Project on the design of an inertial measurement unit to be used in aerospace vehicles

Oriol Casamor Martinell

Directors : Joseba Quevedo and Manel Soria

Escola Tècnica Superior d'Enginyeries Industrial i Aeronàutica de Terrassa

A thesis submitted for the degree of

Grau en Enginyeria de Vehicles Aeroespacials

Terrassa, June 2015

This document contains : Memory

Acknowledgements

I offer my sincerest gratitude to my tutor, Joseba Quevedo Ph.D., who has supported me throughout my thesis with his patience and knowledge. Also to the co-tutor, Manel Soria Ph.D., for challenging and motivating me to get the best of myself and feel passionate with anything that I am working on.

To my colleagues in the students association EUROAVIA Terrassa , which have given all their support to me in my project, and specially Arnau Miró who has taught me much more than many teachers in the university.

To Terrassa Rocket Team for giving me their logistic support and experience to launch the rocket.

To the university, for creating the Inspire3 program, which has been a greenhouse for passionate students to go further the classes and the exams.

Abstract

Aiming to obtain aerospace vehicle flight parameters and engine characteristics, a DCM and magnetometer fusion algorithm is introduced in addition to a Kalman filter, in order to estimate position, velocity and acceleration in the vertical axis. This strategy is tested in the case of a model rocket, and the drag coefficient and thrust curve are calculated. The attitude of the vehicle is also estimated with fewer outcomes. Finally, It is described how the system could be upgraded so that trajectory in 3D can be obtained and therefore a better knowledge of the flight parameters.

Contents

List of Figures

List of Tables

Abbreviations

DCM Direction Cosine Matrix

FSR Full Scale Range

GND Ground

GPS Glopal Positioning System

IDE Integrated Development Environment

IMU Inertial Measurement Unit

INS Inertial Navigation System

MTOW Maximum Take Off Weight

PCB Printed Circuit Board

PWM Pulse Width Modulation

UAV Unmanned Air Vehicle

RMS Root Mean Square

SMD Surface Mount Device

TRT Terrassa Rocket Team

TOW Take Off Weight

ZFW Zero Fuel Weight

Part I

Theoretical approach

1 Introduction

1.1 Aim

The aim of this project is to develop a low cost inertial measurement unit in an electronic integrated system to be mounted in small aerospace vehicles such as a model rocket, a radio controlled airplane or sailplane, or a multirotor. Through the processing of the accelerations and rotations, it is possible to estimate the trajectory and speed of the vehicle. With this information, the performance of the vehicle can be analyzed and flight mechanics parameters can be estimated in an non invasive way, in comparison with other methods such as test stands or wind tunnels.

1.2 Scope

- Develop an electronic integrated data logger of accelerations, rotations, pressure and temperature.
  - Develop a data logging software for the electronics
  - Develop calibration functions for the sensors
  - Design and solder a prototype of the hardware
  - Design and order a PCB of the hardware
  - Validate, test and redesign if necessary the PCB
  - Develop a comprehensive documentation of the electronics so that in a future it can be used by other students in the future.
- Develop a system of calibrating the accelerometers, with known acceleration.
- Develop a system of calibrating the gyroscopes, with known angular velocity
- Develop software to be ran in a computer to process data files generated by the data logger
  - Sensor data files
  - Calibration data files
- Develop software to be ran in a computer to validate, filter and integrate the sensor data.
- Validate software results with a known trajectory case
- Estimate flight mechanic coefficients of the vehicle, preferably a model rocket.
- Obtain the thrust curve of the engine, in the case of model rocket, .
- Get conclusions about the quality of low cost sensors

1.3 Requirements

Geometry The most limiting case is the rocket model, therefore the embedded system must be able to fit in a cylinder of 4 cm of diameter and 10 cm of length, since they are the dimensions of the available model rocket.

Sample rate The minimum sample rate of the data logger depends on the response of the magnitude which is measured. The magnitude which changes fastest, the most critical, is the acceleration in the longitudinal axis of the rocket. Fortunately, TRT had some static tests of the engine that we will use in the experiments, ESTES-D12-7[1]. Results of TRT are shown in Figure 1.1. The thrust of the rocket according to the static measures of TRT (blue), the average value of this measures (green), and the manufacturer specifications of the engine (red).

According to Isermann (1989), if this process is considered a step response, the minimum sample period of the acquisition system can be calculated as follows,

$$T_s = \frac{T_{0.95}}{16}$$

Where $T_{0.95}$ is the required time to get 95% of the stable final value.

Because it takes approximately $0.175\,s$ to achieve 95% of the stable thrust, which is about $8\,N$, the calculation gives a required sample rate of, at least, $91,46\,Hz$. It will be considered to be $100\,Hz$.

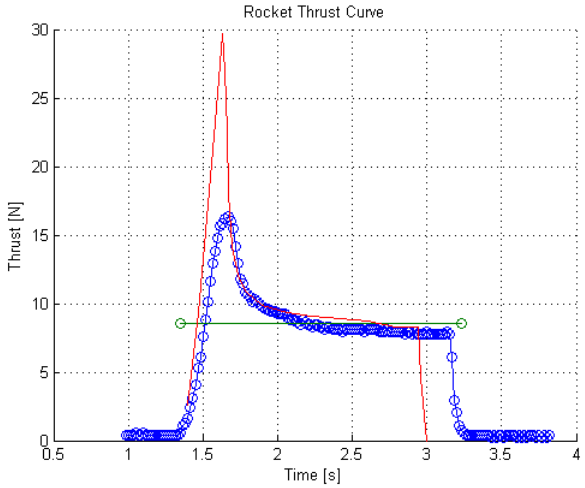

Figure 1.1: Engine thrust according to static test (blue) and manufacturer data (red)

Cost We are aiming to find an economic solution.

1.4 Usefulness of the project

This project has to be able to give accurate information to an engineer about a vehicle performance, not only relate to the flight mechanic parameters such as drag coefficient or stability derivatives but also a accurate description of the engine characteristics. One could test this in a wind tunnel, in the case of the vehicle itself, or in a static test stand, in case of analyzing the engine. The ability to get conclusions from flight data not only means that measures are done in the exact conditions in which the vehicle will operate, but also that it is not necessary to develop expensive test facilities.

1.5 Background

Motivation and justification This project comes from the model rocketry group of students in the university, TRT (Terrassa Rocket Team). When testing engines in a static test stand, one questions if the results would be similar in the engine in launch conditions. In addition,

[1] http://www.estesrockets.com/

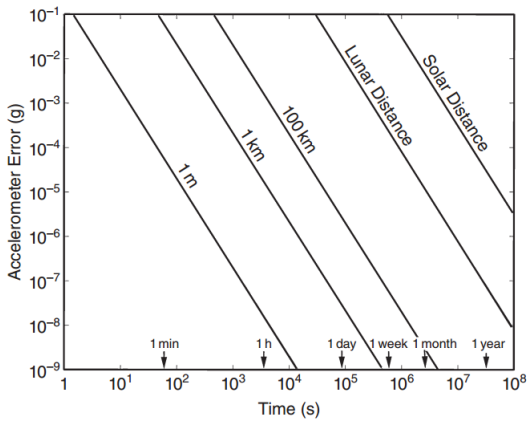

Figure 1.2: Navigation error depending on time and accelerometer error. Source: Grewal et al. (2013)

the cost decrease of the electric components in the last few years has become an opportunity for hobbyist to develop their own projects with very limited cost. This has arrived to a projects such as TRT, which is used to working with microcontrollers in prototyping boards such as Arduino[2] and Mbed[3].

Main principles Then, there is need to design a non invasive technique to get the kinematic description of the vehicle. With an accurate description of the position, velocity and accelerations over time, and estimating the mass of the rocket - notice that the mass of the rocket decreases as the engine burns - it is possible to estimate the summation of all the forces acting in the body. Then, if the drag is estimated by means of correlations, it is possible to obtain the rocket thrust curve. The drag can also be studied in the launch after the engine is no longer burning. It would be interesting in order to study the effect of different configuration of fins in the flight. Nevertheless, since rocket launches depend on the availability of the rocketry team, other possibilities are being considered, such as using in the system on a radio controlled airplane. It would be specially interesting to use it in a sailplane, the absence of engine vibrations is a good environment for the sensors.

Advantages and disadvantages As it has been said before, IMU's are a non invasive way of studying motion of bodies. In the case of the rocket, there are no more consequences than the addition mass and the reduction of the available pay load volume inside the vehicles. Nevertheless, IMU's have one main disadvantage, if the sensors have a bias error, the estimation of the position over time has an error increases with time. To compensate this difficulty, inertial measurements should be combined with measures that do not accumulate error. In this case, a barometer is used in order to get the height of the vehicle. Another possibility is using a GPS, with the advantage that information in three dimensions is obtained.

The precision of the sensor must be chosen according to the acceptable error and operation time, as seen in Figure 1.2.

Critical points of the project Apparently, the project can be carried out without major problems, however it has to be taken into consideration critical aspects that could go wrong.

- Electronics can be difficult to deal with, sometimes hardware does not work as expected and it is necessary to carefully study the libraries. In the case of this problem arose, an

[2]http://www.arduino.cc/
[3]https://mbed.org/

interesting option would be using analog sensors.

- Hardware can be lost or damaged, and not taking into consideration this possibility can bring to important planning problems.
- Rocket launching depends on the rocketry team, therefore it is important to be able to do the as soon as possible, to give them the required flexibility to plan it.

2 State of the art

INS and data fusion INS data fusion with other sensor technologies such as GPS, magnetometer or barometer is not a new subject and many research have been carried, Tailanian et al. (2014); Zwirello et al. (2013).

This data integration have been implemented in the last years in several types of applications such as person tracking, UAV positioning and control and robot navigation. The INS can provide continuous and reliable navigation determination. However, drift in position and velocity are the main drawbacks of this system . However, other complementary sensors can be used to correct these errors, as an aiding system because it provides long-term stability with high accuracy.

Two main approaches of the technology are usually carried out,

- For systems considering inertial sensors and magnetometer are only used to estimate orientation.This technology is not designed to operate in systems with high linear accelerations,since the accelerometer is used in order to estimate the orientation.Yongliang et al. (2008); Hu et al. (2011)
- The other approach considers GPS in addition to the IMU, in order to estimate both orientation and trajectories.Hall et al. (2008); Zhang et al. (2005)

Low cost sensors Low cost sensors are one of the main objects of the project, high cost sensors are not considered. This have become increasingly popular in the last few years specially because of the miniaturization of mobile phone components.

Many recent publications consdier this technology, both from the manufacturing point of view, as explained in Warnasch and Killen (2002), for example, and applications as seen in publication such as Lou et al. (2011).

Model identification from flight data As indicated by Horn (2008), aircraft system identification is a higly versatile procedure for rapidly and efficiently extracting accurate dynamic models of an aircraft from the measured response to specific control inputs.

This strategy which is increasingly popular has been successfully implemented in aircraft characterization and optimization, not only in fixed wing aircraft Liu et al. (2011) but also in helicopters Wu (2014). Most of this technologies is being implemented in small scale or model vehicles, as seen in Taha et al. (2011), which may assemble the strategy which will be carried out in this project.

Since in this project we are working with a model rocket with no control, we would be in the specific case of the dynamic response of a vehicle with fixed controls.

In flight thrust measurement In order to determine the thrust of an engine in a air vehicle, different approaches are available, most of the explained in reference books such as 456 (1979) and Covert (1985).

- By means of an strain gauge coupled to the engine, as shown by Conners et al. (1998) and John S. Orme (1999). However, this technology is complex and expensive to implement in aircraft.
- By means of measuring the pressure on the nozzle, in the case of air breathing engines, which has been used for a long time as seen in Salmon (1966).

In any case, it is agreed that it is not possible to measure both thrust and drag at the same time.

3 Description of the solution

In order to fulfill the aim of the project, the process that will be carried out consists on, as shown in Figure 3.1.

1. Developing the software to process the data. It must be able to process the binary data file, estimate the attitude and trajectory of the vehicle, and proceed with the final calculations in order to obtain accurate information we are aiming to compute. It will developed in MATLAB. Since computing time is not critical, this high level software environment will be used. Also, the important amount of built-in mathematical functions, community support and easy debugging are advantages of this option.

2. Developing the electronic system. This will developed using an Arduino Nano board (Atmega 328) and a matrix of sensors IMU, barometer and magnetometer that will save the data into a microSD card. After the design has been tested, it will be implemented in a PCB.

3. Launching the rocket while logging sensor data from electronic board. At least two launches will be done in order to compare and validate the results. This will be done with the support of TRT.

4. Process the data with the developed software in MATLAB, after the launch has been carried out. Obtain flight parameters such as drag coefficient and stability derivatives and a characterization of the engine with its main figures.

5. Obtain results and conclusions , evaluate hypothesis made during the project and write the final document. Prepare documentation so that TRT or anyone else could use this project for the same purpose or improving it.

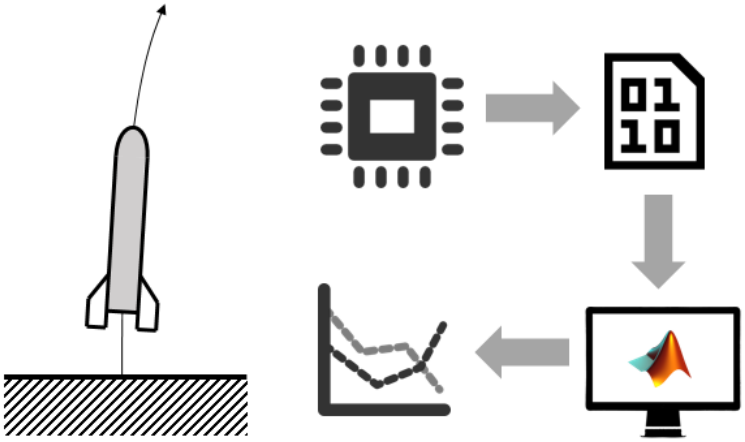

Figure 3.1: Project description scheme

4 Inertial navigation system

The aim of this section is to estimare the attitude and trajectory from the inertial sensors in the strapdown system. Two main approaches of the attitude problem are available, by directly updating the direction cosine matrix or by means of an attitude quaternion. The first option has been used because of its simplicity, despite of the higher computational cost, which is not critical in this study because the data processing is done after the mission.

History of the technology According to Grewal et al. (2013), INS are a product of the cold war between the Soviet Bloc and NATO Allies. The need for developing guidance for long range delivery systems motivated well-funded programs to develop this technology. INS had already been used in German guided missiles during the second world war, but only to be used on open loop, in the sense that control was used only to follow a pre-programmed trajectory without feedback related to trajectory errors. In February 1953, it was the first successful demonstration of acceptable inertial navigation performance over a representative mission distance, on a flight from Bedford, Massachusetts to Los Angeles, California, a distance of about 2250nm, aboard a World War II-vintage Boeing B-29 bomber.

4.1 Direction cosine updating algorithm

The algorithm has been developed based on Savage (2008); Weston (2005).Any vector measured in the body frame, $\mathbf{v_b}$ can be expressed in the earth frame, $\mathbf{v_e}$ by means of a rotation matrix C_{eb}, as shown in Eq 4.1. Thus the rotation matrix has the information of the attitude of the vehicle. Determining the rotation matrix in each time step gives a description of the attitude of the vehicle. The attitude variation can be determined by the rotations of the frame, which are measured by the gyroscopes. Then, the problem is to update the cosine matrix at each time step knowing the previous cosine matrix and the angular velocity between time steps, $C_{n+1} = f\left(C_n, \boldsymbol{\omega}\right)$. This function is the one described in the following algorithm.

$$\mathbf{v_e} = C_{eb}\mathbf{v_b} \tag{4.1}$$

The direction cosine matrix can be updated by solving the following differential equation,

$$\dot{C} = C\Omega \tag{4.2}$$

Over a single time cycle, the solution to the equation may be written as follows

$$C_{n+1} = C_n \cdot exp\left(\int_{t_n}^{t_{n+1}} \boldsymbol{\Omega} dt\right) = C_n A_n \tag{4.3}$$

Where C_n is the direction cosine matrix relating body to earth axis at n^{th} time step, A_n is the matrix relating the cosine matrix a n^{th} time step to the one at $n + 1^{th}$ time step.

