks from the Embedded Coder Support

Package for Texas Instruments C2000 Processors. Fig. (2.5) shows a typical write operation that is performed; in particular, one must enter the address of the specific register to be written (note that this address is conceptually different from the slave I^{2}C address explained above) and the value to write into it. Register maps are found on the datasheets, but the precise sequences of operations to be done for the activation procedures are here reported in Sec. (3.3.1), (3.4.1), (3.5.1) respectively for gyroscope, accelerometer and magnetometer. The "Enable stop condition" option is necessary to send the stop bit from the microcontroller to the slave device, otherwise the current operation is not terminated and it is not possible to perform a new one.



(a) I2C Transmit block


Block Parameters: I2C Transmit

C28x I2C Transmit (mask) (link)

Configures the I2C module to transmit data on the I2C bus.

Parameters

Module: I2C_A

Addressing format: 7-Bit addressing

Slave address source: Specify via dialog

Slave address register: hex2dec('69')

Bit count: 8

☑ Enable stop condition

☐ Enable repeat mode

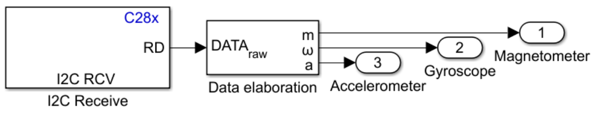
☐ Output transmitting status

(b) I2C Transmit mask parameters

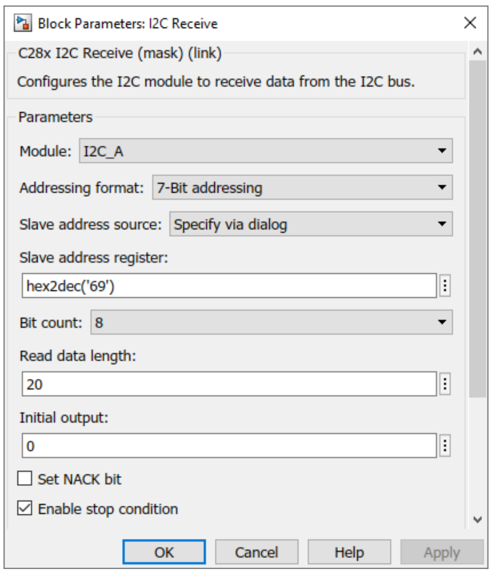
Figure 2.5: Simulink I^{2}C transmit block for TI F28379D LaunchPad with address of register to be written and data to be stored (a) and relative mask with I^{2}C address specification (b) (example data here reported are used specifically to perform an initial soft reset).

The I^{2}C transmit block is used to set up the working mode of the sensors and other important parameters, like the desired output data rate (ODR) and the measurement range, but it is also needed to initiate the reading operation: each time step the transmit block sends the data containing the register to be read to collect the measurements data, without sending the stop condition bit; the I^{2}C receive block shown on Fig. (2.6) can then be used to retrieve the wanted data, sending the stop condition at the end of each read operation. Burst data read is performed, getting in this way the output values of all the three slave sensors with only one read operation; the value of each measurement along one sensor axis is contained in two different registers, one with the 8 most significant bits (MSB) and one with the other 8 least significant bits (LSB) since the TI LaunchPad works in single precision and represents each number with 16 bits. The read length setting the number of registers to be read is thus 18, but it is set to 20 since two more registers giving the sensors hall resistance are present in the data read sequence but relative values are not used for present purposes. In order to read the sensor continuously, a transmit-receive loop is initiated[2].

[2]An issue has been encountered with the transmit-read loop for continuous measurements retrieving: once started, even if the loop is performed for a limited amount of time inside a Simulink Stateflow



(a) I2C Receive block and data elaboration


Block Parameters: I2C Receive

C28x I2C Receive (mask) (link)

Configures the I2C module to receive data from the I2C bus.

Parameters

Module: I2C_A

Addressing format: 7-Bit addressing

Slave address source: Specify via dialog

Slave address register: hex2dec('69')

Bit count: 8

Read data length: 20

Initial output: 0

☐ Set NACK bit

☑ Enable stop condition

(b) I2C Receive mask parameters

Figure 2.6: Simulink I^{2}C receive block for TI F28379D LaunchPad retrieving sensors data and elaborating them (a) and relative mask with I^{2}C address specification and burst data length settings(b).

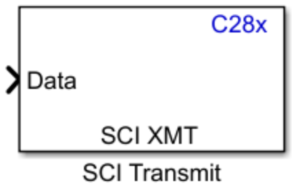
2.6 Serial Communication Interface

Communication between the TI LaunchPad and the personal computer is instead done through the universal asynchronous receiver-transmitter (UART) serial communication interface (SCI), via USB cable. The protocol is, in this case, asynchronous so data can be received correctly from the PC only setting the same baud rate (i.e. the number of bits per second transmitted) for both devices. Tab. (2.3) reports the value set for the present work, as suggested by the user guide [9]. Data transmission is configured through the specific SCI transmit block (Fig. 2.7), setting two reference characters as headings for correct message individuation. Fig. (2.8) shows instead settable parameters and output quantities of the Simulink code to be deployed on the target LaunchPad.

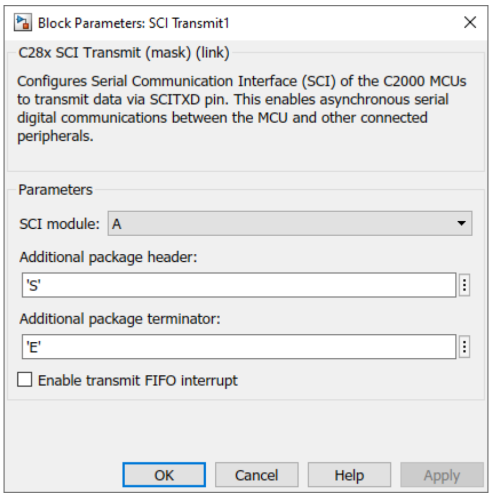
2.6.1 Real-Time Serial Read Algorithm

Once the TI LaunchPad has been programmed using the Simulink blocks described above, the output data stream can be accessed directly from MATLAB by opening the right serial port and setting the correct baud rate. The algorithm that has been developed is able to recognize the package headers "S" (Start) and "E" (End) previously set, read the message in between, convert the received data concatenating the bits and finally obtaining the transmitted values in decimal form. Each value is represented in Simulink with the IEEE 754 standard for binary floating-point numbers [17]; with this convention, each value transmitted has a size of 4 bytes (32 bits) because as mentioned before the

function, it is not possible to perform other different writing operations, for example modify the ODR or perform a soft reset. This issue has not been a limitation for the present work but is highlighted for future development.



(a) SCI Transmit block


Block Parameters: SCI Transmit1

C28x SCI Transmit (mask) (link)

Configures Serial Communication Interface (SCI) of the C2000 MCUs to transmit data via SCITXD pin. This enables asynchronous serial digital communications between the MCU and other connected peripherals.

Parameters

SCI module: A

Additional package header: 'S'

Additional package terminator: 'E'

☐ Enable transmit FIFO interrupt

(b) SCI Transmit block parameters

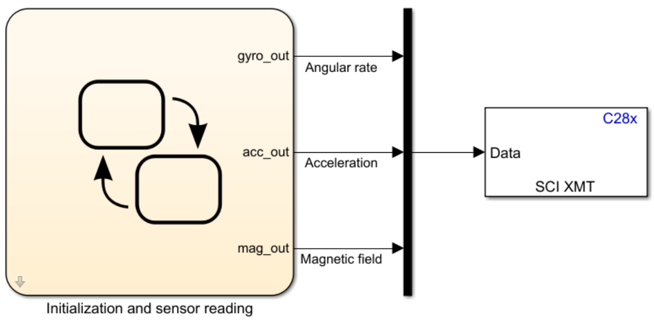
Figure 2.7: Simulink SCI transmit block for TI F28379D LaunchPad for serial data sending (a) and relative mask with headings settings (b).

TI LaunchPad works in single precision. Conversion from binary to decimal is computed consequently. The total number of bytes to be received is a required user input to be set, otherwise correct message cannot be received. With the currently set baud rate a maximum number of 14400 $bytes/s$ can be sent; using a 100 Hz ODR it means that up to 360 variables can be read in real-time, widely sufficient for the present application.

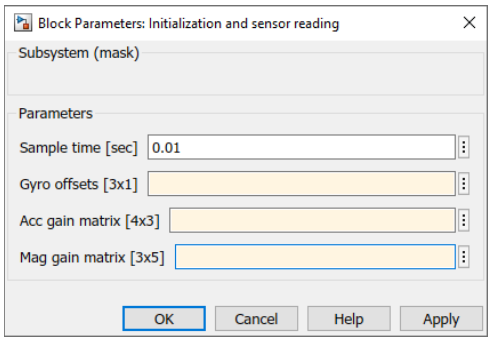
MATLAB algorithm for real-time data read permits to retrieve gyroscope, accelerometer and magnetometer measurements and also to record them for post-processing, allowing in this way to perform all the calibration procedures (Chapter 3), to test attitude reconstruction routines (Chapter 4) and to execute post-processing computations. It will also constitute a fundamental tool to collect and download telemetry data once the DANCER vehicle will be set into the operative mode. In Fig. (2.9) is reported an example of usage of the algorithm for preliminary evaluation of the general behaviour of the attitude estimation filters with a live-updated intuitive plotting; inertial and body-fixed reference frames are visible, and the apparel used to have a 90° reference can be seen.

Parameter	**Value**	**Description**
UART SCI Baud Rate	115207 $baud$	Max bit/s set for serial communication
UART SCI Sample time	100 Hz	ODR set for serial communication
Gyro_ODR	25 Hz	Gyroscope measurements ODR
Acc_ODR	25 Hz	Accelerometer measurements ODR
Mag_ODR	12.5 Hz	Magnetometer measurements ODR

Table 2.3: Main technical parameters set for serial communication and sensors measurements update.



(a)


Block Parameters: Initialization and sensor reading

Subsystem (mask)

Parameters

Sample time [sec] 0.01

Gyro offsets [3x1]

Acc gain matrix [4x3]

Mag gain matrix [3x5]

(b)

Figure 2.8: Simulink Stateflow block containing the code to be deployed on the TI LaunchPad for real-time sensor reading (a) and relative mask with settable parameters (b): sample time is set by default to $t_{sample} = 0.01\ s$; the other parameters must be inserted after having performed the calibration procedure.

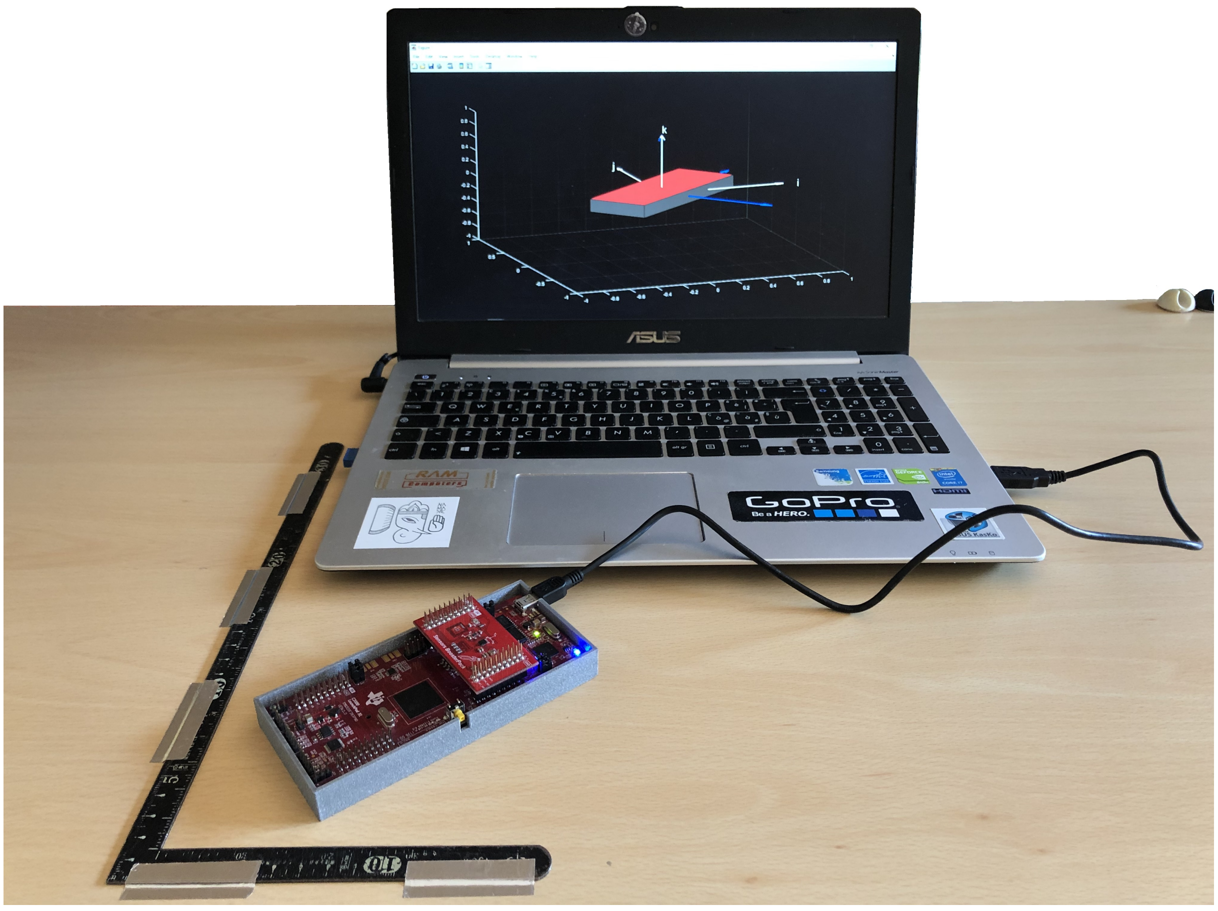


Figure 2.9: Experimental activity showing the MATLAB real-time representation of the estimated attitude using TI LaunchPad together with BOOSTXL module. In the plot are visible the inertial frame (in white) and the body frame (in blue); the tool for 90° referencing can also be noticed.

3 Sensors Calibration

3.1 Chapter Overview

As previously discussed, in order to be able to reconstruct the orientation of the attitude platform of DANCE facility with respect to an inertial reference frame, some vectorial measurements must be available; they are retrieved through the IMU and magnetometer by which each DANCER vehicle is equipped. Nevertheless, low-cost MEMS output cannot be trusted as it comes but requires some adjustment. In the following sections this problem will be firstly addressed in depth; then a solution for sensors calibration will be separately exposed for every single type of the MARG system component: gyroscope, accelerometer and magnetometer.

3.2 Purposes and Needs

Calibration is defined as the process of finding the relationship between a sensor output measure and the parameters to be measured [18], which is performed in order to adjust wrong measurements and bring them as close as possible to the real quantity. Although the vast majority of MEMS sensors which are present on the marked are factory calibrated, the different environmental conditions to which they are exposed during utilization by customers and process variation issues lead to the necessity of applying specific adjustment procedures, in order to correct the sensor reading values and obtain the expected performance. Some MEMS sensors, including the IMU and the magnetometer present on the BOOSTXL expansion pack chosen (Sec. 2.4.1), provide the possibility to execute a fast offset compensation (FOC) which consists in an automated procedure for rapid correction of sensors imprecisions. Anyway, FOC is not a suitable choice for this project for the following main reasons:

- FOC does not imply storing any value on the non-volatile memory of the microcontroller, thus the procedure would have to be repeated every time the system is reset. It means that sensors should be on a controlled position and orientation every time the electronics is switched on, bringing to consequent slow and intricate operative procedures;
- The exact algorithm used to compensate for errors and biases is not specified on the datasheet, so it is not possible to judge how rigorous this procedure is;
- FOC is not available on all sensor power modes;
- High precision is not granted.

Consequently, it is necessary to define one deterministic and rigorous procedure for each one of the three different MEMS sensors that are employed in this project, with the aim of achieving the maximum possible measurement precision, which is by the way limited by the physical characteristics of the sensors themselves.

In literature several different calibration procedures can be found, which are substantially divided into three categories [19]:

- Comparison procedure: the MEMS to be calibrated is used to measure a physical quantity that could be in this case the angular rate, acceleration or magnetic field. The same quantity is measured by a preciser instrument which output can be trusted. By comparing the values read by the two devices, the less precise one can be adequately corrected. With this procedure, the highest accuracy can be reached, but it requires the availability of expensive instruments and/or facilities able to keep the sensor in a strictly controlled environment.
- Substitution procedure: the quantity measured by the sensor to be calibrated is generated by another instrument or facility, in such a way that the output to be measured is known.
- Direct procedure: the device is used to directly measured one quantity and calibrated according to the expected output. This is usually done when the device is still, so that expected measure is easily determined.

Each different procedure has advantages and weak points, briefly summarized in Tab. (3.1); in the present work, as it will be explained in detail, a direct procedure has been adopted for all the sensors used. The main decision driver is, in fact, the lack of availability of expensive equipment, supported by the possibility of reaching acceptable accuracy even with the direct approach if procedure and software are rigorously developed; the present work tries to satisfy this last requirement.

Procedure	**Assets**	**Drawbacks**
Comparison	• High accuracy • Fast & easy to perform	• Expensive equipment needed
Substitution	• Dynamic referencing • Cross axis sensitivity check • Global error control	• Expensive equipment needed • Dedicated facility development
Direct	• Fast & easy to perform • Little or no equipment needed • Acceptable accuracy reachable	• MEMS accuracy not maximized • Prone to experimental errors

Table 3.1: Trade-off features among different calibration procedures.

3.3 Gyroscope

In the following sections, the rationale and the operations for the gyroscope activation, settings, usage and calibration are analyzed in depth.

3.3.1 Activation and Reading Procedure

The procedure for the activation of the BMI160 gyroscope is straightforward and simply requires to set the power mode and configure the desired settings by writing the right values to the correspondent microcontroller registers through the I^{2}C communication bus. Before executing these operations, performing a soft reset is highly recommended. When using the gyroscope in combination with the accelerometer and/or the magnetometer, only one initial soft reset has to be done, otherwise initialization settings are lost. Operations to be done are summarized in Tab. (3.2), specifying the execution order; a delay of minimum 100 ms is set between each read or write operation in order to not cause congestion of the I^{2}C bus. The full scale is set at the minimum value possible since the DANCE facility will be characterized by very low operative angular rates; this setting allows also to have higher resolution in measurements. The output data rate (ODR) is set to be in accordance with magnetometer ODR choice (Sec. 3.5), but the exact same value for BMI160 gyroscope is not available, so it is taken the minimum possible one.

Order	Register name	Address	Write value	Purpose
1	PMU_MODE	0x7E	0x6B	Trigger soft reset
2	PMU_MODE	0x7E	0x15	Set gyro normal mode
3	GYR_CONF	0x42	0x26	Set ODR at 25 Hz
4	GYR_RANGE	0x43	0x04	Set $R_G = 125°/\text{s}$ full scale
5	DATA_8	0x0C	-	Read gyro measurements data

Table 3.2: Gyroscope activation and reading procedures.

3.3.2 Calibration: State of the Art

As already explained in Sec. (3.2), many different procedures exist for MEMS calibration, belonging to different categories. In the particular case of a gyroscope, the most simple comparison procedure would require a highly stable rotating platform capable of holding the device in a steady-state condition, with a known angular velocity; in this case it is possible to find not only the offsets which characterize the gyroscope when still, but also to check error during measurements. For the present work, a trade-off evaluation among the possibilities brought to the choice of relying on the third category of procedures, as further explained by the following considerations. The first main reason is the current unavailability at the DAER laboratory of a facility which can rotate in a very stable and precise manner around one axis, equipment which is generally very expensive. The other reason that confirms the direct procedure as a good approach is that the measurement offsets of the gyroscope can be experimentally computed by holding the device in a still position, as described more in detail in the next section. In this case, the advantages regard the simplicity of such procedure, which is inexpensive, fast, simple and brings to acceptable results, as it turned out from the functional tests. Values are in fact calibrated with an error on the measured angular rate which is $\leq 0.025°/s$, when the device is set still. The expected error on measurement during operations is instead already reported on the gyroscope datasheet [15], and the dynamic bias is estimated during the attitude reconstruction process (see Chapter 4 for attitude estimation algorithms).

3.3.3 Procedure

The process consists in keeping the gyroscope in a stationary position and measuring the output value; knowing that in an ideal case the angular rate measured must be zero allows to determine the constant bias which affects the sensor. In fact, in this condition, the sensor is subjected only to the Earth rotation, but its value of approximately $\omega_{earth} = 7.3 \cdot 10^{-5} rad/s$ is way under the sensitivity of the instrument, so it is not influencing the output. Of course, the presence of noise $n_{G,nD}$ is unavoidable and must be taken into account: typical values can be retrieved directly from the datasheet of BMI160 [15]. Assuming that for frequencies in the range lower than 1 kHz it can be treated purely as an ergodic random process with zero mean (white noise), the expected noise standard deviation in degrees per second can be computed with Eq. (3.4). For one axis:

$$\sigma_{G,n} = n_{G,nD} \cdot \sqrt{f_{G,sampling}} \tag{3.1}$$

where, from the Shannon-Nyquist theorem:

$$f_{G,sampling} \leq \frac{ODR}{2} \tag{3.2}$$

Being the gyroscope output data rate set to 25 Hz, the maximum disturbance expected on each axis results to be:

$$\sigma_{G,n} = 0.025\ °/s \tag{3.3}$$

Stability is in fact checked by ensuring that the variation of the measured angular rate is lower or at least equal to noise disturbance; mean variation is found by computing the standard deviation $\boldsymbol{\sigma}_G$ for the uncalibrated values:

$$\boldsymbol{\sigma}_G(i) = \sqrt{\frac{1}{N_{meas}} \cdot \sum_{i=1}^{N_{meas}} \Big(\boldsymbol{\omega}_{uncal}(i) - \overline{\boldsymbol{\omega}}_{uncal}(i)\Big)^2} \tag{3.4}$$

A recap on the statistical parameters used in the present work can be found in Appendix A.1.

In Tab. (3.3) the requirements for the stability check and the validation of the calibration procedure are set. $\boldsymbol{\sigma}_{G,cal}$ is computed with the same equation of the standard deviation (Eq. 3.4) but taking as reference the expected value of null angular rate on each axis instead of considering the mean of the measurements; it will be referred to as "standard deviation of gyroscope's calibrated values" even if this definition is not rigorously correct. The reason why the parameter is computed in this way is that in the ideal case the mean coincides with the zero expected value. In this way, it is possible to verify if the calibration procedure has been successful and validate the results. GYRO-REQ-1 is checked during every application of the calibration procedure, while GYRO-REQ-2 is used to validate the procedure itself, so it is not strictly necessary to be checked on each application.

As pointed out, the important factor to take care of during the procedure is the angular rate of the gyroscope, which must be kept as close to zero as possible. Regarding instead the physical orientation of the sensor with reference to the gravity vector, no particular needs are present. Normally the measurement is affected by the gravity field by a quantity identified as *g sensitivity*, which is reported on the datasheet; nevertheless keeping the

GYRO-REQ-1	The gyroscope sensor can be considered still if $\boldsymbol{\sigma}_G(i) \leq \sigma_{G,n}$ or, equivalently, if $\boldsymbol{\sigma}_{G,raw}(i) \leq 27\ LSB/s$ for every i, with the gyro specified settings ($\sigma_{G,n} = 0.025\ °/s$).
GYRO-REQ-2	The calibration procedure can be considered successful if $\boldsymbol{\sigma}_{G,cal}(i) \leq \sigma_{G,n}$ for every i, with the gyro specified settings ($\sigma_{G,n} = 0.025\ °/s$).

Table 3.3: Gyroscope calibration requirements.

sensor still this quantity acts as a noise disturbance which is intrinsically considered and compensated by the offsets computations. To prove this fact the calibration has been experimentally carried out with the device set in various orientations (example in Fig 3.3), showing none sensible difference in results as shown in Tab. (3.5). Moreover, the BMI160 gyroscope included in the BOOSTXL sensor pack is a high-quality device which shows very little g sensitivity [15] and, for DANCE applications, angular rates to be measured are typically very small so external acceleration plays a secondary role. For the final calibration the device is kept horizontally (i.e. with the z axis aligned with the g vector) because it is the same orientation at which it will be subject during its use at the beginning of each experiment once posed on the top of a DANCER AP; it is also the simplest configuration to keep the gyroscope still.

The MATLAB code for real-time data reading through serial communication (Sec. 2.6.1) has been modified appositely for the calibration procedure in order to take a user-specified number of measurements N_{meas}, get raw data and convert them into an angular rate $[°/s]$ by multiplication with a gyro scaling factor obtained as:

$$Sf_G = R_G \cdot 2^{-15} \tag{3.5}$$

where R_G is the range of values set during the configuration procedure. In fact, working in single precision, each measurement is represented in the microcontroller processor as a 16 bits binary number, in which the first bit is dedicated to the sign; as a consequence, the values read belong to the interval $[2^{-15}\ 2^{15}]$ that must be scaled to the specified R_G. After that, the algorithm performs the operations that are hereafter reported; an extensive explanation is mandatory since they represent the fundamental logic passages followed in the calibration procedure. According to the order of execution, one finds:

1. Computation of the average using the scaled measurements;
2. Estimation of the noise variance $\boldsymbol{\sigma}_G$ with Eq. (3.4);
3. Check if the value of $\boldsymbol{\sigma}_G$ satisfies the GYRO-REQ-1 imposed, otherwise gyroscope is not stable enough and measurements must be repeated;
4. Find the gyro offsets vector

   $$\boldsymbol{O}_G = [O_{G,x},\ O_{G,y},\ O_{G,z}] \tag{3.6}$$

   as the difference between the mean value and the expected null value.

Some more passages are then needed for the validation of the procedure, and are performed after having inserted the offsets in a different validation script:

1. Read gyro data and perform the calibration using Eq. (3.7) with the computed offsets;
2. Computation of the standard deviation of calibrated value $\boldsymbol{\sigma}_{G,cal}$ considering the expected null vector instead of the mean in Eq. (3.4);
3. Check that $\boldsymbol{\sigma}_{G,cal}$ satisfies GYRO-REQ-2 on each axis.

The calibration procedure can be validated and considered successful if, taking each component of $\boldsymbol{\sigma}_{G,cal}$ computed on last passage, GYRO-REQ-2 is satisfied. Gyroscope output is then granted to be characterized by an error lower than 10^{-1} $[°/s]$ when set in a still position.

Once the offsets have been computed, they are inserted in the read routine (Sec. 2.6.1) and subtracted to the raw measurements before the range scaling, in order to compensate the static bias. In fact the measured angular rate in degrees per second is finally computed with Eq. (3.7).

$$\boldsymbol{\omega}_{meas} = [\boldsymbol{\omega}_{uncal} - \boldsymbol{O}_G] \cdot Sf_G \tag{3.7}$$

3.3.4 Test, Results and Validation

The calibration procedure described has been experimentally tested several times, with different range of values for N_{meas} parameter and with the gyroscope kept in different conditions. Example results are reported in Tab. (3.4) and (3.5). What is found is that, since the device is theoretically kept still, by increasing the number of measurements the final precision in offsets computation is not increased. In fact adding more and more data in such steady-state condition only permits to observe noise oscillations on a longer period of time, but the mean value does not change sensibly. On the other hand, a minimum of few hundreds of measurements is needed to have a sufficient amount of data on which perform statistical computations. For these reasons N_{meas} is suggested to be set arbitrarily between a value of 1000 and 10000. Here even the case with $N_{meas} = 100$ is reported for completeness and also because, technically speaking, it satisfies the imposed tolerances. Of course, all the above discussion holds only if the requirement for stability is satisfied.

N_{meas}	σ_G	$O_{G,x}$	$O_{G,y}$	$O_{G,z}$	$\sigma_{G,cal}$
100	[0.024 0.025 0.019]	-97	195	675	[0.024 0.023 0.028]
1000	[0.024 0.023 0.021]	-97	200	674	[0.022 0.021 0.027]
5000	[0.072 0.067 0.063]	-100	198	680	[0.023 0.022 0.021]
10000	[0.075 0.072 0.086]	-100	198	679	[0.022 0.021 0.022]

Table 3.4: Gyroscope calibration results for different number of measurements.

Even if the variability of offsets may seem high, one must consider that as much as the variation of raw values is ≤ 27 LSB the tolerance is satisfied, and the results do not vary more than $0.025°/s$; offsets are in fact variable inside the explained range even if the test is repeated other times with the same N_{meas}. Since $\boldsymbol{\sigma}_{G,cal}$ is on each direction lower than the tolerance set, GYRO-REQ-2 is satisfied and the procedure validated. In Fig. (3.1) and (3.2) are shown respectively the norm of $\boldsymbol{\omega}$ and its components before and after the calibration procedure; it is evident how measurement have been correctly rectified.

Orientation	N_{meas}	$\boldsymbol{\sigma}_G$	$O_{G,x}$	$O_{G,y}$	$O_{G,z}$	$\boldsymbol{\sigma}_{G,cal}$
0°	1000	[0.024 0.023 0.021]	-97	200	674	[0.022 0.021 0.027]
$90°@x_{axis}$	1000	[0.020 0.020 0.020]	-97	199	672	[0.019 0.023 0.022]
$90°@y_{axis}$	1000	[0.022 0.020 0.019]	-97	200	674	[0.023 0.021 0.020]
$90°@z_{axis}$	1000	[0.022 0.022 0.022]	-95	199	673	[0.021 0.024 0.022]

Table 3.5: Gyroscope calibration results for different orientations.

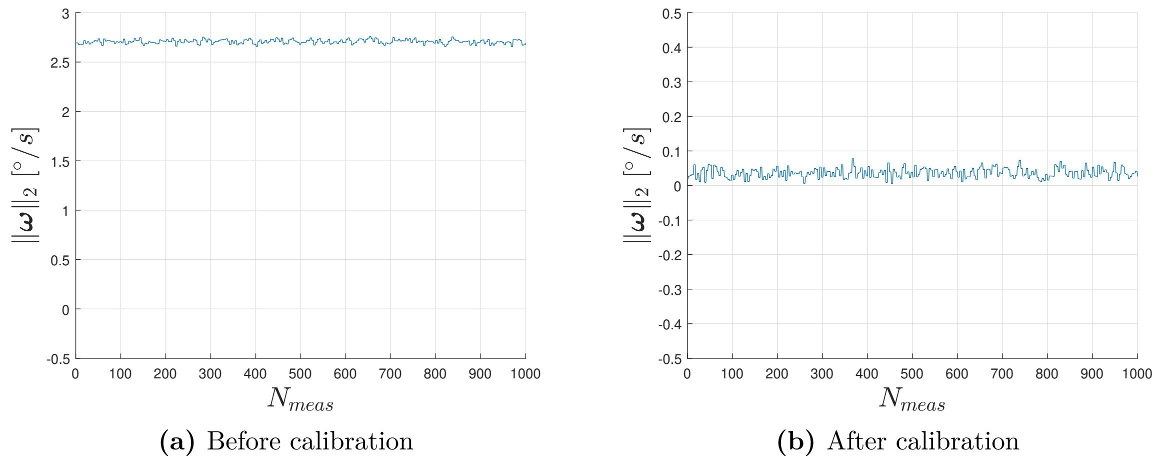


Figure 3.1: Comparison of $\|\boldsymbol{\omega}\|_2$ before and after calibration.

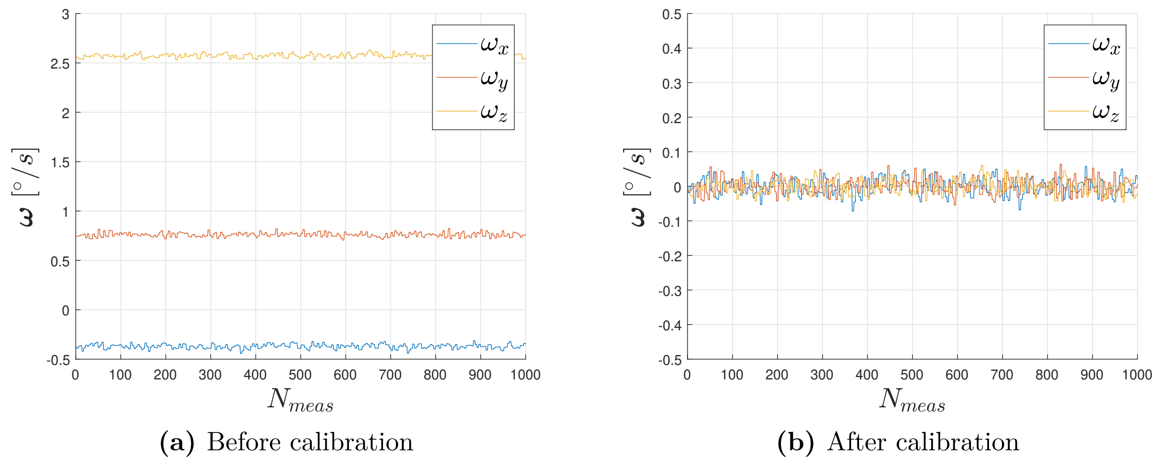


Figure 3.2: Comparison of $\boldsymbol{\omega}$ components before and after calibration.

3.4 Accelerometer

In the following sections, the rationale and the operations for the accelerometer activation, settings, usage and calibration are analyzed in depth.

3.4.1 Activation and Reading Procedure

The procedure to power on and set the correct mode for the BMI160 accelerometer is completely equivalent to the one of the gyroscope (Sec. 3.3.1): passages to be done are reported in Tab. (3.6). Even in this case the full scale is set to the minimum value possible to get the highest resolution; a larger range is not necessary since the device is used to get a normalized vector which contains only the information about the direction of the gravity vector, and not the value of the acceleration itself during the motion, which will be very little. The ODR instead is again chosen to be in line with the magnetometer ODR forced-choice (Sec. 3.5).

Order	**Register name**	**Address**	**Write value**	**Purpose**
1	PMU_MODE	0x7E	0x6B	Trigger soft reset
2	PMU_MODE	0x7E	0x11	Set acc normal mode
3	ACC_CONF	0x40	0x25	Set ODR at 25 Hz
4	ACC_RANGE	0x41	0x03	Set $R_A = \pm 2g$ full scale
5	DATA_14	0x12	-	Read acc measurements data

Table 3.6: BMI160 Accelerometer activation and reading procedures.

3.4.2 Calibration: State of the Art

Comparison procedure, in this case, consists in using a facility which holds the accelerometer on a specified position, knowing the orientation with respect to $\boldsymbol{g}$ with a precision higher than the one of the sensor itself; sensor must be kept still in order to avoid the presence of external acceleration due to motion. As an alternative, ellipsoid fitting can be performed (the same procedure adopted for the magnetometer, explained in Sec. 3.5.3). For such approach, the difficulty is again in holding the device still while taking a high number of measurements in every direction as uniformly as possible, so it would be mandatory to develop a dedicated facility. In the simplest case, offsets can be computed performing the procedure only along the measurement reference axes, while retrieving the error for cross-axis measurements from the datasheet [15]. This last procedure has been adopted for the present work, minimizing the equipment needed while trying to maximize the output accuracy with post-processing software corrections.

3.4.3 Procedure

Even for the accelerometer the trade-off between simplicity and effectiveness, together with the difficulty in finding a preciser instrument for the comparison process, brought to the choice of a direct procedure that can be accomplished without the need of expensive additional equipment and requires few minutes to be carried out. This last is in fact a desirable property since the measurement precision achieved can be lost in time and/or after some usage, so it may be necessary to repeat the calibration more than once. To evaluate when re-calibration is necessary, it is enough to check if the accelerometer output vector $\boldsymbol{a}$ is normalized or not. In order to explain this assertion, it is important to highlight that the useful information that must be retrieved from the accelerometer output is the direction of the gravity acceleration measured in the body-fixed frame and not its absolute value; comparing in fact the obtained direction with an initial reference, the rotation of the body can be determined. For this reason, calibration is designed to have a normalized vector in output from the sensor, so when the two-norm $\|\boldsymbol{a}\|_2$ is different from the unity (considering the specified tolerance) the calibration result is no more valid and the process must be repeated again.

The rationale of the proposed procedure [20] consists in setting the device in six different positions, such that the $\boldsymbol{g}$ vector is alternatively directed on a positive and negative direction of each of the three accelerometer reference axes; on every fixed position several measurements are taken, organized in a matrix form and imposed to be equal to the expected output. The raw values read are firstly scaled into $\boldsymbol{a}_s$ with unit measurement of $[g]$ by multiplication with a scale factor (analogous to what explained in Sec. 3.3.3):

$$Sf_A = R_A \cdot 2^{-15} \tag{3.8}$$

The normalized output $\boldsymbol{a}_n$ can then be obtained by subtracting the offsets and multiplying by the rotation matrix A_R which maps the measurements into a unit sphere. The system to be solved is represented by Eq. (3.9) written for a single measurement, where the 12 unknowns are the elements of the rotation matrix and the three mentioned offsets.

$$\begin{bmatrix} a_{x,n} \\ a_{y,n} \\ a_{z,n} \end{bmatrix} = \begin{bmatrix} A_R \end{bmatrix}_{3\text{x}3} \cdot \begin{bmatrix} a_{x,s} - O_{A,x} \\ a_{y,s} - O_{A,y} \\ a_{z,s} - O_{A,z} \end{bmatrix} \tag{3.9}$$

The vector $\boldsymbol{a}_s$ on the the right-hand side collects scaled measurements, while $\boldsymbol{a}_n$ represents the normalized expected output; measurements will be repeated for $2N_{meas}$ times on single axis (positive and negative orientation). The expected output is imposed to be one among the six unit vectors:

$$[\pm 1,\ 0,\ 0]\ g \tag{3.10a}$$

$$[0,\ \pm 1,\ 0]\ g \tag{3.10b}$$

$$[0,\ 0,\ \pm 1]\ g \tag{3.10c}$$

The system of Eq. (3.9) can be rewritten grouping the 12 unknowns in a single matrix, obtaining the compact form of Eq. (3.11), written again for a single measurement. Repeating the process N_{meas} times for each axis and storing the variables in matrix form, one obtains Eq. (3.12). The solution is then found by applying the least square method so computing the pseudo inverse of W (Eq. 3.13).

$$\begin{bmatrix} a_{x,n} \ a_{y,n} \ a_{z,n} \end{bmatrix} = \begin{bmatrix} a_{x,s} \ a_{y,s} \ a_{z,s} \ 1 \end{bmatrix} \cdot \begin{bmatrix} X \end{bmatrix}_{4\text{x}3} \tag{3.11}$$

$$\begin{bmatrix} Y \end{bmatrix}_{6N_{meas}\text{x}3} = \begin{bmatrix} W \end{bmatrix}_{6N_{meas}\text{x}4} \cdot \begin{bmatrix} X \end{bmatrix}_{4\text{x}3} \tag{3.12}$$

$$X = [W^T \cdot W]^{-1} \cdot W^T \cdot Y \tag{3.13}$$

The fundamental aspect in this process is to define the precision by which each orientation is characterized, in terms of stability and deviation with respect to the $\boldsymbol{g}$ vector. Regarding the last aspect, without having a preciser instrument it is almost impossible to know the exact misalignment between the two vectors, and the value got from the device itself can not be trusted before calibrating it. Even with the availability of a preciser accelerometer or, for example, a cube with faces that are orthogonal with a certain specified tolerance, it would be difficult to place the sensor on the specified directions since it is soldered to the BOOSTXL plug-in module, which is in turn assembled to the TI LaunchPad, so it is not free to move. Moreover, even the assembling can bring to orientation errors. For those reasons, the different orientations are set by "trusting" the board case so by placing the board with the assembled sensors on a stable flat surface (Fig. 3.3), following the directions of the reference frame reported on BOOSTXL user guide [14] and on the device surface (Fig. 2.2); orientation errors are corrected by the calibration procedure itself. In fact the important aspect is, as noted previously, to have a precise reference for the variation of the direction of the acceleration vector and not the value of acceleration itself.

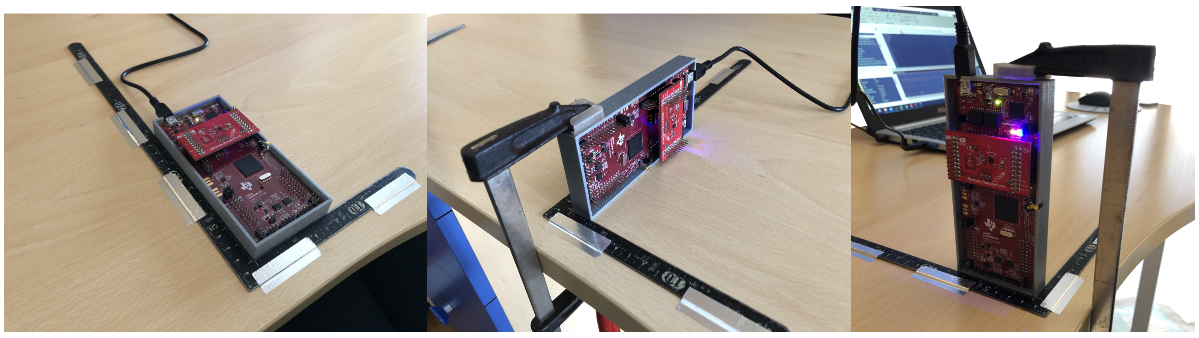

(a) $\boldsymbol{g}$ directed on $-z$ **(b)** $\boldsymbol{g}$ directed on $+x$ **(c)** $\boldsymbol{g}$ directed on $-y$

Figure 3.3: Different LaunchPad and sensors positioning during calibration procedure.

It is nevertheless necessary to check the stability of the accelerometer orientation in every position, to avoid errors due to undesired movements. Each value, as in the case of the gyroscope, is affected by the presence of disturbances that can be retrieved from the datasheet [15] and considered as white noise for frequencies lower than 1 kHz. With analogous computations of Sec. (3.3.3) the maximum deviation in terms of g results to be:

$$\sigma_{A,n} = n_{A,nD} \cdot \sqrt{\frac{ODR}{2}} = 1,1\ mg \tag{3.14}$$

Consequently, the requirements reported in Tab. (3.7) for stability and validation are set. The standard deviation before calibration $\boldsymbol{\sigma}_A$ is computed with Eq. (3.4), equivalently to the gyroscope; again $\boldsymbol{\sigma}_{A,cal}$ is instead found using the same equation, but employing the calibrated values and considering the expected output for each axis on each orientation instead of the reference mean value. Even if it is not a rigorous definition it will be referred to as "standard deviation of the accelerometer's calibrated values" because of the similarity in computing the parameter. The standard deviation is retrieved separately for each orientation because, in this way, there is the possibility to check if only some particular orientations are not passing the standard imposed. Note that ACC-REQ-2 has been relaxed of one order of magnitude with reference to the expected noise value; this is due to the high experimental errors and precision issues that can be encountered in performing the procedure.

ACC-REQ-1	The accelerometer sensor can be considered still if $\boldsymbol{\sigma}_A(i) \leq \sigma_{A,n}$ or, equivalently, if $\boldsymbol{\sigma}_{A,raw}(i) \leq 12\ LSB/s$ on each orientation, with the specified accelerometer settings ($\sigma_{A,n} = 1.1\ mg$).
ACC-REQ-2	The calibration procedure can be considered successful if $\boldsymbol{\sigma}_{A,cal}(i) \leq 10^{-2}\ g$ for each i on each orientation, with the specified accelerometer settings.

Table 3.7: Accelerometer stability calibration condition requirement.

The MATLAB code for real-time data reading (Sec. 2.6.1) has been appositely modified for the calibration, and can be set to take the desired number of measurements N_{meas} for each axis. The algorithm reads raw data through serial communication and scales them by multiplication with Sf_G, then performs the following operations:

1. Start of measurements indicating to the user to set the device on the screen-printed specified orientation among the six possible;
2. Computation of the average using the scaled measurements;
3. Estimation of the noise variance $\boldsymbol{\sigma}_A$ for the current orientation with Eq. (3.4) using the mean value as reference;
4. Check if every component of $\boldsymbol{\sigma}_A$ satisfies the ACC-REQ-1 imposed, otherwise the whole procedure must be repeated;
5. Repeats steps 1-4 for each one of the six directions;
6. Solve the system of Eq. (3.12) with Eq. (3.13) and find the 12 searched calibrating parameters.

The computed X matrix is then inserted into the validation algorithm which performs the last operations:

1. Repeating N_{meas} for each axis and calibrate them by multiplying the augmented matrix of scaled measurements W by the computed X matrix;
2. Computation of the standard deviation of the calibrated values $\boldsymbol{\sigma}_{A,cal}$ for each orientation, using the expected value as the reference instead of the mean value;
3. Check that $\boldsymbol{\sigma}_{A,cal}$ satisfies ACC-REQ-2.

The calibration procedure can be considered successful if, considering $\boldsymbol{\sigma}_{A,cal}$ computed for every single orientation on last passage ACC-REQ-2 is satisfied. In this case, it means that measures are done with the accelerometer in still position have an error lower than 10^{-2} g. Once the matrix of unknowns has been computed, it is inserted in the read routine; real-time read algorithm augments the measurement vector $[a_{x,s}\ a_{y,s}\ a_{z,s}]_{1x3}$ by concatenating 1 as fourth component and obtaining $[W]_{1x4}$, then multiplies it by $[X]_{4x3}$.

3.4.4 Test, Results and Validation

The whole procedure has been tested many times obtaining satisfactory performance, as proven by the following results. Even in this case the N_{meas} parameter is not influencing significantly the final output if it is chosen with a minimum value of 1000; the suggested range is between 1000 and 2000 for a good compromise between the statistic sample dimension and the acquisition time, since N_{meas} must be repeated 6 times. With $N_{meas} = 100$ it happens often to not be able to pass the validation process since X matrix computation is not accurate enough. X is reported in Tab. (3.9) only for one case by way of example, other cases are not reported for readability reasons; mean estimated variance on uncalibrated values, error on calibrated measurements and norm before and after calibration are instead reported for some different cases on Tab. (3.8). Every component of $\boldsymbol{\sigma}_{G,cal}$ satisfies ACC-REQ-2, so the procedure is validated as expected. On Fig. (3.4) the norm of the acceleration measurements before and after calibration is reported; as it can be noticed, calibration improved significantly the normalization of the output vector.

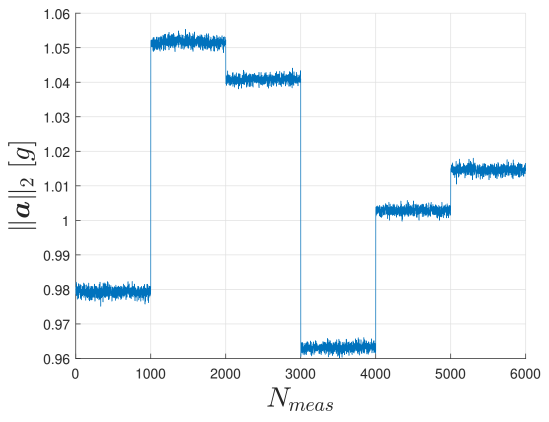
N_{meas}	mean $\boldsymbol{\sigma_A}$	mean $\boldsymbol{\sigma}_{\boldsymbol{A},\boldsymbol{cal}}$	mean $\|\boldsymbol{a}_{cal}\|_2$
1000	[0.0003 0.0004 0.0004]	[0.0012 0.0020 0.0024]	0.999993
1500	[0.0003 0.0004 0.0004]	[0.0016 0.0022 0.0029]	0.999994
2000	[0.0004 0.0004 0.0005]	[0.0004 0.0004 0.0004]	0.999990

Table 3.8: Accelerometer calibration results for different number of measurements.

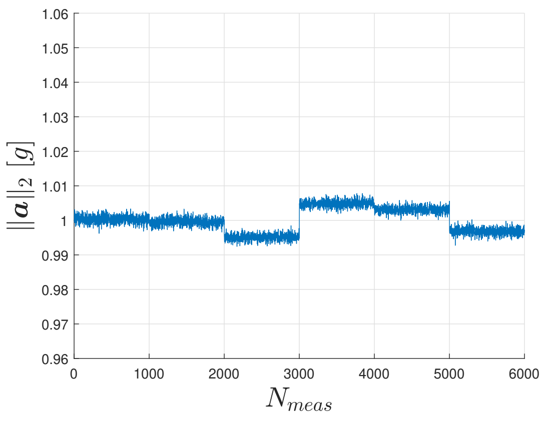
X matrix		
-0.9929	0.0066	0.0100
-0.0109	-0.9990	-0.0124
-0.0048	0.0037	-0.9858
0.0071	0.0474	0.0309

Table 3.9: Calibration matrix for $N_{meas} = 1000$.

The calibration procedure has demonstrated to be useful also to correct the reference frame for acceleration measurements with reference to the gravity vector direction. In fact, initial measurements are not in accordance with the reference system which is reported on the user's guide of BOOSTXL sensors module and on the module surface. By keeping it with the upper face (identified by the presence of reference system drawings and pin indications) opposite to $\boldsymbol{g}$, the measured acceleration should be $[0,\ 0,\ -1]g$ while the actually measured one is $[0,\ 0,\ +1]g$. This may be a factory error; nevertheless, calibration



(a) Before calibration



(b) After calibration

Figure 3.4: Comparison of $\|\boldsymbol{a}\|_2$ before and after calibration on the six orientations evaluated during the calibration procedure ($N_{meas} = 1000$ for each orientation).

is done in order to get $[0,\ 0,\ -1]g$ while the LaunchPad is kept with the upper face opposite to $\boldsymbol{g}$ and then following the orthonormal reference depicted on the BOOSTXL module upper face. In reality, since the $\boldsymbol{a}$ measured is used only as reference for direction variation and not as an absolute measure, the actual initial value is not important as long as it is taken as reference and an orthonormal reference frame is considered, but for consistency with the global and body-fixed DANCER reference frames measure is corrected as explained.

3.5 Magnetometer

The BMM150 magnetometer sensor requires more operations than BMI160 to be used from the point of view of activation, settings and reading procedure. The calibration procedure that has been implemented is simple from a mathematical point of view, but a bit intricate to be performed. All the relevant aspects are analyzed in this section.

3.5.1 Activation and Reading Procedure

The activation procedure is, in this case, not straightforward since the magnetometer cannot be accessed directly: it is connected to the BMI160 as a secondary slave device controlled through SPI or I^{2}C buses. For the actual purposes the usage of I^{2}C bus results to be more convenient, also for consistency with the other devices employed. Indirect writing/reading procedures mean that all the commands given by the user must be written to the particular registers of BMI160 which in turn writes them to the correspondent addresses of BMM150, requiring in this way a lot more operations to be done than normal and thus making the activation procedure more complex. Anyway, once the procedure has been implemented, there could be the need to change only the values for the relevant settings so the overall performance is not affected. In Tab. (3.10) are reported the operations to be done in order at every power on, each one followed by a pause of at least 100 ms. To make the magnetometer work it is required that also accelerometer and gyroscope are in normal mode. The number of repetitions N_{reps} to determine the output of each measure can be manually set according to the precision wanted; since the magnetometer is very sensible to external electromagnetic disturbances, the *high accuracy preset* is chosen by setting $N_{reps} = 83$ for z axis and $N_{reps} = 47$ for x, y axis as indicated in the magnetometer datasheet [16]. With this particular configuration, the recommended ODR is indicated to be set at 20 Hz maximum, so it is imposed at 12.5 Hz since it is the maximum available value that respects the indication.

Order	Register name	Address	Write value	Purpose
1	PMU_MODE	0x7E	0x6B	Trigger soft reset
2	PMU_MODE	0x7E	0x11	Set acc normal mode
3	PMU_MODE	0x7E	0x15	Set gyro normal mode
4	PMU_MODE	0x7E	0x19	Set mag normal mode
5	MAG_IF_0	0x4B	0x26	Set mag I^2C address (slave)
6	MAG_IF_1	0x4C	0x83	Enable set up and burst read
7	IF_CONF	0x6B	0x20	Enable secondary interface
8	MAG_IF_4	0x4F	0x01	Data to put mag in sleep mode
	MAG_IF_3	0x4E	0x4B	Set mag address to be written
9	MAG_IF_4	0x4F	0x04	Data to set N_{reps} for z axis
	MAG_IF_3	0x4E	0x51	Set mag address to be written
10	MAG_IF_4	0x4F	0x0E	Data to set N_{reps} for x, y axis
	MAG_IF_3	0x4E	0x52	Set mag address to be written
11	MAG_IF_4	0x4F	0x02	Data to set mag write address
	MAG_IF_3	0x4E	0x4C	Set mag address to be written
12	MAG_IF_2	0x4D	0x42	Set mag interface data address
13	MAG_CONF	0x44	0x06	Set ODR at 12.5 Hz
14	MAG_IF_1	0x4C	0x00	Enable mag data mode
15	DATA_0	0x04	-	Read mag measurement data

Table 3.10: BMM150 Magnetometer activation and reading procedures.

3.5.2 Calibration: State of the Art

As in the case of the IMU, even for the magnetometer plenty of different procedures are found in literature. Without using a preciser magnetometer for a comparison approach, other possibilities rely on inserting the sensor in a known generated magnetic field (substitution procedure) or are attitude-dependent procedures; a good overview with assets and drawbacks of several methods can be found [21] and it is not here reported to avoid repeatability. Considering performing easiness and the possibility of application without the need for any external equipment as main drivers for the trade-off choice, ellipsoid fitting is the calibration procedure adopted for the present work. The lower precision that may be obtained with respect to other methods is a declared drawback, nevertheless acceptable results are retrieved anyway; in fact, it may not worth using more sophisticated procedures since the MEMS magnetometer used is prone to noisy measurements and disturbances sensibility, so the final result may not change sensibly. A mandatory final remark on the ellipsoid fitting procedure chosen is that it can be applied only because we are interested in having a normalized output that contains direction information and not the value of the magnetic field itself (analogously to the accelerometer case), as further investigated hereafter.

3.5.3 Procedure

Vectorial measurements of an uncalibrated magnetometer, if plotted on the three-dimensional space with a x, y, z orthonormal reference frame, belong to an ideal ellipsoid which center does not coincide with the origin. The calibration procedure consists in mapping the ellipsoid to a unitary radius sphere centered on the origin; this is possible to be done by firstly performing a translation of the center, i.e. finding and subtracting 3 offsets which identify the position of the ellipsoid center, then multiplying by a rotation matrix to align the ellipsoid axes with reference axes, and finally by multiplying by the gains which map it to the expected sphere. In fact (as in the accelerometer case) the magnetometer is not used to retrieve an absolute measure of the Earth's magnetic field, but rather to have information on the variation of the direction of the DANCER AP upon which the sensor is mounted with reference to an initial direction, that in the attitude reconstruction algorithm is taken as the initial device output (Sec. 4.2). For this reason, the calibration is done in order to ensure that the magnetometer measurements are always normalized vectors; computing the two-norm $\|\boldsymbol{m}\|_2$ one can rapidly check how much it differs from the unity and consequently evaluate if re-calibration is necessary. The magnetometer is expected to be the fastest among the MEMS used to lose calibration during its usage, so frequent applications may be necessary; also, the procedure must be repeated every time the device is used on a different location, due to the different environmental conditions.

The calibration procedure [22] requires at first to read the magnetometer raw values which are converted to the right unit of measurement of the magnetic field (Tesla, T) by multiplication with a scaling factor Sf_M (equivalently to Sec. 3.3.1). The gain for measurements on z axis is different than the one for x and y axes since the range of values is different as well so $\boldsymbol{Sf}_M$, in this case, is a vectorial quantity. Also, the range of values R_M can vary for the single axis, as reported in Tab. (3.11); it is evident how measurements on z direction are less accurate. The scaling factor is computed in Eq. (3.15) using the typical range values.

	Minimum $[\mu T]$	**Typical** $[\mu T]$
$R_{M,xy}$	± 1200	± 1300
$R_{M,z}$	± 2000	± 2500

Table 3.11: Magnetometer measurements ranges.

$$\boldsymbol{Sf}_M = [R_{M,xy}\ R_{M,xy}\ R_{M,z}] \cdot 2^{-15} \tag{3.15}$$

Once scaled, values read by the magnetometer belong to an ellipsoid described by the general equation:

$$aX^2 + bY^2 + cZ^2 + 2dXY + 2eXZ + 2fYZ + 2gX + 2hY + 2iZ = 1 \tag{3.16}$$

or, in a matrix form:

$$\Big[D\Big]_{N_{meas}\text{x}9} \cdot \Big[\boldsymbol{v}\Big]_{9\text{x}1} = \Big[I\Big]_{N_{meas}\text{x}1} \tag{3.17}$$

where D represents the matrix in which the measurements are combined and stored , $\boldsymbol{v}$ is the vector of 9 unknowns to be found and I the identity matrix. Once again the system is solved by the means of the least-square approach:

$$\boldsymbol{v} = [D^T \cdot D]^{-1} \cdot D^T \cdot I \tag{3.18}$$

Few more passages are then needed in order to compute explicitly offsets (Eq. 3.19) and gains (Eq. 3.23, 3.24) from $\boldsymbol{v}$, reported below. About the offsets:

$$\begin{bmatrix} O_{M,x} \\ O_{M,y} \\ O_{M,z} \end{bmatrix} = \begin{bmatrix} a\ d\ e \\ d\ b\ f \\ e\ f\ c \end{bmatrix}^{-1} \begin{bmatrix} g \\ h \\ i \end{bmatrix} \tag{3.19}$$

and about the gains, introducing the auxiliary matrixes T and A:

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ O_{M,x} & O_{M,y} & O_{M,z} & 1 \end{bmatrix}, \quad A = \begin{bmatrix} a\ d\ e & g \\ d\ b\ f & h \\ e\ f\ c & i \\ g\ h\ i & -1 \end{bmatrix} \tag{3.20}$$

$$B_1 = T \cdot A \cdot T^T \tag{3.21}$$

$$B_2 = -\frac{1}{b_{44}} \cdot \begin{bmatrix} b_{11}\ b_{12}\ b_{13} \\ b_{21}\ b_{22}\ b_{23} \\ b_{31}\ b_{32}\ b_{33} \end{bmatrix} \tag{3.22}$$

indicating with b the elements of B_1. Axis gains G_M are then retrieved from the eigenvalues ev (that represent the radii of the ellipsoid) of B_2 with Eq. (3.23); rotation matrix R is constituted by column eigenvectors $\vec{\text{v}}$ of B_2 (Eq. 3.24).

$$\boldsymbol{G}_M = \Big[\tfrac{1}{G_{M,x}}\ \tfrac{1}{G_{M,y}}\ \tfrac{1}{G_{M,z}}\Big]^T = \Big[\sqrt{ev_1}\ \sqrt{ev_2}\ \sqrt{ev_3}\Big]^T \tag{3.23}$$

$$R = \Big[\vec{\text{v}_1}\ \vec{\text{v}_2}\ \vec{\text{v}_3}\Big]_{3\text{x}3} \tag{3.24}$$

Since the unknowns are 9, in order to perform the calibration procedure and compute gains and offsets an equivalent minimum number of measurements is needed, on directions that must not be parallel each other in order to avoid singularity conditions. In reality, as the numbers of measurements increases, also the accuracy of the whole procedure is higher because of the fact that the mapping factors are computed on a wider statistic distribution of values. In an ideal case, all the points belonging to the surface of the ellipsoid should be available but, of course, it is a condition impossible to reach since the surface is made by infinite points; what is done in the real case is to take N_{meas} points by manually moving the magnetometer such that measurements are taken as uniformly distributed as possible in all directions. The option of having an autonomous system able to rotate the sensor has been evaluated; in this case the advantage would be the possibility to implement a fixed and highly-repeatable trajectory to be followed by the magnetometer, thus getting measurement always on the same directions and mapping the ellipsoid in a uniform way.

Unfortunately, this idea has to be discarded for one main reason: any electromechanical device generates electromagnetic disturbances which deviate the magnetometer measures because it is very sensible since it must detect Earth's magnetic field (order of 10^{-5} T) and in fact it is characterized by a resolution of 0.3 μT [16]. Thus using a robotic arm or electric motors is not feasible. The alternative is to build a facility which is manually operated and able to perform always the same rotations, in the same order, and to map the whole ellipsoid. Even this possibility has been discarded for the actual purposes, since the time required for designing and building such a complicated facility may not worth the effort: by choosing a very high value for N_{meas} and manually moving the magnetometer a good uniform distribution is obtained (Fig. 3.5c) and results after the calibration are acceptable (Sec. 3.5.4); the only negative aspect is that the procedure is not repeatable with the measurements on the same exact position. For this purpose validation of the whole procedure is done every time at the end of it, by computing the norm of calibrated values and checking the deviation from the unity. The magnetometer is also the less precise sensor among the three that are used since it is the most sensitive to external disturbances, so the attitude reconstruction relies much more on the IMU measurements, as confirmed by the tuning gains found after the optimization of the attitude estimation algorithms on Chapter 4. The requirement in Tab. (3.12) for the validation is set, considering as reference parameter the standard deviation $\sigma_{M,cal}$ computed on the norm of calibrated values using the expected value of 1 as mean reference value, and is compared to the expected value present on the datasheet.

MAG-REQ-1	The procedure for the magnetometer sensor calibration be considered successful if $\sigma_{M,cal} \leq 0.3$ μT with the specified accelerometer settings.

Table 3.12: Magnetometer requirement for calibration procedure.

The MATLAB code for real-time serial read (Sec. 2.6.1) has been appositely modified for the calibration and can be set by the user imposing the desired number of total measurements N_{meas} to be taken. The additional possibility of a real-time plotting of the measured values can be activated, in order to help the user to get a uniform spatial distribution. Operations done by the algorithm are the following:

1. Reading of raw values and scaling them by multiplication with $\boldsymbol{Sf}_M$;
2. Once N_{meas} measurements are done, solving the system of Eq. (3.17) with Eq. (3.18);
3. Explicit computing the offsets, gains and rotation matrix elements;
4. Computing the mean norm of the values after calibration for a fast check.

After that, the unknowns found are inserted into the validation algorithm, which allows to take some random measurements and to compute $\sigma_{M,cal}$ which represents the variance of the norm, which must satisfy MAG-REQ-1 in order to validate the procedure. The validation is done by keeping the magnetometer on the same location, otherwise calibration effects could be lost.

Once the offsets and gains are known and the procedure is validated, their values are inserted into the real-time read routine, which maps the values to the mentioned unit sphere by subtracting the offsets, rotating the ellipsoid and scaling to unit radius with Eq. (3.25) giving the calibrated and normalized magnetic field measurement output.

$$\boldsymbol{m}_{meas} = [\boldsymbol{m}_{raw} - \boldsymbol{O}_M] \cdot R^{-1} \cdot \boldsymbol{G}_M \tag{3.25}$$

In order to ease the insertion of all the computed unknowns in the real-time read routine, the column vectors of gains and offsets, together with the rotation matrix, are stored on a single 5x3 matrix:

$$[\boldsymbol{O}_M \; \boldsymbol{G}_M \; R] \tag{3.26}$$

3.5.4 Test, Results and Validation

The procedure has been carried out several times with different N_{meas} values. In this case, the number of measurements is relevant and should be kept to a minimum value of 20000 to obtain a good estimation of gains and offsets, but the other important aspect to be considered is how uniform the distribution of measurement is. As a general trend, system solution accuracy increases as far as N_{meas} increases if the measurements are evenly distributed. The difficult part of the proposed procedure is, in fact, ensuring that the ellipsoid is described as uniformly as possible. Nevertheless, by using a sufficiently high N_{meas} and moving randomly the magnetometer, good results are obtained. Fig. (3.5) represent the ellipsoid to sphere mapping with 5000 vs 20000 measurements, evidencing the big difference in distribution; on the first case number of available values is too low. Tab. (3.13) reports instead numerical results for the mean norm and error after calibration.

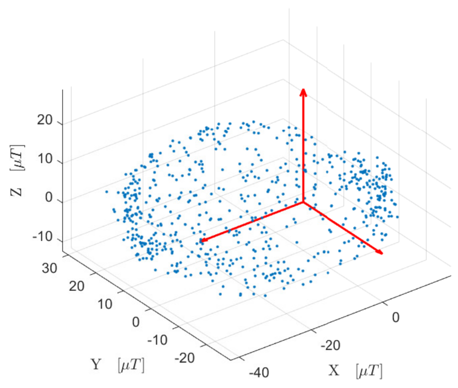
N_{meas}	mean $\|\boldsymbol{m}\|_2$	$\sigma_{M,cal}$ $[\mu T]$
5000	0.9935	0.0537
20000	0.9987	0.02274
35000	0.9997	0.0304

Table 3.13: Magnetometer calibration results for different number of measurements.

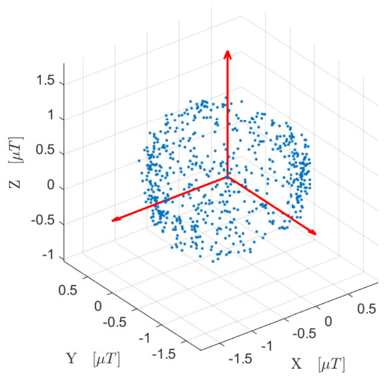
In all the cases MAG-REQ-1 is satisfied and the procedure validated, but it is evident how the standard deviation on calibrated values is higher on the case with $N_{meas} = 5000$.

3.6 Further Development

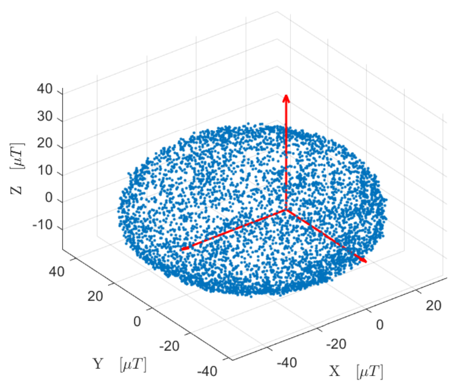
Calibration procedures here proposed for the BOOSTXL IMU and magnetometer are simple and do not require the usage of additional equipment, making them easy and fast to be performed whenever necessary. Despite their simplicity, they turned out to be effective and gave great results with the expected precision. Even if in theory the accuracy obtained should be sufficient for DANCER attitude reconstruction applications, whether or not this is true also in the real case will be determined during functional tests; the final results are however considerably affected also by the type of attitude estimation



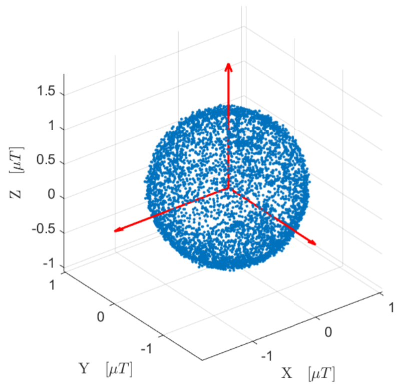
(a) Before calibration ($N_{meas} = 5000$)



(b) After calibration ($N_{meas} = 5000$)



(c) Before calibration ($N_{meas} = 20000$)



(d) After calibration ($N_{meas} = 20000$)

Figure 3.5: Mapping of the ellipsoid to the unit sphere on the origin for magnetometer calibration with different N_{meas} values.

algorithm used. There are a few operations that can be done in order to increase the effectiveness of calibration:

- About the gyroscope, calibration has been performed with respect to a stationary position, i.e. setting the zero reference level for the angular rate measurement. It means that the error for the computation of the offsets and their application is under our control, but instead the tolerance and measurement error during gyroscope motion are retrieved from the datasheet; if it is necessary to increase the accuracy, a comparison procedure with a preciser instrument which rotates at a fixed angular rate must be done. Testing should be performed for each rotation axis and also for cross-axis gains.
- For what concerns the accelerometer, the requirement for the validation has been relaxed of one order of magnitude with reference to the expected noise value, since errors are slightly higher than what it could be ideally obtained, because of experimental errors. If higher precision is needed one must rely again on a comparison procedure. In particular, what can be improved is the precision by which the device is set in every position by the means of a facility able to keep the sensor still and aligned with $\boldsymbol{g}$ measuring the position with higher precision than BMI160 itself; but also with the availability of such facility, orienting may be very difficult since the accelerometer is assembled to the BOOSTXL module and attached to the TI microcontroller, as pointed out previously (Sec. 3.4.3). In that case, more research would be needed.
- For the magnetometer, keeping the same approach, accuracy can be increased only by assessing precise repeatability of measurement and ensuring a uniform and complete mapping of the ellipsoid with an electronic-free rotating device, as already explained (Sec. 3.5.3). But higher precision in calibration is not granted, since the sensor is highly sensitive to external disturbance and the N_{meas} points used is already very high, with great uniformity in distribution of measurements.
- The overall performance in following the proposed procedures can be improved by using a RF unit, in order to discard the presence of the USB cable which can interfere with the board movements, and by supplying the board directly getting the 3.3 V supply voltage from an external battery instead of using the USB cable, with the purpose of avoiding little voltage variations.

All the variables, gains and offsets that constitute the output of calibration operations are then manually inserted into the final attitude determination and control algorithm. The alternative to doing it manually would be to store automatically those variables into the device non-volatile memory (NVM). Several problems have been encountered in evaluating this possibility: at first, writing to the NVM of BMI160 and BMM150 is not possible, since they are characterized by a limited number of write operations of few times [15, 16]. On the other hand, the EEPROM of the F28379D board cannot be accessed easily, since it requires to be written with a single cycle after a complete deletion, meaning that all the values should be stored at the same time and thus it would be not possible to perform calibration for only one sensor but for all the three sensor together. Moreover, to be used in such a way the EEPROM requires to be adequately set in order to simulate the functioning of flash memory, a process that cannot be done in Simulink but only with machine-level programming (like Assembly language). If the manual procedure is too slow to be done every time, future development can rely on the usage of external flash memory. A good choice could be the SPI flash memory [23] reported in Fig. (3.6); it is a low-cost flash memory with 16 M-Bit of free storage that can be used through SPI bus. A total number of 30 numerical variables must be stored for the three sensors; considering that each value is represented with the IEEE 754 convention [17] (32 bit each), the total memory required is 1kBit. The rationale of the procedure consists in performing the calibration, store the output by writing the flash memory which in turn is read every power-on of the microcontroller; after reading, the values are saved in local variables to be accessed during DANCER utilization.

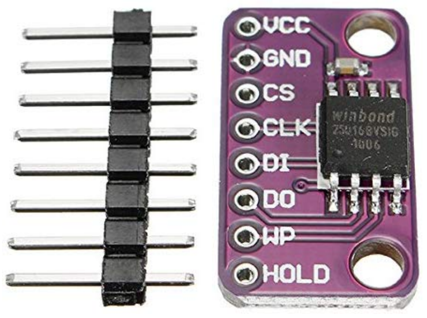


Figure 3.6: External flash memory.

4 Attitude Estimation and Control

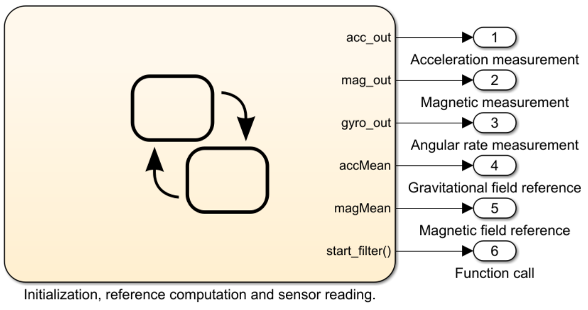
4.1 Chapter Overview

Attitude reconstruction is the process by which the orientation of the body-fixed frame of a dynamical system with respect to a known inertial reference frame is determined, and it constitutes a fundamental part of the GNC system. In fact, without knowing the real-time orientation of the system, it is impossible to control it and to perform relative motion tests. Several different algorithms, called *filters*, give the possibility to combine the sensor measurements and to obtain the system orientation; in this chapter, the problem of attitude reconstruction is exposed and different solutions are proposed, considering also the results of the work done with previous efforts [5] in developing the Kalman filter which is taken into account as a valid alternative.

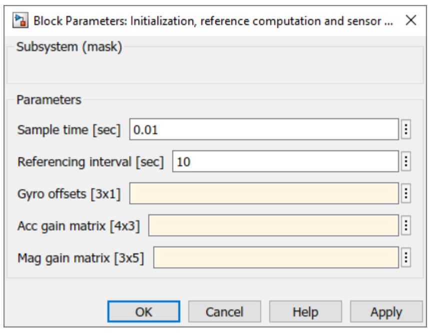
4.2 Software and Sensors Integration

For the final DANCE setup, it is required at first to retrieve in real-time the measurements from the gyroscope, accelerometer and magnetometer present in the BOOSTXL plug-in module employed. The outputs are then processed by the software downloaded to the TI LaunchPad that gives them as input values to the filtering algorithm used to process the quantities and to reconstruct the orientation and angular velocity of the body. For the present purposes of evaluating the performance of the filtering algorithm, the software is developed up to this point; instead for the final operative version, a step forward must be done in order to compute the correct PWM signals needed to control the actuators, starting from the filter outputs.

The algorithm used is the one described in Sec. (2.6.1), with some small modifications. In fact after having performed the activation procedure for all the sensors, during an initial referencing time $t_{ref} = 10\ s$ only accelerometer and magnetometer measurements are retrieved, averaging their values on the whole period of time in order to compute the initial reference for gravitational and magnetic field directions. This procedure is performed every time the microcontroller is switched on or a soft reset is performed by the means of the lateral button (Sec. 2.3.1), for this reason all the times that a new operation is started the device must be kept in a stable and fixed position, otherwise initial reference directions will be affected by errors that can compromise the whole attitude reconstruction process. After t_{ref} has passed, all the sensors are set in normal mode and a function call activates the filtering algorithm which processes the incoming stream of measured data, giving in output the already mentioned quantities. All measurements and reconstructed attitude parameters can be transmitted through the SCI peripheral and be read on MATLAB, in order to be able to test the real-time performance but also to record the values and perform off-line filter tuning and optimization. Fig. (4.1) reports the modified version of the algorithm (with reference to the one of Fig. 2.8), showing the outputs and the setting parameters including the sample time, the initial referencing time



(a)



Block Parameters: Initialization, reference computation and sensor ...

Subsystem (mask)

Parameters

Sample time [sec] 0.01

Referencing interval [sec] 10

Gyro offsets [3x1]

Acc gain matrix [4x3]

Mag gain matrix [3x5]

(b)

Figure 4.1: Simulink Stateflow block for real-time sensors reading with initial reference computation (a) and relative mask with settable parameters (b): sample time and referencing interval are set by default to $t_{sample} = 0.01\ s$ (corresponding to an update frequency of 100 Hz) and $t_{ref} = 10\ s$ respectively; the other parameters must be inserted after having performed the calibration procedure.

for averaging and the offsets computed with the calibration procedure. In particular, the added blocks in the Stateflow chart for the reference computation are shown in Fig. (4.2): the two states are in a parallel configuration, the first getting the sensor outputs while the second averaging them in time in a synchronous way, without the need of storing the collected values.

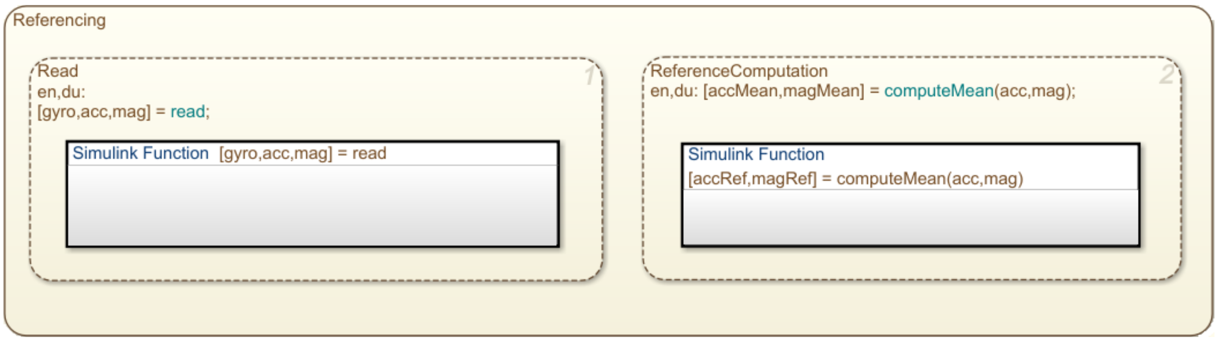

Referencing
Read
en,du:
[gyro,acc,mag] = read;
1
Simulink Function [gyro,acc,mag] = read
ReferenceComputation
en,du: [accMean,magMean] = computeMean(acc,mag);
2
Simulink Function
[accRef,magRef] = computeMean(acc,mag)

Figure 4.2: Simulink Stateflow chart for reference computation.

The Simulink code above described is ready to be attached to one of the different alternative filters that are proposed in the next sections.

4.3 Mahony Filter

The attitude reconstruction filter developed by Mahony et al. [7] is taken into account as an alternative for the DANCE system. The main reason is that the Kalman filter, even if under suitable assumptions it represents an optimal filter (Sec. 4.5), results to be very expensive from the computational point of view, leading to inefficient usage of the real-time hardware resources employed for this project; trade-off among alternative reconstruction algorithms is thus necessary. Secondly, Mahony filter is widely employed for cases in which low-cost MEMS sensors are used [6, 24] and, at the state of the art, it is among the most efficient and reliable filters for such applications [7]. It is termed *complementary* filter for the nature of the equations that are used to elaborate sensors measurements, as it will be explained more in details later on. Different formulation versions exist [7]:

- *Direct* complementary filter and *Passive* complementary filter: non-linear formulations working purely with the quaternion representation, but not very suitable to be used with MEMS sensors.
- *Explicit* complementary filter: the formulation is done in terms of vectorial measurements obtained from an IMU and can be extended for the use with MARG systems. On-line gyroscope bias drift estimation can be included as well.

The explicit version fits the DANCE requirements, since it demands only accelerometer and gyroscope outputs but can be also integrated with readings from a magnetometer. In particular, in order to implement the algorithm, the quaternion formulation appears to be particularly useful since it avoids having singularity problems and enhances computational efficiency. Equations used for the model are now described.