Assuming that the orientation of the rate turn vector $\boldsymbol{\omega}$ remains fixed over the update interval,

$$\int_{t_n}^{t_{n+1}} \Omega dt = [\boldsymbol{\theta}\times] = \begin{bmatrix} 0 & -\theta_z & \theta_y \\ \theta_z & 0 & -\theta_x \\ -\theta_y & \theta_x & 0 \end{bmatrix} \tag{4.4}$$

Where, θ_x, θ_y, θ_z are the components of $\boldsymbol{\theta}$, $\boldsymbol{\theta}$ is the angle vector with direction and magnitude such that a rotation of the navigation frame about $\boldsymbol{\theta}$ through an angle equal to the magnitude of

$\boldsymbol{\theta}$ will rotate the navigation frame from its orientation at time step n^{th} to time step $n+1^{th}$. The vector $\boldsymbol{\theta}$ is computed for each time step n by processing the navigation frame rotation rate data.

Therefore,

$$C_{n+1} = C_n \cdot exp\left[\boldsymbol{\theta}\times\right] = C_n \cdot A_n \tag{4.5}$$

Expanding the exponential term in the expression,

$$A_n = exp\left[\boldsymbol{\theta}\times\right] = I + \left[\boldsymbol{\theta}\times\right] + \frac{\left[\boldsymbol{\theta}\times\right]^2}{2!} + \frac{\left[\boldsymbol{\theta}\times\right]^3}{3!} + ... = I + \sum_{n=1}^{\infty} \frac{\left[\boldsymbol{\theta}\times\right]^n}{n!} \tag{4.6}$$

Which may be also expressed as,

$$A_n = I + \left[1 - \frac{\theta^2}{3!} + \frac{\theta^4}{5!} - ...\right]\left[\boldsymbol{\theta}\times\right] + \left[\frac{1}{2!} - \frac{\theta^2}{4!} + \frac{\theta^4}{6!} - ...\right]\left[\boldsymbol{\theta}\times\right]^2 \tag{4.7}$$

This expression will be used in order to estimate the matrix A_n, which updates the rotation matrix at time step n^{th} to time step $n+1^{th}$.

4.1.1 The computation of $\boldsymbol{\theta}$

The $\boldsymbol{\theta}$ is calculated by processing the data from the strapdown gyroscopes. The vector $\boldsymbol{\theta}$ that updates the direction cosine matrix form time step n^{th} to $n+1^{th}$ can be analytically expressed as follows,

$$\boldsymbol{\theta} = \int_{t_n}^{t_{n+1}} \boldsymbol{\omega} dt \tag{4.8}$$

An expression for $\boldsymbol{\theta}$ under general motion conditions can be derived as shown by Weston (2005) to give,

$$\dot{\boldsymbol{\theta}} = \boldsymbol{\omega} + \frac{1}{2}\boldsymbol{\theta}\times\boldsymbol{\omega} + \frac{1}{\theta^2}\left[1 - \frac{\theta sin\theta}{2(1-cos\theta)}\right]\boldsymbol{\theta}\times\boldsymbol{\theta}\times\boldsymbol{\omega} \tag{4.9}$$

A more practical implementation can be derived by expressing the sines and cosines as a series of expansion and ignoring terms higher than third order, then Equation 4.9 can be written as,

$$\dot{\boldsymbol{\theta}} = \boldsymbol{\omega} + \frac{1}{2}\boldsymbol{\theta}\times\boldsymbol{\omega} + \frac{1}{12}\boldsymbol{\theta}\times\boldsymbol{\theta}\times\boldsymbol{\omega} \tag{4.10}$$

The following algorithm is proposed,

$$\boldsymbol{\alpha} = \int_{t_n}^{t} \boldsymbol{\omega} dt \tag{4.11}$$

$$\delta\boldsymbol{\alpha}_{n+1} = \int_{t_n}^{t_{n+1}} \boldsymbol{\alpha}\times\boldsymbol{\omega} dt \tag{4.12}$$

$$\boldsymbol{\theta} = \boldsymbol{\alpha}_{n+1} + \delta\boldsymbol{\alpha}_{n+1} \tag{4.13}$$

Which is implemented in the following way,

$$\boldsymbol{\alpha}_{n+1} = \frac{\boldsymbol{\omega}_{n+1} + \boldsymbol{\omega}_n}{2} \cdot (t_{n+1} - t_n) \tag{4.14}$$

$$\delta\boldsymbol{\alpha}_{n+1} = \frac{\boldsymbol{\alpha}_{n+1} \times \boldsymbol{\omega}_{n+1}}{4} \cdot (t_{n+1} - t_n) \tag{4.15}$$

4.2 Position

Having estimated the attitude of the rocket in each time step, calculating the position among time is simple.

$$\dot{\boldsymbol{r}}_{n+1} = \dot{\boldsymbol{r}}_n + \int_{t_n}^{t_{n+1}} (\boldsymbol{a} - \boldsymbol{g})\, dt \tag{4.16}$$

$$\boldsymbol{r}_{n+1} = \boldsymbol{r}_n + \int_{t_n}^{t_{n+1}} \dot{\boldsymbol{r}} dt \tag{4.17}$$

Where, $\boldsymbol{a}$ are the accelerations in the earth frame, $\boldsymbol{g}$ is the gravity acceleration in the earth frame.

Which is implemented in the following way,

$$\dot{\boldsymbol{r}}_{n+1} = \dot{\boldsymbol{r}}_n + \left(\frac{C_{n+1} \cdot \boldsymbol{\epsilon}_{n+1} + C_n \cdot \boldsymbol{\epsilon}_n}{2} - \boldsymbol{g} \right) \cdot (t_{n+1} - t_n) \tag{4.18}$$

$$\boldsymbol{r}_{n+1} = \boldsymbol{r}_n + \frac{\dot{\boldsymbol{r}}_{n+1} + \dot{\boldsymbol{r}}_n}{2} \cdot (t_{n+1} - t_n) \tag{4.19}$$

Where,$\boldsymbol{\epsilon}$ are the accelerations in the body frame.

However, this calculations will not be used because a Kalman filter will be implemented, as shown in following sections.

5 Additional sensors

5.1 Magnetometer

As it has been explained in this project, drift on the attitude due to the integration of the gyroscopes should be corrected in some way. The magnetometer has been used in the following way.

The magnetometer gives a 3-component vector that indicates the direction of the magnetic field on the sensor. Assuming that the earth field is constant and that there are no other magnetic fields, the direction of this magnitude can be used in order to estimate the attitude. It is important to notice that an arbitrary rotation has 3 degrees of freedom. However, knowing the orientation of a fixed vector (magnetic field) in an arbitrary frame, has only 2 degrees of freedom. For this reason, using the magnetometer has not the advantages of a complete non drifted attitude sensor.

5.2 Barometer

A barometer measures air pressure. Assuming that the measured pressure only depends on height, the altitude can be estimated as follows,

A force equilibrium is made in a column of air of height dh , as shown in Figure 5.1. The pressure varies along the z-axis so that in a dh increment of height the pressure has increase dp. The gravitational force in the volume is $F_g = S\rho g\, dh$. The, the equilibrium is expressed as follows,

$$pS - (p + dp)\, S - S\rho g\, dh = 0 \tag{5.1}$$

$$-\, dp = \rho g\, dh \tag{5.2}$$

If the ideal gas relation is introduced,

$$p = \rho RT \tag{5.3}$$

$$\frac{dp}{p} = \frac{-g}{RT} dh \tag{5.4}$$

The equation must me solved,

$$\int_{p_0}^{p} \frac{dp}{p} = \frac{-g}{R} \int_{0}^{h} \frac{dh}{T_0 - 0.0065h} \tag{5.5}$$

And the following relation is obtained,

$$p = p_0 \left(1 - 0.0065 \frac{h}{T_0}\right)^{5.2561} \tag{5.6}$$

Inverting this relation, the height as a function of the pressure can be obtained,

$$h\,(p) = T_0 \frac{\left(\frac{p}{p_0}\right)^{1/5.2561} - 1}{0.0065} \tag{5.7}$$

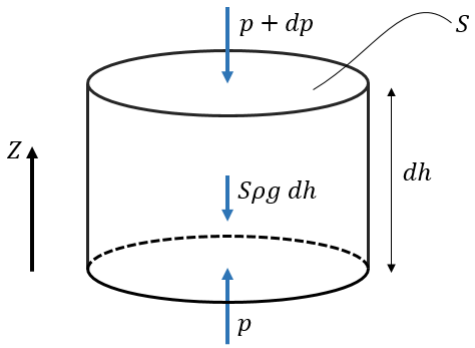

Figure 5.1: Differential column of air

6 Data fusion and global algorithm

6.1 Calibration

Despite of the fact that the specification sheets of the inertial sensors suggest some calculations in order to convert the raw data into useful information, it has been proved that they are not accurate at all. For this reason, it is important to develop a calibration system.

The calibration relation has been assumed to be a linear with an offset, as shown in Equation 6.1 and Figure 6.1.

$$Y = X \cdot m + n \tag{6.1}$$

Where, m is the slope, n_y is the offset, X is the raw measurement, Y is the calibrated value.

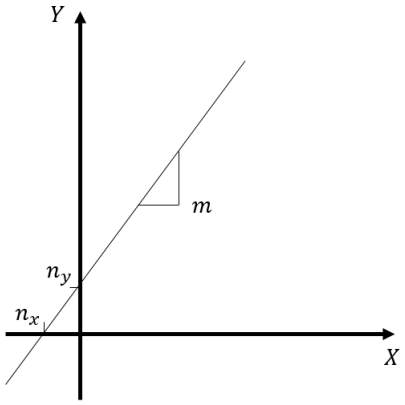

Figure 6.1: Representation of the calibration equation

6.1.1 Accelerometers

The easiest way to calibrate the accelerometers is by means of the gravity, which it is known to be 9.81 $\frac{m}{s^2}$. This is reasonably accurate when the FSR of the accelerometers is low, but it can introduce error in the calculations when the FSR takes values significantly greater than $\pm 1g$. As it has been indicated, for the case of the MPU6050 the minimum is $\pm 2g$.

For this sensors,three axis have to be calibrated, which means 6 constants have to be determined. Imposing that the magnitude of the vector composed by the three axis must be equal to the acceleration of the gravity when the sensor is stationary, the Equation 6.2 must be satisfied.

$$(X_1 \cdot m_1 + n_1)^2 + (X_2 \cdot m_2 + n_2)^2 + (X_3 \cdot m_3 + n_3)^2 - 9.81^2 = 0 \tag{6.2}$$

Therefore, it is necessary to get a set of data from the sensor in, at least, 6 different positions, which are not strictly necessary to be aligned with the gravity vector. For averaging purposes, 50 measures are taken in each position. Then, the 6 values of $m_i$and n_i have to be found according to a least squares algorithm.

Implementation The Arduino software has the data logging part of the procedure, and generates a calibration file that can be processed by a MATLAB function that has a .MAT file as an output, with the value of the constants.

6.1.2 Gyroscopes

The procedure to calibrate the gyroscopes is similar to the one used for the accelerometers, but a system that rotates at a known angular velocity has to be found. A turntable has been used for

the precision of the angular velocity, of 33 or 45 rpm as an standard. Nevertheless, the rotation speed should precisely measured in order to get better results. This has not been done due to lack of time.

Unlike the case of the accelerometer, the offset value of the three axis can be directly determined by means of a measure in stationary conditions. For this reason it is only necessary to estimate 3 more values corresponding to the slope of the calibration function.

Similarly to the accelerometer, Equation 6.3 must be satisfied.

$$(X_1 \cdot m_1 + n_1)^2 + (X_2 \cdot m_2 + n_2)^2 + (X_3 \cdot m_3 + n_3)^2 - w^2 = 0 \tag{6.3}$$

Implementation The implementation in Arduino and MATLAB is equivalent to the one developed with the accelerometer.

6.1.3 Magnetometer

The procedure to calibrate the magnetometer is similar to the one used for the accelerometers and gyroscopes, but in this case there is not need to set the board in specific conditions, the magnetic field of the earth is always available. The user must turn the board in every possible direction while the magnetometer is sampling.

Similarly to the gyroscope, Equation 6.4 must be satisfied.

$$(X_1 \cdot m_1 + n_1)^2 + (X_2 \cdot m_2 + n_2)^2 + (X_3 \cdot m_3 + n_3)^2 - 1 = 0 \tag{6.4}$$

This equation does not ensure that the filtered set of data is unitary in each sample. Therefore, it is necessary to normalize each magnetometer vector,

$$\hat{\boldsymbol{m}}_n = \frac{\boldsymbol{m}_n}{m_n} \tag{6.5}$$

6.2 Filtering

The first step into processing the war data is filtering the noise.

6.2.1 Complementary filter

The basic complementary filter is shown in Figure 6.2. According to Higgins (1975), supposing that there are two signals x and y of some signal z, in which $\overline{z}$ is its estimate. Consider that x has noise in mostly low frequency and y has noise in mostly high frequency. then, a low pass and high pass filter is applied to each of this signals, with transfer functions $G_1(s)$and $G_2(s)$, so that $G_1(s) + G_1(s) = 1$. This filter is easily implemented in the case of an scalar signal in discrete sample times with the following update equation,

$$\overline{z}_{n+1} = x_{n+1} \cdot \alpha + y_{n+1} \cdot (1 - \alpha) \tag{6.6}$$

where $\alpha \in [0, 1]$ is the complementary filter gain which has to be determined to achieve an optimal performance.

This is specially interesting in the case of combining information of gyroscopes with non drifted signals from accelerometers or magnetometers. In this case, $\dot{x}$ is available instead of x, and it is necessary to integrate it over time, as shown in the following equation,

$$z_{n+1} = \left(z_n + \int_{t_n}^{t_{n+1}} \dot{x} dt \right) \cdot \alpha + y_{n+1} \cdot (1 - \alpha) \tag{6.7}$$

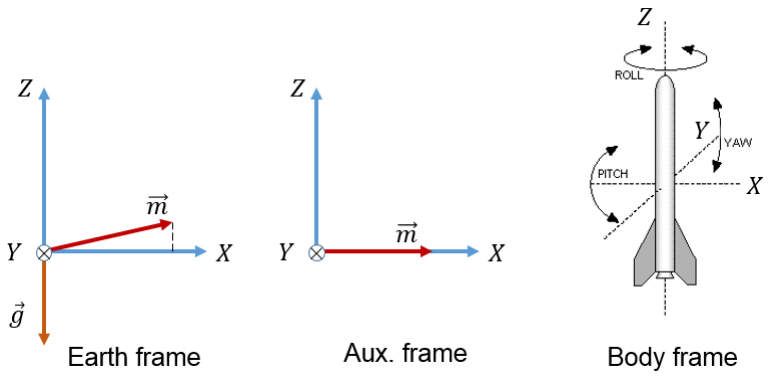

Figure 6.3: Frames scheme

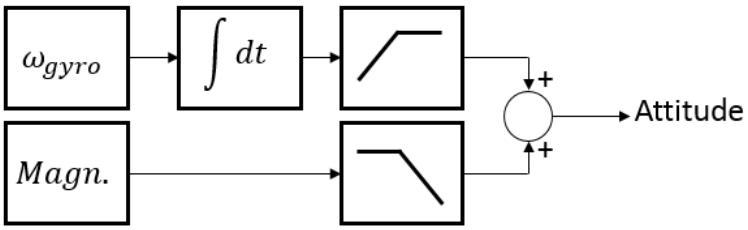

Figure 6.2: Scheme of a complementary filter

6.2.2 Properties of the rotation matrix and frames

As it is widely know form basic algebra, a rotation matrix is used to change the frame in which a vector is expressed. The rotation matrix can be constructed by knowing the expression of the axis in the rotating frame.

First, we will define the frames that are used, as shown in Figure 6.3.

- **Earth**, z-axis in the opposite direction of the gravity, x-axis is the projection of the magnetic field on z-y plane, and y-axis creating a positive system of axis.
- **Auxiliary**, x-axis in the direction of magnetic field, and the other two axis in an arbitrary direction creating a positive system of axis.
- **Body**, z-axis in the axial direction of the rocket, and the other two axis in an arbitrary direction creating a positive system of axis.

The auxiliary frame is defined in order to be able to implement the complementary filter. As it is widely known from basic algebra, the columns of a rotation matrix are the orthogonal unit vectors that define the frame in which it is wanted to transform the vectors, as it is shown in the following example,

$$C_{eb} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}, \quad \mathbf{v_e} = C_{eb}\mathbf{v_b}$$

That means that vectors,

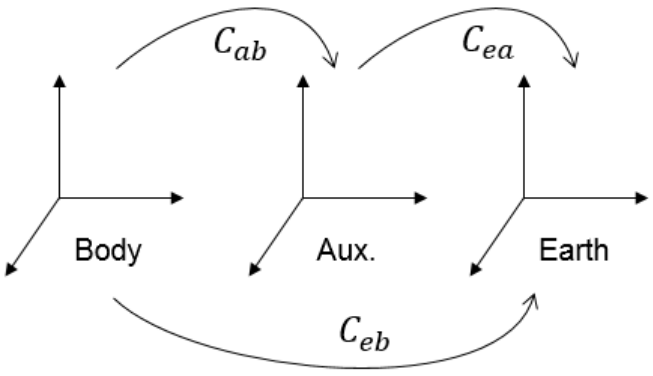

Figure 6.4: Frame relations

$$\begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \end{bmatrix}, \begin{bmatrix} a_{12} \\ a_{22} \\ a_{32} \end{bmatrix}, \begin{bmatrix} a_{13} \\ a_{23} \\ a_{33} \end{bmatrix}$$

are the ones which define the axis of body, expressed in earth frame. Knowing that rotation matrix are orthogonal with unitary norm,

$$C_{eb} = C_{be}^{-1} = C_{be}^{T} \tag{6.8}$$

Therefore,

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \end{bmatrix}, \begin{bmatrix} a_{21} & a_{22} & a_{23} \end{bmatrix}, \begin{bmatrix} a_{31} & a_{32} & a_{33} \end{bmatrix}$$

are the vectors that define the axis of earth, expressed in body frame.
The relations between the three frames are shown in Figure 6.4.

6.3 Kalman filter

6.3.1 Description

In order to combine the information from the accelerometers and barometers, in order to properly estimate height, velocity and acceleration, a Kalman filter has been designed. A Kalman filter is an algorithm that makes optimal estimations of the state of a system, weighting a mathematical description of the model and measurements, so that the statistical error of the estimation is minimum.

The Kalman filter has two different parts, Welch and Bishop (2006). A simple scheme in order to understand the main algorithm is shown in Figure 6.5. A summary of the equations of the process is shown in Figure 6.6.

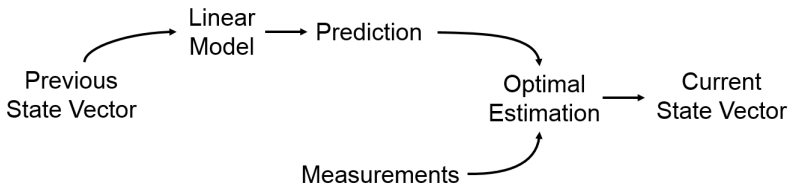

Figure 6.5: Basic description of the Kalman Filter

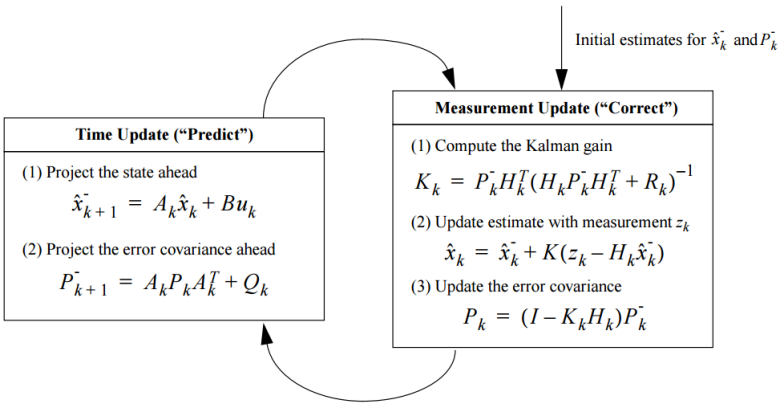

Figure 6.6: A complete picture of the operation of the Kalman filter. Source : Welch and Bishop (2006)

1. The state vector is computed based on the estimations from last iteration, following the described mathematical model. In this case, A and B are the model.

$$\hat{\mathbf{x}}_k^- = A\hat{\mathbf{x}}_{k-1} + B\mathbf{u}_{k-1} \tag{6.9}$$

$$P_k^- = AP_{k-1}A^T + Q \tag{6.10}$$

1. The estimation is updated with the prediction and the measurements.

$$K_k = P_k^- H^T \left(HP_k^- H^T + R\right)^{-1} \tag{6.11}$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + K_k\left(\mathbf{z}_k - H\hat{\mathbf{x}}_k^-\right) \tag{6.12}$$

$$P_k = (I - K_kH)P_k^- \tag{6.13}$$

where: $\hat{\mathbf{x}}$- state vector estimate, $\mathbf{z}$ - measurement vector, A - process matrix, B - control matrix, C - input matrix, P - covariance matrix, K - Kalman gain matrix, Q - process noise covariance matrix, R - measurement noise covariance matrix.

6.3.2 Application

In this case, the analysis that has been implemented is in one dimension, under the following assumption,

- The rocket velocity, speed and accelerations are all in the vertical axis

Knowing that this assumption is false, in the case of a launch of a model rocket, specially in the initial period of flying, it may be accepted in order to get a good approximation of kinematic description of the flight.

In order to implement this filter, Gasior and Gardecki (2014), has been followed as a guide because the aim of the paper is totally equivalent to our case.

From the kinematic point of view, we have information of acceleration and height. This two signals are related with a model, the acceleration is the second derivative of the position with respect to the time, and each one has different noise.

6.3.3 Measurement model

The linear model has the following form,

$$\mathbf{x}_k = A\mathbf{x}_{k-1} + B\mathbf{u}_k \tag{6.14}$$

If we consider acceleration to be constant between each time step, one gets that

$$v_k = v_{k-1} + a_{k-1}T \tag{6.15}$$

$$h_k = h_{k-1} + v_{k-1}T + \frac{a_{k-1}T^2}{2} \tag{6.16}$$

where T is the sample period,
If we fit this equations in the state model, Eq 6.14 , one gets that

$$\begin{bmatrix} h_k \\ a_k \\ v_k \end{bmatrix} = \begin{bmatrix} 0 & \frac{T^2}{2} & T \\ 0 & 1 & 0 \\ 0 & T & 1 \end{bmatrix} \begin{bmatrix} h_{k-1} \\ a_{k-1} \\ v_{k-1} \end{bmatrix} \tag{6.17}$$

Also, the measures can be expressed in the form of,

$$\mathbf{y} = C\mathbf{z}_{k-1} + D\mathbf{u}_k$$

Also, we can fit this to the equations described before,

$$\mathbf{y} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} h_{k-1} \\ a_{k-1} \\ v_{k-1} \end{bmatrix} \tag{6.18}$$

6.3.4 Kalman Filter initialization

Once the model has been described, one knows the matrix A, B, C and D, but matrix P, Q and R are still to be defined.

P - Covariance matrix This matrix represnts how accurate is the model compared to the measurements. The larger the value of P, the more accurate the model is. In our case, the kinematic relations of position, velocity and acceleration are very accurate, then initial P must have large value. To compare with this, imagine that we are modeling the behavior of a process of fermentation. It is very difficult to have an accurate model of this to predict the state value, we would want to take the measures more into consideration, Then, the value of K should be significantly smaller.

In our case, we have begun with $P{=}I \cdot 10^4$.

R - Measurement noise covariance matrix On a stationary condition of measurement, one can calculate the covariance matrix of two variables with its definition. In MATLAB, the function *cov()* is available to compute it.

$$R{=}\begin{bmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \end{bmatrix}$$

Q - Process noise covariance matrix Since the model is not always as accurate as one would want, the matrix Q represents the noise that the model introduces into the filter due to its inaccuracy. Since our model is quite accurate, the noise should be significantly smaller than the measurement noise.

In our case, we have begun with $Q = \begin{bmatrix} k_1 R_{11} & k_1 R_{12} & 0 \\ k_1 R_{21} & k_1 R_{22} & 0 \\ 0 & 0 & \frac{R_{11}+R_{22}}{2} k_1 k_2 \end{bmatrix}$.

where $k_1 = 10^{-2}$ and $k_2 = 10^{-4}$.

This is because the third element in the main diagonal in the matrix is referred to the velocity. Since there is no direct measure of the velocity, it has been found to have good results to decrease the model noise of this variable more than the other variables.

6.4 Global algorithm

6.4.1 Initialization

In order to define the frames, reference vectors have to be defined,

- $\boldsymbol{m}_0$ is the average magnetic field of earth during the stationary period before the mission
- $\boldsymbol{g}_0$ is the average gravity field of earth during the stationary period before the mission

Initial C_{ab} rotation matrix The initial C_{ab}rotation matrix is constructed as follows,

$$\boldsymbol{v}_1 = \boldsymbol{m}_0,\ \boldsymbol{v}_2 = \boldsymbol{v}_1 \times \begin{bmatrix} 1 & 0 & 0 \end{bmatrix},\ \boldsymbol{v}_3 = \boldsymbol{v}_1 \times \boldsymbol{v}_2 \tag{6.19}$$

$$C_{ab}\,(t=0) = \begin{bmatrix} \boldsymbol{v}_1 \\ \boldsymbol{v}_2 \\ \boldsymbol{v}_3 \end{bmatrix} \tag{6.20}$$

It is important to notice that the direction of $\boldsymbol{v}_2$ is arbitrary, the only condition that has to satisfy is being orthogonal to $\boldsymbol{v}_1$.

Initial C_{eb} rotation matrix The initial C_{eb}rotation matrix is constructed as follows,

$$\boldsymbol{v}_1 = \boldsymbol{m}_0 - \frac{\boldsymbol{g}_0 \boldsymbol{m}_0^T \boldsymbol{g}_0}{g_0^2} \tag{6.21}$$

$$\boldsymbol{v}_3 = \boldsymbol{g}_0 \tag{6.22}$$

$$\boldsymbol{v}_2 = \boldsymbol{v}_3 \times \boldsymbol{v}_1 \tag{6.23}$$

$$C_{eb}\,(t=0) = \begin{bmatrix} \boldsymbol{v}_1 \\ \boldsymbol{v}_2 \\ \boldsymbol{v}_3 \end{bmatrix} \tag{6.24}$$

Notice that Equation 6.21 is used in order to calculate the projection of the vector $\boldsymbol{m}_0$ on the plane orthogonal to $\boldsymbol{g}_0$.

C_{ea}rotation matrix From basic knowledge of algebra, it is know that,

$$C_{eb} = C_{ea}C_{ab} \tag{6.25}$$

$$C_{ea} = C_{eb}C_{ab}^{T} \tag{6.26}$$

Notice that this matrix is not time dependent because both the earth and auxiliary frame are fixed.

6.4.2 Processing

In this section, the algorithm to process the data is being described, with the aim to clarify which equations are being used, without mixing them with its derivation. A simple scheme of the global algorith is shown in Figure 6.7. To sum up, we are using a DCM algorithm in order to get the orientation of the rocket. This DCM agorithm uses information from the gyroscopes and the magnetometer. Once the rotation matrix is known at each time step, it is possible to have all the vector magnitude in earth frame. Then, using the z component of the vector the Kalman filter is used in order to have more accurate description of the acceleration and the altitude and estimate the velocity.

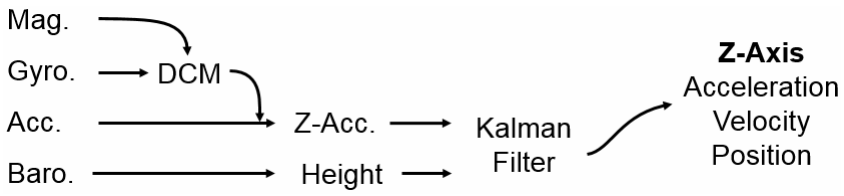

Figure 6.7: Global scheme of the algorithm

Attitude estimation from gyroscopes According to the algorithm described before, the rotation matrix of the body is estimated by means of the gyroscope. A crucial part of the program is the initial conditions of the rotation matrix. One could think that the most useful way would we estimating the matrix C_{eb}. However, if we want to combine the gyroscope data with the magnetometer data, it is necessary to estimate the matrix C_{ba}, referring to the auxiliary frame, in which the x-axis is aligned with the earth magnetic field.

$$\boldsymbol{\alpha}_{n+1} = \frac{\boldsymbol{\omega}_{n+1} + \boldsymbol{\omega}_n}{2} \cdot (t_{n+1} - t_n) \tag{6.27}$$

$$\delta\boldsymbol{\alpha}_{n+1} = \frac{\boldsymbol{\alpha}_{n+1} \times \boldsymbol{\omega}_{n+1}}{4} \cdot (t_{n+1} - t_n) \tag{6.28}$$

$$\boldsymbol{\theta} = \boldsymbol{\alpha}_{n+1} + \delta\boldsymbol{\alpha}_{n+1} \tag{6.29}$$

$$\theta = \sqrt{\theta_x^2 + \theta_y^2 + \theta_z^2} \tag{6.30}$$

$$[\boldsymbol{\theta}\times] = \begin{bmatrix} 0 & -\theta_z & \theta_y \\ \theta_z & 0 & -\theta_x \\ -\theta_y & \theta_x & 0 \end{bmatrix} \tag{6.31}$$

$$A_n = I + \left[1 - \frac{\theta^2}{3!} + \frac{\theta^4}{5!} - ...\right][\boldsymbol{\theta}\times] + \left[\frac{1}{2!} - \frac{\theta^2}{4!} + \frac{\theta^4}{6!} - ...\right][\boldsymbol{\theta}\times]^2 \tag{6.32}$$

This expansion has to be limited to a certain number of terms, which may be a parameter of the program.

$$C_{n+1} = C_n A_n \tag{6.33}$$

Complementary filter Because we are estimating the rotation matrix C_{ab}, the third row of the matrix is equal to the x-axis of the auxiliary frame expressed in the body frame. This means that the measures unitary vector of the magnetic field should be equal to the the third row of the matrix. Under this conditions a complementary filter has been used.

$$C_{n+1}(1, 1:3) = C_{n+1}(1, 1:3) \cdot \alpha + \boldsymbol{m} \cdot (1 - \alpha) \tag{6.34}$$

Normalization Most strapdown attitude computation techniques periodically employ self-consistency correction algorithms as an outer-loop function for accuracy enhancement. If the basic attitude data is computed in the form of a DCM, the self-consistency check is that the rows should be orthogonal to each other and equal to unity in magnitude. The procedure is suggested by Savage (2008).

$$C_n = C_n + \frac{1}{2}\left(I - C_n C_n^T\right) C_n \tag{6.35}$$

Kalman filter First, the acceleration vector is rotated to the earth frame,

$$\mathbf{a}_n^- = C_n \mathbf{a}_n \tag{6.36}$$

Then, the measure matrix is arranged with the height calculated with the barometer and the z-component of the acceleration.

$$k_n = \begin{bmatrix} h_n \\ a_{n_Z} \end{bmatrix} \tag{6.37}$$

According to what it has been described in the Kalman filter section, the equations are as follows.

$$\hat{\mathbf{x}}_n^- = A\hat{\mathbf{x}}_{n-1} + B\mathbf{u}_{n-1} \tag{6.38}$$

$$P_n^- = AP_{n-1}A^T + Q \tag{6.39}$$

$$K_n = P_n^- H^T \left(HP_n^- H^T + R\right)^{-1} \tag{6.40}$$

$$\hat{\mathbf{x}}_n = \hat{\mathbf{x}}_n^- + K_n\left(\mathbf{z}_n - H\hat{\mathbf{x}}_n^-\right) \tag{6.41}$$

$$P_n = (I - K_n H)P_n^- \tag{6.42}$$

6.4.3 Post processing

Once the processing is finished, the rotation matrix are changed so that they are referred to the earth frame,

$$C_{be_n} = C_{ba_n} C_{ae_n} \tag{6.43}$$

7 Performance analysis

The main aim of this project is analyzing the performance of the rocket with the processed data taken on board. First, all the available information will be summed up,

- z,and its derivatives, referring to the vertical position of the vehicle.
- the Euler angles and it derivatives, referring to the attitude of the vehicle.