Gyroscope measurements are affected by the unavoidable presence of noise n and a bias error term b_{err} which is, in the general case, time-varying. Expressing measurement in the body frame B with reference to a fixed inertial frame I, the output is modelled as:

$$^{B}\boldsymbol{\omega}_{meas} = {}^{I}\boldsymbol{\omega} + \boldsymbol{b}_{G,err} + \boldsymbol{n}_{G} \tag{4.1}$$

The bias term will be estimated in real-time during algorithm execution. For the accelerometer, analogous considerations lead to the formulation of Eq. (4.2), where $\boldsymbol{v}$ represents the velocity of the system in the inertial frame and A the attitude matrix expressing the relative position of the two reference frames, i.e. the rotation matrix from the inertial reference frame to the body fixed one $^{B}_{I}R$, explicitly written in Eq. (4.4).

$$^{B}\boldsymbol{a}_{meas} = A(^{I}\dot{\boldsymbol{v}} - {}^{I}\boldsymbol{g}) + \boldsymbol{b}_{A,err} + \boldsymbol{n}_{A} \tag{4.2}$$

DANCER vehicle will be characterized by very slow movements which translate into very low accelerations, thus $\dot{\boldsymbol{v}} \approx 0$ and the biggest contribution is given by the gravity acceleration. The bias that characterizes the accelerometer is not estimated. Magnetometer mathematical model is very similar:

$$^{B}\boldsymbol{m}_{meas} = A \cdot {}^{I}\boldsymbol{m} + \boldsymbol{m}_{dist} + \boldsymbol{n}_{B} \tag{4.3}$$

where the external magnetic disturbances that can be possibly present are included in the $\boldsymbol{m}_{dist}$ term. Writing the rotation matrix explicitly with quaternions later defined, we obtain Eq. (4.4).

$$^{B}_{I}A = \begin{bmatrix} 2q_1^2 - 1 + 2q_2^2 & 2(q_2q_3 + q_1q_4) & 2(q_2q_4 - q_1q_3) \\ 2(q_2q_3 - q_1q_4) & 2q_1^2 - 1 + 2q_3^2 & 2(q_3q_4 + q_1q_2) \\ 2(q_2q_4 + q_1q_3) & 2(q_3q_4 - q_1q_2) & 2q_1^2 - 1 + 2q_4^2 \end{bmatrix} \tag{4.4}$$

The equations of the explicit complementary filter with on-line bias estimation that are used are here reported in Eq. (4.8) with the quaternion formulation. Components of the quaternion vector $[\boldsymbol{q}]_{4x1}$ are defined with Eq. (4.5a), where e_1, e_2, e_3 are the components of $\boldsymbol{e}$ which represent the direction of Euler axis, and γ a general rotation around it. Quaternions are characterized by the fundamental property of Eq. (4.5b), and the identity element is represented by $\boldsymbol{q} = [\pm 1;\ 0;\ 0;\ 0]$. Eq. (4.5c) defines instead the quaternion conjugate $\boldsymbol{q}^*$ of $\boldsymbol{q}$.

$$q_1 = cos\Big(\frac{\gamma}{2}\Big);\ q_2 = e_1 sin\Big(\frac{\gamma}{2}\Big);\ q_3 = e_2 sin\Big(\frac{\gamma}{2}\Big);\ q_4 = e_3 sin\Big(\frac{\gamma}{2}\Big) \tag{4.5a}$$

$$q_1^2 + q_2^2 + q_3^2 + q_4^2 = 1 \tag{4.5b}$$

$$\boldsymbol{q}^* = [q1;\ -q2;\ -q3;\ -q4] \tag{4.5c}$$

We define also the operator:

$$\mathrm{p}(\boldsymbol{x}) = [0;\ \boldsymbol{x}] \tag{4.6}$$

and, introducing the representation $\boldsymbol{q} = [s;\ \overline{\boldsymbol{q}}]$, the quaternion product between two generics quaternion vectors $\boldsymbol{q}_1$, $\boldsymbol{q}_2$ is defined as:

$$\boldsymbol{q}_1 \otimes \boldsymbol{q}_2 = \begin{bmatrix} s_1 s_2 - \overline{\boldsymbol{q}}_1^T \overline{\boldsymbol{q}}_2 \\ s_1 \overline{\boldsymbol{q}}_2 + s_2 \overline{\boldsymbol{q}}_1 + \overline{\boldsymbol{q}}_1 \times \overline{\boldsymbol{q}}_2 \end{bmatrix} \tag{4.7}$$

At the (k) time step, indicating with the *hat* notation the estimation of the represented quantity, with $\boldsymbol{v}_i$ a generic sensor measurement and with $\boldsymbol{v}_{i,ref}$ the initial reference in the inertial frame relative to the same measured quantity, one obtains:

$$\begin{aligned} \boldsymbol{\omega}_{err,k} &= -\mathrm{vex}\Bigg(\sum_{i=1}^{n} \frac{K_i}{2}\Big(\boldsymbol{v}_i \boldsymbol{v}_{i,ref}{}^T - \boldsymbol{v}_i^T \boldsymbol{v}_{i,ref}\Big)\Bigg) \\ &= -\mathrm{vex}\Bigg(\frac{K_a}{2}\Big(\boldsymbol{a}_k \boldsymbol{a}_{ref}^T - \boldsymbol{a}_k^T \boldsymbol{a}_{ref}\Big) + \frac{K_m}{2}\Big(\boldsymbol{m}_k \boldsymbol{m}_{ref}^T - \boldsymbol{m}_k^T \boldsymbol{m}_{ref}\Big)\Bigg) \end{aligned} \tag{4.8a}$$

$$\dot{\hat{\boldsymbol{b}}}_k = -K_{Int} \boldsymbol{\omega}_{err,k} \tag{4.8b}$$

$$\dot{\hat{\boldsymbol{q}}}_k = \frac{1}{2} \hat{\boldsymbol{q}}_{k-1} \otimes \mathrm{p}(\boldsymbol{\omega}_{meas,k} - \hat{\boldsymbol{b}}_k + K_p \boldsymbol{\omega}_{err,k}) \tag{4.8c}$$

where the *vex* operator denotes the inverse of the cross product matrix, defined with Eq. (4.9), as clarified by Eq. (4.10).

$$\Omega \times = \begin{bmatrix} 0 & -\Omega_3 & \Omega_2 \\ \Omega_3 & 0 & -\Omega_1 \\ -\Omega_2 & \Omega_1 & 0 \end{bmatrix} \tag{4.9}$$

$$\mathrm{vex}(\Omega \times) = \Omega \tag{4.10}$$

Eq. (4.8) represents Mahony filter mathematical expression. As it can be noticed, at each time step at first the error term $\boldsymbol{\omega}_{err}$ is estimated with Eq. (4.8a) and is used to compute the deviation of a vectorial measurement coming from the accelerometer and/or the magnetometer with respect to a reference vector $\boldsymbol{v}_{ref}$, which can be taken as the transformation to the body frame of the gravity acceleration and local magnetic field:

$${}^B\boldsymbol{a}_{ref} = {}^B_I A\ {}^I\boldsymbol{g} \tag{4.11a}$$

$${}^B\boldsymbol{m}_{ref} = {}^B_I A\ {}^I\boldsymbol{B_m} \tag{4.11b}$$

Of course ${}^{B}_{I}A$ is computed each time step with the updated quaternion estimate, while ${}^{I}\boldsymbol{g}$ and ${}^{I}\boldsymbol{B_m}$ are constant.

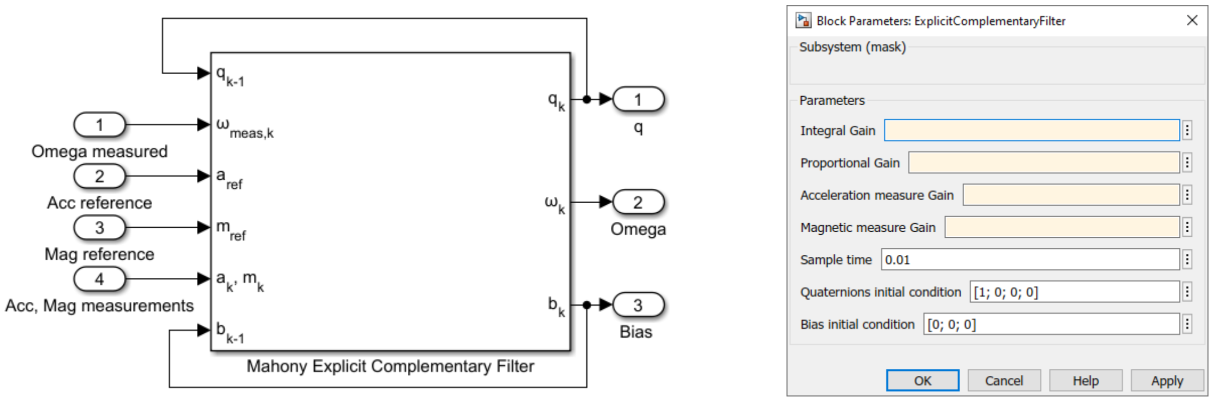
In particular, in order to have reliable values for the local physical quantities, initial sensor measurements are taken as reference (see Sec. 4.2). In fact acceleration and magnetic field measurements are used to find a unit vector that contains information on the direction and not on the absolute value of the inspected quantity; the direction must then be compared to an initial reference in order to compute the relative rotation of the body-fixed reference frame with respect to the initial position. The error term (Eq. 4.8a) is then used to correct the angular rate measurement $\boldsymbol{\omega}_{meas,k}$, together with the bias term estimated from the integration of $\dot{\hat{\boldsymbol{b}}}_k$ relative the previous time step. In fact, at each time interval the attitude and bias derivatives are integrated and in this way propagated, permitting an estimation of those quantities in real-time; integrated values are then corrected with current measurements in order to be used for the successive step. The parameters by which the filter behaviour can be controlled and sharpened are the four gains K_p, K_{int}, K_a and K_m; respectively, the proportional and integral gains rule convergence velocity and oscillations by changing error and bias contribution to the correction of the attitude estimation, while instead the last two terms weight respectively the acceleration and magnetic field measurements for the error computation (a short summary is found in Tab. 4.1).

Parameter	**Description**	**Effects**
K_p	Proportional gain	rules the global error contribution in correcting angular rate measurements
K_{int}	Integral gain	rules the bias contribution in correcting angular rate measurements
K_a	Accelerometer gain	weights the contribution of accelerometer output in error computation
K_m	Magnetometer gain	weights the contribution of magnetometer output in error computation

Table 4.1: Description of tuning parameters used in the Mahony filter.

4.3.1 Implementation

The Mahony filter equations described have been implemented both in MATLAB and Simulink. This last is needed in order to create the model that has to be downloaded to the TI LaunchPad and is, in fact, developed taking care of using blocks and functions compatible with the Embedded Coder. The MATLAB code, which is completely equivalent to the model built for the microcontroller, is instead needed to perform efficiently the optimization process to find the best tuning parameters, as explained later in Sec. (4.7). In Fig. (4.3) the main system block for the Simulink implementation is reported, in order to clarify which are the input and output quantities; this block can be directly connected to a simulator or integrated to the final DANCER software. As mentioned previously, the initial references for the magnetic and gravitational fields in the inertial frame are taken from sensors measurements by averaging the output for $t_{ref} = 10\ s$, and for this reason the sensors must be kept on a stable and fixed position every time the microcontroller is switched on or reset, in order to get a suitable reference. From the system block it is immediate to notice how the estimated quaternion and the gyroscope bias drift at the time step (k) are used in a feedback loop since they are needed for the integration during the successive time step. Initialization for both parameters is then necessary, relative values are reported in Tab. (4.2). The complete implementation of the filter is instead reported in Fig. (4.4).



Integral Gain	
Proportional Gain	
Acceleration measure Gain	
Magnetic measure Gain	
Sample time	0.01
Quaternions initial condition	[1; 0; 0; 0]
Bias initial condition	[0; 0; 0]

(a) **(b)**

Figure 4.3: Main system block of implemented Mahony filter on Simulink with the indication of input, output and feedback loop quantities (a) and user input parameters to be set (b).

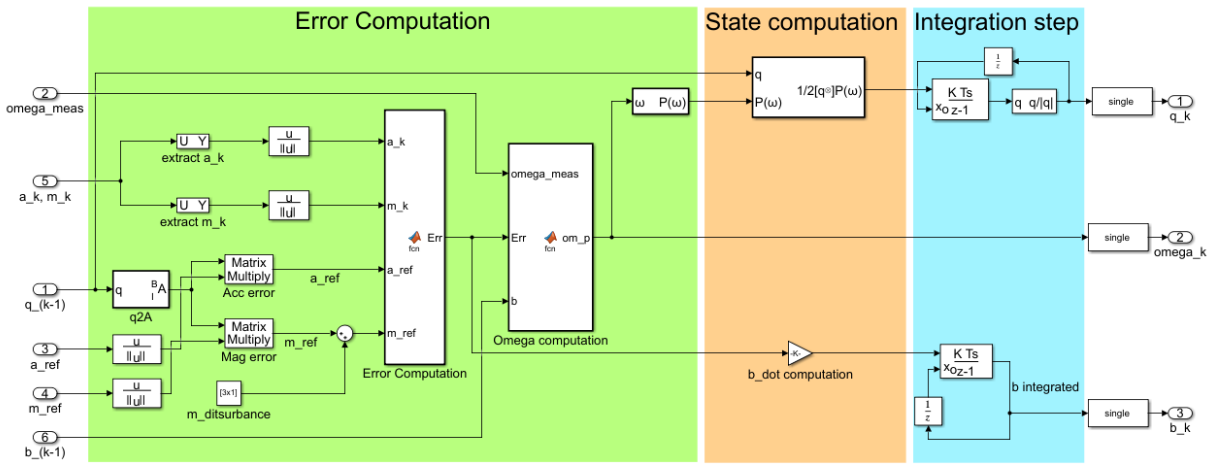


Figure 4.4: Complete Simulink implementation of the Mahony filter.

Parameter	Initial value	Description
$\boldsymbol{q}$	[1; 0; 0; 0]	Attitude quaternion
$\boldsymbol{b}$	[0; 0; 0]	Bias drift of gyroscope measurement

Table 4.2: Initialization of parameters for kinematics integration of the Mahony filter.

4.4 Madgwick Filter

The filtering procedure proposed by Madgwick et al. [6, 25] is another alternative here considered for the attitude reconstruction. It is a computationally efficient algorithm suitable to be used with inertial measurement units (IMUs) as well as integrating an IMU with a tri-axis magnetometer (MARG), fitting the case of the present work. Using a quaternion representation, the algorithm is able to apply a sensor-fusion technique in order to combine data coming from the accelerometer and magnetometer to estimate the gyroscope measurement error in the form of a quaternion derivative, including bias drift estimation and magnetic disturbance compensation. The filter is moreover described as effective also at low sampling rates (order of 10 Hz) [6], advantage that makes it appropriate to be used with the current BOOSTXL sensors settings and for real-time applications. Adopting the same quaternion notation of Sec. (4.8), main filter equations used for the implementation are hereafter precisely described.

The quaternion product of Eq. (4.7) can be used to compute compound rotations: a generic vector $\boldsymbol{v}$ is rotated from the body reference frame B to the inertial one I with Eq. (4.12), or equivalently with Eq. (4.13) by using the rotation matrix of Eq. (4.4).

$$ {}^{I}\boldsymbol{v} = {}^{I}_{B}\boldsymbol{q} \otimes {}^{B}\boldsymbol{v} \otimes {}^{I}_{B}\boldsymbol{q}^{*} \tag{4.12}$$

$$ {}^{I}\boldsymbol{v} = {}^{B}_{I}A^{T} \cdot {}^{B}\boldsymbol{v} \tag{4.13}$$

We can now define problem of Eq. (4.14a), where the objective function f defined in Eq. (4.14b) represents the quantity to be minimized in order to reconstruct the real-time attitude at each time step. The Madgwick filter is in fact the formulation of an optimization problem where the searched solution is the estimated orientation of the body at the $(k-1)$ step $\hat{\boldsymbol{q}}_{k-1}$, which rotates the correspondent vectorial measurement in the body frame $\boldsymbol{b}_i$ and aligns it with a predefined reference in the inertial frame $\boldsymbol{r}_i$ [24].

$$ \min_{\forall \hat{\boldsymbol{q}}_{k-1} \in \mathbb{R}^4} f\Big(\hat{\boldsymbol{q}}_{k-1},\ \boldsymbol{r}_i,\ \boldsymbol{b}_i\Big) \tag{4.14a}$$

$$ f\Big(\hat{\boldsymbol{q}}_{k-1},\ \boldsymbol{r}_i,\ \boldsymbol{b}_i\Big) = \hat{\boldsymbol{q}}^{*}_{k-1} \otimes \boldsymbol{r}_i \otimes \hat{\boldsymbol{q}}_{k-1} - \boldsymbol{b}_i \tag{4.14b}$$

The formulated problem has not a unique solution whenever only one field measurement is used: if for example the gravity field direction is known on the inertial frame and measured in the body frame by the accelerometer, the two reference frames can be aligned but the solutions are infinite and represented by all the possible rotations around an axis parallel the measured field. For this reason magnetic and gravity field measurements are combined together, in order to provide a complete solution for the searched estimated quaternion $\hat{\boldsymbol{q}}$. By combining the different sensors outputs the new objective function can be defined with Eq. (4.15) formed by a two-blocks matrix, making clear the reference frame of each quantity to avoid ambiguities. The filter is used in its generalized formulation, taking the two vectors ${}^{I}\boldsymbol{a}_{ref}$, ${}^{I}\boldsymbol{m}_{ref}$ as initial references respectively for gravitational and magnetic field; sensors measurements are indicated with the same notation as the Mahony filter (Sec. 4.3).

$$ f_{a,m}\Big({}^{I}_{B}\hat{\boldsymbol{q}}_{k-1},\ {}^{I}\boldsymbol{a}_{ref},\ {}^{B}\boldsymbol{a}_i,\ {}^{I}\boldsymbol{m}_{ref},\ {}^{B}\boldsymbol{m}_i\Big) = \begin{bmatrix} f_a\Big({}^{I}_{B}\hat{\boldsymbol{q}}_{k-1},\ {}^{I}\boldsymbol{a}_{ref},\ {}^{B}\boldsymbol{a}_i\Big) \\ f_m\Big({}^{I}_{B}\hat{\boldsymbol{q}}_{k-1},\ {}^{I}\boldsymbol{m}_{ref},\ {}^{B}\boldsymbol{m}_i\Big) \end{bmatrix} \tag{4.15}$$

The optimization problem can be solved in different ways, but the simple and computationally efficient solution proposed in [6] and reported in [24] uses a gradient descent algorithm that is able to find the local minimum by the means of computing the gradient of the objective function through the Jacobian matrix; this type of approach (i.e. Newton method) would require an iterative procedure for the search of the minimum, but with appropriate approximations it can be avoided. Detailed mathematical assumptions under which the algorithm is valid and additional passages can be found in [6] and are not here reported for readability reasons. At the end, the direction of the quaternion-derivative error $\dot{\hat{\boldsymbol{q}}}_{\varepsilon,k}$ relative to the estimated quaternion at time step $(k-1)$ is found with Eq. (4.16), where ∇f is defined in Eq. (4.17) for the case in which both magnetometer and accelerometer measurements are used.

$$\dot{\hat{\boldsymbol{q}}}_{\varepsilon,k} = \frac{\nabla f}{\|\nabla f\|} \tag{4.16}$$

$$\nabla f = \begin{cases} J_a^T({}^I_B\hat{\boldsymbol{q}}_{k-1},\ {}^I\boldsymbol{a}_{ref}) f_a\left({}^I_B\hat{\boldsymbol{q}}_{k-1},\ {}^I\boldsymbol{a}_{ref},\ {}^B\boldsymbol{a}_i\right) \\ J_{a,m}^T({}^I_B\hat{\boldsymbol{q}}_{k-1}, {}^I\boldsymbol{m}_{ref}) f_{a,m}\left({}^I_B\hat{\boldsymbol{q}}_{k-1},\ {}^B\boldsymbol{a}_i,\ {}^I\boldsymbol{m}_{ref},\ {}^B\boldsymbol{m}_i\right) \end{cases} \tag{4.17}$$

Where, in an equivalent way to Eq. (4.15), the "combined" Jacobian is defined as the following block matrix:

$$J_{a,m}\left({}^I_B\hat{\boldsymbol{q}}_{k-1},\ {}^I\boldsymbol{a}_{ref},\ {}^I\boldsymbol{m}_{ref}\right) = \begin{bmatrix} J_a^T({}^I_B\hat{\boldsymbol{q}}_{k-1},\ {}^I\boldsymbol{a}_{ref}) \\ J_m^T({}^I_B\hat{\boldsymbol{q}}_{k-1},\ {}^I\boldsymbol{m}_{ref}) \end{bmatrix} \tag{4.18}$$

Explicit matrix formulation can be found again on [6]. Fusion process permits to use the computed error direction on the estimation in combination with the usual quaternion kinematics of Eq. (4.19).

$$\dot{\hat{\boldsymbol{q}}}_{\omega,k} = \frac{1}{2}\Omega(\boldsymbol{\omega}_k) \cdot \hat{\boldsymbol{q}}_{k-1} \tag{4.19}$$

But before getting to the final filter equations, additional aspect for the gyroscope bias drift estimation must be analyzed. The Mahony filter approach permits to estimate the time-variable bias by the means of the integral error, and a similar concept is applied here for the Madgwick filter: at first an estimate of the error at the (k) step $\boldsymbol{\omega}_{\epsilon,k}$ is found with Eq. (4.20), which derives from Eq. (4.19). After that, error is integrated to obtain the bias $\boldsymbol{\omega}_{bias,k}$ of Eq. (4.21) and used to correct the angular rate measurements with an appropriate gain ζ, finally obtaining the corrected angular velocity $\boldsymbol{\omega}_{c,k}$ of Eq. (4.22).

$$\boldsymbol{\omega}_{\epsilon,k} = 2 \cdot \hat{\boldsymbol{q}}^*_{k-1} \otimes \dot{\hat{\boldsymbol{q}}}_{\epsilon,k-1} \tag{4.20}$$

$$\boldsymbol{\omega}_{bias,k} = \zeta \int \boldsymbol{\omega}_{\epsilon,k} dt \tag{4.21}$$

$$\boldsymbol{\omega}_{c,k} = \boldsymbol{\omega}_{meas,k} - \boldsymbol{\omega}_{bias,k} \tag{4.22}$$

The set of equations used for the Madgwick filter can now be retrieved (note that Eq. (4.23a) is the same of Eq. (4.16), but it is here reported again in order to have a complete summarized view on main filter equations):

$$\dot{\hat{\boldsymbol{q}}}_{\varepsilon,k-1} = \frac{\nabla f}{\|\nabla f\|} \tag{4.23a}$$

$$\boldsymbol{\omega}_{c,k} = \boldsymbol{\omega}_{meas,k} - \zeta \int \Big(2 \cdot \hat{\boldsymbol{q}}^{*}_{k-1} \otimes \dot{\hat{\boldsymbol{q}}}_{\epsilon,k-1}\Big) dt \tag{4.23b}$$

$$\dot{\hat{\boldsymbol{q}}}_{\omega,k} = \frac{1}{2}\Omega(\boldsymbol{\omega}_{c,k})\hat{\boldsymbol{q}}_{k-1} \tag{4.23c}$$

$$\dot{\hat{\boldsymbol{q}}}_{k} = \dot{\hat{\boldsymbol{q}}}_{\omega,k} - \beta\dot{\hat{\boldsymbol{q}}}_{\varepsilon,k} \tag{4.23d}$$

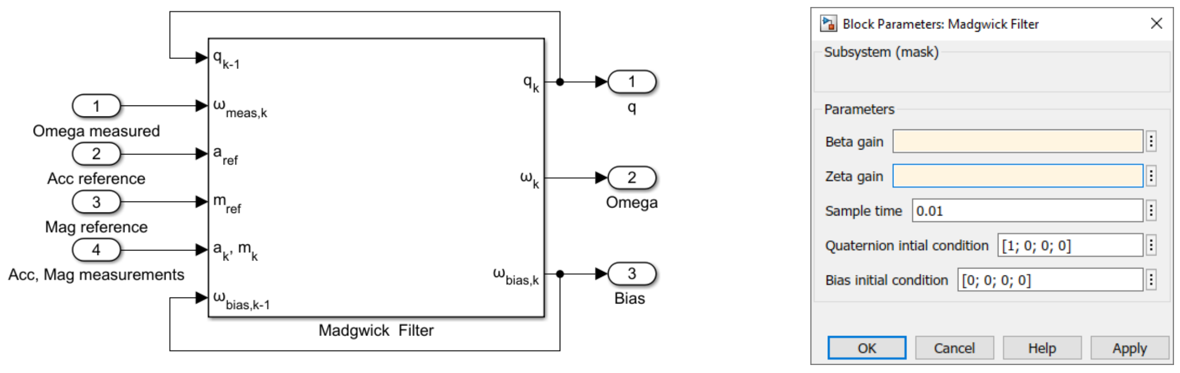
Eq. (4.23d) represents the core of the fusion algorithm: the estimated error direction found solving the optimization problem with the gradient descent algorithm and the estimated gyroscope bias found by the means of integral error are fused together to compute the time derivative of the quaternion at the step (k), which is then integrated to finally retrieve the reconstructed attitude of the body. As it can be noticed, the filter convergence and behaviour can be tuned with the two constant parameters β and ζ, as summarized in Tab. (4.3). Differently from the Mahony filter, in this original formulation the contributions of accelerometer and magnetometer measurements are not weighted but used directly into the filter equations. For this reason, an alternative version is here proposed, introducing two additional gain K_a and K_m and permitting to trust differently the outputs of the sensors, analogously to the Mahony filter. To the author knowledge, no evidence of this approach is explicitly found in literature; even if benchmark testing has shown no particular difference in results, this different formulation is kept into account since in the real case (i.e. testing the algorithm with the DANCER vehicle in working conditions) results may be different.

Parameter	Description	Effects
β	Proportional gain	rules the estimated error direction contribution in the fusion equation
ζ	Integral gain	weights the bias contribution in correcting angular rate measurements
K_a	Accelerometer gain	weights the contribution of accelerometer output in error computation
K_m	Magnetometer gain	weights the contribution of magnetometer output in error computation

Table 4.3: Description of tuning parameters used in the Madgwick filter in the original version (on top) and with the proposed alternative formulation (adding the two gains on bottom).

4.4.1 Implementation

The same considerations of Sec. (4.3.1) about Mahony filter implementation are valid also for the Madgwick case, so the general approach is not described again. MATLAB code is used in order to perform the tuning optimization, while the Simulink model is used to program the TI LaunchPad. The main system block evidencing input and output quantities is reported in Fig. (4.5), while the whole system is instead represented in Fig. (4.6). The parameters that need to be initialized and their specific values are reported in Tab. (4.4). The alternative version with four gains has been implemented only in MATLAB.



Subsystem (mask)

Parameters

Beta gain

Zeta gain

Sample time 0.01

Quaternion intial condition [1; 0; 0; 0]

Bias initial condition [0; 0; 0; 0]

(a) **(b)**

Figure 4.5: Main system block of implemented Madgwick filter on Simulink with the indication of input, output and feedback loop quantities (a) and mask with user input parameters to be set (b).

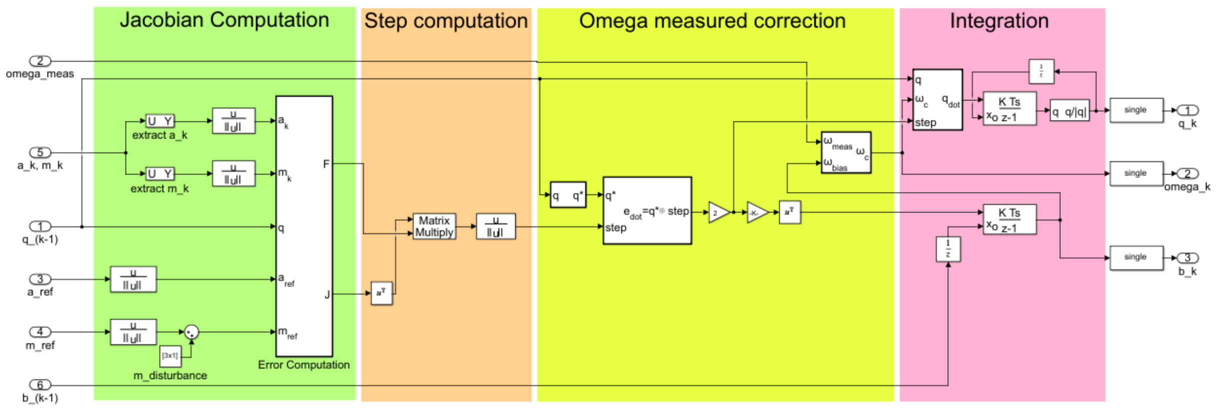


Figure 4.6: Complete Simulink implementation of the Madgwick filter.

Parameter	Initial value	Description
$\boldsymbol{q}$	[1; 0; 0; 0]	Attitude quaternion
$\boldsymbol{\omega}_{bias}$	[0; 0; 0]	Bias drift of gyroscope measurement

Table 4.4: Initialization of the parameters needed during the integration of the kinematics for the Madgwick filter.

4.5 Kalman Filter

The last alternative considered for the algorithm which purpose is to estimate the attitude and angular rate is the Kalman filter; this name is used to indicate a wide family of filters, with many different formulations. They represent the optimal filters if the following hypotheses are verified [26]: the noise which is entering the dynamical system can be assumed to be a zero-mean Gaussian noise (i.e. white noise), the noise covariance is known exactly and the mathematical model used matches the real system. The optimality feature is nevertheless lost when dealing with nonlinear problems, like in the case of attitude reconstruction; the most common approach of applying Kalman equations trying to maintain the optimality for such kind of problems is referred to as *Extended Kalman Filter*. The rationale is to linearize the error dynamics equation with a first-order Taylor expansion around an initial point by assuming that the estimated state is sufficiently close to the real one [26]. The filter formulation here proposed is the *Multiplicative* Extended Kalman Filter (MEKF), of course in its discrete-time form in order to be used in combination with digital systems. The MEKF uses a multiplicative error quaternion which can be reduced to a three-components vector since the first-order approximation performed under the mentioned assumptions gives $\delta q_1 \approx 1$, so $\delta \dot{q}_1 = 0$ when propagating its kinematics. In this way, the quaternion normalization is granted even during linear measurements update. The process of attitude estimation using this filter can then be summarized in two conceptual phases, similarly to the other filters proposed in this thesis work:

- Update: the attitude quaternion $\hat{\boldsymbol{q}}_{k-1}^-$ and the angular rate $\hat{\boldsymbol{\omega}}_{k-1}^-$ estimated from the previous time step are corrected (i.e. updated) using the input sensors measurements, obtaining $\hat{\boldsymbol{q}}_{k-1}^+$ and $\hat{\boldsymbol{\omega}}_{k-1}^+$.

- Propagation: the post-update estimates are propagated integrating the quaternion kinematics and the system dynamics, obtaining $\hat{\boldsymbol{q}}_k^-$ and $\hat{\boldsymbol{\omega}}_k^-$ for the next update step.

It is worth to highlight that the MEKF can not be considered as an optimal filter, since the optimality is lost because of the linearization process.

All filter equations and complete mathematical model are not here reported, since they have been exhaustively explained in the previous studies done for the DANCE project [5] (a detailed explanation of all the Kalman filter formulations can instead be found in [26]). The Simulink model of the filter has in fact been already developed and is used in the present work in order to compare the overall performance with the computationally lighter filters of Mahony and Madgwick, with the purpose of checking the differences in the overall performance and predispose alternative attitude estimation algorithms for the optimization of future development. Fundamental parameters used to tune the filter are reported in Tab. (4.5); these gains are used to compute noise covariance matrix Q_k with Eq. (4.24a) and the sensor noise covariance matrix R_k with Eq. (4.24b). Accelerometer and magnetometer variances can be in theory estimated directly from the datasheet [15, 16] with Eq. (3.14), while other two parameters are in general unknown (see Appendix A.1 for a summary on statistical indicators used).

$$Q_k = diag([\sigma_{\eta,v}^2 \; \sigma_{\eta,u}^2]) \tag{4.24a}$$

$$R_k = diag([\sigma_{acc}^2 \; \sigma_{mag}^2]) \tag{4.24b}$$

Generally speaking even the values found on datasheet relative to the noise variances of the sensors could not match exactly the real ones, so estimating also these parameters during the optimization process may result more convenient from the point of view of the filter performance.

Parameter	Description	Estimated Value
$\sigma^2_{\eta,v}$	Noise variance for the angular velocity drift	/
$\sigma^2_{\eta,u}$	Noise variance for the angular velocity drift rate	/
σ^2_{acc}	Accelerometer noise variance	$1.125{\cdot}10^{-6}$ $[g]$
σ^2_{mag}	Magnetometer noise variance	$5.625{\cdot}10^{-13}$ $[T]$

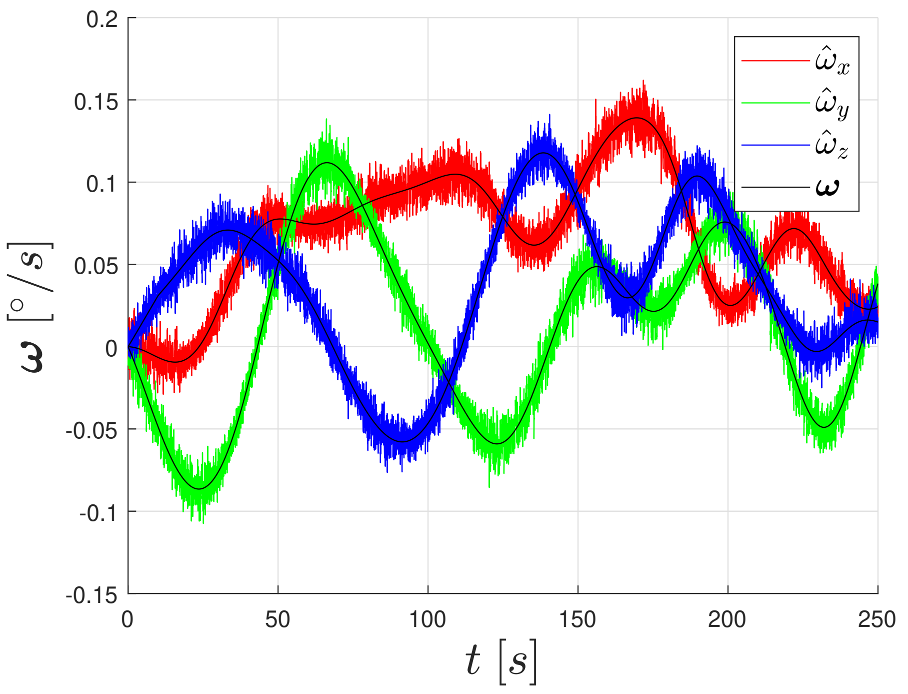
Table 4.5: Description of tuning parameters used in the Kalman filter.

4.6 Simulation Test

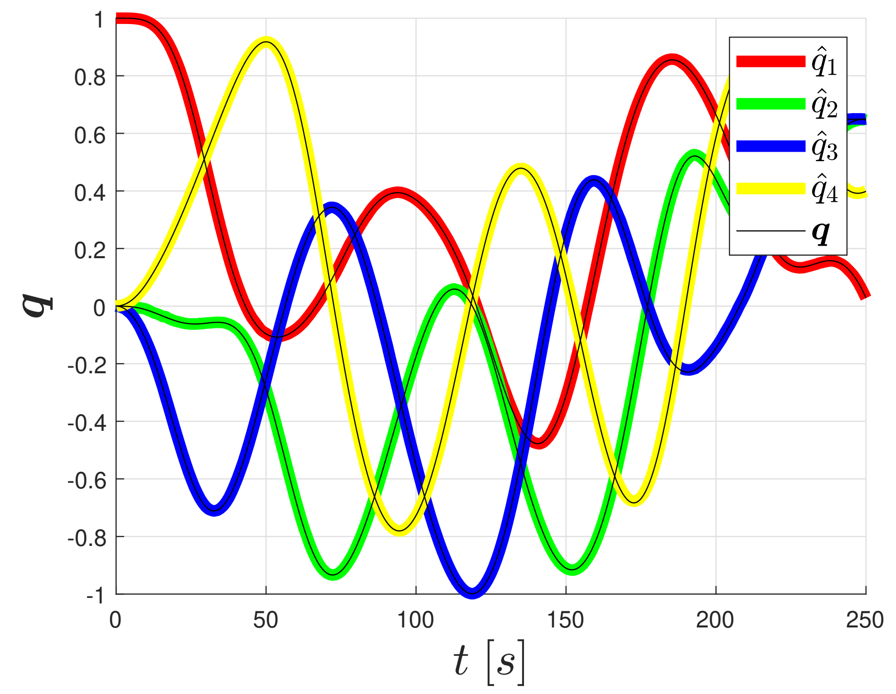
All the three filters described in this chapter have been tested at first with a dummy simulator which generates fake sensor measurements. The purpose of this kind of simulation is purely to verify the correctness of the mathematical equations implemented, debug errors and check on the whole the correct working capability of the filters. The advantage of this approach is being able to operate with a Simulink model which is much lighter and faster with reference to the complete DANCER simulator, that can eventually be tested on a second moment once filtering process has already been verified and the optimal tuning parameters have been computed. Nevertheless, this kind of simulations can test the right mathematical behaviour of the filter but can not constitute a proof that the attitude estimation algorithms are working well also in real life, but for the first-mentioned purpose the present test is considered sufficient; simulation with the complete DANCER model could be an additional feature, but not strictly necessary from this point of view.

The dummy simulator generates noiseless values for the angular rate, used to integrate the Euler equations describing the dynamics of an ideal body and retrieve the true attitude matrix, which is used as a reference. Angular velocity signal is then modified adding the presence of white noise and given in input to the implemented filters, together with simulated accelerometer and magnetometer values obtained in the same way; values used for the noise power of accelerometer and magnetometer σ_{acc} and σ_{mag} can be retrieved from Tab. (4.5), while for the gyroscope is computed equivalently obtaining $\sigma_{gyro} = 4.320 \cdot 10^{-4}$ *rad* (see Sec. 4.5). The output attitude matrix and angular rate can then be compared to the true ones, permitting to evaluate the general behaviour of the filter. Main results are here reported for all the three filters: Fig. (4.7) for Mahony, Fig. (4.8) for Madgwick and Fig. (4.9) for Kalman.

Analysis of the results shows no particularly big differences among the three filters about the attitude quaternion estimation; the angular rate presents instead different amplitudes of oscillations around the true value. Filters output can surely be adjusted with the tuning gains, but it makes no sense in this context since the simulation is different from the real case, and the purpose was here to verify only the correct reconstruction of the wanted quantities, an objective that has been achieved with successful results.