The rocket is modeled as rigid solid, with known mass properties such as mass, center of gravity and inertia, as shown in Figure 7.1. This solid has interactions with the surrounding air,characterized by some coefficients and functions that relate conditions to external aerodynamic forces and moments. The thrust of the motor is also considered as a force.

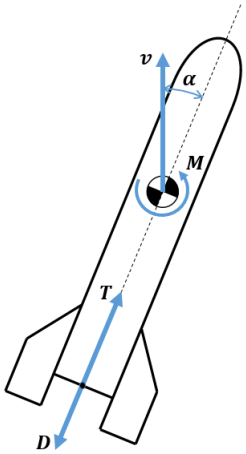

Figure 7.1: Rocket as a rigid solid scheme

7.1 Longitudinal stability

The analysis of the longitudinal stability in the case of a model rocket requires a transient study. In order to model the phenomenon, the pitch has to be related to the aerodynamic forces.If we do a one dimension analysis,

$$\frac{dH}{dt} = M \tag{7.1}$$

$$H = I \cdot \omega \tag{7.2}$$

Deriving the expression over time,

$$\frac{dH}{dt} = \frac{dI}{dt} \cdot \omega + \frac{d\omega}{dt} \cdot I \tag{7.3}$$

Two contributions to the moment are considered, according to general knowledge of flight mechanics, for example Howard (1997).

- M_α, moment due to angle of attack
- $M_{\dot{\alpha}}$, moment due to variation of angle of attack
- No external moments are considered, for example air perturbations

which leads to the following equation,

$$\dot{I} \cdot \dot{\alpha} + \ddot{\alpha} \cdot I = M_\alpha + M_{\dot{\alpha}} \tag{7.4}$$

If a linear study of the problem is carried out, the moment can be modeled as follows,

$$M_\alpha = qSlC_{m\alpha} \cdot \alpha \tag{7.5}$$

$$M_{\dot{\alpha}} = qSlC_{m\dot{\alpha}} \cdot \dot{\alpha} \tag{7.6}$$

where,

- $$q = \frac{1}{2}\rho u^2 \tag{7.7}$$
- S is the characteristic section if the rocket, the cross section area of the tube
- l is the length of the rocket

If we introduce this relations in Equation 7.4,

$$\dot{I} \cdot \dot{\alpha} + \ddot{\alpha} \cdot I = qSlC_{m\alpha} \cdot \alpha + qSlC_{m\dot{\alpha}} \cdot \dot{\alpha} \tag{7.8}$$

And rearranging the terms, the following differential equation is presented,

$$\ddot{\alpha} + \dot{\alpha}\left(\frac{\dot{I} - qSlC_{m\dot{\alpha}}}{I}\right) + \alpha\left(\frac{-qSlC_{m\alpha}}{I}\right) = 0 \tag{7.9}$$

However, considering the available information the equation may be rearranged such that the therms that have to be estimated are isolated,

$$\left(\ddot{\alpha} + \frac{\dot{\alpha}\dot{I}}{I}\right) + \left(\frac{-\dot{\alpha}qSl}{I}\right) \cdot C_{m\dot{\alpha}} + \left(\frac{-\alpha qSl}{I}\right) \cdot C_{m\alpha} = 0 \tag{7.10}$$

$$A + B \cdot x_1 + C \cdot x_2 = 0 \tag{7.11}$$

where,

$$A = \ddot{\alpha} + \frac{\dot{\alpha}\dot{I}}{I} \tag{7.12}$$

$$B = -\frac{\dot{\alpha}qSl}{I} \tag{7.13}$$

$$C = -\frac{\alpha qSl}{I} \tag{7.14}$$

Notice that the coefficients A, B and C are known in each time step but not constant over time, since they depend on the velocity, Equation 7.7 and the derivative of the inertia. Using Equation 7.11, the stability derivative can be estimated, by considering this equation in each time step and finding the values of x_1 and x_2 which minimize the squared error.

Notice also, that Equation 7.9can be simplified if the engine is not burning, by eliminating the terms related to the derivative of the inertia,

$$\ddot{\alpha} + \dot{\alpha}\left(\frac{\not{I} - qSlC_{m\dot{\alpha}}}{I}\right) + \alpha\left(\frac{-qSlC_{m\alpha}}{I}\right) = 0 \tag{7.15}$$

$$\ddot{\alpha} + \dot{\alpha}\left(\frac{-qSlC_{m\dot{\alpha}}}{I}\right) + \alpha\left(\frac{-qSlC_{m\alpha}}{I}\right) = 0 \tag{7.16}$$

And therefore, Equation 7.10 can be rearranged into a simpler form,

$$(\ddot{\alpha}) + \left(\frac{-\dot{\alpha}qSl}{I}\right) \cdot C_{m\dot{\alpha}} + \left(\frac{-\alpha qSl}{I}\right) \cdot C_{m\alpha} = 0 \tag{7.17}$$

It is important not to forget some hypothesis and considerations related to this equations,

- In general, the stability derivatives are a function of angle of attack, control-surface angle(s), Mach number, Reynolds number, thrust coefficient, and dynamic pressure Howard (1997). For this reason, applying the equation in a period of time in which the speed is changing is not strictly correct. However, it is possible to use a shorter time period so that the velocity change is reduced.
- External perturbations are neglected. This is possible considering that the analysis is done once the rocket has certain speed, therefore external perturbations are neglected in front of the aerodynamics moments.
- The thrust of the motor is considered to be aligned with the center of mass of the rocket,therefore it does not produce moment.

7.2 Axial acceleration

As it has been said in the beginning, the function $z\,(t)$ and its derivatives are known. Therefore the acceleration of the rocket over time is available.

$$F = \frac{d\,(mv)}{dt} = \dot{m}v + ma \tag{7.18}$$

$$-T + ma = -W - D \tag{7.19}$$

$$ma = T - W - D \tag{7.20}$$

Similarly to the procedure done before, the drag can be expressed as follows,

$$D = qS \cdot C_D \tag{7.21}$$

Then, the final equations can be arranged as,

$$a - \frac{T}{m} - g - \frac{qS}{m}C_D = 0 \tag{7.22}$$

In this equation,T and C_D are unknown.
Similarly to the stability analysis,

- C_D is principally function of the configuration shape, thrust coefficient, and Mach number Howard (1997). A classical example of this dependency in the case of an sphere is shown in Figure 7.2, Schlichting (1960).

Both the function $T\,(t)$ and the C_D coefficient can be estimated in two simple steps,

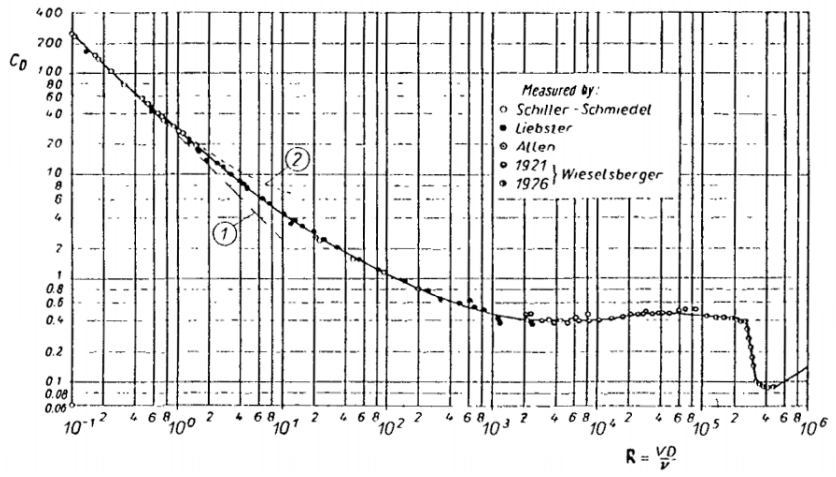

Figure 7.2: Drag coefficient for spheres as a function of the Reynolds number according to Stokes Theory (1) and Oseen's Theory (2) .

1. The C_D is estimated in a time period after the burn of the motor. With this $T = 0$, and $C_D$can be estimated using a least-squares method, then

$$C_D = \frac{m\,(g - a)}{qS} \tag{7.23}$$

1. Assuming that C_D is constant over time, the only unknown during the rocket burnout is the thrust function, which can be directly estimated rearranging Equation 7.22.

$$T\,(t) = ma - mg - qSC_D \tag{7.24}$$

Part II

Practical approach

8 Hardware design and implementation

8.1 Introduction

The aim of this section is to describe the developed hardware platform that is used in order to get the physical magnitudes that are measured. These magnitudes are the following,

- Acceleration (3-axis)
- Angular Speed (3-axis)
- Magnetic field (3-axis)
- Pressure
- Temperature

The first two magnitudes represent the inertial information. From a mathematical point of view, a complete description of this magnitudes over the time are enough in order to determine the trajectory of any motion. However errors due to bias, non linearity, hysteresis, etc, of inertial sensor result make them useless if they are not combined with no drift information about orientation and position. In order to correct drift in orientation, the magnetic field is used. The barometer is used also to correct drift in height. Temperature is necessary in order to correct the pressure.

8.2 Components

According to the cost requirements and capabilities, the following components have been chosen in order to prepare the electronics.

8.2.1 Arduino Nano V3

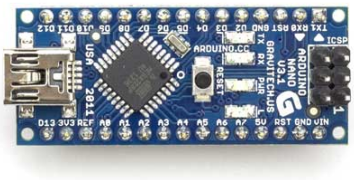

Figure 8.1: Arduino Nano V3

The Arduino Nano is a small, complete, and breadboard-friendly board based on the ATmega328. Specifications are shown in Table 8.1. This board has the advantage of having all components required for the microcontroller to work. However, if the same components were placed in a PCB in addition to the sensor components, the size of the whole system could be reduced, but it would be more difficult to design, manufacture and assemble. The component is shown in Figure 8.1.

8.2.2 GY-87

This is a assembled breakout board that includes the three following sensors, in addition to the required components such as resistors and capacitors. For this reason, it is theoretically plug-and-play. The three components share the I2C bus, from which the microcontroller can communicate with each one of the sensors, by means of an address. Further information about the I2C protocol can be found easily. The component is shown in Figure 8.2.

Table 8.1: Arduino Nano V3 Specifications

Microcontroller	Atmel ATmega328
Operating Voltage (logic level)	5 V
Input Voltage (recommended)	7-12 V
Input Voltage (limits)	6-20 V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	8
DC Current per I/O Pin	40 mA
Flash Memory	32 KB of which 2 KB used by bootloader
SRAM	2 KB
EEPROM	1 KB
Clock Speed	16 MHz
Length	45 mm
Width	18 mm
Weight	5 g

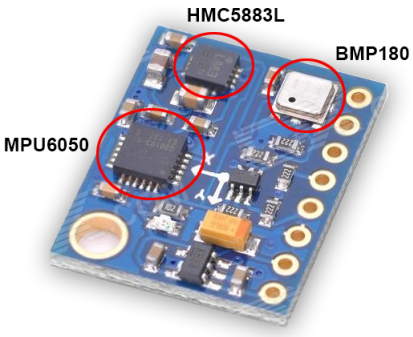

Figure 8.2: GY-87 breakout board

Table 8.2: MPU6050 Main Specifications

Gyroscope	
Full-Scale Range	± 250 deg/s
	± 500 deg/s
	± 1000 deg/s
	± 2000 deg/s
ADC Word Length	16 bits
Output Data Rate	8000 Hz
Accelerometer	
Full-Scale Range	± 2 g
	± 4 g
	± 8 g
	± 16 g
ADC Word Length	16 bits
Output Data Rate	1000 Hz

Table 8.3: HMC5883L Main Specifications

Full-Scale Range	± 1-8 Gauss
ADC Word Length	12 bits
Output Data Rate	75 Hz Continuous Mode
	160 Hz Single Measurement Mode

MPU6050 The MPU-6050 is a 6-axis MotionTracking device designed for the low power, low cost, and high performance requirements of smartphones, tablets and wearable sensors. It combines a 3-axis gyroscope and a 3-axis accelerometer on the same silicon. The device can access external magnetometers or other sensors through an auxiliary master I^2C bus, allowing the devices to gather a full set of sensor data without intervention from the system processor. Specifications are shown in Table 8.2.

HMC5883L The Honeywell HMC5883L is a surface-mount, multi-chip module designed for low-field magnetic sensing with a digital interface for applications such as lowcost compassing and magnetometry. Applications for the HMC5883L include Mobile Phones, Netbooks, Consumer Electronics, Auto Navigation Systems, and Personal Navigation Devices. Specifications are shown in Table 8.3. Two different sampling configurations are available. first, continuous mode, in which the ADC continuously samples data and stores it in the register. In the single measurement mode, conversion begins when it is told by the I2C master, the microcontroller, and information is available when a Data Ready pin is HIGH.

Table 8.4: BMP180 Main Specifications

Pressure Range	300 - 1100 hPa
RMS noise	0.02 - 0.06 hPa depending on the mode
	0.17 m - 0.5 m depending on the mode
Output Data Rate	222 - 39 Hz depending on the mode
ADC Word Length	32 bits for pressure
	16 bit for temperature

Figure 8.3: Batteries

BMP180 It is a digital barometric pressure sensor of Bosch Sensortec, which enables applications in advanced mobile devices, such as smart phones, tablet PCs and sports devices. It follows the older sensor BMP085 but brings some improvements, like the smaller size and the expansion of digital interfaces. The available libraries of BMP085 are also compatible with this sensor, since the communication protocol is exactly the same as the older one. Main specifications are shown in Table 8.4.

8.2.3 Batteries

Two batteries, shown in Figure 8.3, are used to power the board. Each battery has one cell of 3.7V, so it is necessary to connect them in series. Main specifications of the battery are shown in Table 8.5.The board has been tested with the batteries, in logging conditions ,ad is shown to be working more that 1.5 h.

8.3 Development of the board

The development of the board is a complex process that requires several iterations, from a prototype to a final version.

Table 8.5: Battery Main Specifications

Voltage	3.7V
Capacity	175 mAh
Weight	6 g
Dimensions	56 x 12 x 6mm

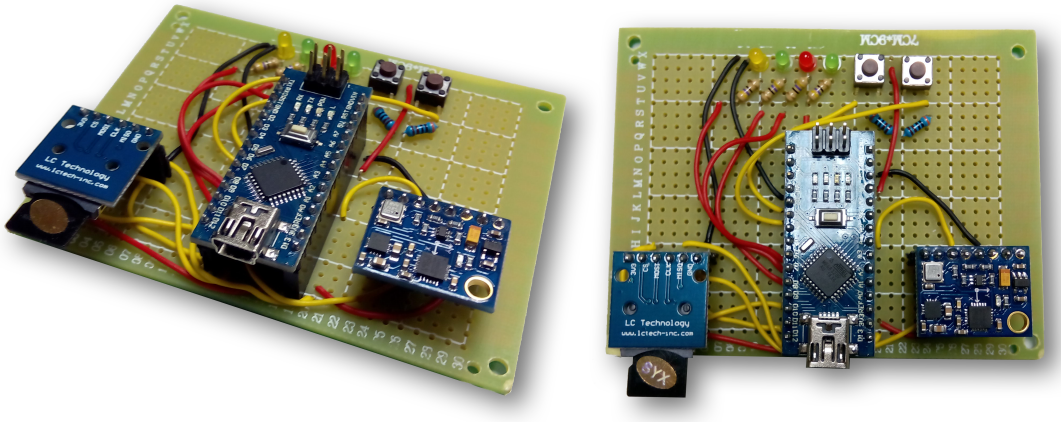

Figure 8.4: Prototype board

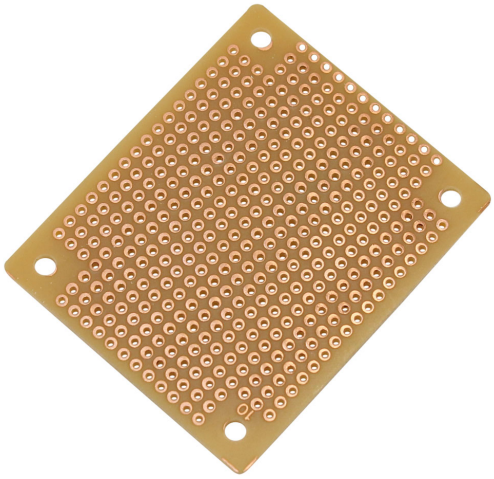

Figure 8.5: Raw perfboard

8.3.1 Prototype

In order to test the components and develop the first version of the software, a prototype was made. This prototype was designed without taking into account the dimension restrictions because it was just for testing and debugging. This board is shown in Figure 8.4. The circuit is soldered in a perfboard, which is a surface with a matrix of holes in which the components are soldered in one side and connected with cables. An example of a raw perfboard is shown in Figure 8.5.

All possible components that might be useful were placed in this board, despite of the fact that some of them were no used in the PCB version of the board,

- Arduino Nano
- micro SD reader
- IMU breakout board
- 4 leds (2 green, 1 red and 1 yellow)
- 2 push buttons

8.3.2 PCB v.0

After the connections were checked in the prototype and the first version of the Arduino software was developed.

A first design of the PCB board was developed, shown in Figure 8.6, using a free software called Fritzing. The computer design of the PCB is shown in Figure 8.7. Once the board design was finished,the circuit was sent to the online shop of Fritzing[4].

In order to simplify, only the very crucial components were included,

- Arduino Nano
- micro SD reader
- IMU breakout board
- 3 leds (1 green, 1 red and 1 yellow)
- 1 push buttons
- Expansion female header with unused digital and analog pins to be used in a future projects.

Problems and solutions After a long time of debugging, some important errors and necessary improvements were identified,

- In the design, the push button was connected to the digital pin of the Arduino and to 5V, without any resistor in the circuit. This was because it was thought that it was possible to enable an internal pull-down resistor of the Arduino. However, it is only possible to enable an internal pull-up resistor. For this reason, in the next design the button should be connected to the digital pin of the Arduino and to ground.
- Many problems arose when the board was working. All examples for each component (IMU, SD card) worked fine separately, however, when all components were working at the same time there was unexpected results. The data save into the file was not consistent, specially with the magnetometer and barometer. Strange values appeared at a regular intervals. After hard work on debugging, some bugs were found in the software, but they did not solve the problems. Because the problem arose when the components working simultaneously, it seemed that there was a problem with power supply. In the design, both components were powered on the 3.3V bus of the Arduino. and it is definitely a bad strategy. The SD card has high current peaks which may affect the IMU sensors and also the 3.3V regulator of the Arduino is an internal one of the USB communication chip, with a maximum current supply of 150 mA. Two decisions were taken,
  - In the next board, IMU sensors and SD reader must be powered by different buses.
  - An external 3.3V regulator must be used inf the board, with higher maximum current supply.
  - Capacitors must be placed in the power supply of the SD reader and the IMU sensors.
  - The connections of the board should be shorter and without small angles, in order to avoid electromagnetic interferences.

[4]http://fab.fritzing.org/fritzing-fab

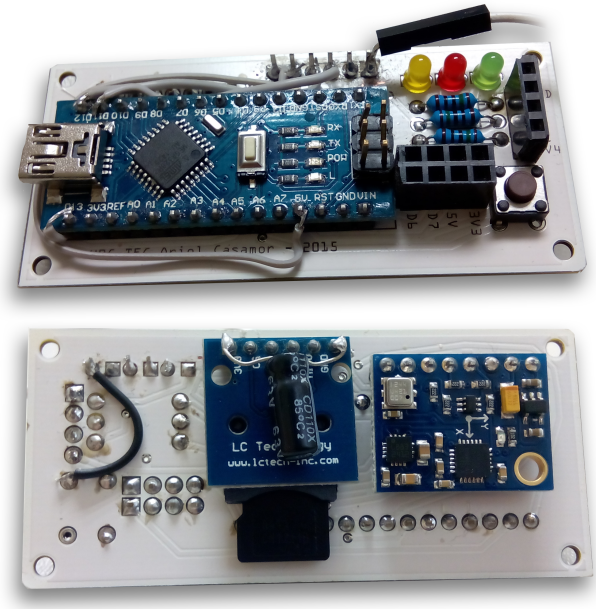

Figure 8.6: First version of the board after some modifications

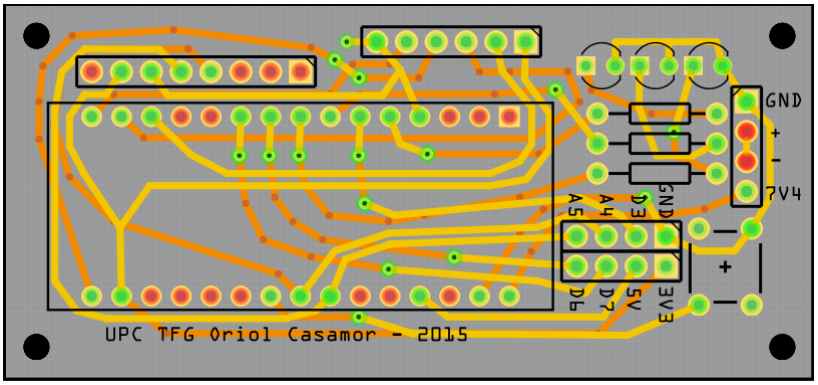

Figure 8.7: PCB computer design of the PCB v.0

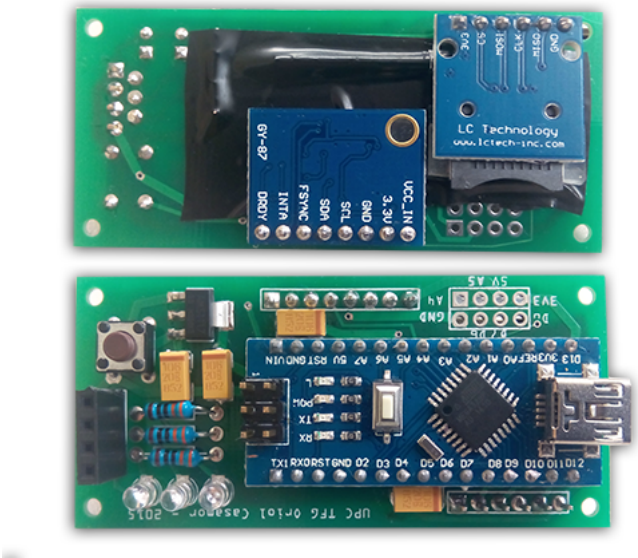

Figure 8.8: Second version of the board

8.3.3 PCB v.1

With all the knowledge and decisions taken using the previous board, the circuit design was totally renewed. It is shown in Figure 8.10. The assembled board is shown in Figure 8.8. In this case the board was ordered in China to be manufactured, in roder to minimize costs. The final circuit is shown in the Appendix.

The main modifications are listed below,

- Push button connected to ground instead of +5V.
- Power tracks (5V, 3.3V, and GND) are wider than the signal tracks where possible.
- IMU sensor placed next to the I2C pins of the Arduino and microSD card reader placed next to the SPI pins of the Arduino. With this two changes, tracks are significantly shorter and with less turns.
- Capacitors are placed in the power tracks next to the IMU and microSD reader, with a capacity of 10uF, in order to avoid problems with high current peaks.
- External 3.3V regulator included. The specific components name is LM1117, with a maximum output current of 1A, and according to the specifications sheet, it is placed next to two condensers, as shown in Figure 8.9.

Both the capacitors and the regulator are SMD (Surface Mount Device), which means that they are directly soldered to the board, without using cabled or pins. An analysis of the electronic performance of the board and a test in order to see the effectiveness of the design is detailed in Appendice C.

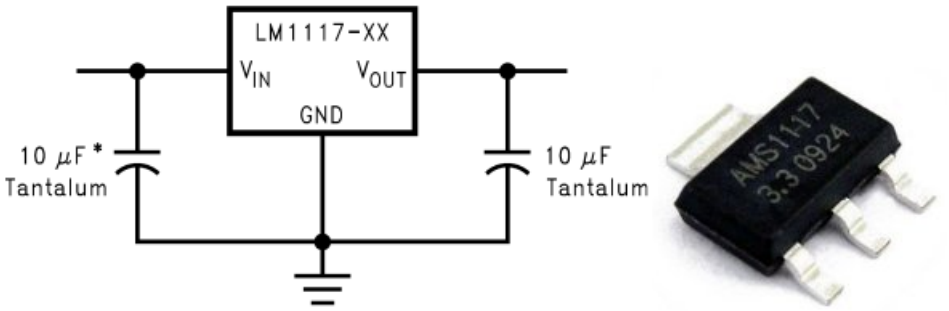

Figure 8.9: LM1117 as a fixed output regulator

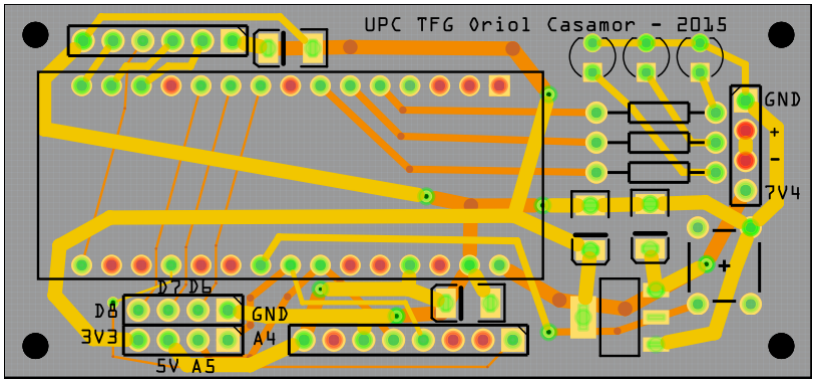

Figure 8.10: PCB computer design of the PCB v.1

9 Software implementation

9.1 Board

The data logger board, based on Arduino, can be programmed using the Arduino IDE in C language. The software has two different sections, calibration and logging. All the different options are selected using only a push button. Using software and the internal timer, the program can identify if it is a short or long click, according to the times defined in the program. The function also considers a minimum time to avoid button bouncing in the transition between on and off, as shown in Figure 9.1. In order to give feedback to the user, three color leds have been used. The finite state machine representation of the whole software in the board is shown in Figure 9.2.

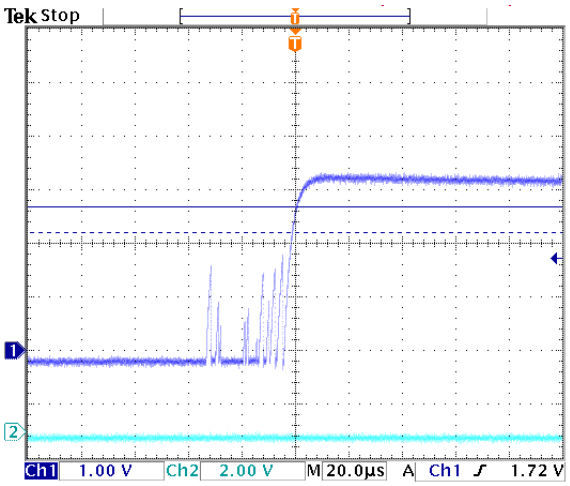

Figure 9.1: Button bouncing effect

9.1.1 Setup

When the board is turned on, the setup function is executed. This function contains all the required actions that must be completed before the board being ready to operate in normal mode. The procedure is as follows,

1. Pin mode setup
   (a) Led pins are set as a digital output
   (b) ChipSelect pin, required for the SD card, is set to output mode according to the library indications.
   (c) Button is set as a digital input, and the internal pull-up resistor enabled
2. Sensors initialization
   (a) I2C communication is enabled
   (b) MPU6050 is started
   (c) HMC5883L is started and set to continuous mode
   (d) SD card initialization. Success on this operation is represented with red or green led turning on.

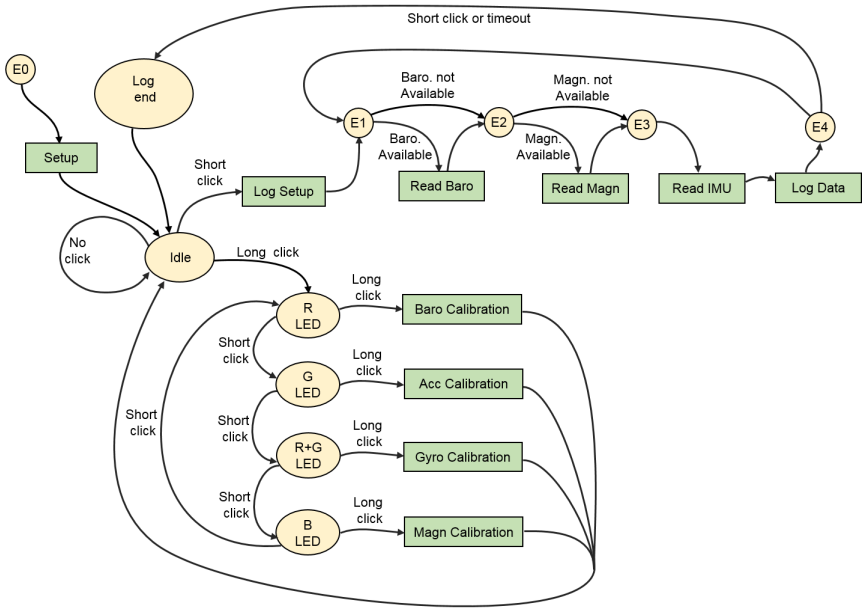

Figure 9.2: Finite state representation of the board software

9.1.2 Calibration menu

When the board is in idle state, a long press initializes the calibration menu. In this menu, by short clicking the press button the user can switch between barometer, accelerometer, magnetometer and gyroscope calibration. The selection is shown by turning on the corresponding led, red, green and blue. Then, when the button is long-pressed, the calibration function for the corresponding sensor is activated.