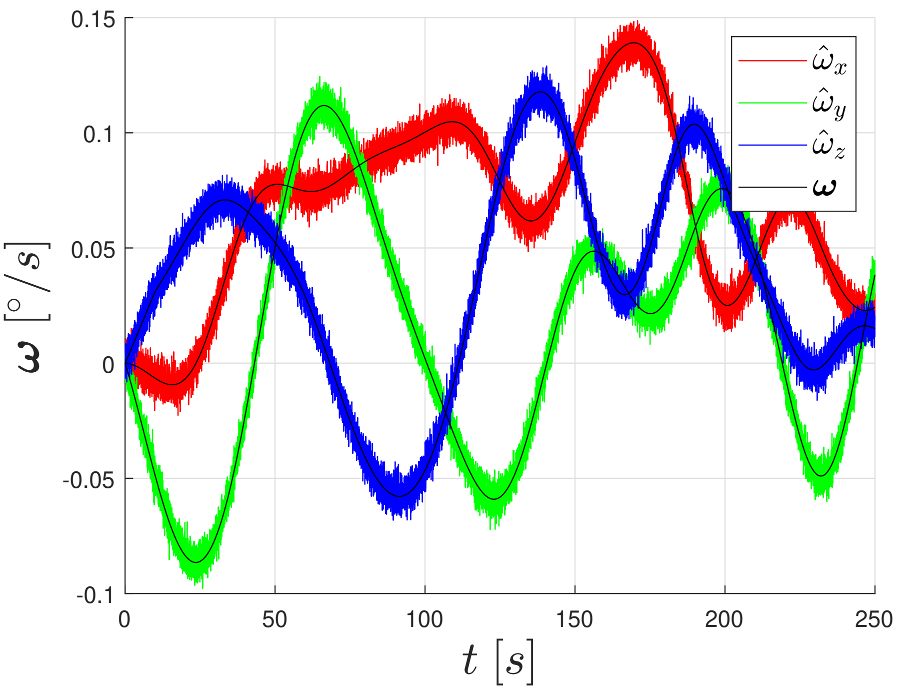


(a) Angular rate

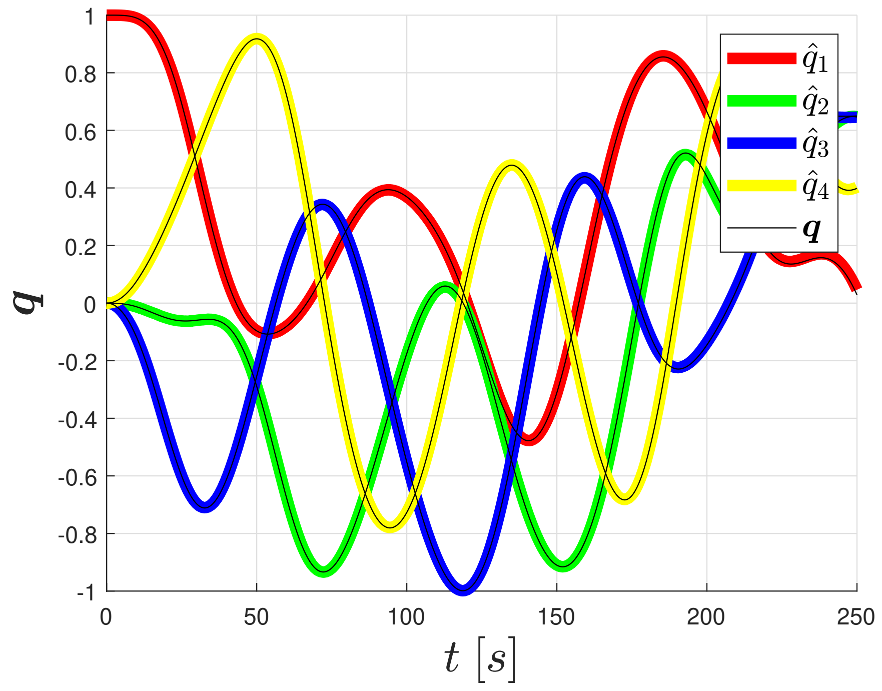


(b) Attitude quaternion

Figure 4.7: True (in black) vs. estimated (indicated with the *hat*) quantities resulted from the simulation with the Mahony filter.

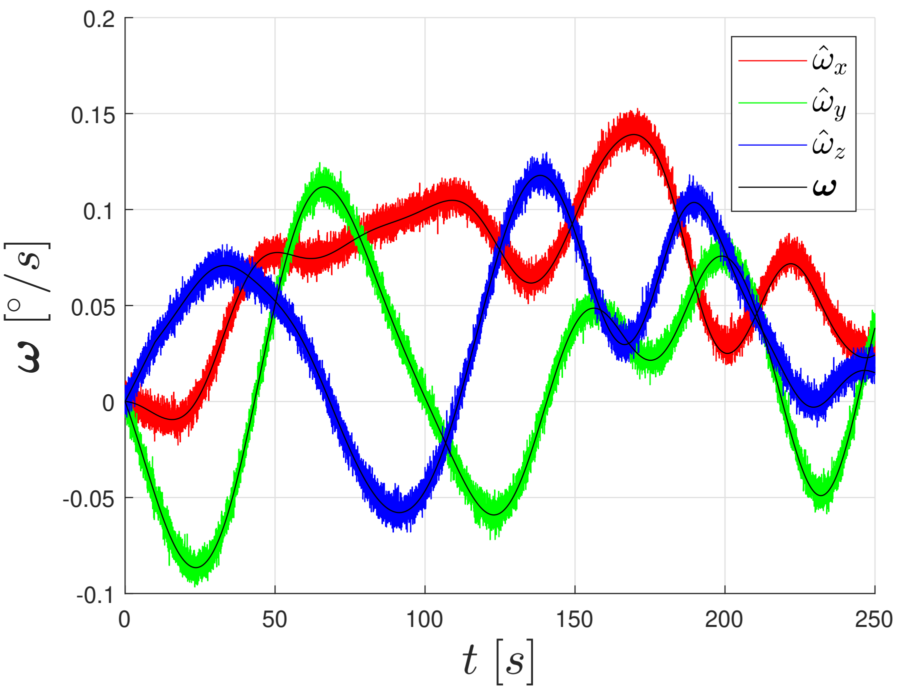


(a) Angular rate

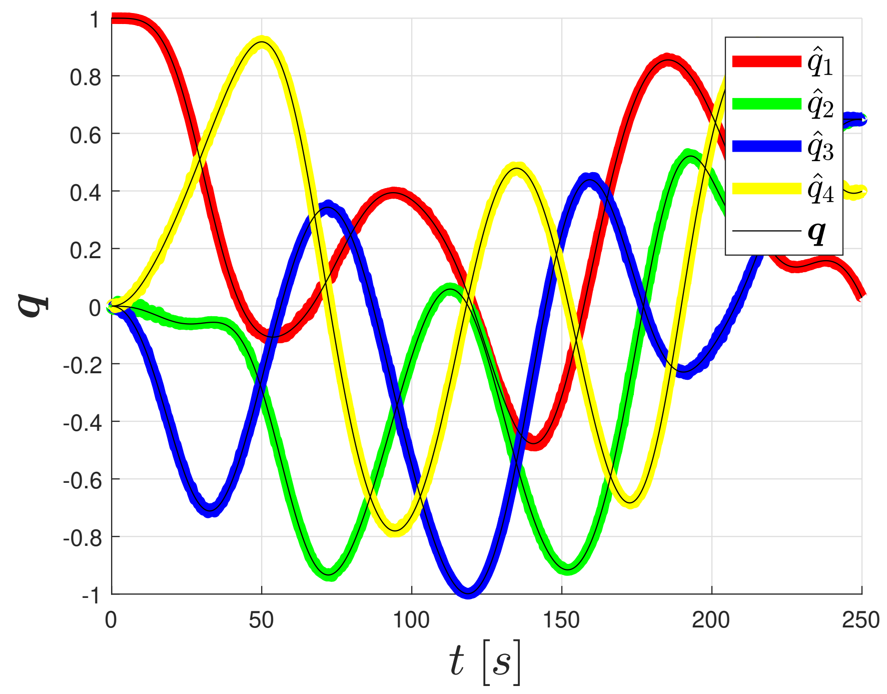


(b) Attitude quaternion

Figure 4.8: True (in black) vs. estimated (indicated with the *hat*) quantities resulted from the simulation with the Madgwick filter.



(a) Angular rate



(b) Attitude quaternion

Figure 4.9: True (in black) vs. estimated (indicated with the *hat*) quantities resulted from the simulation with the Kalman filter.

4.7 Tuning & Optimization

Each filtering algorithm needs to be tuned assigning to the specific gains some values which ensure at first that the considered filter is working properly when used in combination with the set of sensors used, and secondly that its behaviour is optimal in terms of error minimization. In particular, the following features are searched:

- Fast convergence and precise following: algorithm has to converge to the right output values (evaluated with respect to an absolute reference) as fast as possible, in order to be able to follow the body orientation with small errors even when sudden changes in position are performed. This feature can be easily spotted by moving randomly the device and bringing it back to the initial position, evaluating how fast the Euler angles are converging to the null value. This kind of behaviour is in general tuned by the means of the proportional gain: increasing its value the rate of convergence increases, but could lead the filter to diverge when fast movements are performed.

- Stability over time: correct following and convergence imply that values are not diverging from the expected value even with long simulations, because of possible problems of error accumulation.

- Little oscillations: convergence to a specific value or following a parameter variation can be not direct, but instead some oscillations around the final value can be present and should be minimized as far as possible.

- Low drift: bias drift errors must be taken into account with on-line estimation of the gyroscope bias in order to be able to correct its drift and minimize the error.

Unfortunately, no particular procedures exist in order to identify the optimal gains and thus a pragmatic approach must be adopted as suggested in [7, 6, 24]. Trial & error has, in this case, revealed to be not accurate enough, especially for the Mahony filter which is characterized in its general version by four tuning parameters that increase considerably the number of possible combinations. For these reasons, the algorithms have been implemented in MATLAB, in order to be able to run the simulation several times with different values combinations and find the one that best fits the imposed constraints. The complete optimization process is described hereafter.

The first important aspect is to define a reference trajectory that represents the expected real attitude of the body, in order to have a "true" reference to which the output of the filter can be compared, permitting in this way to evaluate the estimation error; minimization of this last parameter is the main driver for the optimal gains searching procedure. It is a common practice to quantify the filter effectiveness by computing the error in terms of the root means square (RMS) value [6, 24]: indicating with α_i a generic Euler angle, the RMS is found with Eq. (4.25), i.e. considering the difference between the expected value for the particular angle and the one obtained from the filter output.

$$RMS_{error} = \sqrt{\frac{\sum_{i=1}^{n} \left(\alpha_{i,filter} - \alpha_{i,reference}\right)^2}{n}} \tag{4.25}$$

The real issue in this procedure is to determine the reference track, in fact for this purpose an apparatus for attitude determination preciser than the filtering algorithm would be required. At the state of the art, one effective approach that is used [6, 24] is to exploit an optical system that, by the means of some visual markers applied to the observed body, is capable of determining its dynamical orientation with very high precision. This kind of technology has not been developed yet at the DAER laboratory, so dynamical referencing is not possible; performance of the filter can be evaluated only when the sensors are in a static position which orientation can be estimated with reference to an initial direction. The adopted referencing sequence is the following, indicating with θ, ϕ and ψ respectively a rotation around x, y and z:

1. The TI LaunchPad equipped with the sensors is switched on (or reset if it was already operating);
2. The device is kept still for a minimum period of time given by $t_{ref} = 10\ s$, during which the initial reference direction for gravitational and magnetic fields are computed;
3. After t_{ref} has passed, the MATLAB routine for real-time serial reading (Sec. 2.6.1) is executed and measurements of gyroscope, accelerometer and magnetometer begin to be recorded;
4. The expected output computed in terms of Euler angles in the initial position is set to 0° for all θ, ϕ and ψ;
5. The device is rotated alternatively to $+90°$ and $-90°$ around each axis. On each position it is let still for a minimum amount of time of 5 s, and the consequently expected value for the relative angles is set;
6. The device is brought back to the initial position, so the expected output is again null for all the three Euler angles; the device is let in this position for a longer period of at least 30 s, in order to ensure and verify (in post-processing) that the filter converges back to zero.

After having completed the procedure above described all the measurements of the sensors are recorded and, knowing the time history of the physical position of the device, references are set for each configuration; Fig. (4.10) shows the sensors output for the particular simulation used for tuning. As mentioned, it is possible to define a "true" value only for the selected positions, since in all the other time instants (i.e. when the device is moving or in a position different from 90°) the real orientation is unknown and can be determined only if an external apparatus is used. The issues to be faced in order to define the $\pm 90°$ are the same encountered during the accelerometer calibration procedure (Sec. 3.4.3) so a similar solution is adopted (Fig. 3.3). Since the tolerance by which rotation of 90° is characterized can not be quantified with precision, even the resultant RMS_{error} value can not be considered as an absolute parameter to evaluate the filter performance, but only examined as a general indicator of the filter behaviour; only the RMS_{error} computed on the initial position (with null Euler angles) can be trusted as a "real" value, but it is valid only for this restricted and static case. Nevertheless taking an angle as reference does not mean to strictly impose that the filter output must be exactly equal to that value in a particular position, but rather it means to choose the filter gains

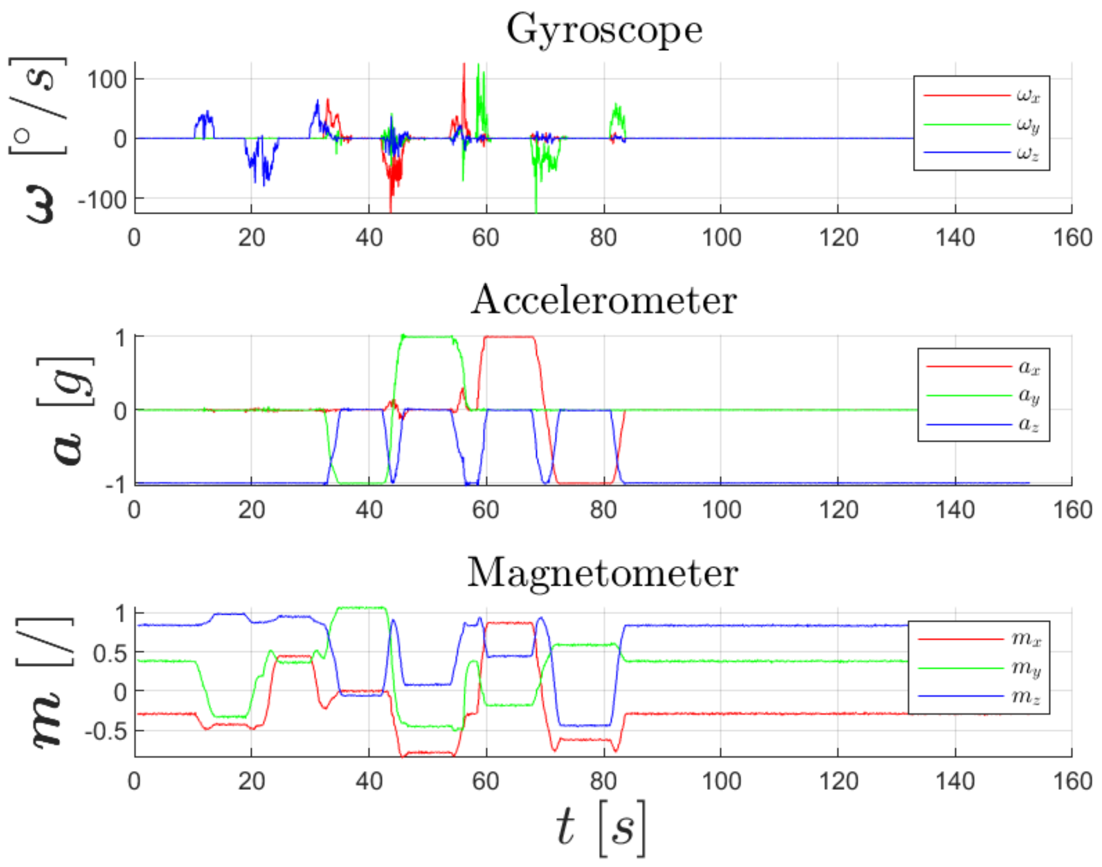


Figure 4.10: Sensor measurements collected during the experimental test in which the device has been rotated alternatively to ±90° around each axis and used to perform the tuning procedure (note that magnetic field lacks of unit of measurement, since its output is normalized).

that better approximate the imposed behaviour; so if the filter is working properly, it should converge to the right angle value even if it is a bit different from the reference.

The filter that has to be tuned, implemented within a MATLAB script, is then executed many times using in input the collected data from the sensors together with the gravitational and magnetic fields initial directions; Euler angles obtained from the filter output and corresponding to the time intervals in which the position of the device is known are compared to the reference track previously set and the RMS_{error} value is computed. Each time the filter is run, a different combination of the tuning gains is used and a different error value is retrieved; by choosing the parameters that lead to the minimum error, the optimal tuning is found, with the explained procedure limits. Two approaches have been tested to solve the minimization problem:

- The first one consists in manually setting the ranges for the gains to be tested by defining a vector containing a set of values for each parameter, trying all the possible combinations, saving the RMS_{error} and directly finding the minimum. This approach has turned out to be not effective at all: the procedure is very slow (especially with four tuning parameters) and the gains tested are limited not only by the range imposed but also by the specific values tested.

- The adopted approach for the current minimization problem solution exploits the use of a genetic algorithm, that results to be way more efficient. Even in this case the initial range of values must be specified for each parameter set; different simulations are thus performed successively refining the imposed domain. The biggest limitations of using a genetic algorithm are that at first there is no mathematical certainty that it will converge to the global minimum, and secondly that even repeating the simulation with the same imposed ranges the minimum value found may vary; it is anyway an efficient approach as shown by the lower error found at each successive attempt.

Results about each different filter for the most significative iterations performed with the genetic algorithm are reported on the next sections.

From the Euler angles representation reported hereafter as a result of the application of each filter it can be noticed that when θ approaches a value $\pm 90°$, the attitude estimation is not reliable and in fact the other two angles ϕ, ψ show a different behaviour from the expected one, since they should be kept to a zero value (Fig. 4.11, 4.14, 4.16, 4.17). This phenomenon is known as "gimbal lock" which consists in the loss of a degree of freedom when two of the sensor axis are in a parallel configuration. Nevertheless, this issue is not a concern for DANCE vehicle because mathematically it affects only the Euler angle representation of the attitude, that is characterized by the presence of singularity conditions; using quaternions singularities are removed. Secondly, DANCE attitude platform will never reach pitch angles around 90°, since it is physically impossible for the way the vehicle is designed (Sec. 1.2), and it is moreover not required for operative purposes.

4.7.1 Mahony Filter Results

The tuning parameters for the Mahony filter were previously discussed and can be found on Tab. (4.1). Results obtained executing the genetic algorithm are now exposed; several iterations and attempts with different values range has been done, but only the most significative are here reported in Tab. (4.6). Initial values were decided both by a trial & error approach and by looking at the typical results obtained in other applications [6, 24]. Instead Fig. (4.11) shows the reconstructed attitude in terms of Euler angles and quaternions concerning the performed test, for which optimal gains relative to the last iteration have been used; the estimated angular velocity is reported as well.

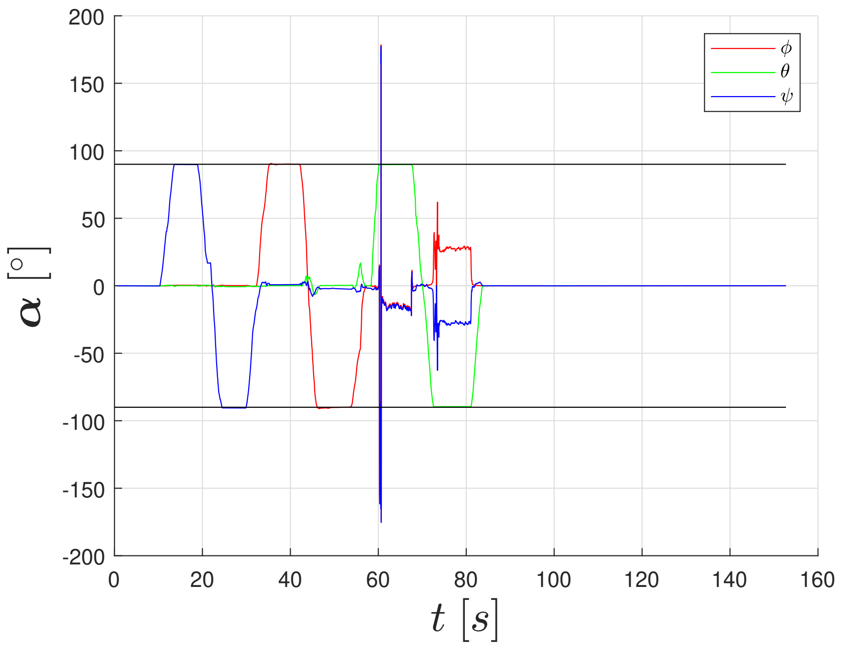
Even if not perfect, the filter shows a good trend, which is the best result that has been obtained after a lot of attempts. This is nevertheless valid if very slow movements are performed and the assumption made in Sec. (4.3) is verified, according to which accelerometer can be approximated to measure only the gravity acceleration; otherwise, estimation errors are bigger and convergence to the right value (for example bringing the device back to the null Euler angles position) is very slow. This is explained by the fact that the proportional gain is kept low since DANCER operative conditions will be characterized by very small angular rates; even this small value causes the filter to have some oscillations, so higher values will probably have a worse effect.

The main reason of not having a perfect behaviour is probably to be searched in the weakness of the procedure itself, that is able to give only some references on selected static positions; to achieve better results an external apparatus such as an optical system is surely required. Of course, other sources of errors are contributing as well, like sensor

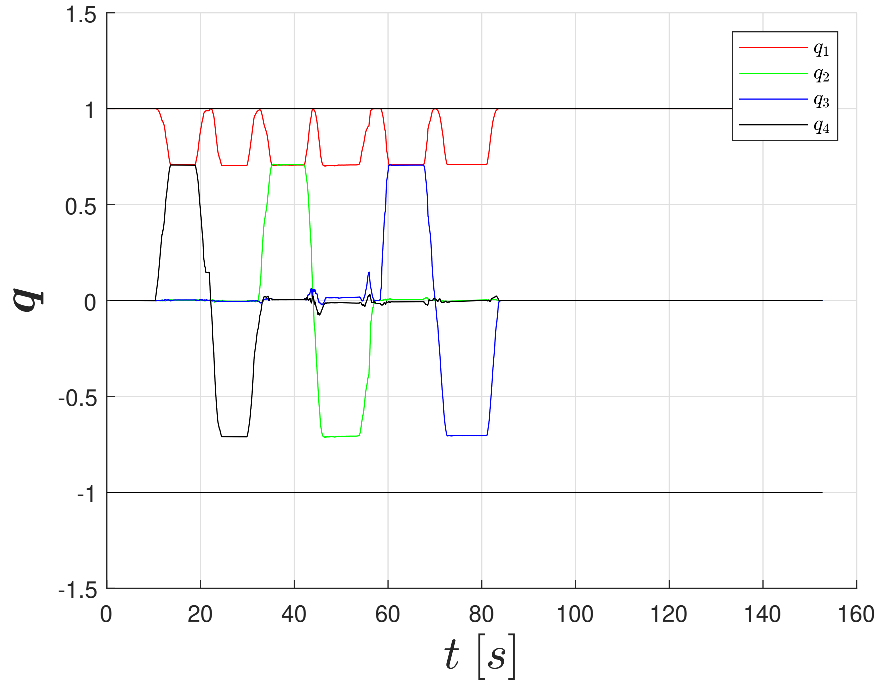
	Parameter	Range	Result
Iteration 1	K_p	[0.0 5.0]	1.9070
$RMS_{error} = 0.4701$	K_{int}	[0.0 1.0]	0.1009
	K_a	[0.0 6.0]	5.9999
	K_m	[0.0 1.0]	$4.3452 \cdot 10^{-3}$
Iteration 2	K_p	[0.5 2.0]	1.2170
$RMS_{error} = 0.4554$	K_{int}	[0.0 1.0]	$6.5434 \cdot 10^{-2}$
	K_a	[5.0 10.0]	7.9964
	K_m	[0.0 1.0]	$6.6631 \cdot 10^{-3}$
Iteration 3	K_p	[0.5 1.0]	0.8985
$RMS_{error} = 0.1117$	K_{int}	[0.0 0.1]	$9.5830 \cdot 10^{-2}$
	K_a	[6.0 10.0]	7.3636
	K_m	[0.0 0.1]	$1.3989 \cdot 10^{-2}$

Table 4.6: Mahony filter tuning parameters results for different successive iterations.

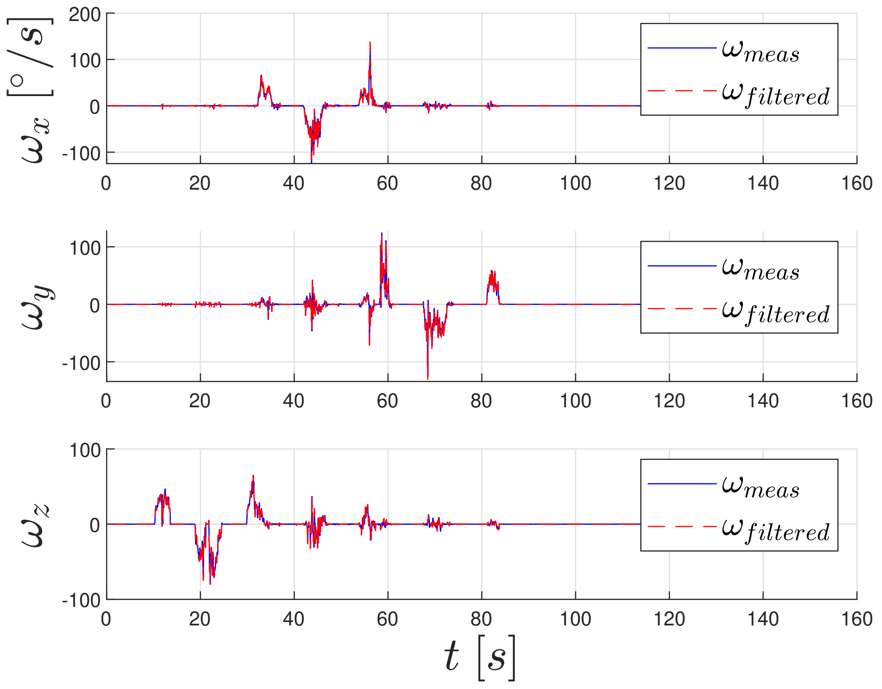
inaccuracy, experimental errors, noise disturbance, calibration errors, temperature and supply voltage sensibility. From Fig. (4.11) it is not visible, but with a close look the yaw component of Euler angles ψ seems to diverge from the null position, when in reality it is oscillating around it with a very long period, as demonstrated from a longer simulation reported in Fig. (4.12) in which it is evident that the filter does not diverge, proving instead that it is working properly; anyway it shows many oscillations, highlighting that the overall performance can surely be improved by adopting the suggested more rigorous procedure. This last test has been performed by recording in real-time the output values directly from the software deployed to the TI LaunchPad.



(a) Euler angles



(b) Quaternions



(c) Filtered vs measured angular rate

Figure 4.11: Attitude reconstruction and angular rate estimation using the Mahony filter with the optimal gains found on last iteration.

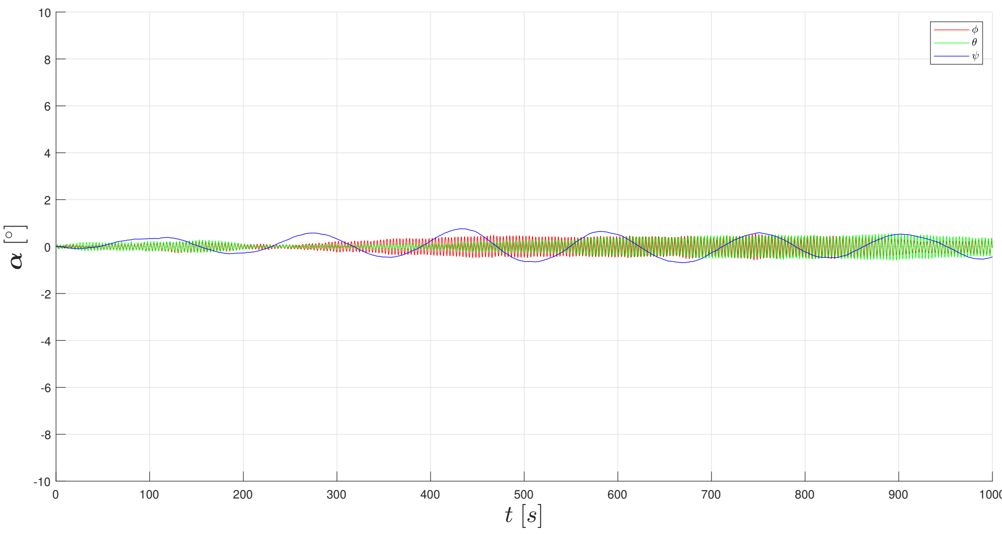


Figure 4.12: Mahony filter stability test with the device kept in still position for a long period of 10000 s ($\approx$ 16 min), oscillations around the null equilibrium position are evident, ψ presents a longer period with reference to ϕ, θ.

4.7.2 Madgwick Filter Results

The Madgwick filter behaviour can be adjusted with the tuning gains explained on Tab. (4.3) and optimization results are here reported. Ranges for the first iteration were appositely chosen to be very large with reference to the expected value of β and ζ (<1) in order to test the general behaviour and verify the correctness of expected results. Using two parameters it is possible to visualize the error surface obtained with different combinations of gains (Fig. 4.13), that shows the complexity of the problem domain and the tendency to have a minimum error towards little values of β and ζ, as proven by the optimization results.

As a general trend, with respect to Mahony the filter, the algorithm shows a bigger error in estimating the right orientation value, especially for the pitch and roll axes (θ and ϕ) but is characterized by greater stability over time as demonstrated by the long simulation test reported in Fig. (4.15), obtained using the software downloaded to the TI LaunchPad and recording output values. Error minimization using the modified version with the two additional gains has shown to not change in a sensible way the filter behaviour since comparable results are obtained as can be noticed from the value of the RMS_{error} (Tab. 4.7 versus Tab. 4.8) and from the reconstructed quaternions and Euler angles (Fig. 4.14 versus Fig. 4.16); for the modified version only the reconstructed attitude is reported since the angular rate comparison is not significative. In this case, also a different approach has been tried: since the Mahony filter has a better behaviour in terms of the error when the Euler angles approach ±90°, the Mahony output quaternions have been taken as an absolute dynamical reference for the tuning; the minimization driver is thus the relative error with reference to the reference reconstructed attitude. Even with this approach the results obtained were comparable to the other two cases and can be found in Appendix B.1.

The same considerations exposed for the Mahony filter tuning are valid even in this

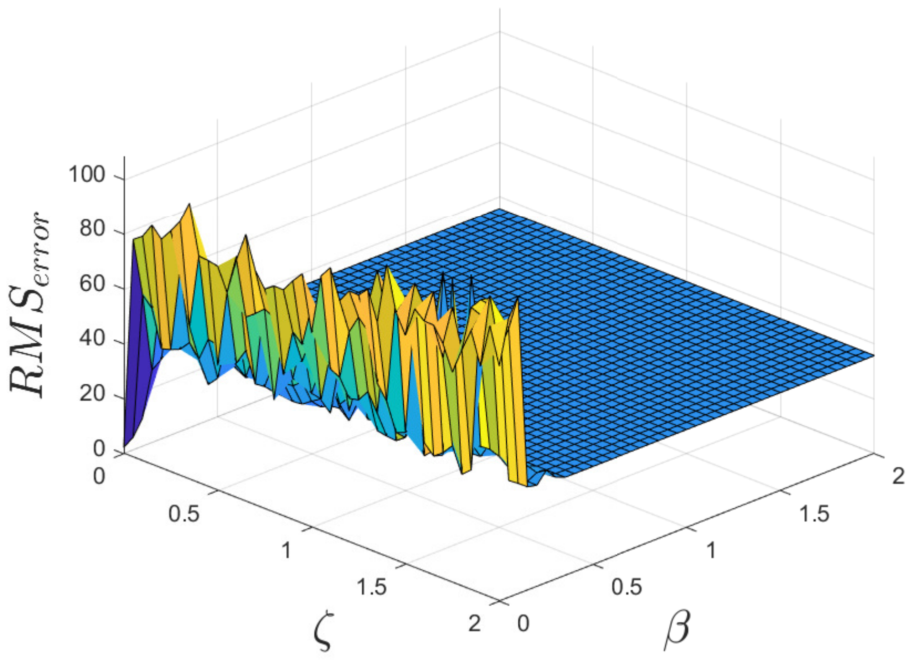
	Parameter	Range	Result
Iteration 1	β	[0.0 2.0]	$1.1577{\cdot}10^{-1}$
$RMS_{error} = 20.6884$	ζ	[0.0 2.0]	$3.1698{\cdot}10^{-2}$
Iteration 2	β	[0.0 1.0]	$9.3899{\cdot}10^{-4}$
$RMS_{error} = 0.9550$	ζ	[0.0 1.0]	$1.0037{\cdot}10^{-8}$
Iteration 3	β	[0.0 10^{-2}]	$9.3897{\cdot}10^{-4}$
$RMS_{error} = 0.9550$	ζ	[0.0 10^{-4}]	$1.0132{\cdot}10^{-8}$

Table 4.7: Madgwick filter (in original version) tuning parameters results for different successive iterations (note that iterations 2 and 3 are very similar).

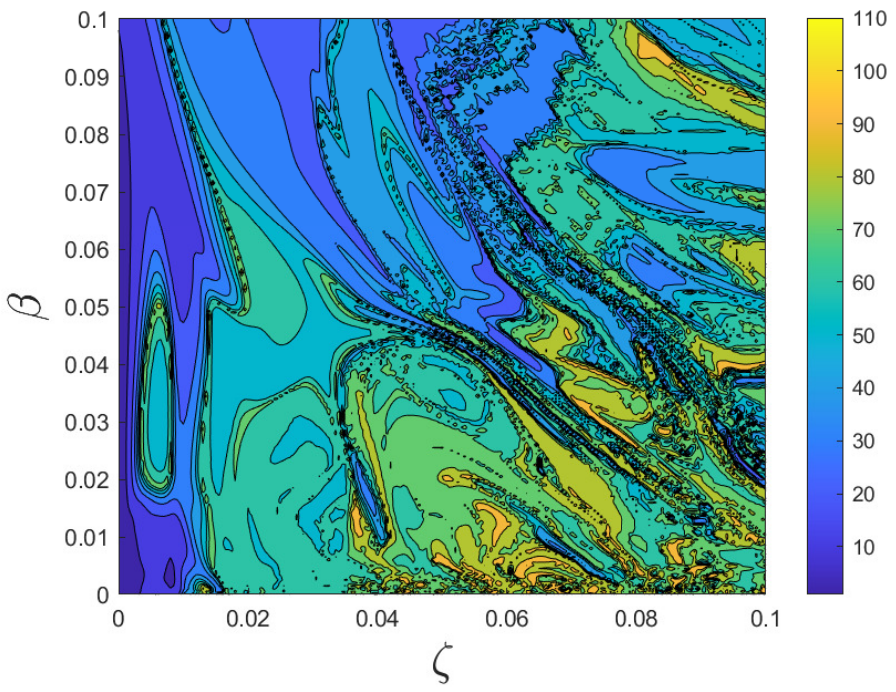
	Parameter	Range	Result
Iteration 1	β	[0.0 2.0]	$8.9379{\cdot}10^{-4}$
$RMS_{error} = 0.9636$	ζ	[0.0 2.0]	$6.9097{\cdot}10^{-6}$
	K_a	[0.0 5.0]	1.4072
	K_m	[0.0 2.0]	0.9137
Iteration 2	β	[0.0 1.0]	$9.7653{\cdot}10^{-4}$
$RMS_{error} = 0.9555$	ζ	[0.0 0.1]	0.0000
	K_a	[0.0 2.0]	1.0890
	K_m	[0.0 1.0]	$3.9063 \cdot 10^{-3}$
Iteration 3	β	[0.0 10^{-3}]	$9.3810{\cdot}10^{-4}$
$RMS_{error} = 0.9550$	ζ	[0.0 10^{-5}]	$8.8619{\cdot}10^{-8}$
	K_a	[0.5 1.5]	1.3542
	K_m	[0.0 1.0]	0.9682

Table 4.8: Madgwick filter tuning parameters results for different successive iterations, introducing the two additional gains for accelerometer and magnetometer measurements K_a, K_m.

case: to find a truly optimal combination of the parameters a more precise and dynamical reference is needed, obtained for example by the means of an optical tracking system.

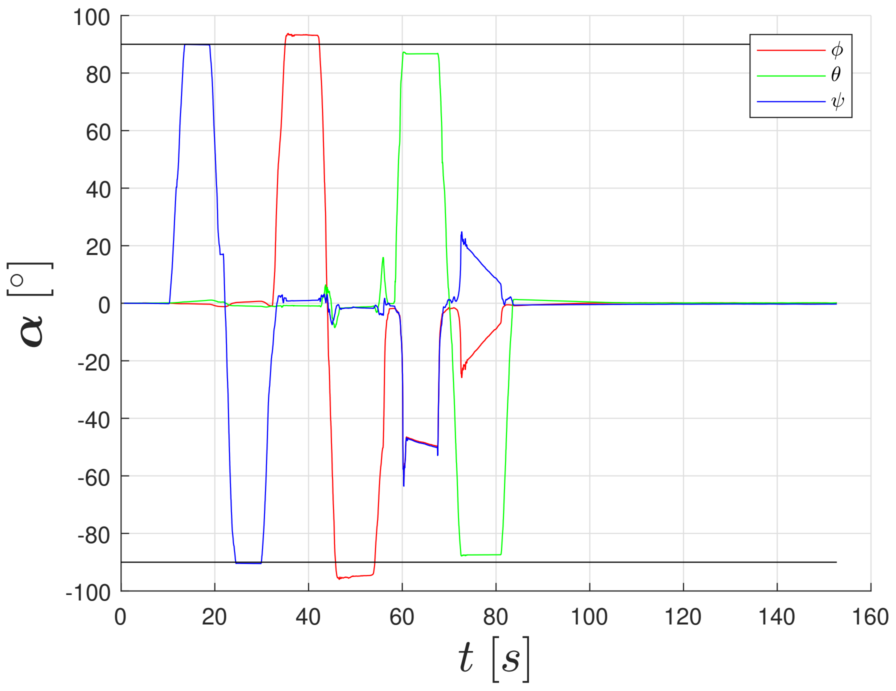


(a) Surface plot for β, $\zeta \in [0\ 2]$

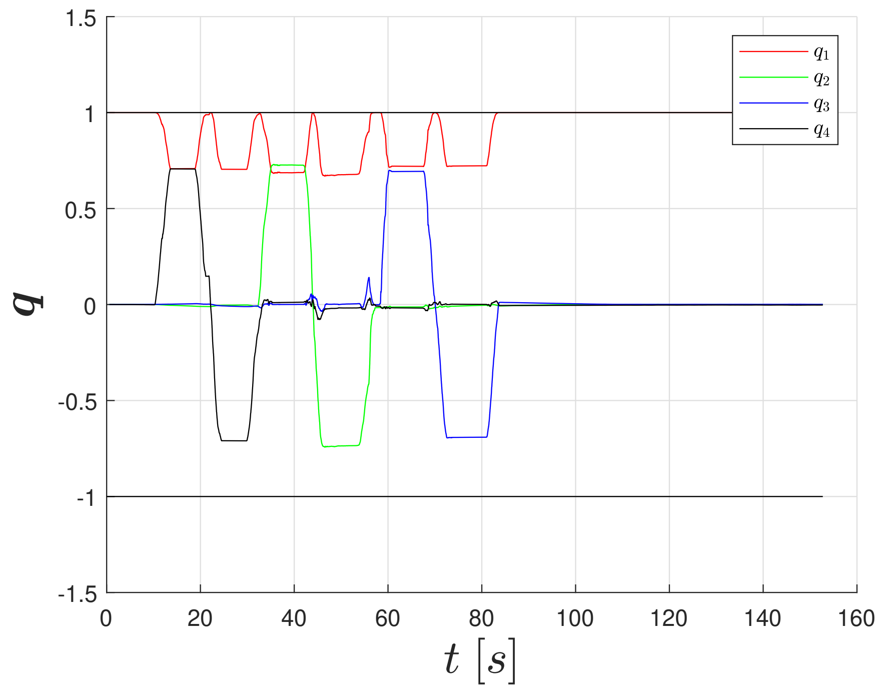


(b) Contour plot refinement for β, $\zeta \in [0\ 0.01]$

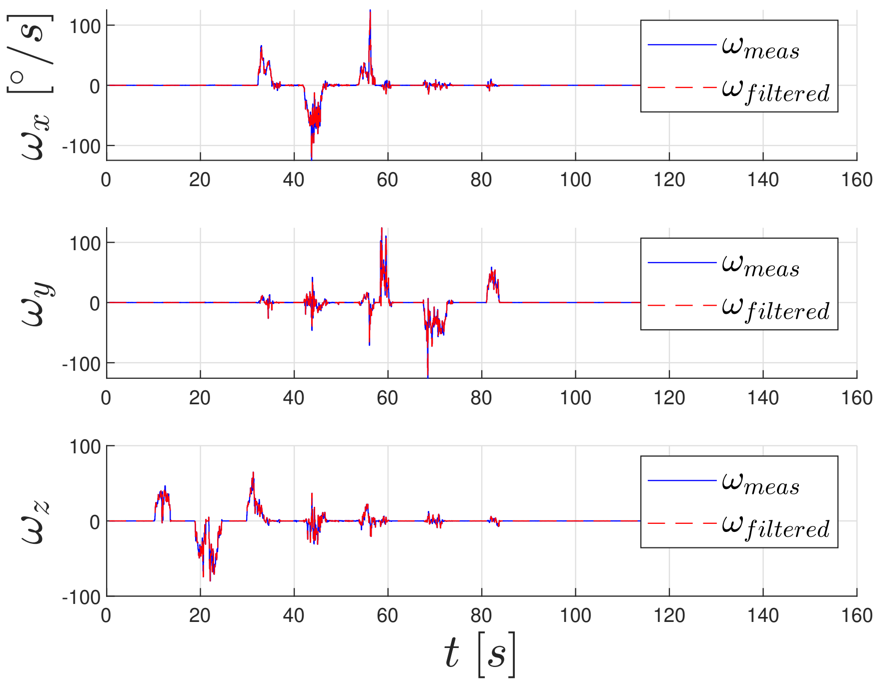
Figure 4.13: Locus of points obtained plotting the RMS_{error} for different combinations of gains β, ζ tuning the Madgwick filter; minimum error is found for low values of both parameters.



(a) Euler angles



(b) Quaternions



(c) Filtered vs measured angular rate

Figure 4.14: Attitude reconstruction and angular rate estimation results using the Madgwick filter with the two optimal gains β, ζ.

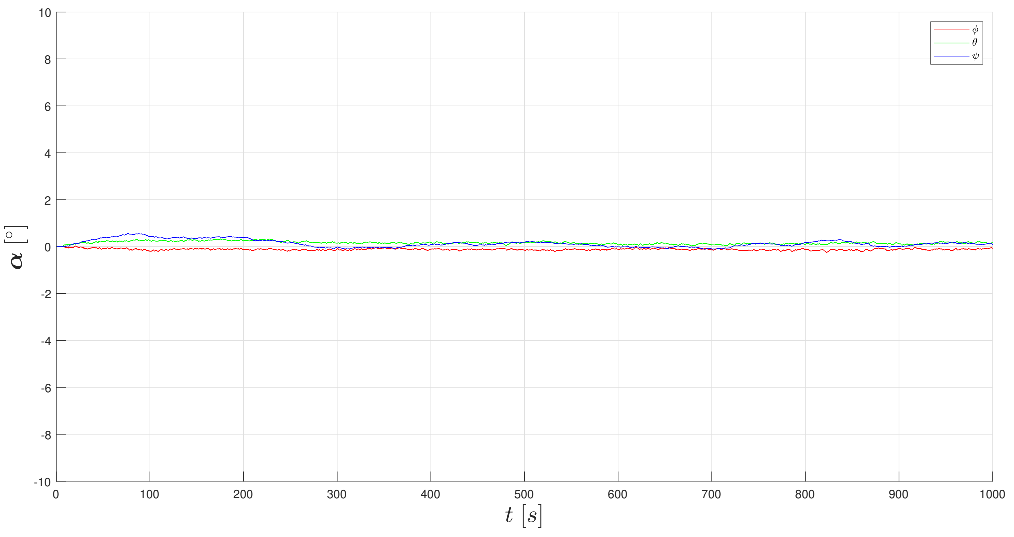
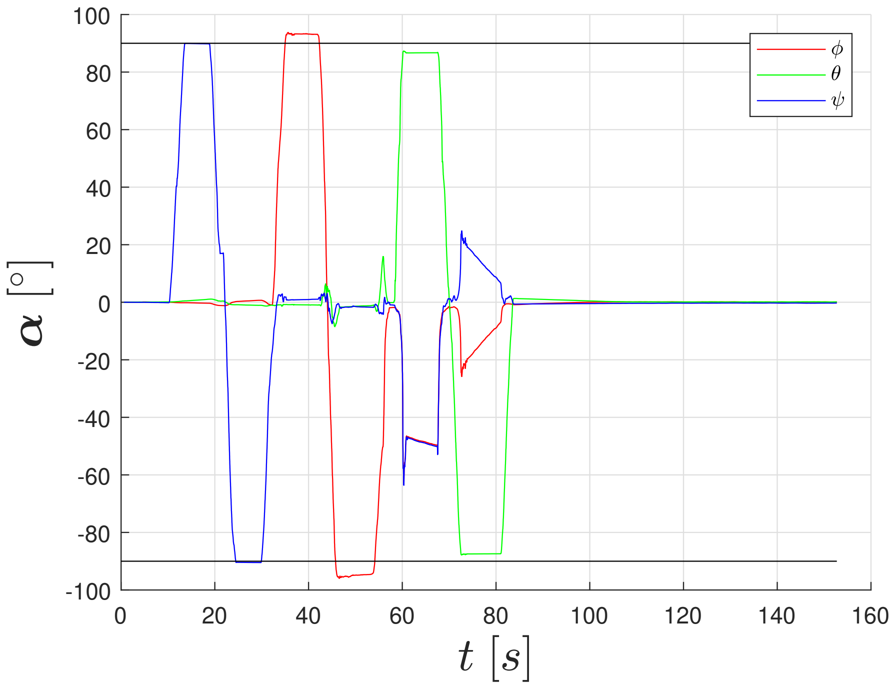
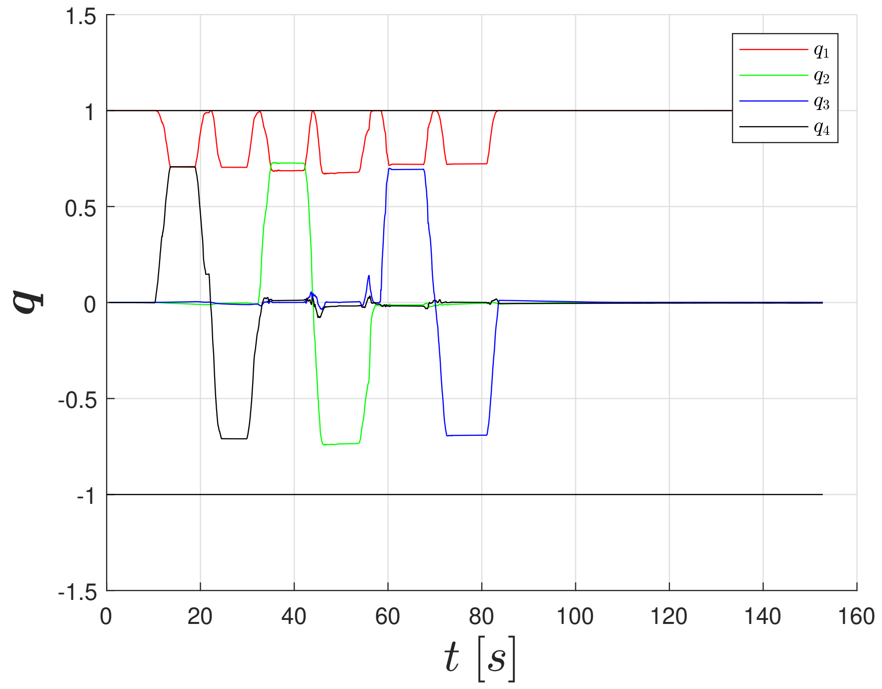


Figure 4.15: Madgwick filter stability test with the device kept in still position for a long period of 10000 s ($\approx$ 16 min).



(a) Euler angles



(b) Quaternions

Figure 4.16: Attitude reconstruction results using the Madgwick modified version with the four optimal gains β, ζ, K_a, K_m; no particular improvements are found with reference to the original version.

4.7.3 Kalman Filter Results

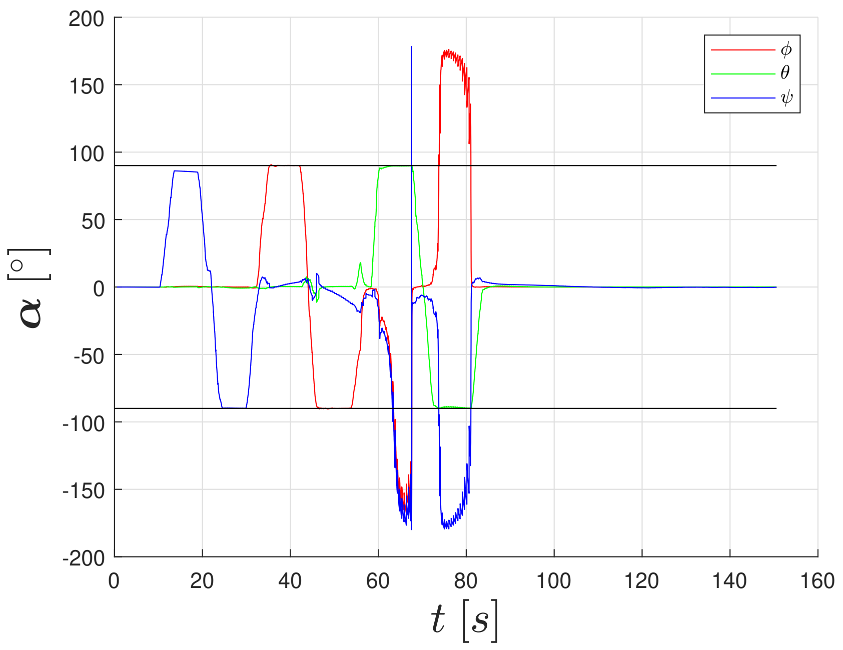
The gains to be found in order to optimize the Kalman filter are reported on Tab (4.5); as mentioned before, the noise variances relative to sensors measurements σ_{acc} and σ_{mag} can be estimated directly from the sensors' datasheets, while the other two parameters $\sigma_{\eta,v}$ and $\sigma_{\eta,u}$ are unknown and must be computed with the optimization process. In reality, what is found more convenient is to tune all the four parameters directly with the optimization process, for two main reasons: the first one is that the values found on the datasheet could be not precise, while the second is about the fact that when real measurements are taken other environmental effects and experimental errors can interfere, leading to very different results from the expected ones. Finally, and most importantly, overall better results are obtained. The gains values found on the last iteration are in fact several orders of magnitude bigger with reference to the predicted ones, but initially expected values are only theoretical and do not constitute a constraint.

The only algorithm used for the Kalman filter is the Simulink code that was already available at the beginning of the present work; the genetic algorithm is implemented to run the simulation directly from Simulink, even if this approach is way more inefficient in terms of computational time with reference to using a MATLAB version of the algorithm. A final remark has to be made about the referencing used for this optimization procedure: because of the aforementioned gimbal lock problem, in this case some iterations of the genetic algorithm result to be ill-conditioned and cause the simulation to stop. For this reason, the Kalman filter is tuned taking as reference the attitude quaternion estimated with the Mahony filter, which gave acceptable results. This approach is to be considered as equivalent to the other proposed procedure which uses a direct Euler angles reference, since both strategies were tested during the Madgwick filter tuning, obtaining no noticeable differences in results, as already explained and proven in Sec. (4.7.2). Reconstructed attitude and angular velocity are reported in Fig. (4.17), while the stability test (analogously to previous sections) is reported in Fig. (4.18).

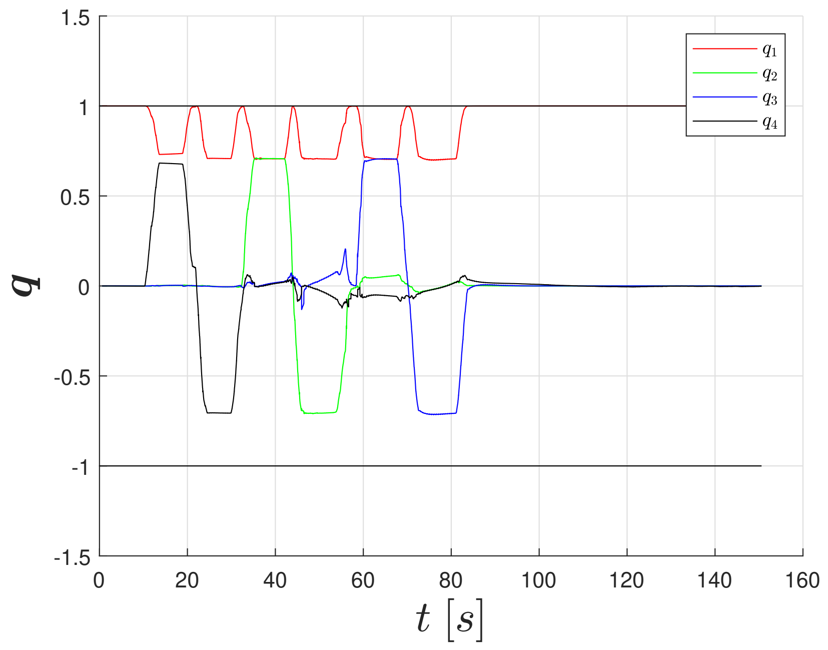
	Parameter	**Range**	**Result**
Iteration 1	$\sigma_{\eta,v}$	[0.0 2.0]	0.7627
$RMS_{error} = 0.7710$	$\sigma_{\eta,u}$	[0.0 2.0]	$5.9280{\cdot}10^{-3}$
	σ_{acc}	[0.0 5.0]	0.9078
	σ_{mag}	[0.0 5.0]	2,2298
Iteration 2	σ_v	[0.0 1.0]	0.6861
$RMS_{error} = 0.7516$	σ_v	[0.0 10^{-2}]	$11241{\cdot}10^{-3}$
	σ_{acc}	[0.0 1.0]	0.7568
	σ_{mag}	[0.0 3.0]	1,8246
Iteration 3	σ_v	[0.0 10^{-2}]	$6.0036{\cdot}10^{-4}$
$RMS_{error} = 0.0192$	σ_v	[0.0 10^{-2}]	$1.0010{\cdot}10^{-4}$
	σ_{acc}	[0.0 10^{-2}]	$4.8120{\cdot}10^{-3}$
	σ_{mag}	[1.0 4.0]	3.8001

Table 4.9: Kalman filter tuning parameters results for different successive iterations.

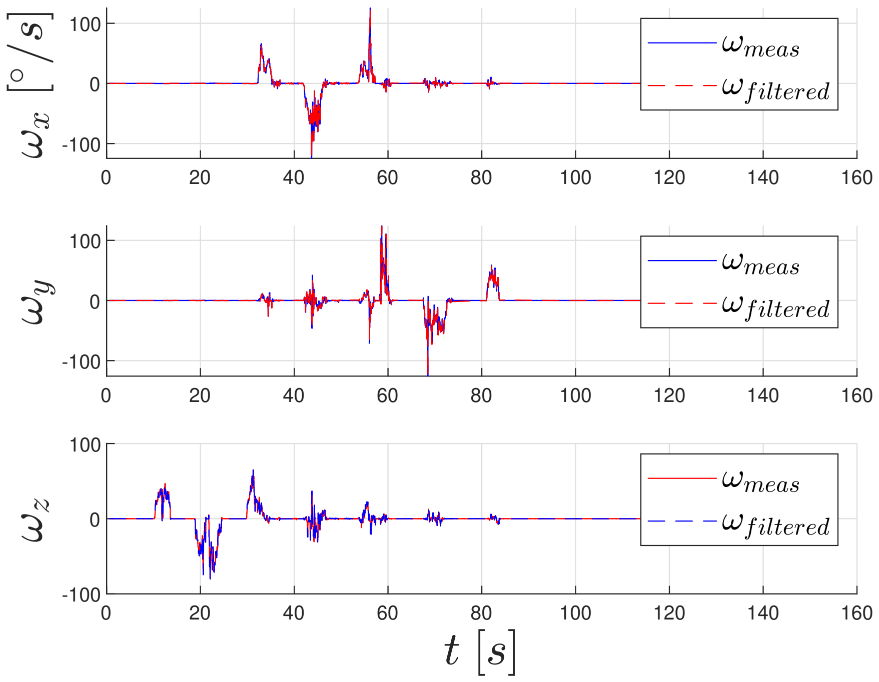
Attitude estimation results are not as good as expected from a Kalman filter. Besides the tuning procedure limits, there is another important cause: the *update* part of the algorithm should be performed each time step to have the best behaviour possible, but the computational cost is really high since it involves the inversion of a 6x6 matrix for the optimal gain computation. The update step is in fact imposed to be performed each 50 time steps; the BOOSTXL sensors are set to work with an ODR of 12.5 to 25 Hz so it is equivalent to assert that the new gain is updated after 7 to 13 time steps, according to the measurements update. A lower update interval is hard to use for performance reasons.



(a) Euler angles



(b) Quaternions



(c) Filtered vs measured angular rate

Figure 4.17: Attitude reconstruction and angular rate estimation results using the Kalman filter with the optimal gains found on last iteration.

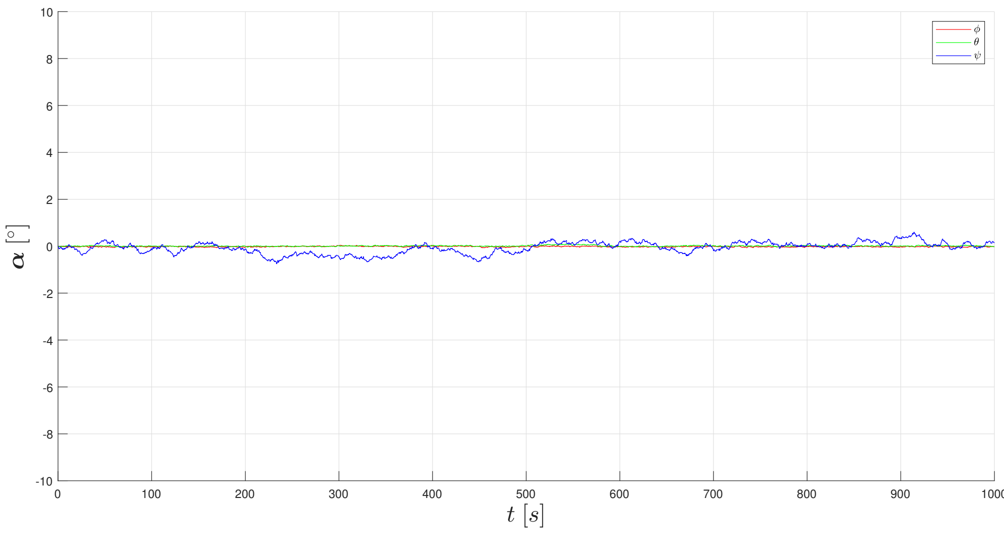


Figure 4.18: Kalman filter stability test with the device kept in still position for a long period of 10000 s ($\approx$ 16 min).

4.8 Conclusions & Further Development

Precise attitude estimation is not a trivial task, especially when dealing with low-cost MEMS sensors that provide noisy and inaccurate measurements. A big variety of filtering algorithms exist at the state of art, three of the most common choices for this kind of applications has been implemented, tested and optimized in order to be used with the DANCER hardware. All the three filters have shown to work properly inside the simulator environment, giving acceptable results even without a particular precision in tuning. In the real case, i.e. when used with the BOOSTXL sensors pack and the TI LaunchPad, the behaviour is appreciably different and a rigorous optimization process, like the one proposed in this work, is needed to have good estimations. The limitations of the results obtained have been remarked more than once: the optimal gains that have been found are computed considering a static reference, meaning that filters are optimized to converge to a known value when moved and set in a still position. The risk of the proposed procedure is in fact that the optimal gains for convergence are found, but they may be not optimal when the body is moving since the behaviour in the dynamical case could be different and not as good as expected. Dynamic referencing is surely a desirable feature, but need the use of a dedicated optical facility for movements-tracking. Anyway, the process of filtering with MEMS sensors itself could result to be too inaccurate for testing GNC algorithms; cameras and optical trackers may be, in fact, used to directly reconstruct the attitude of the vehicle, as suggested in previous works [4, 5].

After the optimization process, all the tested filters have demonstrated to be able to estimate the attitude of the body, but some aspects can be highlighted, as a general trend of their behaviour:

- Mahony filter shows a precise convergence to the expected value of $\pm 90°$, paying it with low stability around a still position which is reconstructed with remarkable periodic oscillations in the estimated Euler angles, as proven by the long simulation (Fig. 4.12).
- Madgwick filter estimates the precise value of θ, ϕ with an error of some degrees, which is not a desirable behaviour especially for the DANCE facility since to test formation flight algorithms an high precision in attitude estimation is required. It shows instead great stability when the device is still.
- Kalman filter has demonstrated to well approximate the θ and ϕ angles, with some bigger imprecisions on the ψ component which shows a slower convergence. Some major oscillations in ψ over the long period are then found with the still test (Fig. 4.18), showing a "dashed" line behaviour instead of a continuous one. This is surely due to the update part which is not performed continuously but every 50 time steps, to enhance efficiency in computation.

For all the filters the angular velocity is estimated with satisfactory results.

The suggested next step is to test the algorithms with the complete DANCER simulator, imposing a known trajectory and evaluating error in reconstructing it with the alternative filters proposed. It is nevertheless true that none simulation can forecast with precision the behaviour in the real case, for which experimental tests are surely needed. From this point of view, the Kalman filter is theoretically the best choice since, once tuned estimating noise variances, the continuous update of the gains (intrinsic in the algorithm) theoretically ensure good performance; nevertheless, as here demonstrated, in order to have a computationally efficient algorithm its quality can be compromised. Mahony and Madgwick filters are instead less computationally expensive, since they require a less number of operations to be done, making them more suitable for real-time applications. All the three filters are currently kept as good and valid alternatives, since for the actual knowledge it is not possible to make a final choice, and it could also be interesting to test the DANCER vehicle behaviour with different algorithms. At any case, the availability of implemented codes and optimization procedures are surely a mandatory requirement for future tests, satisfied by the present work; refinements and preciser tuning can then be made on the developed codes, with time-saving procedures.

5 Attitude Control Actuators Design & Modelling

5.1 Chapter Overview

The primary purpose of using the electronic hardware system discussed in previous chapters is to control displacements and movements of DANCER vehicle while recording and keeping track of them in order to save telemetry data. Without an adequate equipment of actuators, the whole vehicle would behave uncontrollably and would be practically useless; that is the reason why it is fundamental to choose and size a proper set of actuators. DANCE actual design provides the presence of three reaction wheels controlled by electrical motors and twelve nozzles fed by a pressurized circuit and controlled with the aid of pressure regulators and solenoid fast-switching valves. In this chapter the need of re-designing the inertial mass of the reaction wheels is firstly addressed, proposing a solution for the realization and the final assembling. Moreover, the nozzles chosen in previous works [4, 5] have been tested experimentally at different pressures and with different working cycles, elaborating a mathematical model able to overcome the divergences arising from the real behaviour with reference to ideal expected one; results are here exposed and commented in details.

5.2 Reaction Wheels

Reaction wheels are a fundamental hardware part for the control of the DANCE facility. Their principal purpose is to keep the AP in an equilibrium condition before starting to perform maneuvers, which are instead mainly driven by the thrusters. It is nevertheless true that also the reaction wheels can contribute to performing small rotations of the body. The possibility of implementing an automatic dynamic balancing system equipped with moving masses able to keep the center of mass of the AP coinciding with its center of rotation had already been evaluated during previous works [4, 5], but at the actual state of the art is developed only in a conceptual way and thus not usable; several difficulties would have to be faced with this approach, starting from the inertia of the needed actuators that would disturb the DANCER vehicle attitude control. Balancing is in fact done manually before starting to utilize the facility, resulting in high imprecisions; that is why reaction wheels begin acting as soon as the system is switched on. Their effect is necessary to counteract the gravity acceleration and keep the AP on the desired angular configuration before starting maneuvering. At the state of the art only three reaction wheels are mounted on the vehicle but, on the final version, they should be changed to a total number of four, organized in a pyramidal configuration. The mathematical description of such configuration is not here reported but can be found in Sec. (3.4.5) of [5].

One reaction wheel is made by two components: an electric motor (Fig. 5.1) that is used to actuate and control the rotation, and a cylindrical mass attached to its shaft, which provides the right moment of inertia during the rotation; the final assembly mounted on the support system is shown in Fig. (5.2). The actuators used are furnished by Maxon Group and are electronically communicated (EC) brushless motors with maximum absorbed power of 30 W [27, 28, 29]. The integrated MILE incremental encoder uses an inductive angle measurement to generate output quadrature signals that permit to identify the rotational direction during the operative usage. Motors are also equipped with an internal Hall sensor that, by measuring variations of the magnetic field, is able to determine the angular rate, permitting in this way to get a feedback signal and allowing a closed-loop control. In Sec. (5.2.4) programming and connection procedure are described in further details.

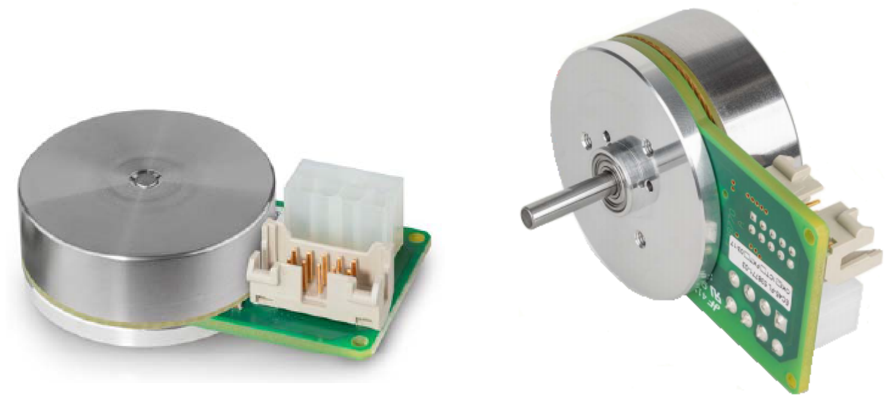


Figure 5.1: Maxon EC 45 flat brushless motor with integrated MILE encoder and Hall sensor.

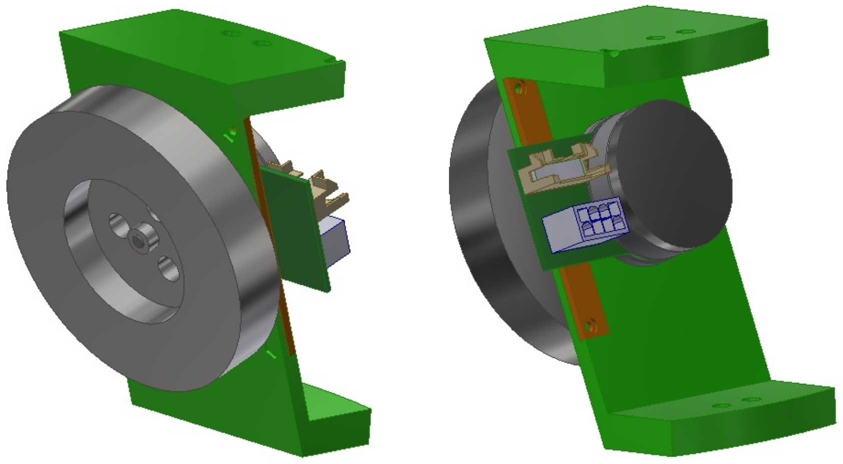


Figure 5.2: Reaction wheel assembly and structural support to be mounted on DANCER AP.

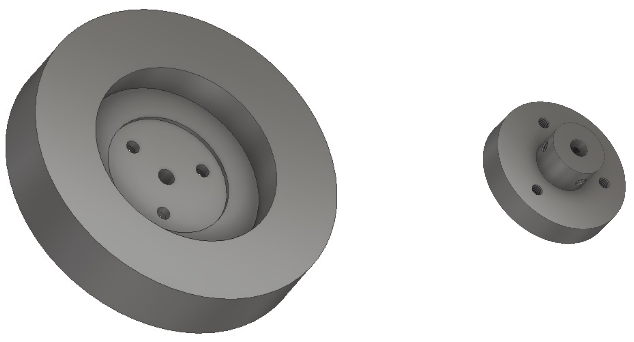
5.2.1 Design

In order to realize the reaction wheels with the right inertia to provide the requested angular momentum, each brushless motor must be equipped with a cylindrical mass that has to be mounted on the shaft. The flywheel must be manufactured with the right precision in terms of mass distribution to not exceed the maximum torque to be applied to the shaft itself because of unbalances; this parameter is known as radial run-out. The state of the art design of the flywheel has been re-evaluated for two main reasons:

- The old design established the presence of two components: the main flywheel and a smaller mass used for the assembling with the motor shaft (Fig. 5.3). By the means of a screw, the flywheel retainer was fixed to the shaft keeping the main mass in position. This configuration is not suitable since it does not allow to reach high precision in terms of mass distribution, causing the explained radial run-out problem that, for high rotational speeds, would lead to an engine failure.
- Inertia of the actual mass is too low. The problem of saturation has been already accounted in [5]: even with a small center of mass offset of the AP, during simulations the saturation (i.e. maximum rotational speed of the electric motors that corresponds to the maximum storage of angular momentum) was reached after 20 to 30 seconds, with consequent de-saturation requirements and obvious complications in controlling the system.

The first issue is addressed by re-designing the flywheel as a single compact piece and expressing the relative requirement for its realization. The maximum allowable center of mass misalignment can be computed easily [30]: the radially directed centrifugal force associated to the centripetal acceleration is given by Eq. (5.1), where M_{fw} $[kg]$ is the total flywheel mass, $\boldsymbol{\omega}_m$ $[rad/s]$ is the angular velocity of the motor shaft (and consequently of the flywheel, supposing a no-slip condition) and d_{mis} $[mm]$ is the center of gravity misalignment with respect to the center rotation axis, which actually sets the constraints for the flywheel realization.

$$F_{cf} = M_{fw} \cdot d_{mis} \cdot \|\boldsymbol{\omega}_m\|^2 \tag{5.1}$$



(a) Flywheel **(b)** Flywheel retainer

Figure 5.3: Previous flywheel design made by two different pieces in order to facilitate the assembling with the brushless electrical motor.

From Eq. (5.1) it is immediate to notice the inverse proportionality between M_{fw} and d_{mis} once the other values are fixed. On one hand, the mass must be kept as low as possible in order to relax the requirement on the center of mass offset; on the other hand, the inertia must be maximized to overcome the saturation problem exposed. Additional constraints are imposed by the actual design of the structure of DANCER vehicle, which does not allow to enlarge excessively the radius because of physical space problems. The parameters of interest for the motor are found on its datasheet [29] and are reported here on Tab. (5.1).

Parameter	**Value**	**Description**
$\lVert\boldsymbol{\omega}_m\rVert$	10000 RPM	Maximum shaft angular rate
F_{cf}	18 N	Maximum axial load (5 mm from the flange)

Table 5.1: Maxon EC brushless motor parameters of interest for flywheel design.

Keeping into account the aspects mentioned above, the design was carried out by increasing both the radius and the thickness of the flywheel, obtaining almost twice the previous inertia relative to the z rotation axis by maximizing the mass distribution as far as possible from the central axis. Two equivalent design concepts are proposed and reported in Fig. (5.4). The three symmetrical holes are needed for the mounting of the reaction wheel assembly with the proper structural support that connects it to the DANCER AP; the two proposals are almost identical from an inertial and dimensional point of view, what changes is the shape of the holes. In the first case (Fig. 5.4a) realization may be easier since it requires only a drill, but holes must be in any case done with high symmetry and more difficulties may be encountered during the final assembling. The second approach (Fig. 5.4b) is more flexible but requires high precision laser cutting. In both versions, the central part has been lightened, while the external part of the middle cylindrical hole has been extended in order to increase the contact area with the motor shaft; the external circular crown has been stiffened as well. Physical parameters of both designs are reported in Tab. (5.2) and the consequent requirement for the center of mass accuracy is made explicit in Tab. (5.3). Both design versions are meant to be realized in steel: aluminium was evaluated but, because of the reduced density, almost twice the actual dimension would be required to reach the same inertia, and thus was discarded. The reason why it was taken into account is that it is characterized by different ferromagnetic properties which would lead to an almost zero disturbance for the magnetometer but, unfortunately, with the actual DANCER design, it is not feasible.

Design version	**Mass** $[kg]$	**Inertia** $\boldsymbol{I_z}$ $[kg \cdot mm^2]$	$\varnothing$ $[mm]$	$\boldsymbol{d_{mis}}$ **max** $[\mu m]$
Old (Total mass)	0.445	346.695	75	37
New (a)	0.627	686.795	87	27
New (b)	0.624	686.257	87	27

Table 5.2: Flywheel physical attributes of previous and actual design alternatives.

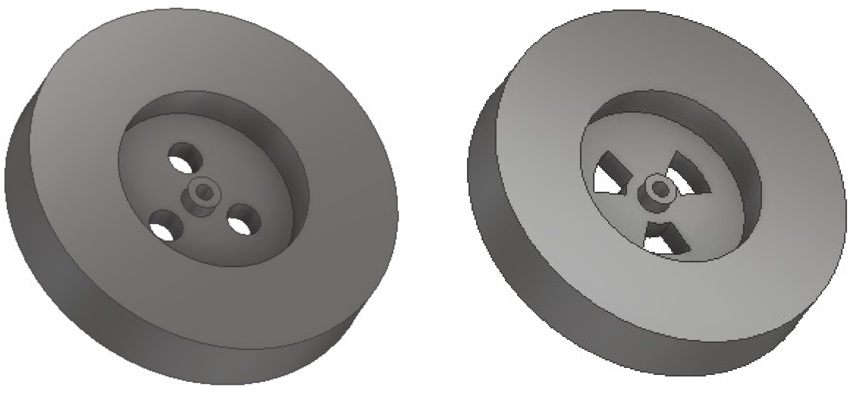


Figure 5.4: Actual flywheel design equivalent alternatives characterized by only one massive component.

RW-REQ-1	The flywheel must be realized with a tolerance on the center of mass misalignment with reference to the central rotation axis lower than the maximum d_{mis} reported in Tab. (5.2).

Table 5.3: Center of mass requirement for flywheel realization.

5.2.2 Flywheel Assembling

As mentioned, the central part of the flywheel has been extended, because the contact area between the motor shaft and the flywheel must be maximized. In fact one drawback of realizing the flywheel as one single piece is that it must be forcibly coupled with a press-fit procedure, discarding in this way the possibility of removing it if necessary and requiring a new design of the reaction wheel support structure; in this type of coupling the static friction is the main driver parameter, that explains why the contact area must be as large as possible. In order to determine the amplitude of the tolerance for the interference fit, at first the minimum interference required for a torque transmission without slipping must be determined. Relation between the no-slip torque T_{ns} $[N \cdot mm]$ and radial force F_R $[N]$ is given by Eq. (5.2), indicating with d_{nom} $[mm]$ the nominal diameter of the shaft. The radial force can be computed with Eq. (5.3) knowing the static friction coefficient between the shaft and the flywheel μ, the lateral contact area A_{lat} $[mm^2]$ and the minimum pressure to guarantee the no-slip condition P_{max} $[Pa]$.

$$T_{ns} = F_R \cdot \frac{d_{nom}}{2} \tag{5.2}$$

$$F_R = \mu \cdot P_{max} \cdot A_{lat} = \mu \cdot P_{max} \cdot \pi d_{nom} L_{contact} \tag{5.3}$$

Finally, the relation between the minimum pressure that is able to grant the wanted condition P_{min} $[Pa]$ and the searched radial interference δ $[\mu m]$ is given by Eq. (5.4a), indicating with r $[mm]$ the nominal radius, with E $[Pa]$ the Young's modulus and with ν the Poisson's ratio; the subscript f refers to the flywheel, while s refers to the shaft.

Since both shaft and flywheel are made of stainless steel[3], we can approximate that the material properties are the same, simplifying the relation into Eq. (5.4b).

$$P_{min} = \frac{\delta}{\frac{r}{E_f}(1+\nu_f) + \frac{r}{E_s}(1-\nu_s)} \tag{5.4a}$$

$$= \frac{\delta}{2 \cdot \frac{r}{E}} \tag{5.4b}$$

From the motor datasheet [29] the maximum torque can be found; since the reaction wheels must work up to the largest angular velocity supported without slipping, this torque is used in Eq. (5.2) to retrieve the relative radial force. Minimum pressure for this condition is then computed with Eq. (5.3) and finally the minimum radial interference to be ensured is determined by the means of Eq. (5.4b). All the parameters used are summarised in Tab. (5.4), while Tab. (5.5) presents the results obtained, with the consequent requirement for the press-fit reported in Tab. (5.6). An additional requirement must then be set in order to ensure a minimum distance between the motor flange and the flywheel (Tab. 5.7) to guarantee the correct assembling with the supporting structure and to avoid touching during operative conditions, as explained in next Sec. (5.2.3).

Parameter	**Value**	**Description**	**Source**
T_{max}	$54.8\ mN \cdot m$	Motor maximum nominal continuous torque	[29]
$L_{contact}$	$10\ mm$	Axial contact length	(designed)
d_{nom}	$3.995\ mm$	Shaft nominal diameter	[29]
μ	0.74	Steel-steel static friction coefficient	[31]
E_f	$200\ GPa$	Flywheel Young's modulus	[32]
ν_f	0.305	Flywheel Poisson's ratio	[33]
E_s	$200\ GPa$	Shaft Young's modulus	[32]
ν_s	0.305	Shaft Poisson's ratio	[33]

Table 5.4: Physical parameters for press fit calculations and design.

Parameter	**Value**	**Description**
P_{min}	$2.16 \cdot 10^6\ Pa$	Minimum pressure to grant no-sleeping
F_R	$100.13\ N$	Radial force associated to P_{min}
δ_{min}	$0.052\ \mu m$	Minimum radial interference for no slipping

Table 5.5: Resulted requirements for press fit design.

The nominal radius of the shaft reported on the datasheet [29] is $d_{nom} = 4^{-0.005}_{-0.01}\ mm$ with the indication of the upper and lower limits resulted from manufacturing imprecisions. In order to find the tolerance classes for the hole-shaft coupling with the international standard ISO notation [34] at first the part that acts as the base must be indicated.

[3]The flywheel has not been realized yet but it should be made of stainless steel; if a different material is used, the computation must be repeated. Nevertheless, minimum interference is satisfied with a high tolerance with the quality classes chosen, so it should not be a concern.

RW-REQ-2	The press fit coupling between the flywheel and the motor must be performed with tolerance classes for the hole of the flywheel and the shaft of the motor that ensure a minimum radial interference δ_{min} reported in Tab. (5.5).

Table 5.6: Press fit requirement for flywheel and motor coupling.