The barometer function stores the calibration values that are available in the sensor EEPROM memory into a file.

The accelerometer function needs the user to put the board in six different positions. The user must wait for the green led to be turned on, then when the button is pressed, after a small delay, 50 samples of acceleration are taken and stored into a file. When the sensor is sampling, the red light is turned on so that the user knows that he must not move the board.

The gyroscope calibration functions works equivalently to the accelerometer, but there is four different sets of data, instead of six. The first one corresponding to a non-rotation state, and the three other ones corresponding to a rotation with known acceleration in thee different axis.

When any of the calibration functions is finished, the programs return to idle mode.

9.1.3 Data logging

Data logging is by far one of the most tricky parts of the project. Running each sensor separately and at low sample rate is easy, but sampling all the sensors at high frequency makes the process far more difficult.

The procedure for the logging setup and the logging loop is as follows,

Setup

1. File creation

   Data files are titled according to the following format, DATAX.BIN, where X is an integer. Every new file continues the numeration of the highest file available.

2. Barometer call

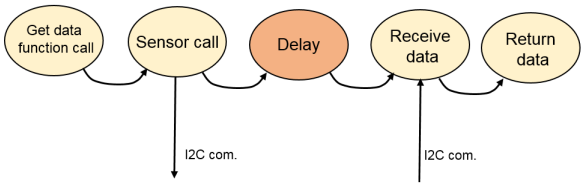

Figure 9.3: Barometer library scheme

The barometer call for a data conversion is executed.

Data logging loop

1. Barometer pooling. As it has been said before, a timer is checked in order to know if the conversion is ready to be read.
2. Magnetometer pooling. The magnetometer is called if a period specified time has elapsed from the last call, in order to limit the sample rate.
3. IMU. The inertial measures, accelerometer and rotations are sampled at each cycle.
4. Turning on or off the green led after a determined number of cycles, so that the user has a feedback of the logging rate.

9.1.4 Debug mode

In the source code file header, it can be chosen whether to enable serial outputs to debug the program or not, as shown below. As it is indicated, enabling this mode can cause the program to being unstable, because serial outputs have an important effect in SRAM memory of the microprocessor. Despite the SRAM memory of the microcontroller being 2kb, when few free memory is available the program may have unexpected results or reset. Reset in Arduino caused by low SRAM is difficult to diagnose because there is not any indicator of this happening.

```
#define DEBUG 1
#undef DEBUG // Comment this line to enable debugging output in Serial
                // Debug mode may be unstable due to low free SRAM memory
```

9.1.5 Libraries

BMP085 One of the first problems that was found when programming the board was the barometer library. This library is not efficient when other sensors have to be logged at the same time, because it handles the conversion time of the sensor with an intern delay function, which means that the program is stopped when the barometer is sampling, as shown in the Figure 9.3.

The library needed to be modified, and the delay function was replaced with a timer, so that at each loop of logging the time of execution of the sensor call is checked. If elapsed time is greater than the required time of conversion, pressure and temperature are logged, otherwise this step is skipped. The required time for the sensor to get a data conversion is clearly indicated in the sensor specifications.

MPU6050 The library for the IMU sensor is one belonging to the I2Cdev library collection.[5].

HMC5883L The library for the magnetometer is one published in bildr.org[6]

[5] http://www.i2cdevlib.com/devices/mpu6050#source
[6] http://bildr.org/?s=hmc5883l

9.1.6 Data file

A crucial part of the program is how data is stored in files. Two main possibilities are available, ASCII files and binary files.

In ASCII files, numbers are stored as characters and usually separated by an special character such as a comma. Each character occupies 1 byte, 8 bits. The advantage of this system is that the user can open the file and read it directly using any text editor, for debugging purposes. The main disadvantage is that numbers require more memory to be stored, and therefore more time to be written in the SD card.

In binary files, numbers are stored the exact way that they are in the SRAM memory in the microcontroller, with zeros and ones. This files cannot be read with a text editor, but need to be processed with a program that must know which is the format of the file, as it will be clearly seen in the example.

Because of the sample rate being a basic requirement of the project, the second option was chosen. In the program, a structure to store data has been created as it is shown here,

```
struct Data {
        int16_t ax, ay, az, gx, gy, gz;
        uint16_t temp;
        uint32_t pres, dta, dtb;
        int16_t mx, my, mz;
};
```

This structure occupies 256 bits, 32 bytes, and it is saved into the SD card in one step like,

```
myFile.write((byte *) &myData, sizeof(struct Data));
```

In each data loop, the structure variables are modified. mx, my and mz, corresponding to the magnetometer values are set to zero if there is no magnetometer read in that iteration. The same happens with the pressure.

In order to reduce the size of the structure, one possibility would be to store differences of time instead of the absolute time. The disadvantage is that the size of the variable must be chosen according to the sample rate. The lower the sample rate the larger the variable. In addition, if data is stored in milliseconds or microseconds is also chosen according to the sample rate, so that each time step is significant. The higher the sample rate the more resolution the variable should have, and so more memory. This two effects implies that when sample rate can be variable, according to the user settings, different sizes of the variable should be chosen. For this reason this option has not been used.

9.1.7 Timeout

The logging can be stopped in two ways. First, by short clicking the press button. It can be also stopped if a timeout period has been elapsed. This timeout is set in the program header and avoid unnecessary data logging after the mission has finished.

9.1.8 Button function

One of the main objectives when designing the board was making it as simple as possible. For this reason, it was necessary to implement an easy way to navigate through the menus. A one button system was implemented. By means of a software function inspired by an example [7], the program can identify if the user has clicked the button shortly or a long time, and also to avoid button bouncing.

[7] http://jmsarduino.blogspot.com.es/2009/05/click-for-press-and-hold-for-b.html

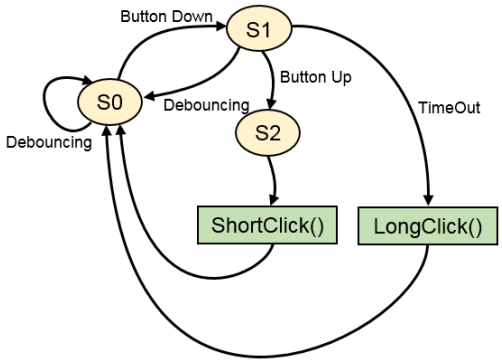

Figure 9.4: Finite state representation of the button function

9.2 Post processing

Once the experiment has taken place and a set of data has been logged, it is necessary to process this information in order to go through the performance analysis. Two different steps are necessary, first, converting the binary data file into variables in the MATLAB environment, and after processing this data.

10 Testing

10.1 Test case #1

A sample case has been designed in order to validate part of the post processing software. This consists in a known trajectory which can be compared to the output of the program in order to test if all the calculations are properly implemented. This test is only useful for analyzing inertial data, accelerometers and gyroscopes.

For this purpose, a turntable has been used. The sensor has been placed at the end of the disk, as shown in Figure 10.1. The motion consists in the following parts,

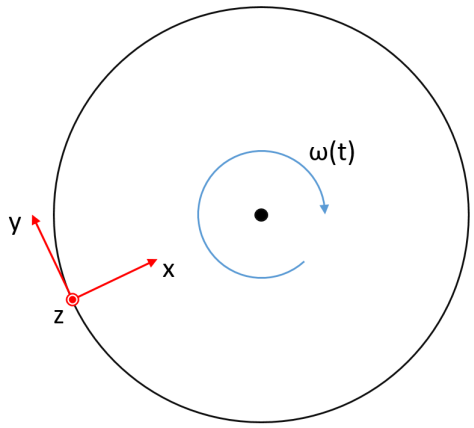

Figure 10.1: Scheme of the test case #1

1. Stationary
2. Accelerating
3. Constant angular speed
4. Decelerating
5. Stationary

With this motion, the expected output should be something similar as the plots shown in Figure 10.3, while the real data from the experiment is shown in Figure 10.4. Some conclusions can be easily found without processing any data. For this experiment, the full scale range of the accelerometers is set to $\pm 2g$ and the full scale range of the gyroscopes to $\pm 500^{\circ}/s$.

- Accelerometers have an important noise component
- Gyroscopes have also noise but it is significantly less than the accelerometers.
- There is a misalignment of the axis in relation with the gravity. This is easily seen by means of analyzing the oscillations that appear in the accelerometers. These are produced by the variation of the angle between the sensor axis with the gravity vector, as seen in the example in Figure 10.2. This is also confirmed because the rotation vector is seen with two a main component in z-axis and two smaller components in x and y axis.

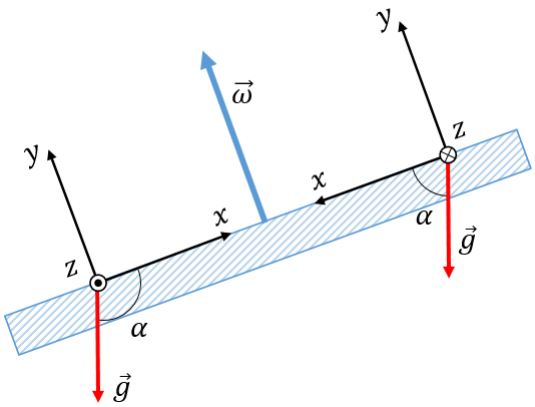

Figure 10.2: Example of misalignment of the accelerometers axis with respect to the gravity

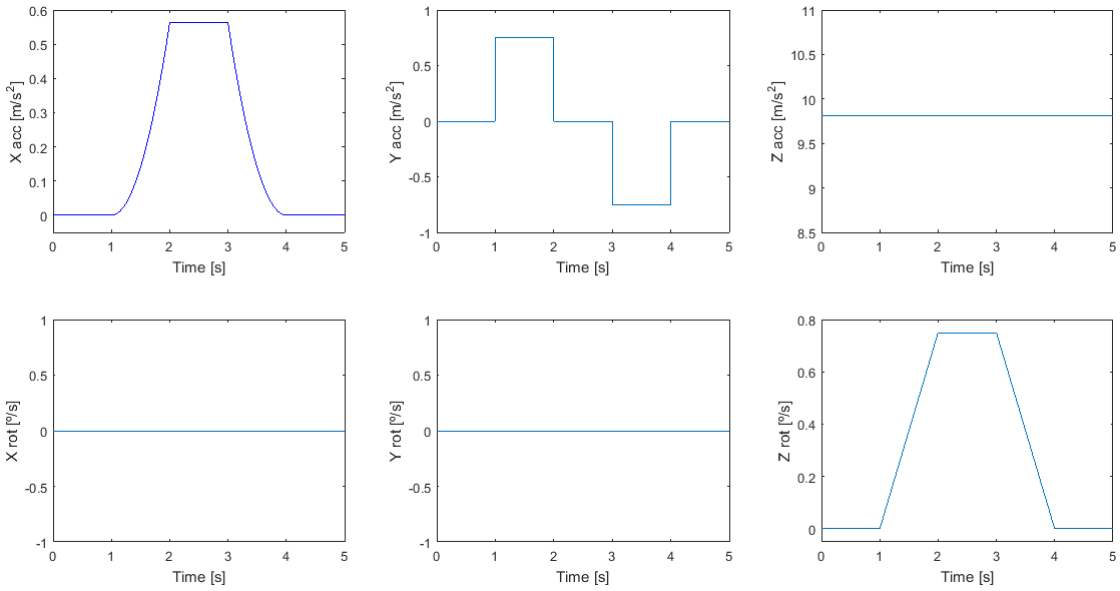

Figure 10.3: Sample output from inertial sensors in Test Case#1

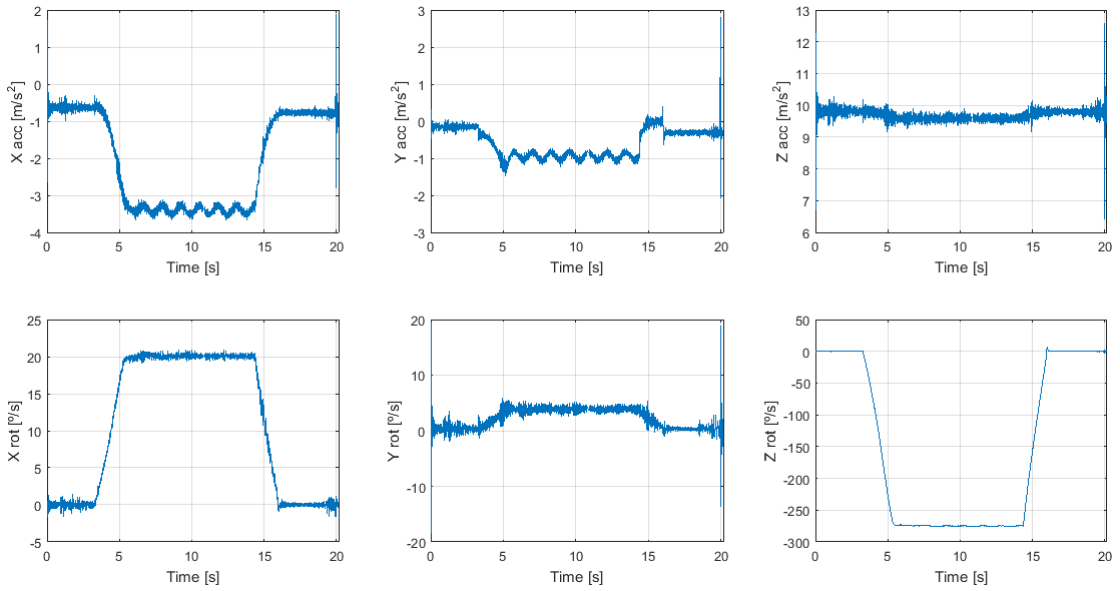

Figure 10.4: Inertial Raw Data from Test Case#1

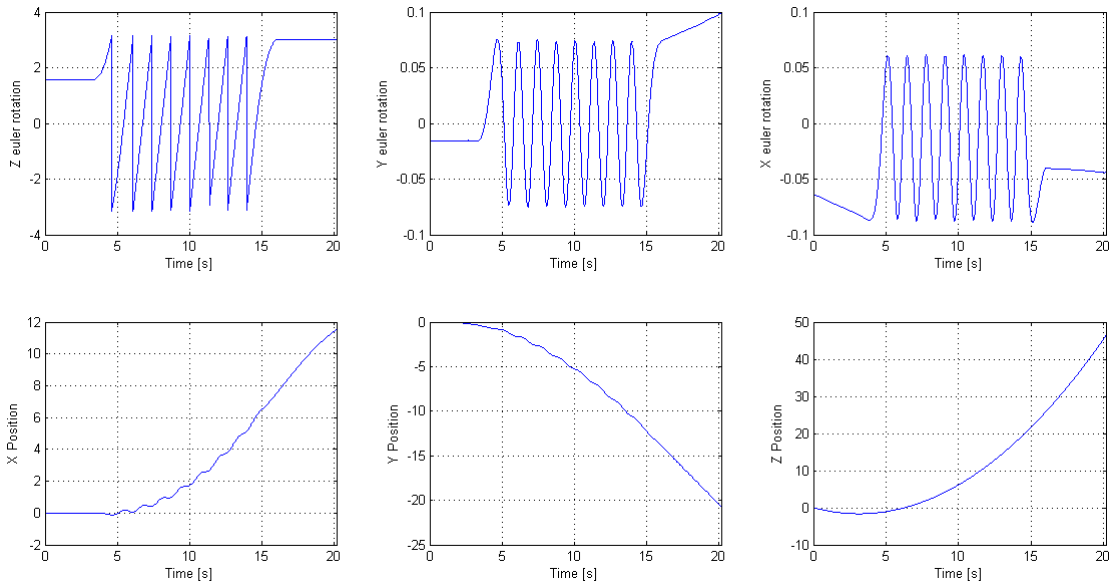

Figure 10.5: Integrated data from Test Case #1

10.1.1 Sensibility of the sensor bias error

In order to test the inertial processing, without considering any other input from other sensors, the data obtained in the Test Case #1 has been integrated, to obtain the data on Figure 10.5. It seems clear that mainly because of the bias error on the accelerometers the error in the position increases with a quadratic rate. In order to confirm this hypothesis, an algorithm has been used, that finds bias error in gyroscopes and accelerometers that minimize some error function. The expected result would be a sinusoid curve in the trajectories.