Typically, for the case of an interference fit, the reference part is taken to be the shaft [35]; the upper limit of its diameter is assumed as reference value and, for this reason, the nominal diameter is indicated to be, in an equivalent way, $d_{nom} = 3.995^{+0.000}_{-0.005}\ mm$. Considering the tolerance defined in such way the class $h5$ for the shaft is determined. The letter indicates the position of the tolerance, in this case h represents a unilateral inferior deviation different from zero and a null upper deviation, while the number indicates the degree of tolerance, corresponding in this case to the 0.005 mm given. The couple nearest to the typical industrial applications [35], indicating with the capital letter the class of the hole, is:

$$\varnothing 3.995\ P6/h5$$

that, in an explicit way, corresponds to:

$$\begin{aligned}\text{Shaft: } \varnothing 3.995\ h5 &= \varnothing 3.995^{+0.000}_{-0.005}\\ \text{Hole: } \varnothing 3.995\ P6 &= \varnothing 3.995^{-0.009}_{-0.017}\end{aligned}$$

With these classes the minimum interference results to be 0.004 mm while the maximum is 0.017 mm thus satisfying RW-REQ-2 with a wide tolerance. A positive aspect of such configuration is that the assembling procedure can be carried out by hand with the help of a mechanical press, a mallet and, if needed, heating the flywheel to temporary increase the hole diameter and/or freezing the shaft with liquid nitrogen to decrease its diameter during short-term application [35]. The maximum applicable axial stress for the motor is again found on the datasheet [29] and corresponds to 53 N if the motor is supported, or 1 kN if instead the shaft is directly sustained [36]. One alternative would be to keep the flywheel fixed and insert the shaft by pressing on the bottom part of its shaft, caring that their axes are correctly aligned during the whole procedure.

5.2.3 Structural Support Design

The structural support needs to be reconsidered in order to grant the satisfaction of the new constraints imposed for the manufacturing and the assembling of the flywheel. The main concern regards the necessity of performing the press-fit coupling between the flywheel and the shaft of the motor, which does not allow performing the old assembling strategy. In particular, it consisted in: inserting the shaft into the apposite hole present on the structural support, fix the three screws to keep the motor attached to it and finally mount the flywheel. With the press-fit constraint, the flywheel must instead be mounted on the shaft *before* being coupled to the structural support. Starting from the mentioned issue, the design has been re-valuated and adapted to the new model; at the moment it is developed only in a conceptual way, permitting to visualize a solution for the problem. Before the final realization, a structural test is highly suggested in order to correct in advance possible problems most probably correlated to vibrations generated in operative conditions. Some different solutions have been evaluated, reporting here only the most relevant; three different conceptual approaches are individuated:

- **Solution 1.** The first solution actually permits to keep the old design and it is the best from a structural point of view since the support is conserved in its integrity and does not need to be composed by different assembling pieces. The biggest drawback is that the press-fit coupling must be performed with the shaft already inserted in the predisposed hole, eliminating the possibility to disassembling and re-assembling, if necessary or if the support breaks. Fig. (5.5) represents the old support design.
- **Solution 2.** The opposite strategy to the first one exposed consists in designing the support with a lateral aperture as small as possible that allows the motor to be set in position after the assembling with the flywheel; an additional part is then added in order to reinforce the open part of the structure. In this case, disassembling possibility is granted, even if the structural capabilities of load-carrying are weakened. Attention must be paid during the press-fit coupling in order to leave the right interspace between flywheel and motor, which must be higher than the support thickness, otherwise assembling is not possible, but the requirement set in order to ensure safe operative conditions (Tab. 5.7) already grants this necessity. A small hole is also added in order to permit the assembling of the top screw for the additional piece since it is in a position that may be difficult to reach with a screwdriver. Fig (5.6) and Fig. (5.7) clarify the configuration proposed.
- **Solution 3.** A third hybrid solution is proposed, characterized by the presence of an additional assembling part in which the shaft must be inserted before the press-fit coupling. The additional piece is then mounted to the other part of the structural support, ensuring a physical constraint by the means of the same three screws used to fix the motor; two more screws are added to have on the whole a more robust structure. Even in this case a part of the structure cannot be removed after the interference fit is performed, but it is a less cumbersome smaller piece with reference to the first solution and the possibility to remove the flywheel-motor assembly from the main structural support is ensured. An additional hole for assembling purpose is needed, as for solution 2. Fig (5.8) and Fig. (5.9) clarify the configuration proposed.

In all the cases the thickness of the support is maintained to a value of 5 mm and thus an additional requirement must be set to grant a minimum interspace distance d_{int} between the flywheel and the support itself, to avoid touching conditions. The requirement is expressed in terms of the distance between the flywheel surface facing the motor and the flange of the motor, and is reported in Tab. (5.7) and clarified by Fig. 5.10. With the present knowledge, the best choice is surely constituted by the second solution, for the main reason that it allows to mount and dismount the motor-flywheel assembly from the supporting structure every time it is necessary. From a structural point of view, a finite element analysis must be performed in order to verify the stress withstanding capability; with the stress analysis results, if the chosen design results to be not stiff enough, the trade-off between the proposed alternatives may be reconsidered and one of the other two different solutions may be adopted. Another aspect to keep into account if selecting the first or the third option is that if for any reason the additional piece inserted to the shaft (inserted before the press coupling) breaks, flywheel and motor must be decoupled in order to substitute the broken piece, or the second solution must be adopted forcibly. Moreover, performing the interference coupling may result in a more difficult and inconvenient process; all those aspects were considered in indicating the second solution as the best choice. All the parts are meant to be manufactured using 3D printing technology; old supports were realized in Polylactic Acid (PLA) but in order to increase resistance to fatigue other materials should be used, like Acrylonitrile Butadiene Styrene (ABS) or Polyethylene Terephthalate (PET), as suggested in [5]. In all the different cases the three screws used to fix the motor can be accessed through the opposite holes present on the flywheel.

RW-REQ-3	The press fit coupling between the flywheel and the motor must be performed leaving an interspace distance d_{int} of at least 7 mm measured as the minimum distance from the motor flange to the flywheel surface which is facing the motor (Fig. 5.10).

Table 5.7: Press fit requirement to ensure a minimum distance between the rotating flywheel and the supporting structure in operative conditions.

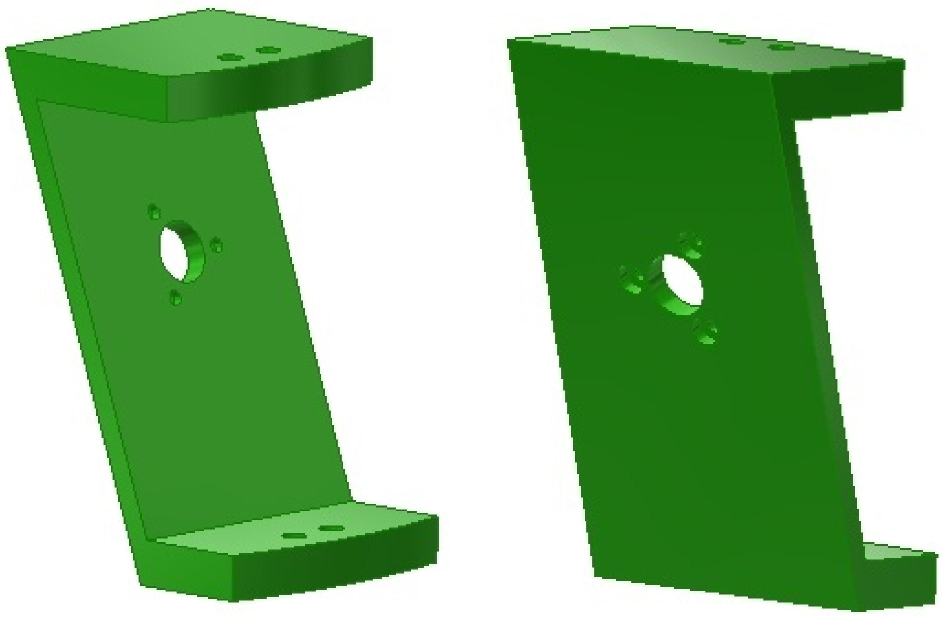


Figure 5.5: Old reaction wheel structural support design (solution 1).

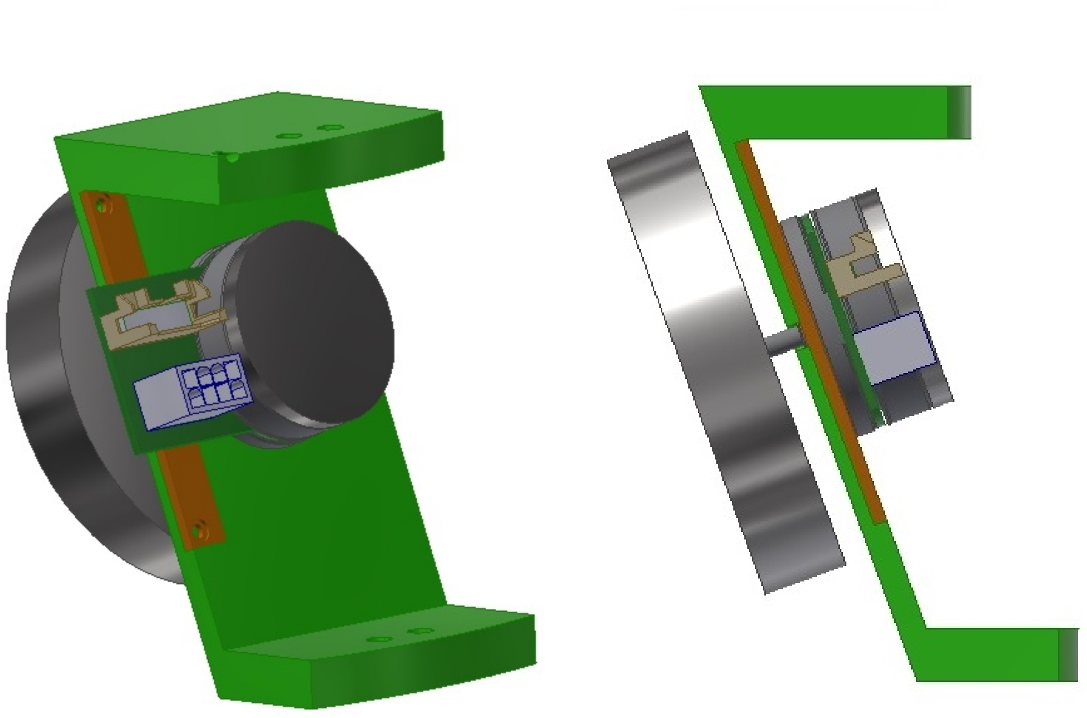


Figure 5.6: Reaction wheel assembly concept with lateral aperture and additional reinforcing part (solution 2).

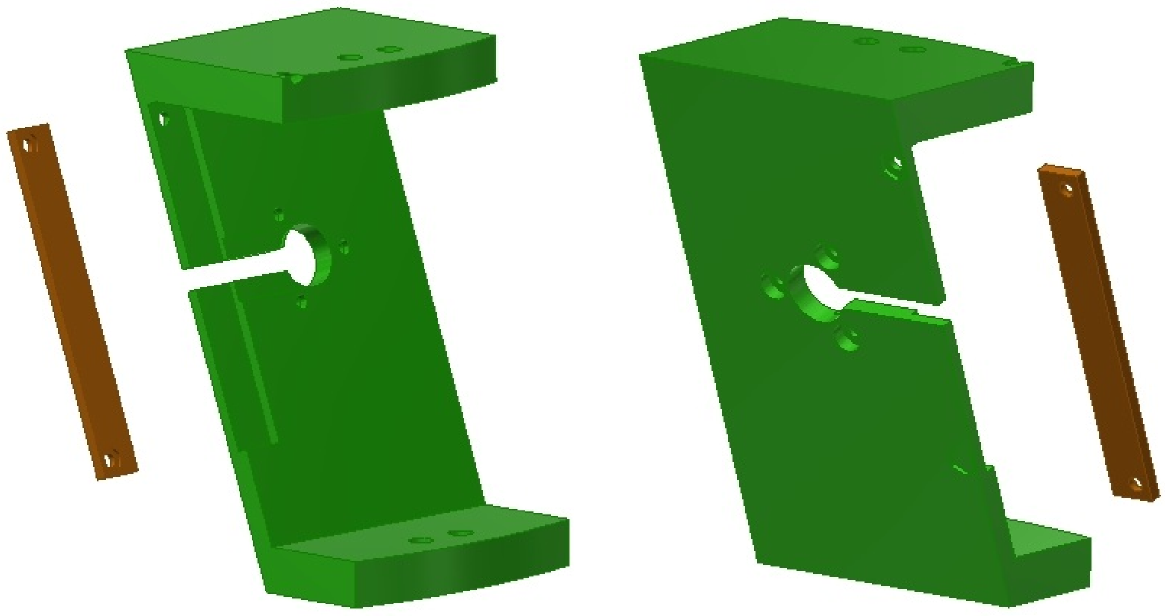


Figure 5.7: Exploded view of reaction wheel's structural support with lateral aperture and additional reinforcing part (solution 2).

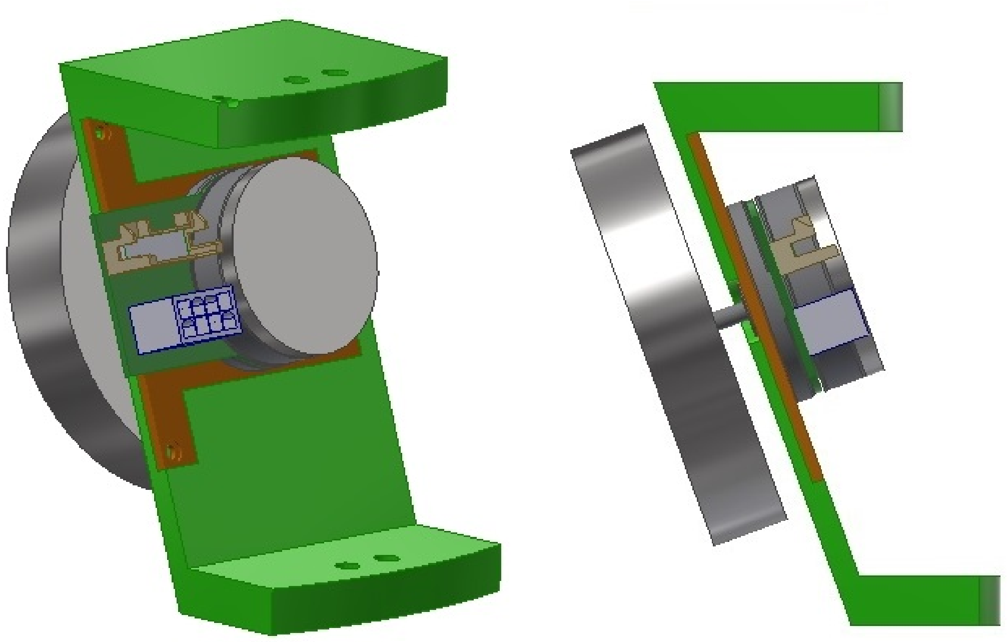


Figure 5.8: Reaction wheel assembly concept with additional assembling part attached to the motor (solution 3).

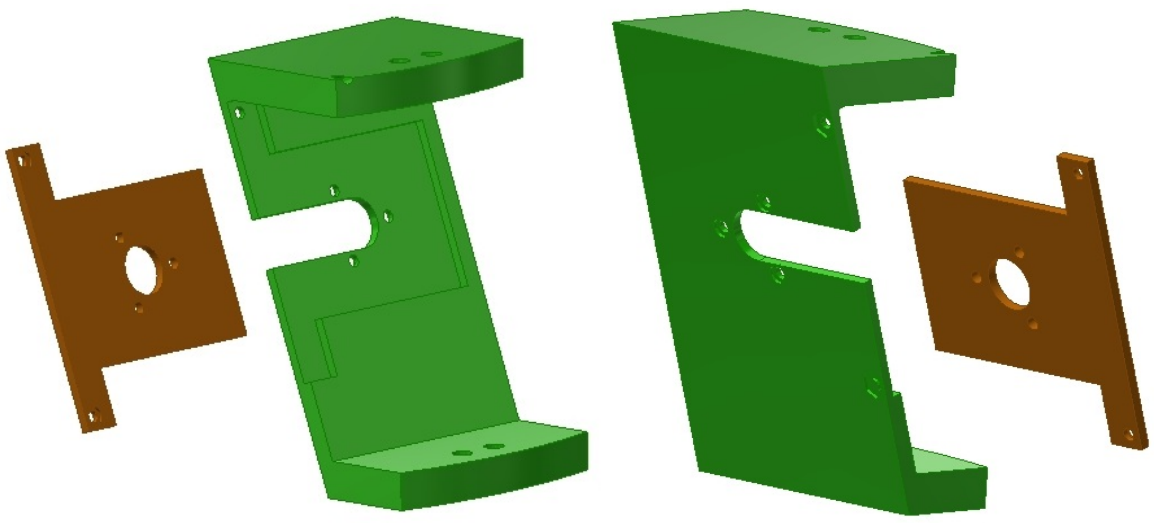


Figure 5.9: Exploded view of reaction wheel's structural support with additional assembling part attached to the motor (solution 3).

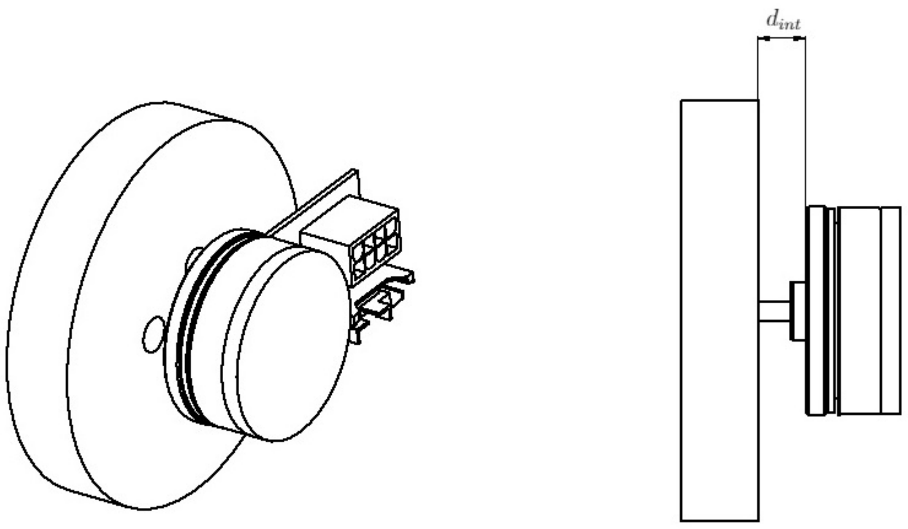


Figure 5.10: Minimum distance requirement technical view.

5.2.4 Motor and Driver Connection

The electrical motors used to control the inertial mass rotation of the reaction wheels can not be connected directly to the TI LaunchPad, but need an additional intermediate hardware component, which is the ESCON Module 24/2 furnished by Maxon Group (Fig. 5.11). The driver is programmed by the means of the associated Escon Studio software which allows configuring the correct parameters to control the EC motor; the rotational velocity of the shaft can then be controlled in a closed-loop by specifying an input current limit with a pulse-width modulation signal coming directly from an ePWM general output pin of the TI LaunchPad [11]. The angular velocity of the motor is then measured by the means of the integrated Hall sensor that gives signal feedback. A simple block representation of the closed-loop control is reported in Fig. (5.12).

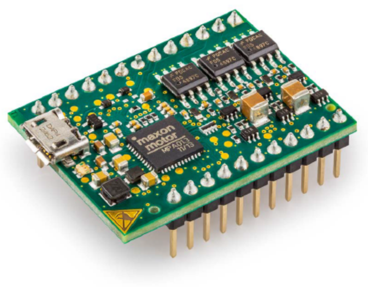


Figure 5.11: Maxon ESCON module 24/2 used to control the electrical motors of the reaction wheels.

Wiring sketch for controller and motor connection can be found on the relative datasheets [27, 37]; the driver needs to be fed with an external source of +24 V DC voltage, while

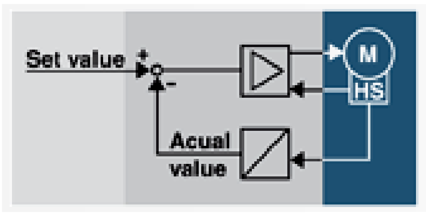


Figure 5.12: Block representation of the closed loop control of the Maxon EC motor (M indicates the motor, while HS represents the integrated Hall sensor). *Credits: Maxon Group Escon Studio.*

the motor nominal operating voltage of $+5V$ is directly retrieved connecting it to the driver. During experimental activities at the DAER laboratory, connections have been at first made by the means of a breadboard and several jumper pins, in order to test the functionality. The circuit has been successively soldered on a stripboard obtaining a prototype that facilitates the connections (Fig. 5.13): the driver is directly inserted on the predisposed pins while a series of Euroblock-type terminal blocks allow inserting the cables and transmitting input/output signal from/to the motor, by following the indications reported on Fig. (5.14). A sketch representation of the printed circuit is also reported.

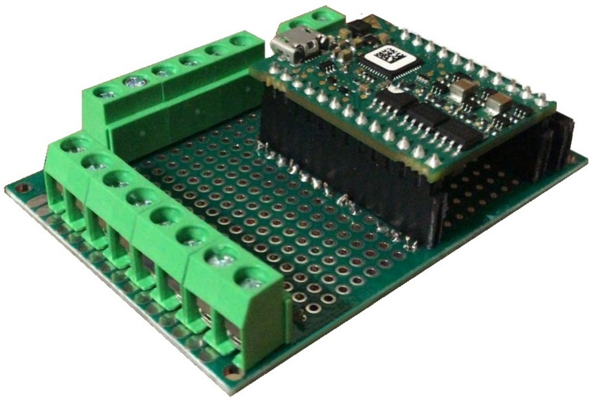
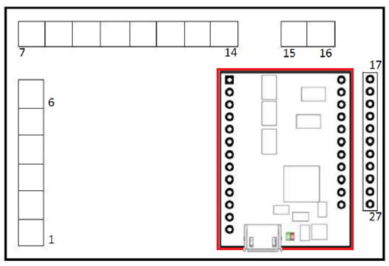


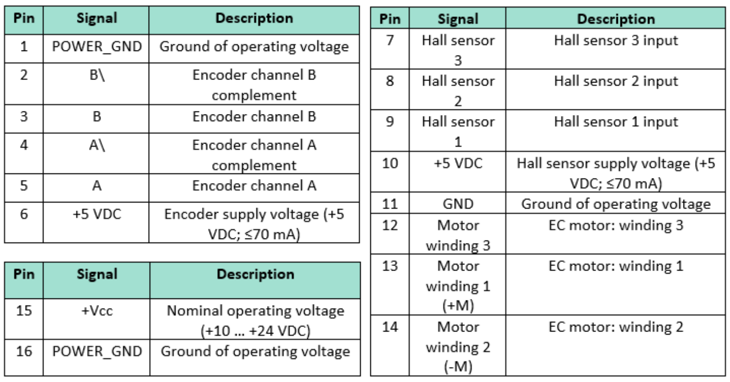
Figure 5.13: Prototype of the wiring for the connection between the Escon 24/2 module and the EC motor.

5.3 Thrusters

At the current design state, DANCE is equipped with twelve thrusters; in particular, four aluminium blocks are present, each one characterized by three internal cylindrical holes to which nozzles are mounted (Fig. 5.15). Nozzle model identified in previous work [5] and used for the present analysis is the Silvent MJ40 [38]; nozzles allow to accelerate a pressurized gas and to get a force opposite to the gas expulsion direction by the action-reaction principle. In this case, the gas selected for the working conditions is air, pressurized at about 300 *bar* and stored on four vessels (two mounted on the TP and the other two mounted on the AP). A pressure regulator [39] separates the high-pressure



(a)



Pin	Signal	Description
1	POWER_GND	Ground of operating voltage
2	B\	Encoder channel B complement
3	B	Encoder channel B
4	A\	Encoder channel A complement
5	A	Encoder channel A
6	+5 VDC	Encoder supply voltage (+5 VDC; ≤70 mA)

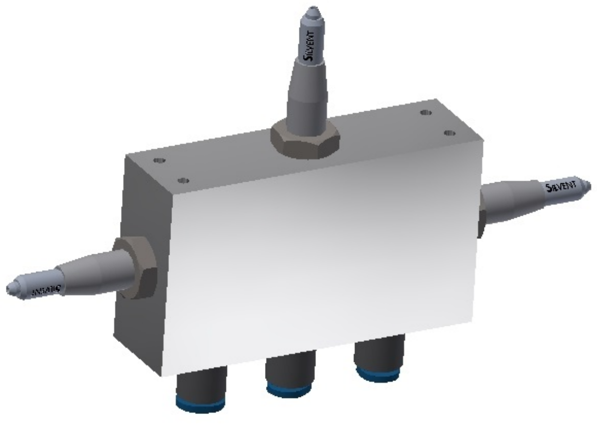
Pin	Signal	Description
15	+Vcc	Nominal operating voltage (+10 … +24 VDC)
16	POWER_GND	Ground of operating voltage

Pin	Signal	Description
7	Hall sensor 3	Hall sensor 3 input
8	Hall sensor 2	Hall sensor 2 input
9	Hall sensor 1	Hall sensor 1 input
10	+5 VDC	Hall sensor supply voltage (+5 VDC; ≤70 mA)
11	GND	Ground of operating voltage
12	Motor winding 3	EC motor: winding 3
13	Motor winding 1 (+M)	EC motor: winding 1
14	Motor winding 2 (-M)	EC motor: winding 2

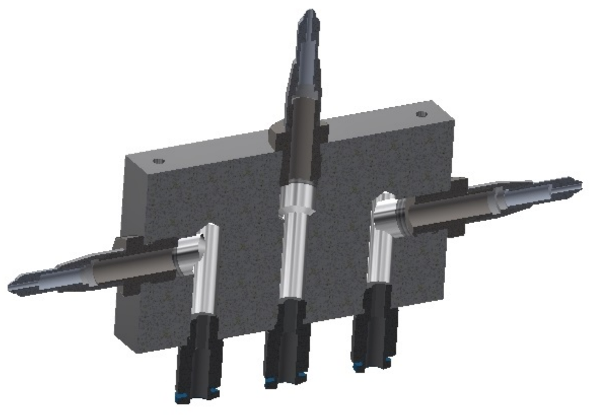
(b)

Figure 5.14: Sketch of the prototype circuit (a) with the numbered Euroblock-type terminals and the assembled Escon 24/2 module (highlighted in red), with the relative pins description (b). Pins 1 to 6 must be connected to the encoder, pins 7 to 14 to the Hall sensor, pins 15 and 16 are dedicated to the power supply, pins 17 to 27 are dedicated the input signals and their description can be found directly on the datasheet [37].

circuit connected to the vessels and the low-pressure circuit that feeds the thrusters, with working conditions lower than 10 *bar*; to control the final pressure of the gas that goes into the nozzles and the magnitude of the force retrieved an additional digital pressure regulator is needed, controlled directly from the TI LaunchPad. Twelve fast switching solenoid valves (one for each nozzle) are then present in order to control the gas output stream and thus the impulse obtained.



(a) Full view



(b) Section view

Figure 5.15: Thrusters model composed by a block aluminium, three holes and three nozzles.

Thrusters are clearly a fundamental part of the set of actuators of which the DANCER vehicle must be provided; they are used for attitude control and maneuvering in combination with the reaction wheels.

5.3.1 Pneumatic Tests

The already available thrusters have been tested with different working pressures of 2, 3, 4, 5, 5.5 and 6 *bar*. The test measured the resultant force given by the action-reaction principle by the means of the Futek LSB200 load cell [40], with the final purpose of computing the total real impulse obtained with different PWM duty cycles and refine the linear model used to compute the thrust versus time variation during real-time control of the DANCER platforms. What is searched is, in fact, a function that takes in input the pulse-width modulation period $pwmPrd$ and the pulse-width length $pWidth$ giving as a result the total impulse obtained I_{tot} $[N \cdot s]$, as described by Eq. (5.5). A detailed description of the test facility and setup can be found in Sec. (4.2) of [5].

$$I_{tot} = f(pWidth,\ pwmPrd) \tag{5.5}$$

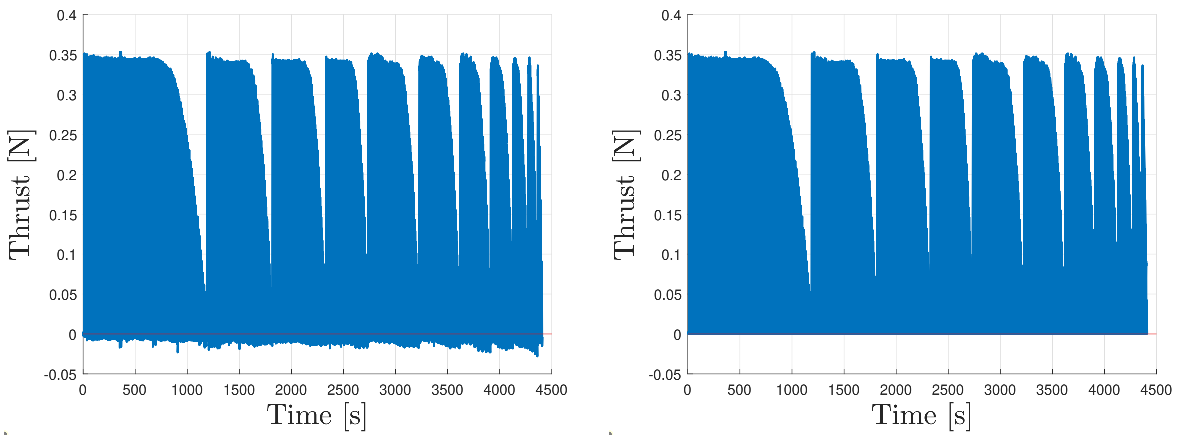
Explaining the tests more in-depth, the working pressure P_{work} is defined as the pressure at which the pneumatic circuit is subjected, that corresponds to the steady-state pressure present in the proximity of the nozzle of the thruster if, for a first evaluation, we neglect tube losses. For each particular pressure, 11 different periods have been experimented, where the pulse-width period $pwmPrd$ $[s]$ refers to the time interval of the PWM signal (in output from the TI LaunchPad) after which the cycle begins again. To every period corresponds a different pulse-width $pWidth$ $[s]$, which represent the time during which the control valve is open and thus the thruster is working. Adopting MATLAB vectors notation, values used for the period are represented by Eq. (5.6): first value tested has a unitary value, then starting from 250 ms the value is lowered to 100 ms with a decrement of 0.050 ms, and then again from 0.0875 ms to 0.0250 ms with steps of 0.0125 ms, with the purpose of testing some different range of values that will be likely used during DANCER control to generate thrusts of the order of 1 N or lower. In an equivalent way Eq. (5.7) delineates the 106 different pulse-widths $[ms]$ that are tested for each period, and Eq. (5.8) describes instead the number of pulses (i.e. the number of repetition of the imposed cycle) correspondent to each $pwmPrd$ value.

$$pwmPrd = [1, 0.250 : -0.050 : 0.100, 0.0875 : -0.0125 : 0.0250]\ s \tag{5.6}$$

$$pWidth = [500, 250 : -25 : 125, 100 : -1 : 2]\ ms \tag{5.7}$$

$$nPulses = [10, 20, 20, 20, 40, 40, 40, 40, 40, 40, 40] \tag{5.8}$$

The measurements were firstly filtered by cleaning noise contribution up to the order of magnitude imposed by tol_{filter} parameter (Tab. 5.8) and successively integrated with MATLAB using a trapezoidal method. An example of the measurements before and after filtering can be found in Fig. (5.16), while all the other cases are reported in Appendix (C.1.1). The load cell used is not so precise thus some pike disturbances that are higher than the tolerance set for filtering are present and need to be taken into account when designing the integration algorithm; higher values of the tolerance mean that more noise is rejected, but even a small part of useful data is wasted, so the value chosen is a good compromise resulted from the trade-off between these two aspects. In terms of temporal resolution, measurements relative to a $pWidth$ lower than 10 ms are not considered, since disturbances in this case resulted to be too high because of the slow pneumatic response to the electric input. Filtered values are reported in Fig. (5.17), from which the steady-state level of the thrust relative to a particular pressure can easily be noticed and retrieved.

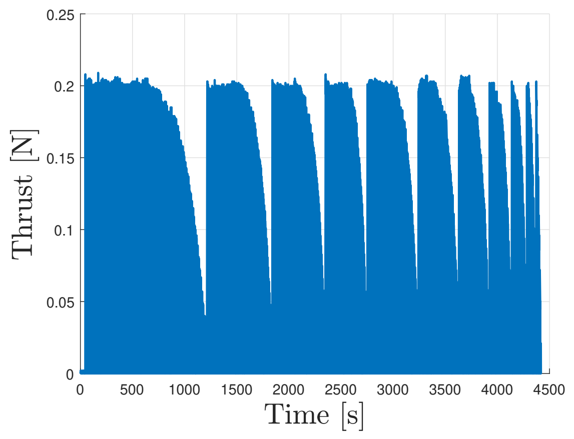


(a) Rough data **(b)** Filtered data

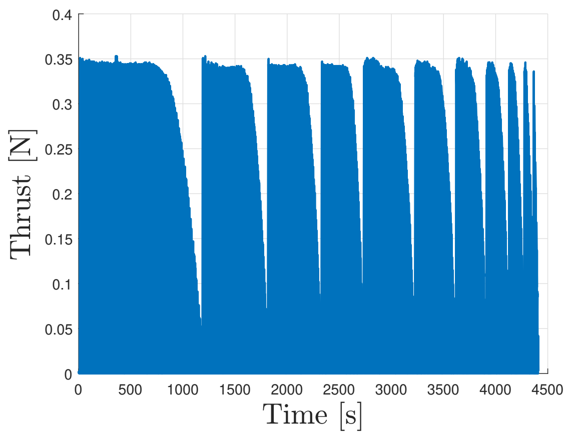
Figure 5.16: Measurements without (a) and with (b) application of noise filtering for $P_{work} = 3\ bar$, using the tolerance tol_{filter}.

5.3.2 Impulses Integration Algorithm

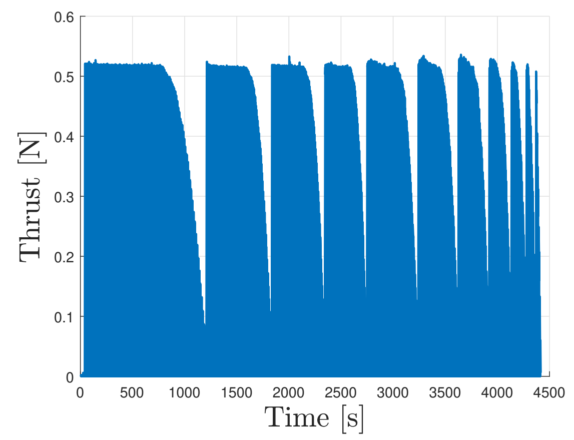
As mentioned, thrust records were integrated to find the underlying area corresponding to the impulse; to this purpose, the regularity of temporal intervals imposed cannot be trusted since the real periods and pulse-widths measured do not reflect precisely the ones set by the Simulink code, leading to a problem of error accumulation if this approach is selected. The solution is found using an imposed threshold tol_1 to find initial and final time instants for the integration of the different thrust measurements; this tolerance must be set higher than the noise in order to avoid wrong computations. Of course, the lower the tolerance the more precise are the results (and vice-versa), but the value is however limited by instrument precision. In this way, when the thrust is found to be higher than the threshold and inside the correct interval of time instants, the initial point of the ramp-up transient and the final point of the ramp-down tail of measured thrust are identified, giving correct references for the final integration; location of these points is refined right after by scrolling time instants respectively backwards and forward in order to catch the points t_0 and t_f in which the thrust value is zero, thus getting the right extremes of each pulse-width. Since the zero level could be not reached exactly, even in this case a threshold tol_2 is set as reference value. Noise disturbance must be additionally discarded since even a small contribution of the order of ms could compromise the whole result; thus to account the time intervals in which the thrust should be zero, each singular pulse is isolated with the logic described above, even if the pulse is repeated in the same way $nPulses$ times for every ($pWidth$, $pwmPeriod$) couple (Fig. 5.18) and, theoretically, integration could be performed only on the first one and its value multiplied for the number of repetitions to find the total area. Fig. (5.19) clarifies the logic of the algorithm, even if it is not in scale: the represented tolerance is higher than the one actually imposed only for readability reasons. After having found the extreme points, each isolated pulse is integrated with a trapezoidal mathematical method; pulses characterized by the same parameters are summed up in order to find the total impulse. At the end, for each $pwmPrd$, the total impulse I_{tot} relative to each $pWidth$ tested is available.



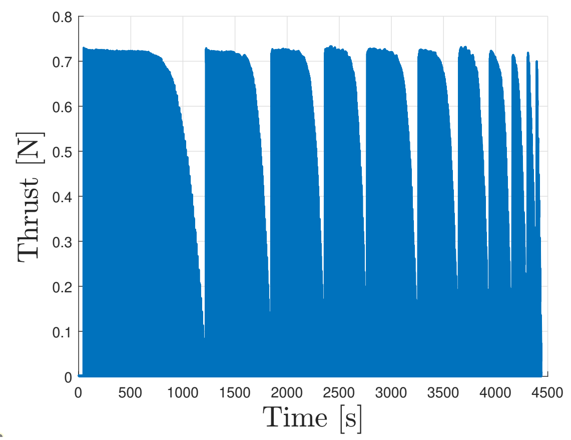
(a) $P_{work} = 2$ bar



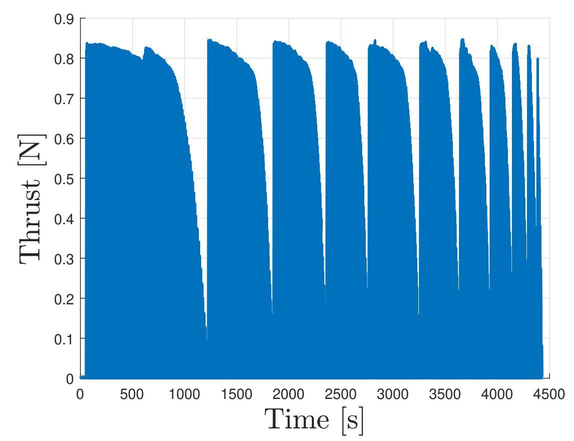
(b) $P_{work} = 3$ bar



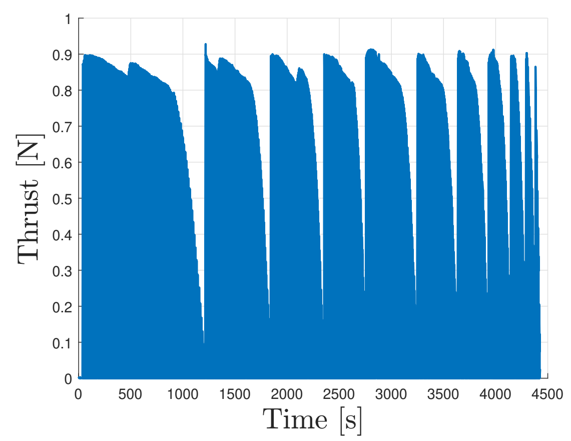
(c) $P_{work} = 4$ bar



(d) $P_{work} = 5$ bar



(e) $P_{work} = 5.5$ bar



(f) $P_{work} = 6$ bar

Figure 5.17: Filtered thrust records for different working pressures.

Parameter	Value	Units	Description
tol_{filter}	$1{\cdot}10^{-3}$	$[s]$	Tolerance for initial noise filtering
tol_1	$2{\cdot}10^{-2}$	$[s]$	Threshold for finding extreme points of each impulse
tol_2	$1{\cdot}10^{-4}$	$[s]$	Threshold for extreme points location refinement

Table 5.8: Tolerances and threshold values set for the elaboration of recorded data.