Initial bias minimization In this first case, the condition is to minimize the displacement and rotations inf the first seconds, when the sensor is stationary. The results are shown in Figure 10.6. The results clearly show the great influence of the bias error in the final results. Despite the results not being acceptable, the improvement in the error in the position at the end of the motion is considerable. This also suggests how important is to make a calibration just before the mission, so that the sensors are in the right conditions of temperature, for example, which may have a noticeable influence on the output.

Initial and final bias minimization In this second case, more information was introduced into the problem. The displacement in the trajectory is minimized both at the beginning and in the end of the motion. The results are shown in Figure 10.7. The displacements at the end of the trajectory are even better that in the previous case. The conclusions that result from this test is that the more information about the model that is introduced in the problem, the better the results. However, it is not clear if in the case of the rocket or any other vehicle, it is possible to introduce such kind of additional information.

10.2 Rocket Launch

First attempt

Second attempt The analysis of this launch is covered in a separated section for clarity.

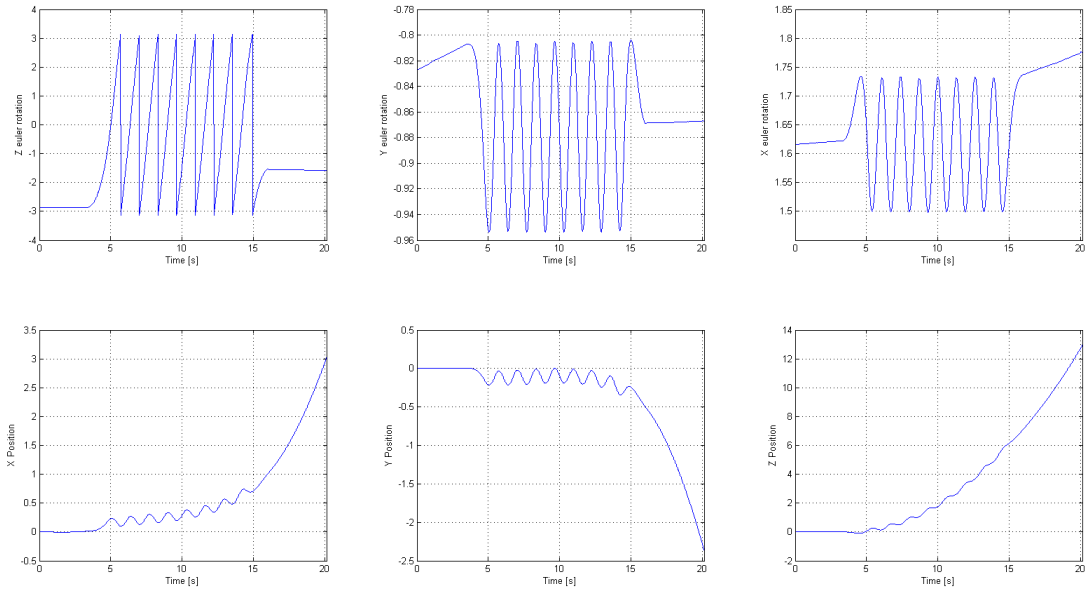

Figure 10.6: Integrated data from Test Case #1 with initialization

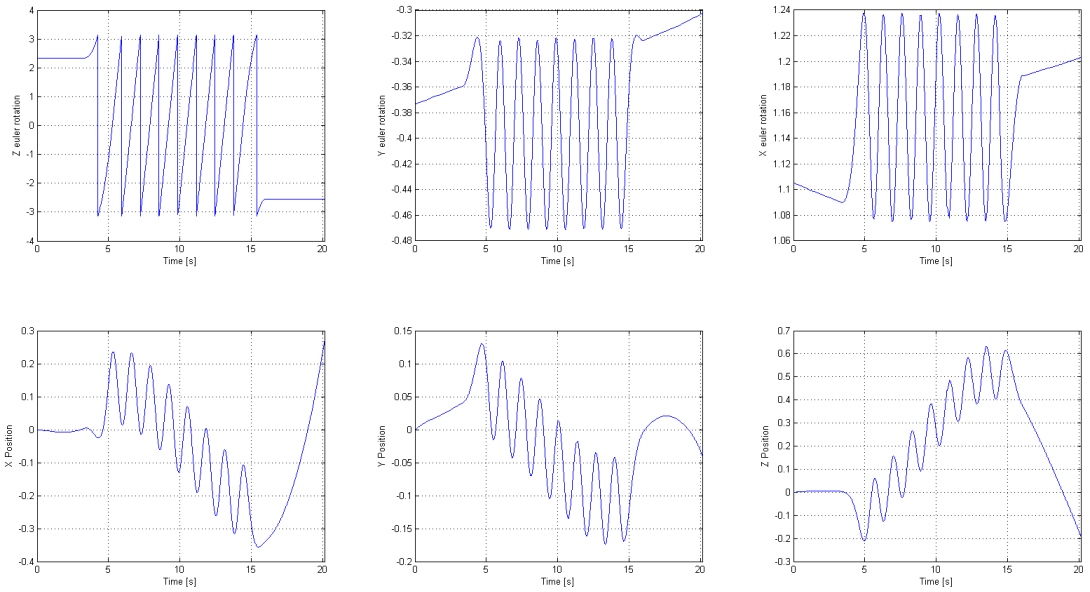

Figure 10.7: Integrated data from Test Case #1 with initialization both in the beginning and in the end

11 Launch analysis

The rocket to test is one of the TRT. It is shown in Figure 11.1. It was built by some members last course and only needed some modifications to be able to bring the electronics. The red section and the nose con are the payload vessel. Two 3D printed parts were designed in order to fix the electronics properly to the structure. The top part can be seen in Figure 11.1. Once the rocket was finally assembled, the geometric and mass properties have been calculated. The engine, Estes D12-7 is designed to lift rocket up to 226 g, however, being the maximum acceleration that the sensor can handle 16g, it its important to limit the acceleration by adding some mass. Some additional weight was added in the payload vessel for this purpose. All data is summarized in Table 11.1. With this, the theoretical maximum acceleration is,

$$F = m \cdot a = 29.73N = 0.195\,kg \cdot a \;\rightarrow\; a = 152.46\,\frac{m}{s^2} = 15.54g \tag{11.1}$$

An important test to do before launch, despite of any previous calculation is to make a stability test. This test consists on attaching a cord in the position of the center of gravity of the rocket. Then, force to rocket to describe circles in the air. If the rocket swings without turning the stability is correct. The test is shown in Figure 11.2.

During the launch, the rocket burns for 1.6 s, according to the specifications of the engine, then it has a free flight trajectory for 7 s, and after the rocket has a little explosion backwards that ejects the parachute and the payload vessel.

A first attempt was made on May 3th. The launch was successful but when the payload vessel had to be ejected, the joint of the parachute with the payload failed and it fell into the ground. There was not important damage nor in the structure or the electronics,but the log file was corrupted, probably due to the final shock.

Table 11.1: Specifications of the rocket and engine

Rocket Weight	
MTOW	226 g
TOW	195 g
ZFW	170 g

Engine Specs	
Maximum Thrust	29.73 N
Propellant Weight	21,1 g
Thrust Duration	1.65 s
Delay	7 s

11.1 Launch #1

A second attempt was made on May the 4th. The connection of the parachute with the payload vessel was reinforced and the recovery was successful. There is no pictures of the launch. There was a moderate wind, and this means that the rocket with the parachute may fall quite fare from the launch point. For this reason, it was considered to be important to be near the expected landing point rather to be close to the launch to record.

The picture of the first launch is shown in Figure 11.3.

11.1.1 Geometry, conditions and mass

In order to proceed with the calculations, the following data has been used,

- Rocket diameter $D = 4\,cm = 4 \cdot 10^{-2}\,m$

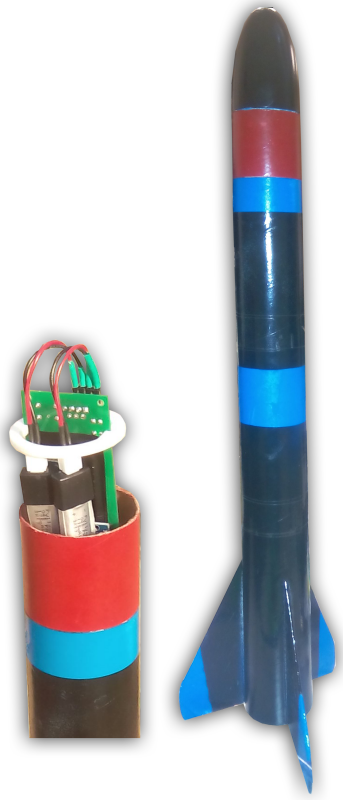

Figure 11.1: Test rocket

Figure 11.2: Stability Test

Figure 11.3: First launch picture

- Rocket characteristic area, is the cross sections of the rocket, $S = \frac{D^2}{4} = 4 \cdot 10^{-4}\, m^2$
- Rocket characteristic length, is the length of the rocket, $L = 40\, cm = 4 \cdot 10^{-1} m$
- Initial mass, $m_i = 195\, g = 1.95 \cdot 10^{-1}\, kg$
- Final mass, $m_f = 170\, g = 1.7 \cdot 10^{-1}\, kg$
- Air temperature, $T = 30^{\text{o}}\text{C} = 303.15\, K$, according to the barometer initial measurement.
- Air pressure, $p = 9.66 \cdot 10^4\, Pa$, according to the barometer initial measurement.
- Air density, $\rho = \frac{P}{287 \cdot T} = 1.1103\, \frac{kg}{m^3}$, according to Eckert and Drake (1972).
- Air viscosity, $\mu = \frac{2.5393 \cdot 10^{-5} \cdot T}{273.15} \cdot \frac{1}{1+122/T} = 2.0095 \cdot 10^{-5}\, Pa \cdot s$, according to Eckert and Drake (1972).
- Reynolds number, $Re = \frac{\rho U L}{\mu}$, the density and viscosity are considered to be constant.

11.1.2 Raw data

First, we will take an overview on the raw data logged by the board.

- The inertial data is shown in Figure 11.4. The three top plots represent the linear acceleration in each axis, while the bottom plots represent the rotations. As it can be seen, the y-axis is the one mainly contained in the longitudinal axis of the rocket. Also it is clearly seen that approximately in $t = 7.3s$ there is the rocket ignition.
- The barometer and temperature data is shown in Figure 11.5.The data is cropped from approximately 5 seconds before the engine ignition until the rocket apogee. The left plot is not an indicator of the air temperature but the sensor temperature. It increases due to the usual heating of operation. It is also seen that the rocket reaches approximately 200 m from the ground.

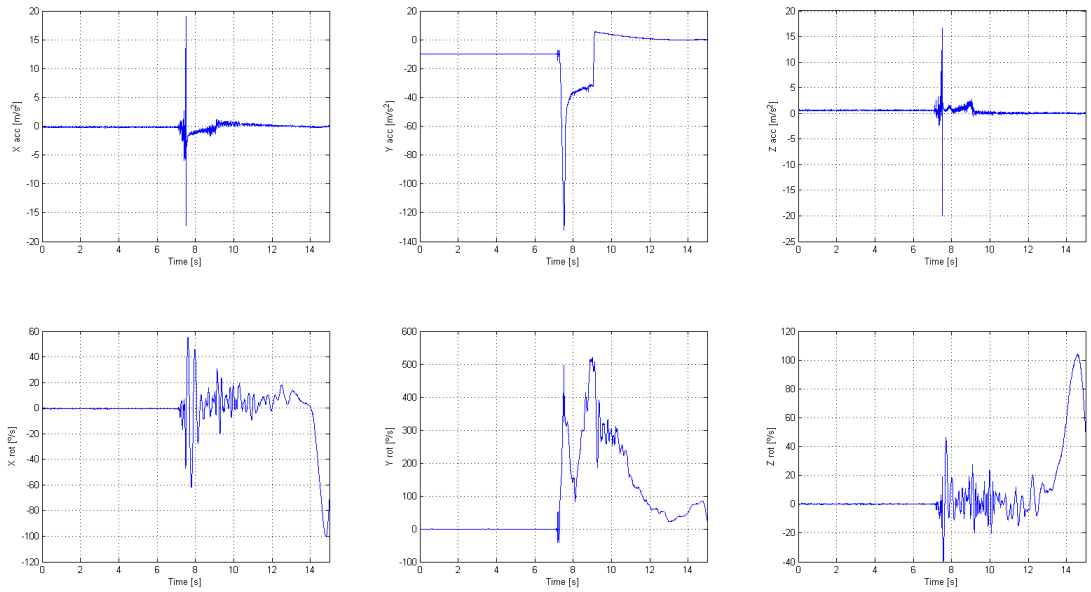

Figure 11.4: Launch #1 inertial raw data

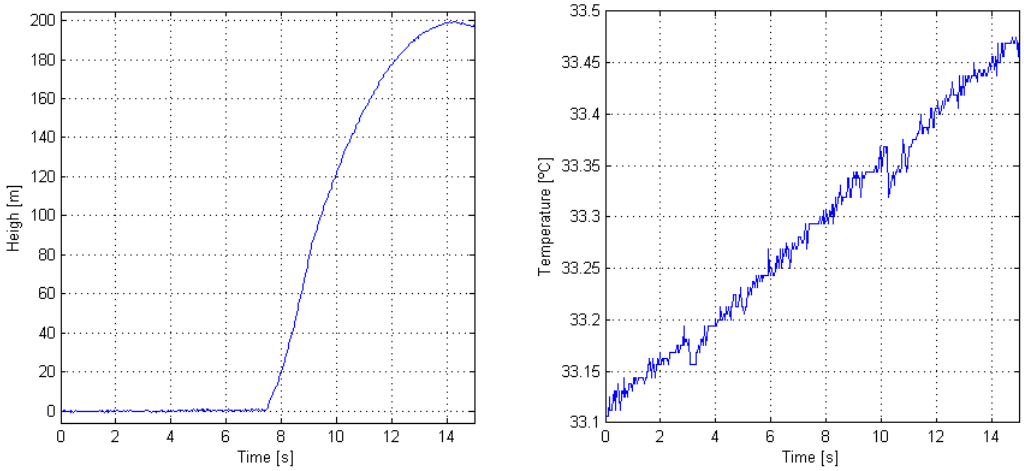

Figure 11.5: Launch #1 barometer and temperature raw data

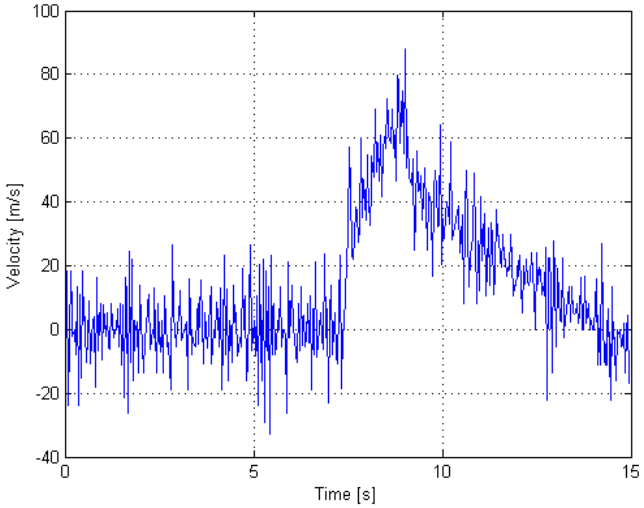

Figure 11.6: Launch #1 numerical differentiation of the position to obtain velocity

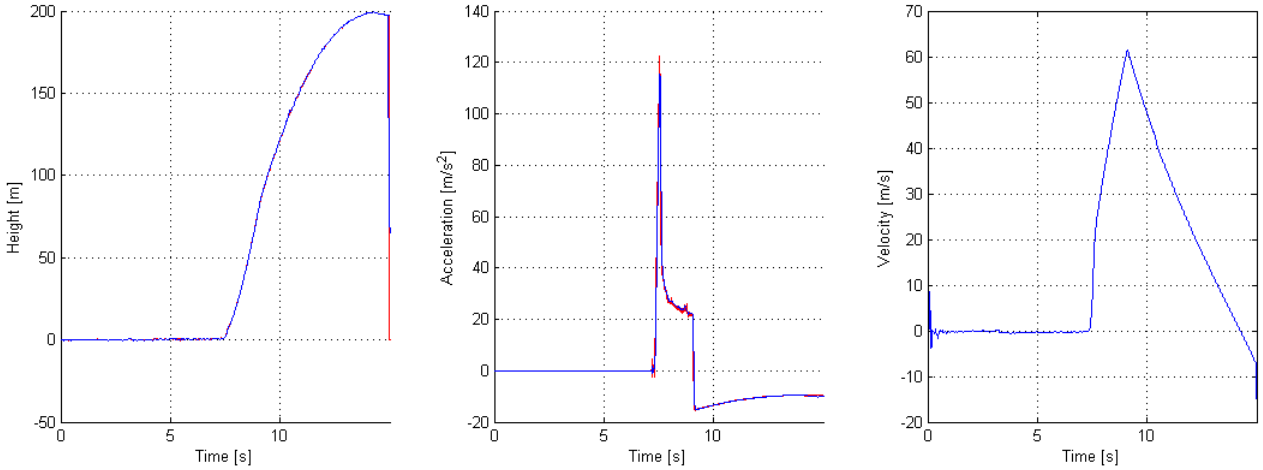

Figure 11.7: Launch #1 filtered height and acceleration, and velocity estimation (blue) and raw data (red).