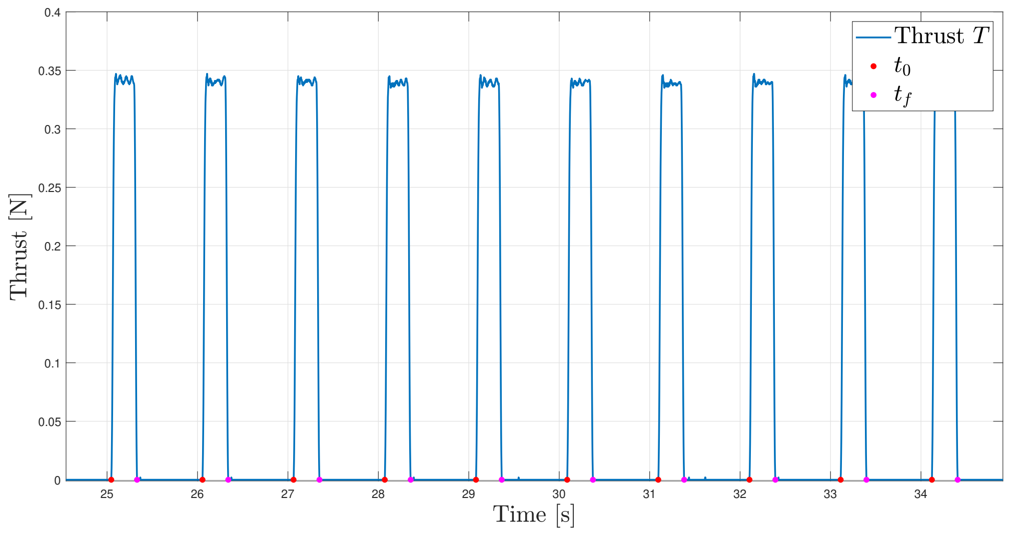


Figure 5.18: Algorithm logic: individuation of t_0 and t_f for the isolation of each impulse characterized by same pulse-width and PWM period (in the figure $P_{work} = 3\ bar$, $pWidth = 500\ ms$, $pwmPeriod = 1\ s$).

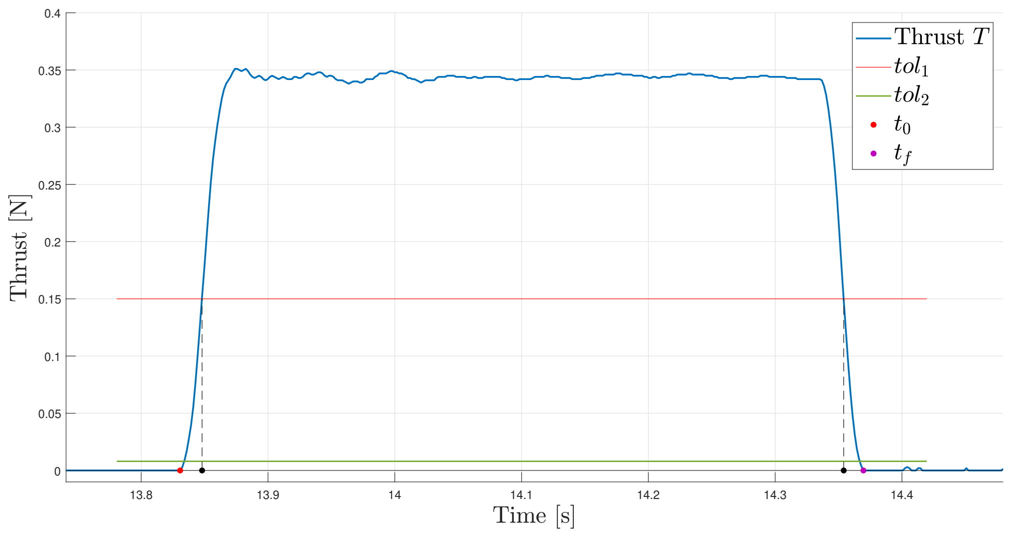


Figure 5.19: Algorithm logic: if the thrust (in blue) is higher than tol_1 (in red), then the correspondent time instant is taken as the starting point to look backward for t_0, untill the thrust is lower than tol_2 (in green); equivalent process for t_f is repeated forward (not in scale).

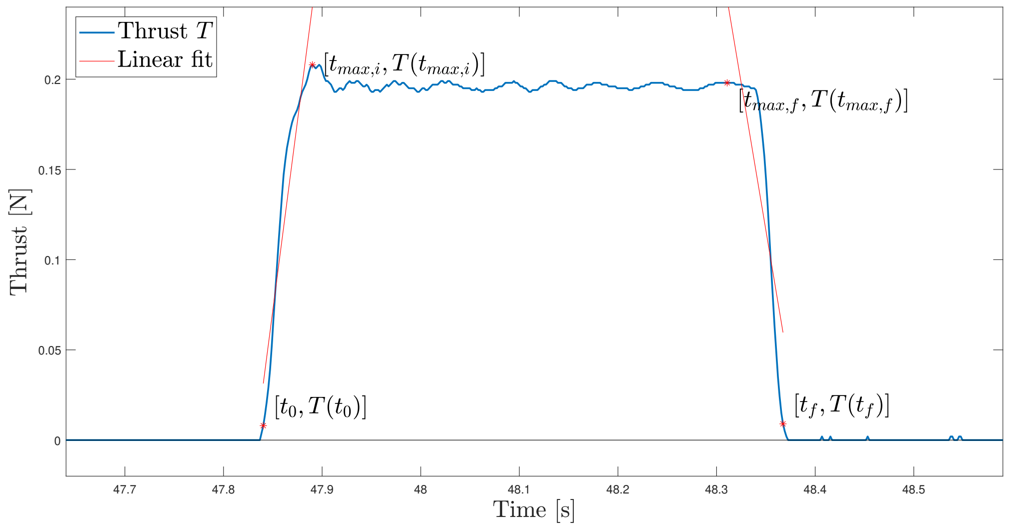
5.3.3 Linear Model Computation

An analytic mathematical model is then required in order to compute thrust and total impulse with reference to the pulse-width which is imposed during DANCER control (Eq. 5.5); model must take into account errors that are present in the real case, such as tail transients, overshoots and fluctuations. Previous work analysis [5] identified a linear model as a good approximation for the ramp-up and ramp-down transients interpolation, and used a simple trapezoidal pattern to compute the final impulse. This model is still considered valid but needed more refinement, in particular for what regards coefficients and corrective factors computation. Tests were in fact performed with a lot more intermediate points by increasing the number of pulse-widths tested for each period; also the working pressure range was tightened with more values in-between. Thrust records are then processed, bringing to the results exposed in this section. Please note that the pictures used to describe the model are here reported by way of example for one particular pressure and not for all the cases tested; additional material can be found in Appendix (C.1).

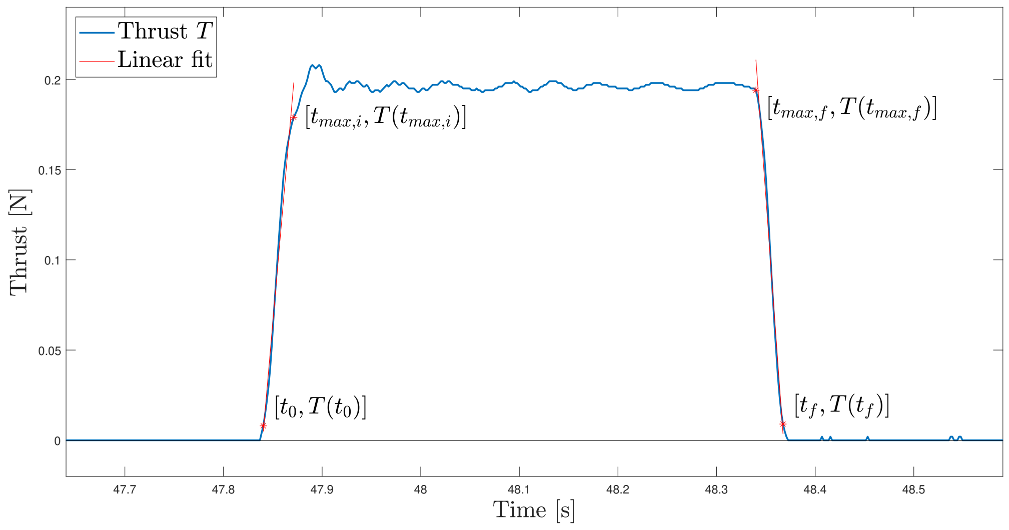
In order to analyze data only the case with $pwmPrd = 1\ s$ is taken into account since the purpose is to find a model that is able to approximate the area of the single separate impulse; analysis could be performed only on the first impulse corresponding to each $pWidth$ value but it is done for all the available data (as explained every pulse-width is in fact tested for $nPulses$ identical repetitions), with the aim of increasing the reliability of statistical computations. Four reference points are then fundamental: the extremes time instants t_0 and t_f that were found previously and two other points $t_{max,i}$ and $t_{max,f}$ which respectively individuate the final point of the ramp-up transient and the initial point of the ramp-down tail (Fig. 5.20). These four points are in fact necessary to define references for the interval of values inside which the linear interpolation is performed. $t_{max,i}$ is found with a different logic than before: research starts forward from thrust value in t_0 and, whenever thrust variation between the actual point and the previous one is higher than the one computed in the previous iteration, the last point is taken as $t_{max,i}$, since it means that thrust values are stabilizing after the transient part. Same process is repeated for $t_{max,f}$ but going backward from t_f. This algorithm logic resulted to be very effective and necessary to be applied; the alternative of looking for a relative maximum thrust value in a specified range will in fact lead to a wrong point identification because of the presence of initial overshoot and oscillations, as clearly evident from Fig. (5.20) that illustrates different approaches results. A first-degree linear fit is then performed on the points of the data set that are between the two initial points to find the ramp-up coefficient a_{ru} for the analytical model and between the last two points, computing a_{rd} for the ramp-down.

As far as the pulse-width lowers, $t_{max,i}$ and $t_{max,f}$ get closer, up to the point in which they ideally coincide; this situation occurs when the pulse-width is so low that the thrust has not the physical time to stabilize around a steady-state value, resulting in one single spike (Fig. 5.21). In this case, errors in total impulse computation are higher but the model is able to correct them, as it will be explained next.

After all the coefficients have been computed with the linear interpolation fit, what is used for the final model is an average value for a_{ru} and a_{rd}. Another important aspect must be considered, which regards the length of the impulse: the real pulse-width does not coincide with the imposed one but is actually longer, because of the actuation time in the real case. Additional time parameters are then needed for the final model since



(a) $t_{max,i}$ and $t_{max,f}$ computed as local relative maxima



(b) $t_{max,i}$ and $t_{max,f}$ computed considering thrust variation

Figure 5.20: Linear model interpolation. Bad results obtained by searching local maxima (a) versus better interpolation considering differences in thrust variation (b).

the geometrical area is not delimited exactly by the four points computed before, that were useful only to find interpolation coefficients. It is in fact necessary to know also the mean ramp-up and ramp-down time transients t_{ru} and t_{rd}, which are then used later on. An outline of the fundamental time parameters is represented in Fig. (5.22); the real pulse-width length is almost equal to the imposed one increased by the final tail transient, but not exactly since in the model mean values are used. Note that the real $pWidth$ is unknown, but estimated through the summation of the imposed $pWidth$ with the mean $pWidth_{diff}$ computed. The time interval in which the thrust value is considered in steady-state is instead called t_{steady}.

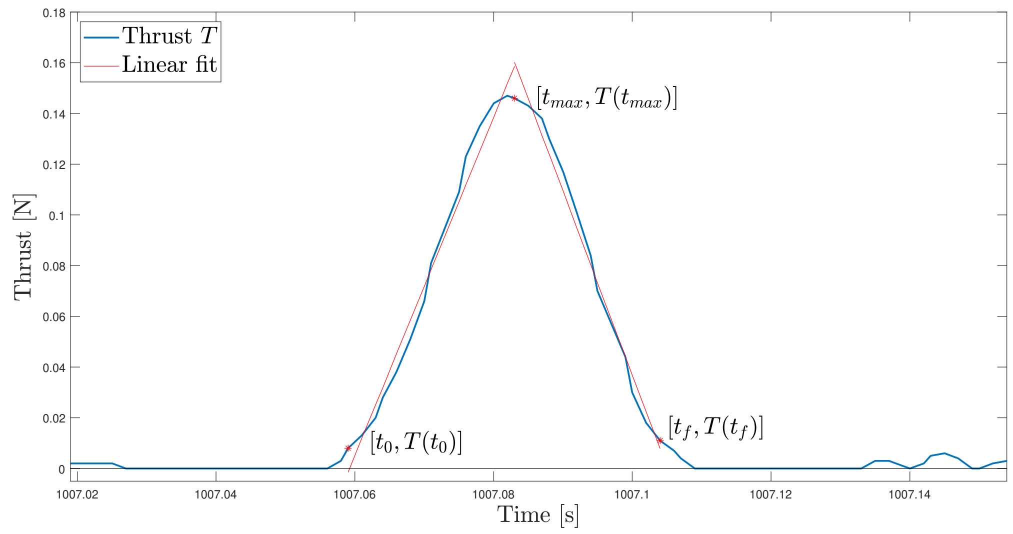


Figure 5.21: Interpolation for $t_{max,i}$ and $t_{max,f}$ coincident.

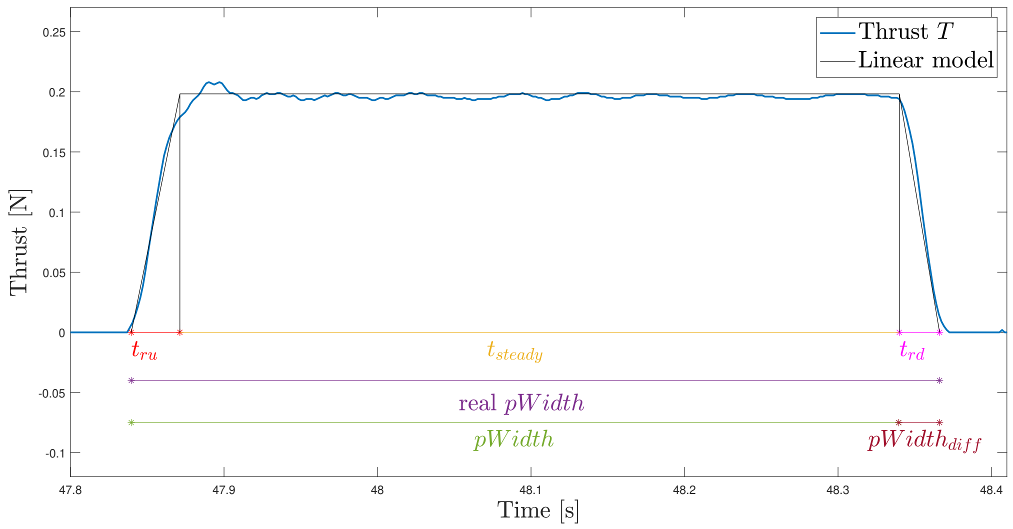


Figure 5.22: Outline of time parameters used in the linear model: ramp-up time transient t_{ru}, ramp-down time transient t_{rd} and and steady state time interval t_{steady}; imposed pulse-width $pWidth$, real pulse-width *real* $pWidth$ and their difference $pWidth_{diff}$.

The area (that corresponds to the impulse searched) is finally computed through the geometrical model adopted, represented in Fig. (5.23), which uses constant coefficients a_{ru} and a_{rd} and constant time transients t_{ru} and t_{rd} (Tab. 5.9). Actually "trapezoidal area" is not a proper definition, since it is computed as the sum of the area of the two triangles increased by the one of the middle rectangle; the reason is that using the parameters described the steady-state value of the thrust in the model does not necessarily coincide with the product $|a_{rd}| \cdot t_{ru}$, that represents the height of the second triangle. The mathematical model can be summarized in Eq. (5.9a), but it is valid only if the *real* $pWidth$ is lower than the sum of t_{ru} with t_{rd}, otherwise Eq. (5.9b) must be used since the spike shape is obtained.

$$T(t) = \begin{cases} a_{ru} \cdot t & \text{if } 0 \le t \le t_{ru} \\ a_{ru} \cdot t_{ru} & \text{if } t_{ru} < t < (t_{steady} + t_{ru}) \\ -|a_{rd}| \cdot t & \text{if } (t_{steady} + t_{ru}) \le t \le real\ pWidth \end{cases} \tag{5.9a}$$

$$T(t) = \begin{cases} a_{ru} \cdot t & \text{if } 0 \le t \le t_{ru} \\ -|a_{rd}| \cdot t & \text{if } t_{ru} \le t \le real\ pWidth \end{cases} \tag{5.9b}$$

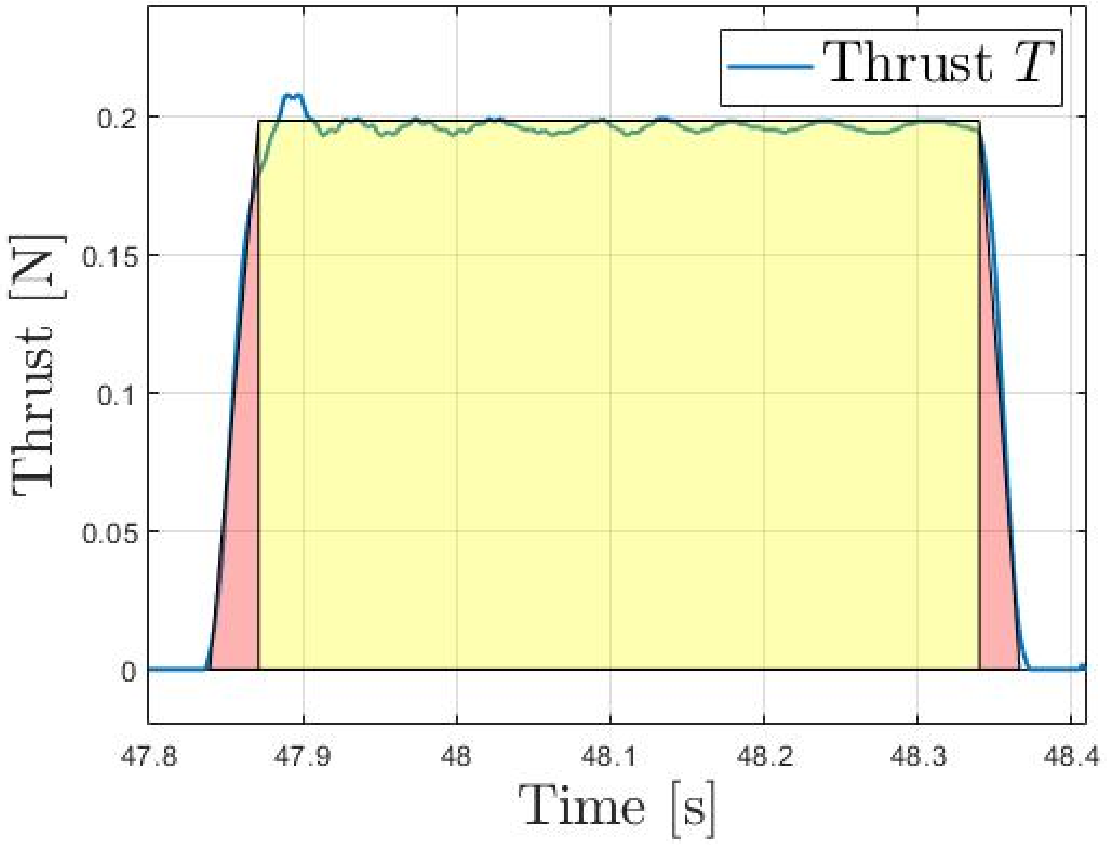


Figure 5.23: Geometrical model adopted to compute the total impulse.

The model adopted brings to a good approximation of the single impulse, but once applied to the whole set of data errors are higher (Fig. 5.24a); that is why some corrective factors are needed, found by computing linear regression parameters (Eq. 5.10) between the area resulted from the integration, considered as the real area, and the area found with the proposed model, which is the approximated one. A first-degree approach has been chosen by evaluating the relation between areas, that clearly shows a linear trend (Fig. 5.25). By applying Eq. (5.11) with the coefficients r_1, r_2 that satisfy Eq. (5.10) the model is able to correct imprecisions and keep the final average error below a threshold of 1% (Fig. 5.24b); some overshoots may be present mainly because of the high variability of recorded data, the low sensibility of the load cell and experimental errors that can compromise some results, but are only a few isolated cases. The trapezoidal model with constant parameters is applied also when the $pWidth$ is very low and the thrust assumes a spike shape, even if the computational error is in this case bigger: as a general rule, by using the proposed model for the area computation, the error obtained is inversely proportional to the $pWidth$ length. Switching to a "pyramidal" approach for lower pulse-widths, where the area is computed only as the sum of two triangles so without the steady-state part, would of course bring to better local results but would also change the trend of the relation between integrated and geometrical area, invalidating the possibility of performing a simple linear regression. For these reasons, the thrust can be retrieved with the model of Eq. (5.9), but the area is always computed with the geometrical model represented in Fig. (5.23) and corrected with the computed regression coefficients. Nevertheless regression is performed on two different range of values, setting the discriminating value for the imposed pulse-width at 25 ms; this value is chosen by looking at the trends of interpolation coefficients a_{ru} and a_{rd} (Fig. 5.26), which show a significative different behaviour for $pWidth < 25\ ms$; corrective factors in this case are indicated as r_1^{25} and r_2^{25}. Tab. (5.9) contains the results that must be saved in order to use the model for real applications.

$$A_{model} = r_1 \cdot A_{integrated} + r_2 \tag{5.10}$$

$$A_{model,\ corrected} = \frac{1}{r_2}(A_{model} - r_2) \tag{5.11}$$

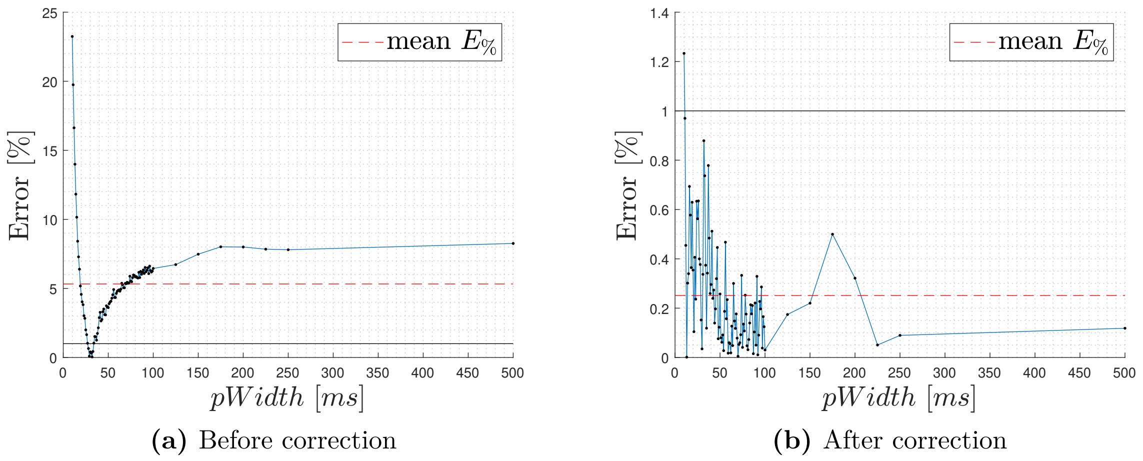


Figure 5.24: Relative error between total impulse computed with the model and the real one computed with integration of recorded data before (a) and after (b) regression corrective factors are applied ($P_{work} = 3\ bar$). After correction, mean error (dashed line in red) is always <1% (in black).

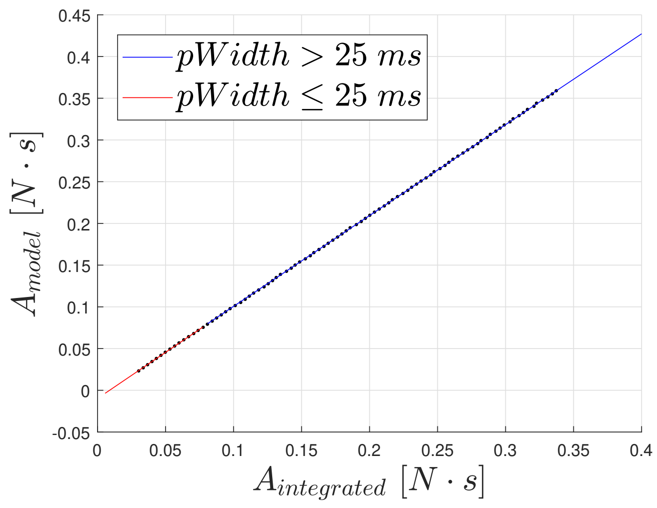


Figure 5.25: Relation between the integrated area and the area found with the geometrical model with regression representation.

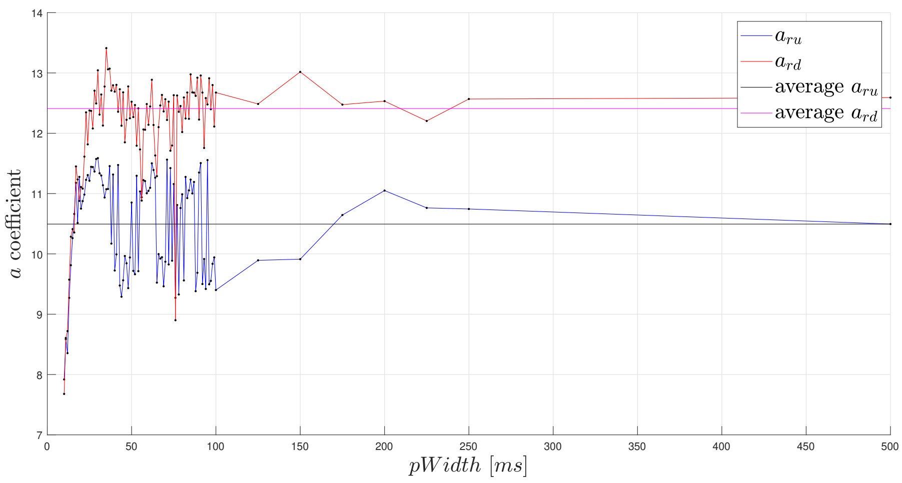


Figure 5.26: Example of linear fit coefficients behaviour ($P_{work} = 3\ bar$); specific values change according to the pressure tested, but the trend remains always similar.

$P\ [bar]$	a_{ru}	a_{rd}	$t_{ru}\ [s]$	$t_{rd}\ [s]$	$pW_{diff}\ [s]$	r_1	r_2	r_1^{25}	r_2^{25}	$\bar{E}_{\%}$
2	5.80	7.00	0.33	0.39	0.03	1.00	0.00	1.13	-0.00	0.34
3	10.50	12.41	0.04	0.03	0.29	1.09	-0.01	1.11	-0.01	0.25
4	14.89	18.43	0.04	0.03	0.03	1.12	-0.02	1.15	-0.03	0.22
5	14.89	18.43	0.04	0.03	0.03	1.12	-0.02	1.15	-0.03	0.22
5,5	23.00	28.83	0.04	0.03	0.03	1.09	-0.02	1.16	-0.04	0.59
6	27.21	30.71	0.03	0.03	0.03	1.04	-0.00	1.08	-0.02	0.87

Table 5.9: Linear model for impulse integration results; note that the mean relative error $\bar{E}_{\%}$ is always lower than 1% (preciser tabulated values with more digits are used for the model).

5.3.4 Multiple Impulses

The last case that has not been analyzed yet is when the pulse-width modulation period imposed is so low that the thrust does not reach the zero level between the successive closing and re-opening of the control valve, resulting in an overlap of the impulses (Fig. 5.27). In theory, this condition would occur only when the period is lower than or equal to the pulse-width length, but this can not be the actual case since such a combination of values was automatically discarded during the tests, by selecting only different combinations. In reality, the fundamental parameter to be checked is the real pulse-width, bringing to the overlapping situation whenever the condition of Eq. (5.12) is verified.

$$pwmPrd \leq real\ pWidth \tag{5.12}$$

The linear model proposed in Sec. (5.3.3) is not able to provide acceptable results for this case; one reason is that linearity could be compromised when impulses are really close each other, giving rise to the necessity of considering a different approach to treat such cases. The other cause is that the relation between the integrated area and the approximated one obtained with the model has a different trend with respect to the cases treated before, so the corrective factors computed for the model are able only to partially fix the value; average relative error is, in fact, kept lower than 10%, but these values are too high to consider the model as reliable.

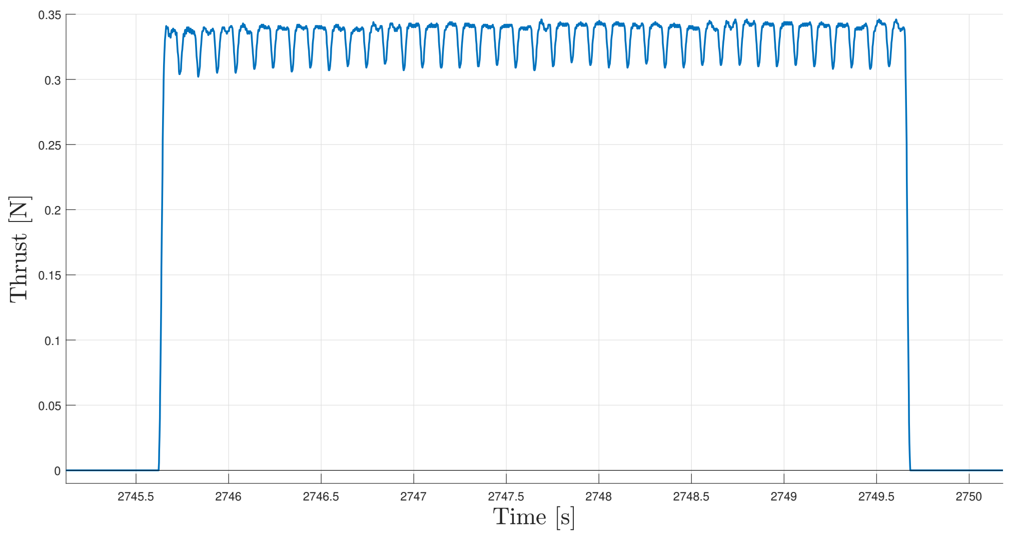


Figure 5.27: Example of impulses overlapping, with $P = 3\ bar$, $pWidth = 100\ ms$, $pwmPrd = 97\ ms$, $real\ pWidth = 126\ ms$.

5.3.5 Conclusions

The approach used in order to compute a mathematical model for the impulse integration has been extensively explained in a very detailed way, for the purpose of giving a clear complete picture of the approximations done. Nevertheless, it is mandatory to recap the fundamental passages needed to apply the model, which result to be the few simple operations reported hereafter.

The input and output quantities are expressed by Eq. (5.5). At first, knowing the operative pressure P_{work}, it is possible to retrieve the relative coefficients from Tab. (5.9). Given then the *pWidth* parameter, the corresponding *real pWidth* can be computed with Eq. (5.13).

$$real\ pWidth = pWidth + pW_{diff} \tag{5.13}$$

The next step consists in checking if the hypotheses of the model are verified and the specific case to be considered. In all the conditions, initially the area (i.e. the total impulse) is computed with Eq. (5.14) that corresponds to what depicted in Fig. (5.23). After that, the area value is corrected with Eq. (5.11) using the suitable coefficients, which are different according to the cases summarized in Tab. (5.10). In this way, the final corrected area is computed.

$$t_{steady} = real\ pWidth - (t_{ru} + t_{rd}) \tag{5.14a}$$

$$A_{model} = \frac{1}{2}(a_{ru} \cdot t_{ru}) + a_{ru} \cdot t_{steady} + \frac{1}{2}(a_{rd} \cdot t_{rd}) \tag{5.14b}$$

Case	Corrective coefficients
$pWidth > 25\ ms$	r_1, r_2
$pWidth \leq 25\ ms$	r_1^{25}, r_2^{25}

Table 5.10: Different sets of corrective coefficients to be used in total impulse computation according to the specific case.

Finally, the *pwmPrd* is used to check the condition of Eq. (5.12): if it is satisfied, then the impulses are overlapping. The model proposed computes the total impulse with the exact same passages explained above, but results are retrieved with a mean estimated error $< 10\%$, thus not reliable.

If instead one wants to compute the time variation of the thrust in time and is not interested in the final impulse, Tab. (5.11) is used as reference.

Case	Model equation
$real\ pWidth > (t_{ru} + t_{rd})$	(5.9a)
$real\ pWidth \leq (t_{ru} + t_{rd})$	(5.9b)

Table 5.11: Different model equation to be used for the computation of thrust time variation according to the specific case.

5.4 Further Development

The analysis performed has permitted to advance the state of the art of the actuators used for the control of the DANCE facility, proceeding with the work that will shortly bring to the possibility of performing functional tests that involve the whole vehicle, or at least the AP. Main features of the innovations introduced are here summarized in detail, giving proposals and suggestions for future development, in order to have a linear continuation of the project:

- **Flywheel:** the new design proposed fits the actual constraints in terms of the available space in the structural configuration of DANCER vehicle and increases the inertial mass helping solve the angular momentum saturation issue. Specific requirements have been set in order to facilitate the correct future realization and assembling with the motor. About the precision on the mass distribution, the tolerance declared by the manufacturer has to be forcibly trusted since no testing equipment is available at the DAER laboratory; in fact measuring the center of mass offset would require an electromechanical balancer able to determine the axial force during the flywheel rotation and compute the relative radial run-out with a sensibility on the order of μm. Without such preliminary verification, functional tests with fully assembled reaction wheels must be done carefully, since they could lead to an engine failure. Once flywheels are ready, the press-fit for the motor assembling must be performed respecting the maximum loads and requirements discussed in Sec. (5.2.2).

- **Motor and driver wiring:** The connections between electrical components have been facilitated by soldering prototype circuits, also reducing the required space for their positioning on the AP and increasing the overall robustness. For a future setup, machine printing of the circuit could be the final solution if the budget permits it.

- **Reaction wheels structural support:** The plastic part needed to support the reaction wheels in operative conditions and to attach them to the DANCER AP has been re-evaluated in the light of the necessity of performing the press-fit coupling between the flywheel and the motor's shaft. Different solutions have been proposed, explaining in detail advantages and drawbacks for each one of them; with the actual knowledge the best solution is indicated, but the trade-off choice is not confirmed since a stress analysis has not been performed yet and is highly recommended before manufacturing the part.

- **Thrusters:** Pneumatic tests were performed with a lot more different configurations with reference to the previous experiments, allowing refinement of the mathematical model to be used during the DANCER control. Giving in input the selected pulse-width and the period, the model is able to retrieve the real impulse that is obtained with an estimated relative error lower than 1%, by using tabulated values computed on a statistical basis on the data collected during the tests. Anyway, the algorithm has been validated only with the available data, and need to be tested first on the DANCER simulator and successively during real applications; such tests will determine if the model is accurate enough or if it needs even more refinement. Moreover, the model is suitable only if one of the tested pressure (2, 3, 4, 5, 5.5

or 6 *bar*) are used; if different working conditions are necessary, tests may have to be repeated. Algorithm limits are reached also in another case, that happens when multiple overlapping impulses are imposed; in this situation, the estimated relative error is kept below 10%, thus more research is surely required if operating in this condition will be needed.

At the moment the battery packs for the autonomous movement of the TP are not available yet; nevertheless, once reaction wheels will be manufactured and tested, the AP can be finally set into operative mode leaving the supply voltage system connected to an external source. About the thrusters, the components for the pneumatic system are already available and need only to be mounted; burst tests are still required to verify the correct working of the pressurized cylinders before connecting them to the final circuit (for details see [4, 5]).

6 Conclusions and Future Work

6.1 Improvements & Results

On the final section of each chapter that contains the key passages of this thesis work are explained in details the goals achieved, the contributions on the whole progress of DANCE project growth, the limitations on obtained results and the indications for future development, together with suggestions on how to improve the quality of the outcome. With reference to the state of the art at the beginning of the present activity [4, 5], several improvements have been introduced that can be summarized as follows, maintaining the organization order of the thesis:

- The set of MARG sensors to be used in combination with the TI LaunchPad has been selected, adopting the BOOSTXL plug-in module for the present development.
- The software for correct activation, setting, usage and real-time reading of the sensors has been implemented keeping into account the compatibility with the TI LaunchPad and permitting the reciprocal communication. In this way integration between microcontroller and sensors has been successfully achieved.
- The algorithm for collection and recording of telemetry data through the serial communication protocol has been written, consenting real-time analysis as well as post-processing computations.
- Calibration procedures for gyroscope, accelerometer and magnetometer have been defined, with step-by-step specifications for fast performing and usage of the developed software. Procedures are dedicated to the particular sensors set used but can be generalized to any MEMS sensor by partially modifying the code instructions.
- Three alternative solutions are proposed for the attitude estimation problem; for each one, a complete Simulink model has been written for the usage with the integrated electronic hardware.
- MATLAB version of the code is also developed for the two introduced filtering algorithms (Mahony and Madgwick filter), together with an optimization procedure for finding suitable tuning coefficients.
- Design and assembling of the reaction wheels have been re-evaluated, expressing the mandatory requirements for the flywheel manufacturing, testing and facilitating the wiring between the motors and the dedicated drivers and proposing different conceptual solutions for the realization of the new structural support.
- Experimental activities on the pressurized air actuators (thrusters) have been repeated increasing the number of tested conditions; collected data have been elaborated to refine the model and improve the precision on coefficients for the total impulse computation.

6.2 Future Activities

Despite the present advancement and successful results obtained, together with the remarkable previous efforts on which this thesis work is based, much more tasks have to be performed before coming to a final functioning version of the first DANCER vehicle. Nevertheless, even if it is hard to set a precise timeline, passing from the actual state to the first operative tests involving the AP should be pretty straightforward: the very next step to be done consists in the full integration of software and avionics already developed with the actuators for the attitude control. As a general guideline, the algorithms for attitude estimation can be joined with those for reaction wheels and thrusters control (implemented in [5]); after prototype wirings are made, the expected response can be verified. Once results are satisfactory all the devices and components can be assembled on the existing structure of the vehicle, consequently opening the road to the first real experiments of the project in its totality, even if the absolute optical tracking reference is still missing. The best way to ease and accelerate the progress toward the final objective is to give precise suggestions and technical advice on the next small steps to be done (in addition to previous indications [4, 5]), also to ensure better continuity of the work; thus next sections are dedicated to this purpose.

6.2.1 Serial Communication

At the actual stage, the serial communication algorithms for telemetry data downloading are completely operative, but a physical connection via USB cable between the microcontroller and the computer is still necessary; in future applications, it is mandatory to use a RF module for wireless data transmission and consequently to integrate the existing software with the new RF device.

6.2.2 Calibration Procedures

The procedures proposed and implemented for the calibration of the sensors have shown to be effective and to give acceptable results in terms of the expected performance, nevertheless small improvements can be done:

- Precise repeatability of magnetometer sample data collection can be ensured with a purely mechanical device (without any electronic parts) capable of holding the sensor while performing a complete mapping of the three-dimensional ellipsoid with uniformly distributed measurements; a prerequisite for this procedure is the use of the aforementioned RF module.

- Automatic saving for the values gains and offsets on the proposed low-cost external flash memory can be implemented, further speeding up the whole procedure.

- To perform the calibration of a sensor the dedicated software has to be deployed to the TI LaunchPad; after the procedure has been completed, the attitude estimation algorithm must be downloaded again on the microcontroller. This forced software switching can be avoided by integrating all the different software on a single file and predisposing a suitable user input that permits to select to the wanted routine, using, for example, a simple electronic switch or a button and exploiting the LaunchPad LEDs for visual feedback.

Calibration quality for gyroscope and accelerometer can hardly be increased since it would need to perform a comparison procedure with the consequent need of preciser expensive reference instruments, especially if the cross-axis non-linearity of the gyroscope must be tested; in this case, the whole approach here proposed would have to be changed. Keeping the same software, requirements and procedures here developed a small improvement can be done for the accelerometer, thinking about a preciser reference for setting the device in the desired position; as remarked many times in fact the requirement for procedure validation had to be relaxed of one order of magnitude, because of experimental errors.

6.2.3 Attitude Estimation

The three alternative estimation algorithms proposed have not been tested yet with the full DANCER simulator Simulink model; in the author's opinion, it would not add any assets with reference to the results already obtained since real sensor would not be used and the filters' correct mathematical behaviour has already been tested with a lighter simulator. Anyway, it could be an additional fast test to be done, adding more general robustness to the whole project. Moreover, the final trade-off decision among the three filtering algorithms will not be possible up to the moment in which motion tests are done with the real vehicle. What has surely to be done is a preciser tuning of the gain parameters; procedures are already available but the system for dynamic absolute referencing has to be developed. In the final version of the project, cameras must be employed to follow the DANCER motion inside the testbed facility and validate the autonomous attitude control of the vehicle, so the same optical system could be used for the precise adjustment of the filters dynamical behaviour.

6.2.4 Reaction Wheels

With the currently developed design and requirements set, the flywheel can be manufactured. A structural FEM analysis is required for the support of the electric motors before proceeding in its prototype realization with 3D printing technology; after stress analysis has been done, a final choice among the proposed solutions is feasible.

6.2.5 Thrusters

The mathematical model for the thrusters has been refined and it is potentially ready to be used to retrieve the total real impulse estimation starting from the given ideal control input. The model has shown to be valid only in the case in which separate impulses are performed (i.e. when the pulse-width of the electric signal is shorter than the period with the air valve opened) and instead to fail when multiple impulses overlap. This is probably due to the loss of linearity, thus the existing model should be extended if this case turns out to be necessary for the DANCER attitude control.

6.3 Conclusions

The initial objective of contributing in a substantial way to the project development can be considered to be achieved since, as explained and shown, several major improvements have been introduced especially in the use of the avionics and about its software development. Besides the brief description of a general guideline, the specific activities here exposed are strictly related to the continuity and refinement of the results obtained with the present work. However not all the challenges highlighted in previous elaborations ([4], [5]) have already been solved, so for completeness it is mandatory to still take into account those indications.

A

A.1 Statistical Indicators Description

Parameter	**Description**	**Unit of measurement**
σ	Standard Deviation or Root Mean Square	$[\sim]$
σ^2	Variance	$[\sim^2]$
n_{nD}	Noise density	$[\sim/\sqrt{Hz}]$
n_{nD}^2	Noise power	$[\sim^2/Hz]$

Table A.1: Summary of statistical parameters used ("~" represents the specific unit of measurement).

Standard deviation definition for the generic x quantity:

$$\sigma = \sqrt{\frac{1}{N} \cdot \sum_{i=1}^{N} \Big(x(i) - \overline{x}(i) \Big)^2} \tag{A.1}$$

Equivalent "standard deviation for calibrated values" definition:

$$\sigma_{cal} = \sqrt{\frac{1}{N} \cdot \sum_{i=1}^{N} \Big(x_{calibrated}(i) - x_{expected}(i) \Big)^2} \tag{A.2}$$

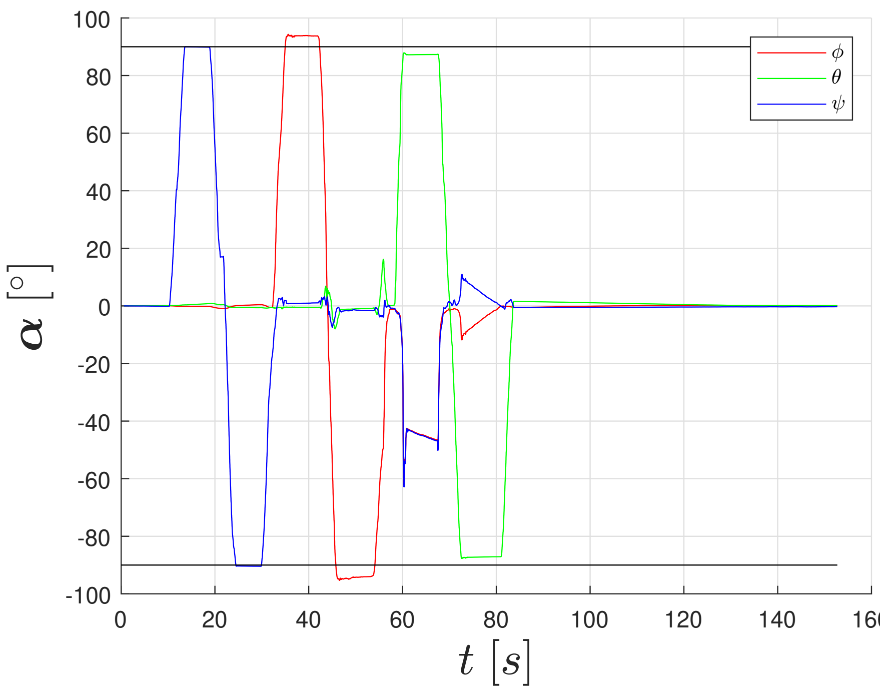
B

B.1 Madgwick Filter Optimization

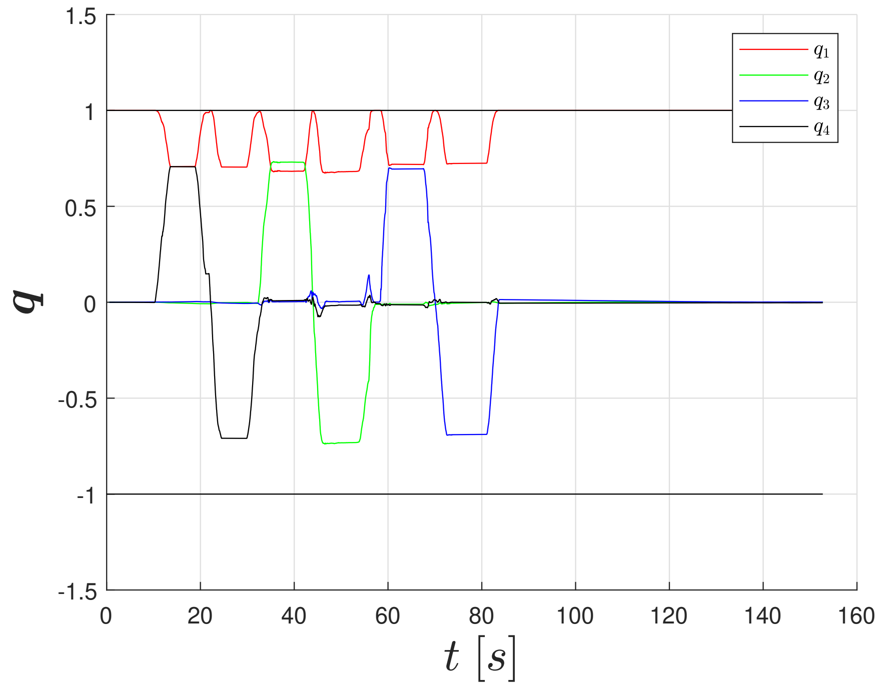
B.1.1 Optimal Tuning

	Parameter	Range	Result
Iteration 1	β	[0.0 2.0]	$0.5541{\cdot}10^{-4}$
$RMS_{error} = 26.7252$	ζ	[0.0 2.0]	$5.2179{\cdot}10^{-3}$
Iteration 2	β	[0.0 0.1]	$6.0368{\cdot}10^{-4}$
$RMS_{error} = 0.9990$	ζ	[0.0 0.1]	$6.4132{\cdot}10^{-6}$
Iteration 3	β	$[0.0\ 10^{-3}]$	$6.3014{\cdot}10^{-4}$
$RMS_{error} = 0.9971$	ζ	$[0.0\ 10^{-5}]$	$2.8316{\cdot}10^{-6}$

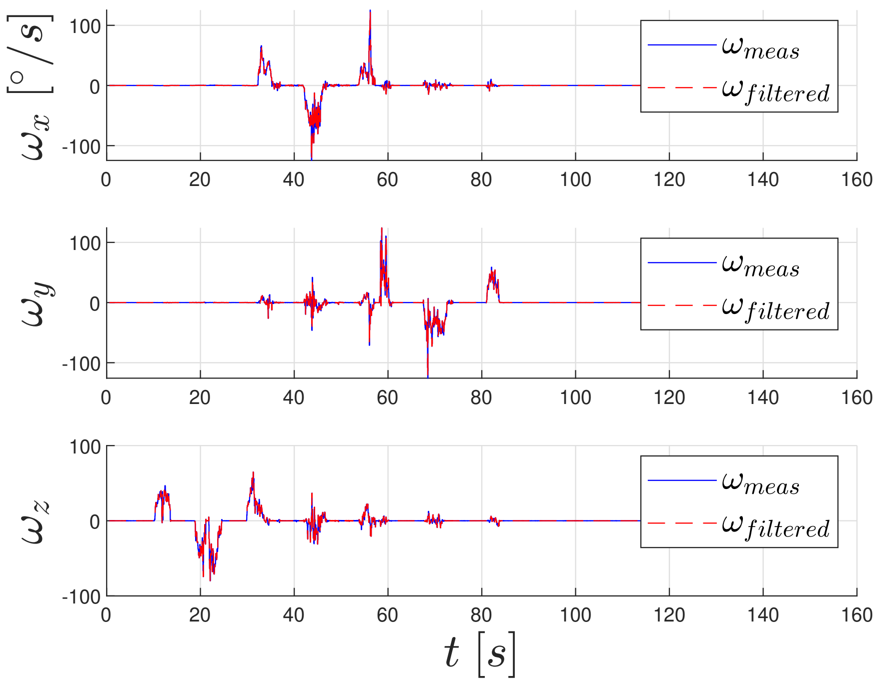
Table B.1: Madgwick filter iterations to find optimal gains β, ζ using the Mahony estimated quaternion as reference for the RMS_{error} computation.



(a) Euler angles



(b) Quaternions



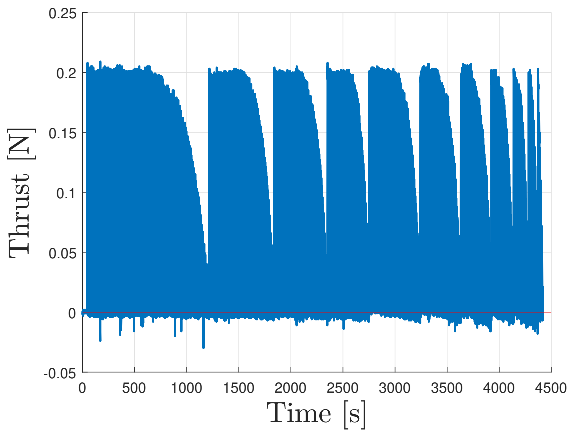
(c) Filtered vs measured angular rate

Figure B.1: Attitude reconstruction and angular rate estimation obtained with Madgwick filter after the optimization of β and ζ found using the the Mahony estimated quaternion as reference for the RMS_{error} computation.

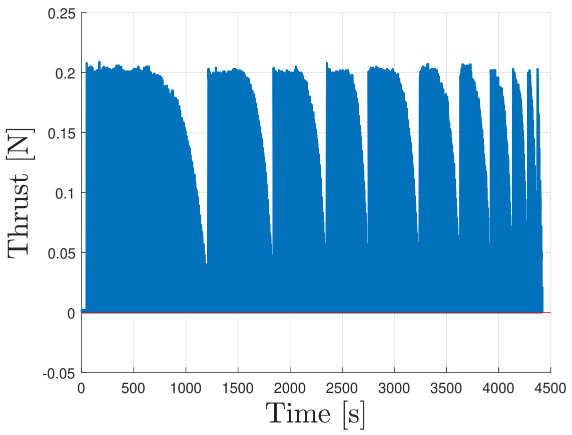
C

C.1 Pneumatic Tests

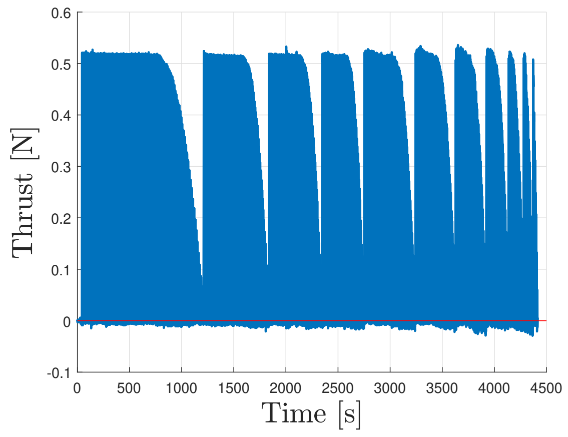
C.1.1 Noisy vs. Filtered Measurements



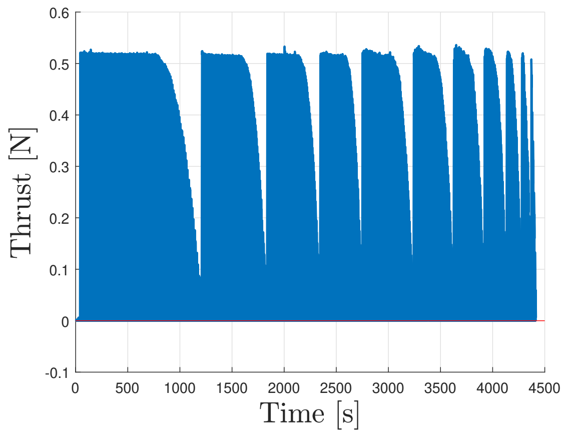
(a) Rough data ($P_{work} = 2\ bar$)



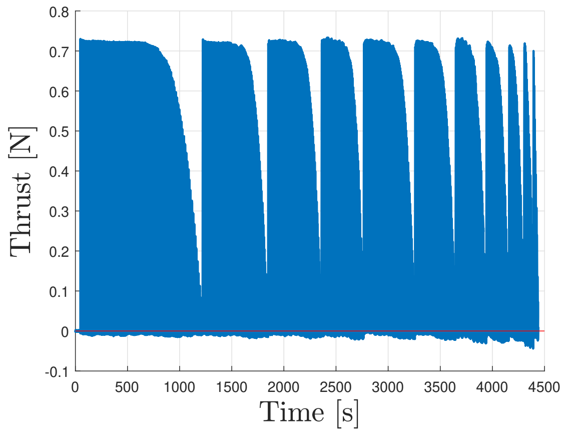
(b) Filtered data ($P_{work} = 2\ bar$)



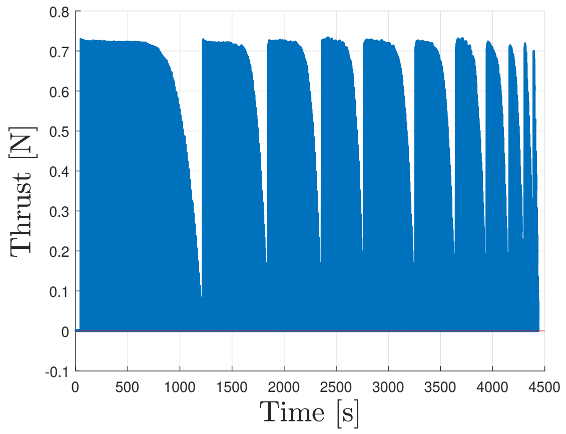
(c) Rough data ($P_{work} = 4\ bar$)



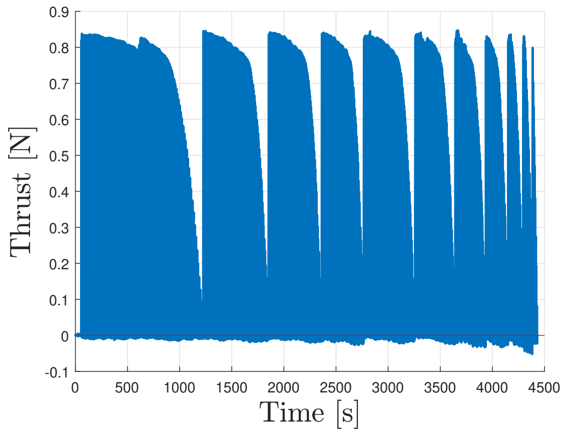
(d) Filtered data ($P_{work} = 4\ bar$)



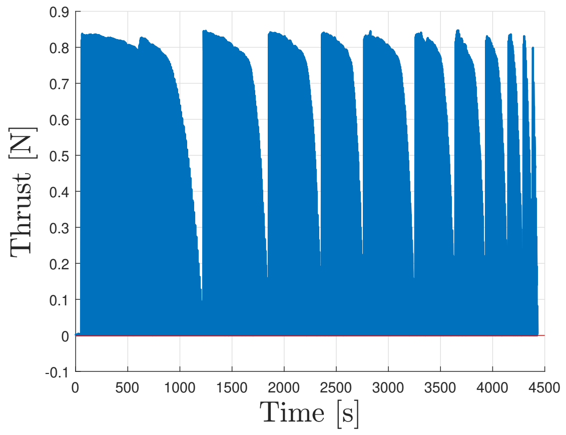
(e) Rough data ($P_{work} = 5\ bar$)



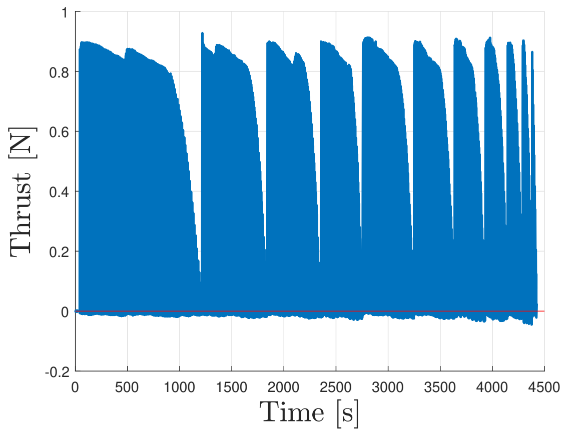
(f) Filtered data ($P_{work} = 5\ bar$)



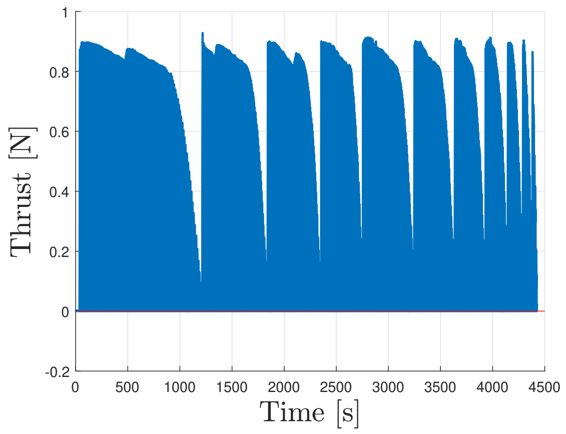
(g) Rough data ($P_{work} = 5.5\ bar$)



(h) Filtered data ($P_{work} = 5.5\ bar$)



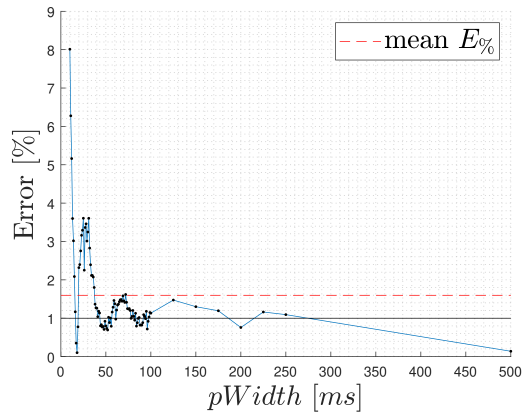
(i) Rough data ($P_{work} = 6\ bar$)



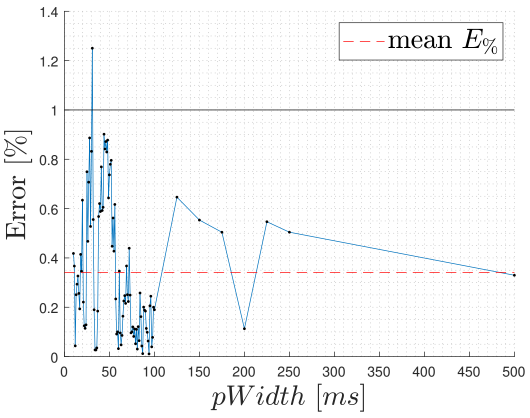
(j) Filtered data ($P_{work} = 6\ bar$)

Figure C.1: Rough and noise filtered measurements for different working pressures, using the tolerance tol_{filter} imposed.

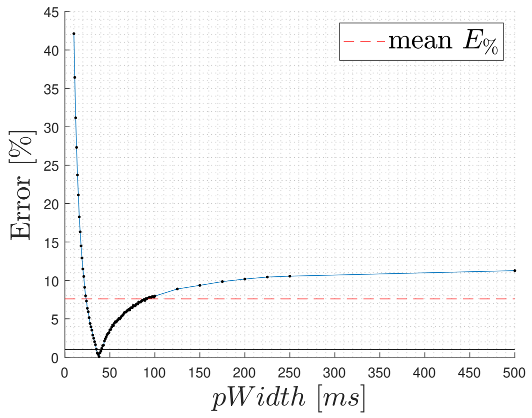
C.1.2 Relative Error in Impulse Computation



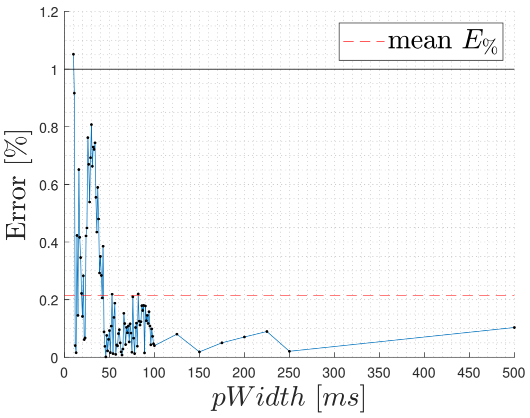
(a) Before correction ($P_{work} = 2\ bar$)



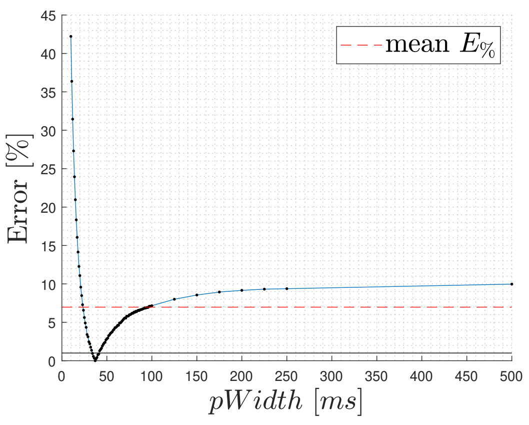
(b) After correction ($P_{work} = 2\ bar$)



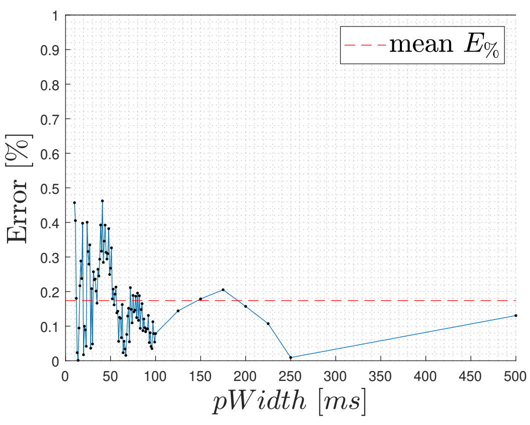
(c) Before correction ($P_{work} = 4\ bar$)



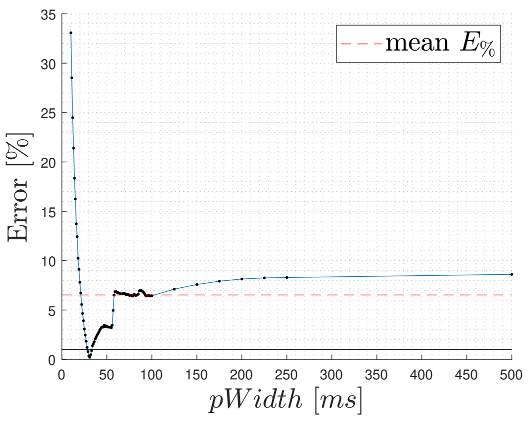
(d) After correction ($P_{work} = 4\ bar$)



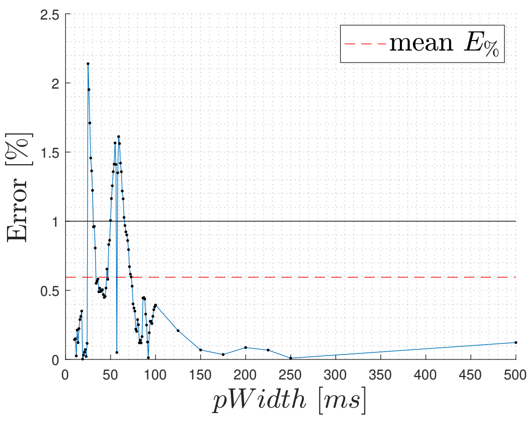
(e) Before correction ($P_{work} = 5\ bar$)



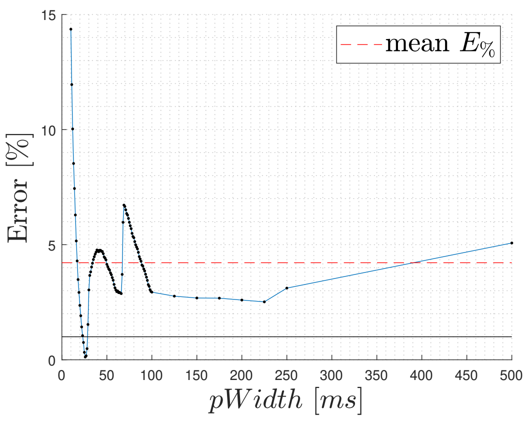
(f) After correction ($P_{work} = 5\ bar$)



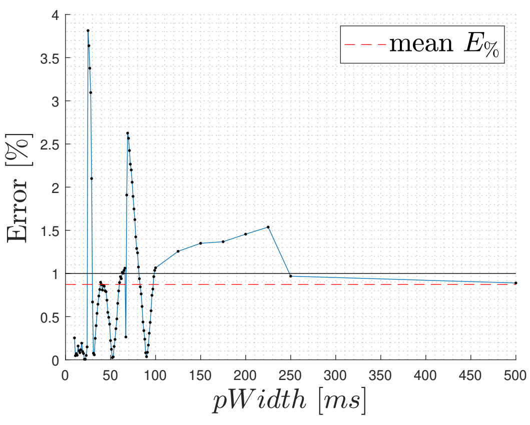
(g) Before correction ($P_{work} = 5.5\ bar$)



(h) After correction ($P_{work} = 5.5\ bar$)



(i) Before correction ($P_{work} = 6\ bar$)



(j) After correction ($P_{work} = 6\ bar$)

Figure C.2: Relative error between total impulse computed with the model and the real one computed with integration of recorded data before and after corrective factors are applied, for different working pressures. After correction, mean error (dashed line in red) is always <1% (in black).

D

D.1 Development Tools for TI LaunchPad

D.1.1 Code Composer Studio IDE

The Texas Instruments C2000 MCU F28379D LaunchPad can be programmed using C, C++ or Assembly programming languages, developing the code to be deployed to the target hardware directly with the Code Composer Studio Integrated Development Environment (IDE). The software is available on the TI website at `http://www.ti.com/tool/CCSTUDIO`, together with the documentation for its usage. Related drivers, libraries and sample codes are included in the TI controlSUITE^{TM} Software Suite for C2000 Microcontrollers, available at `http://www.ti.com/tool/CONTROLSUITE`.

D.1.2 Simulink Embedded Coder for TI LaunchPad

The TI LaunchPad can also be programmed directly from Simulink, in fact this approach has been used in the present thesis work. The rationale is to generate the C, C++ code from a Simulink model and to deploy it to the target hardware (please read the "troubleshooting" section at the end of this page before proceeding with installations).

The first prerequisite is to install both the TI CCS IDE and controlSUITE^{TM} (Appendix D.1.1). The following libraries are then needed, to be installed in order directly from the MATLAB Add-On Explorer:

1. "MATLAB Coder": compiler that allows generating C and C++ code from MATLAB code.
2. "Simulink Coder": compiler that allows generating C and C++ code from Simulink code.
3. "Embedded Coder": adds support for custom targets to the Simulink Coder.
4. "Embedded Coder Support Package for Texas Instruments C2000 Processors": adds support for the specific family of microcontrollers, including the F28379D LaunchPad.

Troubleshooting: a common error encountered during the code building from a Simulink model (after the installation of all the indicated software) is reported with: `"NMAKE : fatal error U1073: don't know how to make ..."` or similar. This is due to the MATLAB (and consequently of all the add-on) installation folder which contains a space in its name (default directory is `C:/user/program files/MATLAB/`); the easiest way to solve this problem is to change the MATLAB installation folder. Other solutions can be: setting the working OS (like Windows) to use short names, create a symbolic link to the MATLAB installation folder or (not suggested) modify the template makefile. For more information, please see the dedicated MathWorks® support page "Build Process Support for Folder Names with Spaces or Special Characters".

Acknowledgement

Nel concludere la mia carriera universitaria vorrei poter ringraziare tutte le persone che mi hanno accompagnato in questi anni, ma è sicuramente impossibile farlo in maniera adeguata. Meriterebbero di essere elencate per nome e cognome ma, oltre ad essere una lista molto lunga, conoscendomi finirei sicuramente per dimenticare qualcuno. Cercherò quindi di fare del mio meglio per esprimere tutta la mia gratitudine.

Vorrei innanzitutto dire che per me questo traguardo rappresenta una conquista importante, un obiettivo che sempre mi è sembrato distantissimo ma, con un velo di malinconia, mi accorgo ora che il momento è già arrivato. Sono senza dubbio felice e soddisfatto di essere arrivato fino a qui, e posso riconoscere che senza l'appoggio e la compagnia delle persone a me care sarebbe stato un viaggio indubbiamente più difficile e non così piacevole come fortunatamente lo è stato.

La dedica più grande va sicuramente a mio padre Lucindo e mia madre Manuela, senza i quali tutto ciò non sarebbe stato possibile. Hanno costantemente creduto in me anche quando io stesso ho dubitato delle mie capacità, hanno sostenuto ogni mia scelta giusta o sbagliata che fosse e, da sempre, rappresentano la mia più grande fonte di sicurezza. Senza dimenticare il resto della famiglia: mia sorella Chiara, che per prima ha sempre aperto la strada ed è probabilmente grazie a lei che ho assorbito la voglia di studiare ed impegnarmi, e la nonna Mimi, che trova in ogni situazione il modo di esaltare la mia parte migliore.

Un enorme grazie a tutti i miei amici, prima di tutto perchè mi sopportano e perchè mi hanno permesso di non impazzire mentre preparavo gli esami, e poi perchè sono i protagonisti di tutte le mie esperienze più belle. E' grazie a loro se sono quel che sono oggi e non potrei immaginare la mia vita senza la loro compagnia.

Un grazie speciale va ai miei coinquilini, con i quali ho condiviso più direttamente la vita universitaria e che hanno conosciuto ogni lato del mio carattere.

Grazie anche a tutto il parentame, un sostegno silenzioso ma costantemente presente e con il quale è sempre un piacere condividere i momenti felici.

Infine, ma non meno importante, vorrei ringraziare la mia relatrice prof.ssa Michéle Lavagna per l'opportunità di crescita offertami e per avermi trasmesso la passione per l'ingegneria, tirando le somme di questi anni passati sui libri, Daniele e Stefano per il supporto durante questo lavoro finale, e il Politecnico di Milano per avermi fornito la conoscenza che, spero, mi sarà fondamentale in futuro.

04/2020, Milano

Riccardo D. G.

References

[1] NASA Jpl. "Formation Control Testbed", `https://dst.jpl.nasa.gov/test_beds/index.htm`.

[2] Jimmy Lau, Sanjay Joshi, Brij Agrawal, and Jong-Woo Kim. "Disturbance Filtering and Identification on the Naval Postgraduate School 3-Axis Spacecraft Simulator". *Collection of Technical Papers - AIAA Guidance, Navigation, and Control Conference*, 5, 2005.

[3] University of Padua. "SPARTANS Mini-satellite Simulator", `https://www.cisas.unipd.it/en/attivita/linee-di-ricerca/spartans`.

[4] P. Visconti. "DANCE: Experimental Platform for GNC Testing and Validation" Master Thesis. 2017.

[5] D. Ottolina. "Architectural developments and simulations for DANCE GNC facility" Master Thesis. 2019.

[6] S.O.H. Madgwick. "An efficient orientation filter for inertial and inertial/magnetic sensor arrays", `https://www.x-io.co.uk/res/doc/madgwick_internal_report.pdf`. 2010.

[7] R.Mahony, T. Hamel, J. Pflimlin. "Nonlinear Complementary Filters on the Special Orthogonal Group". IEEE Transactions on Automatic Control, Institute of Electrical and Electronics Engineers, 53 (5), pp.1203-1217, 10.1109/TAC.2008.923738 hal-00488376. 2008.

[8] BeagleBoard. "BeagleBone Black Wireless", `https://beagleboard.org/black-wireless`. 2010.

[9] Texas Instruments. "TMS320F2837xD Dual-Core Microcontrollers", `http://www.ti.com/lit/ds/sprs880l/sprs880l.pdf`.

[10] Texas Instruments. "TMS320F28379D LaunchPad Development Kit", `http://www.ti.com/lit/ml/sprui73a/sprui73a.pdf`.

[11] Texas Instruments. "LAUNCHXL-F28379D Overview", `http://www.ti.com/lit/ug/sprui77c/sprui77c.pdf`.

[12] Texas Instruments. "Code Composer Studio Integrated Development Environment for Texas Instruments Embedded Processors", `http://www.ti.com/tool/CCSTUDIO`.

[13] X. Shi, L. Lu, G. Jin, L. Tan. "Research on the Attitude of Small UAV Based on MEMS Devices", `https://doi.org/10.1063/1.4982459`.

[14] Texas Instruments. "BOOSTXL-SENSORS Sensors BoosterPack Plug-in Module" user's guide, `http://www.ti.com/lit/ug/slau666b/slau666b.pdf`. 2018.

[15] Bosch Sensortec. "BMI160 Small, low power inertial measurement unit" datasheet, BST-BMI160-DS00001-08 `https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bmi160-ds000.pdf`. 2018.

[16] Bosch Sensortec. "BMM150 Geomagnetic Sensor" datasheet, BST-BMM150-DS001-01, `https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bmm150-ds001.pdf`. 2019.

[17] "IEEE standard for binary floating-point arithmetic". *ANSI/IEEE Std 754-1985*, pages 1–20, 1985.

[18] UNI 4546. "Misure e misurazioni, Termini e definizioni fondamentali". 1984.

[19] International Society of Automation (ISA). "Principles of Calibration".

[20] STM Microelectronics. "AN4508 Application note: parameters and calibration of a low-g 3-axis accelerometer", `https://www.st.com/content/ccc/resource/technical/document/application_note/a0/f0/a0/62/3b/69/47/66/DM00119044.pdf/files/DM00119044.pdf/jcr:content/translations/en.DM00119044.pdf`. 2014.

[21] Z. Wu, W. Wang. "Magnetometer and Gyroscope Calibration Method with Level Rotation", DOI:10.3390/s18030748. 2018.

[22] STM Microelectronics. "DT0059 Design tip: Ellipsoid or sphere fitting for sensor calibration", `https://www.st.com/content/ccc/resource/technical/document/design_tip/group0/a2/98/f5/d4/9c/48/4a/d1/DM00286302/files/DM00286302.pdf/jcr:content/translations/en.DM00286302.pdf`. 2018.

[23] Winbond. "Spi flash 3V 16M-BIT serial flash memory with dual and quad SPI" datasheet, W25Q16DV. 2014.

[24] M. Giurato, M.Lovera. "Quadrotor attitude determination: A comparison study", DOI: 10.1109/CCA.2016.7587816. 2016.

[25] S. O. H. Madgwick, A. J. L. Harrison, and R. Vaidyanathan. "Estimation of IMU and MARG orientation using a gradient descent algorithm", DOI:10.1109/ICORR.2011.5975346. 2011.

[26] John L. Crassidis. "Optimal estimation of dynamic systems". 2004.

[27] Maxon Group. "Mile Encoder for EC 45 Flat" catalogue, DocID 1877473-06, `https://www.maxongroup.com/medias/sys_master/root/8837349802014/Productinformation-MILE-EC45flat-en-1877473-06.pdf`. 2019.

[28] Maxon Group. "Encoder MILE, 2 Channels, with Line Driver" catalogue, 256–2048 CPT, `https://www.maxongroup.com/medias/sys_master/root/8831393071134/18-402-EN-nov.pdf`. 2018.

[29] Maxon Group. "EC 45 Flat Brushless 30W Motor (with Hall Sensors)" catalogue, Part No. 339282, `https://www.maxongroup.com/medias/sys_master/root/8833813053470/19-EN-262.pdf`. 2019.

[30] Martin D. Hasha, NASA. "High-Performance Reaction Wheel Optimization for Fine-Pointing Space Platforms", https://ntrs.nasa.gov/search.jsp?R=20160008147. 2019.

[31] Engineering Toolbox (2004). "Friction and Friction Coefficients", https://www.engineeringtoolbox.com/friction-coefficients-d_778.html. Accessed 19/03/2020.

[32] Engineering ToolBox, (2003). "Young's Modulus - Tensile and Yield Strength for common Materials", https://www.engineeringtoolbox.com/young-modulus-d_417.html. Accessed 19/03/2020.

[33] Engineering ToolBox, (2003). "Poisson's ratio.", https://www.engineeringtoolbox.com/poissons-ratio-d_1224.html. Accessed 19/03/2020.

[34] ISO. "Geometrical product specifications (GPS) — ISO code system for tolerances on linear sizes — Part 1: Basis of tolerances, deviations and fits", 286-1:2010, https://www.iso.org/standard/45975.html. 2010.

[35] E. Chirone, S. Tornincasa. "Disegno Tecnico Industriale vol. 2". 2014.

[36] Maxon Group. "Product Information: DC EC Drives up to ⌀10 *mm*" hardware reference, P/N 466023, https://www.maxongroup.com/medias/sys_master/root/8810888331294/MiniatureDrives-OPS-en.pdf. 2013.

[37] Maxon Group. "ESCON Module 24/2, Servo Controller" hardware reference, P/N 466023, https://www.maxongroup.nl/medias/sys_master/8815105572894.pdf. 2018.

[38] Silvent. "MJ40 Air Nozzle" product page, Item 0MJ40169380, https://www.silvent.com/en-eu/products/air-nozzles-en-eu/mj40-air-nozzle/. 2020.

[39] Emerson US. "TESCOM Regulators - Pressure reducing BB-1 Series" datasheet, DBB011761X012, https://www.emerson.com/documents/automation/catalog-bb1-series-pressure-regulator-oem-tescom-en-5385894.pdf. 2013.

[40] Texas Instruments. "Futek MODEL LSB200 Jr. Miniature S-Beam Load Cell" datasheet, https://media.futek.com/content/futek/files/pdf/productdrawings/fsh00102.pdf.

[41] Festo. "Proportional Pressure Regulators VPPM" Datasheet, Item VPPM-8L-L-1-G-14-0L10H-V1P-S1, https://www.festo.com/cat/en-gb_gb/data/doc_ENGB/PDF/EN/VPPM-G_EN.PDF. 2018.