11.1.3 Data fusion

Once the raw data is obtained, it is time to combine the barometer data and the accelerometer data by means of the Kalman filter, as it has been explained in previous sections. The result of this processing is shown in Figure 11.7. Three plots are shown, height, acceleration and velocity. The two first ones are quite similar to the measured data, except for the noise reduction, which can be seen in detail in Figure 11.4. The most valuable output of implementing this filter is obtaining the velocity. If one derived numerically the height, the output would be very noisy, as shown in Figure 11.6. However, both the acceleration and the height give information about the velocity. If both are fusioned properly, a clear description of the velocity can be obtained. This could not have been done without the Kalman filter.

11.1.4 Drag estimation

The first analysis of the flight performance is estimating the drag coefficient. This can be done after the engine burnout and until the apogee because there is only two forces acting on the vehicle, weight and drag. The acceleration in this period is shown in red in Figure 11.8.This is a quite complex problem that must be simplified in order to be able to solve it with the available

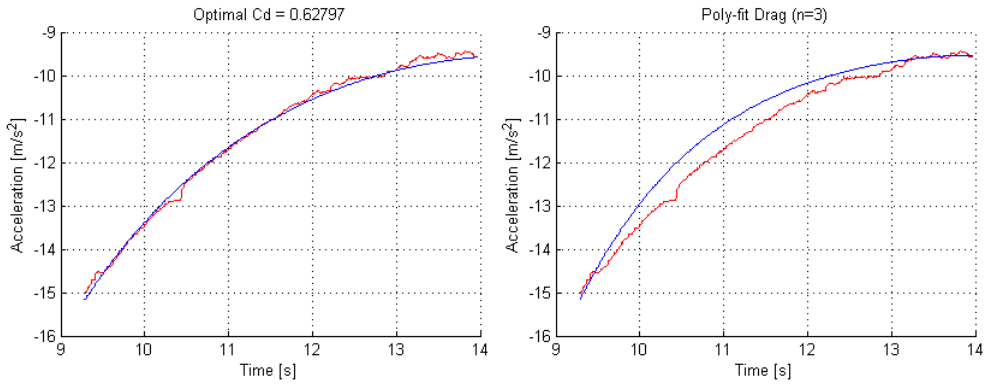

Figure 11.8: Launch #1 acceleration during free flight. Data in red, simulation in blue.

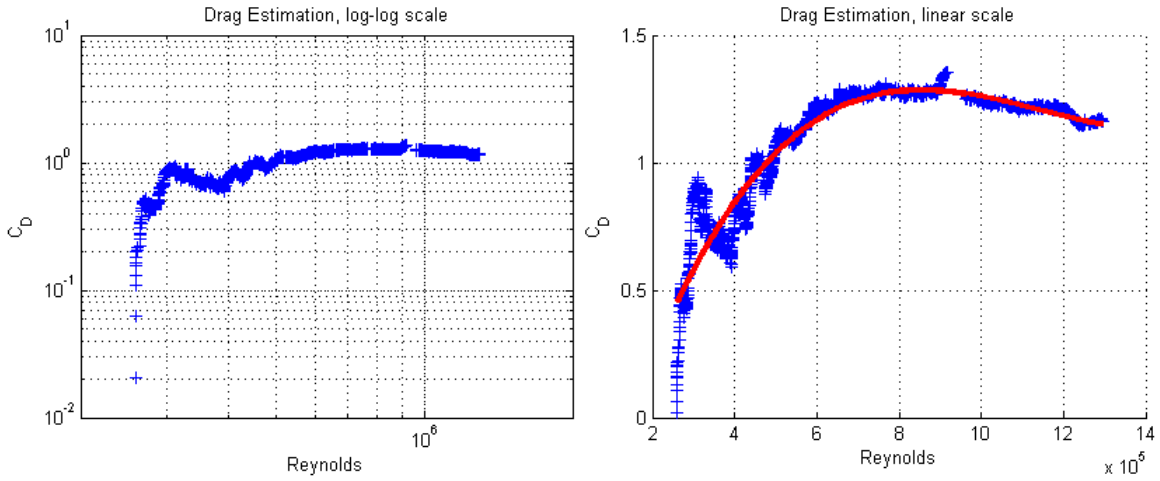

Figure 11.9: Launch #1 estimated drag coefficient of the rocket, computed data (blue) and polynomial fit (red).

data. It is considered that,

- The rocket is flying vertically, as it has been said before. As it has been said in previous sections, height, velocity and acceleration are all in the vertical axis.
- The drag is not dependent on the angle of attack.

With this data, and according to the drag estimation procedure that was explained in Section 7.2, one can obtain the plots on Figure 11.9, in which the drag coefficient is plotted with the Reynolds number. Both plots come from the same data, however the left one is a log-log plot and the right one is a linear plot. The first conclusion that can be obtained when the plots are observed, is that the drag coefficient is dependent with the Reynolds number. However, despite being different case, one could compare this results with the ones in Figure 7.2, corresponding to the drag coefficient of an sphere. One could expect the drag coefficient to decrease with the Reynolds increasing, however the observed relation is the opposite. This is definitively an uncertainty in the results, however, one could think that this relation is not due to a true dependency of the drag coefficient with the Reynolds, but a numerical consequence.

Let's assume the drag coefficient to be constant, According to Eq 7.23, and with a velocity and acceleration conditions such as the ones in Figure 11.10, one could calculate with Eq 11.2 the drag coefficient at any time, except when $V = 0$, where the function can not be evaluated. However, when using real data, the difference $g - a$ is not exactly zero, because of the inaccuracy of the sensors and the noise in the signal. For this reason, the function $C_D(V)$,or $C_D(Re)$, evaluated in its limit when $V \to 0$ tends to $\pm\infty$. This would explain the unexpected tendency of the plot.

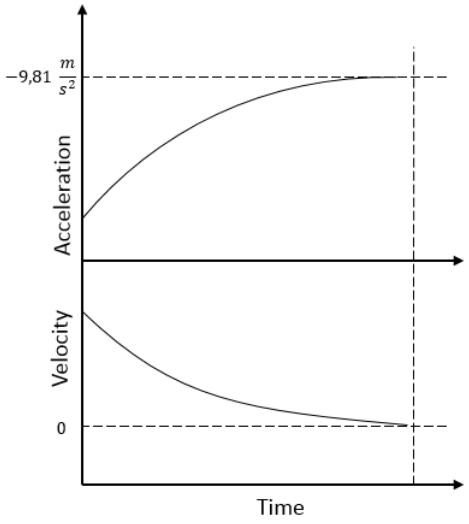

Figure 11.10: Expected velocity and acceleration

$$C_D = \frac{m\,(g-a)}{qS} = \frac{m\,(g-a)}{\frac{1}{2}V^2 S} \tag{11.2}$$

Also, as the velocity increases, the drag coefficient tends to be constant. According to Figure 7.2, in a variation in the Reynolds such as $10^5 < Re < 10^6$ one would not expect a huge variation in the drag coefficient. In order to continue to evaluate the hypotheses of the drag being constant, simulations have been carried out in two different scenarios, constant and variable drag coefficient.

In any case, the differential equation that that has to be solved is,

$$\frac{dV}{dt} = -g - \frac{1}{2m}\rho V^2 S C_D \tag{11.3}$$

In the case of constant drag, the equation is solved in a iterative basis, finding the optimal drag coefficient so that the integrated acceleration and the measured acceleration squared difference is minimum. This results into the drag coefficient being $C_D \approx 0.63$. The result of the simulation is shown in Figure 11.8, in the left plot, red line.

In the case of variable drag, a 3-degree polynomial has been fitted to the points. Then, with the function $C_D\,(Re)$, Eq 11.3 is solved. The result of the simulation is shown in Figure 11.8, in the right plot, red line. As it can be seen, the simulation considering constant drag coefficient is far more accurate that the variable one. From this reason, from this point the drag coefficient will be considered constant as an assumption.

11.1.5 Drag coefficient comparison

There is a free program for model rocket designers called Open Rocket, which is used to estimate rocket performance after the rocket characteristics have been introduced. This software uses correlations in order to estimate parameters such as the drag coefficient. The detailed explanation of the calculations is indicated in the software technical documentationNiskanen (2013). Calculations are carried out assuming Mach number to be $M = 0.1$. The results are shown in Table 11.2 on the following page. The results are surprisingly accurate, however, it is important not to be presumptuous and more test should be realized in order to confirm the calculated values. At least, it is clear that the order of magnitude of the calculated drag coefficient is correct. To sump up, the estimated drag coefficient is $C_D = 0.63$, while the estimated theoretically is $C_D = 0.57$. This means the measurements have a relative error of $\varepsilon_r = 10\%$.

Table 11.2: Contribution to each component to the drag coefficient

	Pressure C_D	Base C_D	Friction C_D	Total Pressure C_D
Nose cone	0.00 (1%)	0.00 (1%)	0.03 (5%)	0.03 (6%)
Body tube	0.00 (0%)	0.12 (22%)	0.17 (30%)	0.30 (52%)
Fin set	0.10 (18%)	0.00 (0%)	0.08 (13%)	0.18 (31%)
Launch lug	0.03 (6%)	0.00 (0%)	0.00 (0%)	0.03 (6%)
Launch lug	0.03 (6%)	0.00 (0%)	0.00 (0%)	0.03 (6%)
Total	0.17(30%)	0.12 (22%)	0.28 (49%)	0,57 (100%)

11.1.6 Thrust curve analysis

The first analysis of the flight performance is estimating the thrust curve. In addition to the assumptions considered before, the new assumption that has been made in order to proceed is,

- The drag coefficient does not depend on the Re nor M during for the flight conditions of the launch.
- Engine exhaust velocity is constant and the pressure in the exhaust plane is the same as the ambient pressure. This is probably not true, and if we were estimating the thrust from this equation the results would be quite inaccurate. However, we are just using this expression in order to get a better approximation of the mass function, which does not have a significant effect in the final computation of the thrust. Therefore, considering Eq 10.4 and 10.6 in Hill and Peterson (1965), the thrust is equal to

$$T = \dot{m} v_e \tag{11.4}$$

where,

- $\dot{m}$ is the mass flow
- v_e is the exhaust velocity of the gases

$$T = ma - mg - qSC_D \tag{11.5}$$

With this assumptions, and following Eq 7.24, which is re-written here for clarity, now it is necessary to estimate the mass of the rocket over time. The first approach could be assuming the mass to decrease at constant rate from m_i to m_f. However, with the assumption made before, one sees that mass flow is proportional to thrust. Considering this, it is possible to estimate the $m(t)$ using a iterative scheme.

1. Assume $\dot{m} = ct$ and obtain $m(t)$
2. Compute $T(t)$ from accelerations
3. Compute $\dot{m}(t) = (m_f - m_i) \frac{T(t)}{\int_{t_i}^{t_f} T\, dt}$
4. $m(t) = m_i + \int_{t_0}^{t} \dot{m} dt$
5. Go to 2 until $max(\mathbf{m}_i - \mathbf{m}_{i-1}) < \delta$

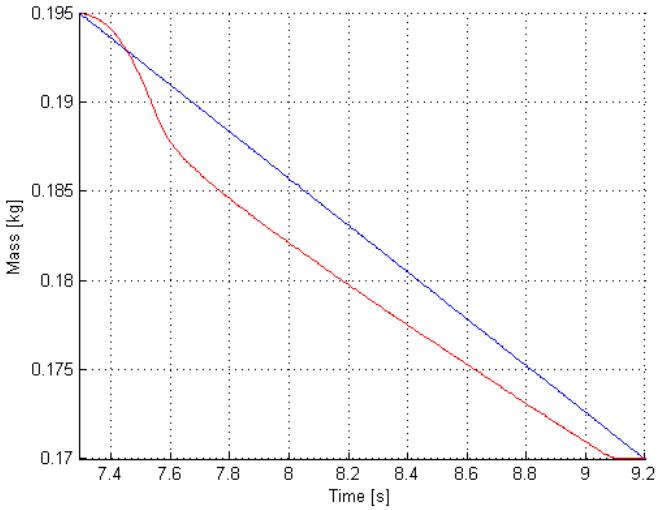

Figure 11.11: Launch #1 corrected mass function

Table 11.3: Launch #1 engine performance comparison

	Manufacturer	Launch
Max. Thrust $[N]$	29.73	23.71
Avg. Thrust $[N]$	10.21	7.60
Impulse $[Ns]$	16.84	14.51
Exhaust Speed $[m/s]$	-	580.31
Burn Time $[s]$	1.65	1.92

This iterative scheme converges in a few iterations. The result is shown in Figure 11.11. Finally, the calculated thrust curve is shown in Figure 11.12, and the performance figures in Table 11.3. The first conclusion that can be seen is that the curve has the expected shape, with a peak at the beginning and a constant period during the rest of the burn time.If we compare the obtained curve with the one given by the manufacturer, one sees that the peak is lower than expected. The same happens with the constant period. However, the burn time is longer in the obtained curve than the manufacturer curve. It is also interesting to evaluate the impulse, which is defined as,

$$I = \int_{t_o}^{t_f} F\, dt \tag{11.6}$$

This is significantly lower than the given by the manufacturer.

Whether these differences are a lack of accuracy in the obtained curve, or if the manufacturer is more optimistic in his specifications than he should is an uncertainty. In the figure, it is also indicated the curve obtained by TRT during a static test. The values also are significantly lower, and differ from the results obtained in the launch. However, since we do not know for certain the conditions in which these measures were taken, we can not take this curve as a reference.

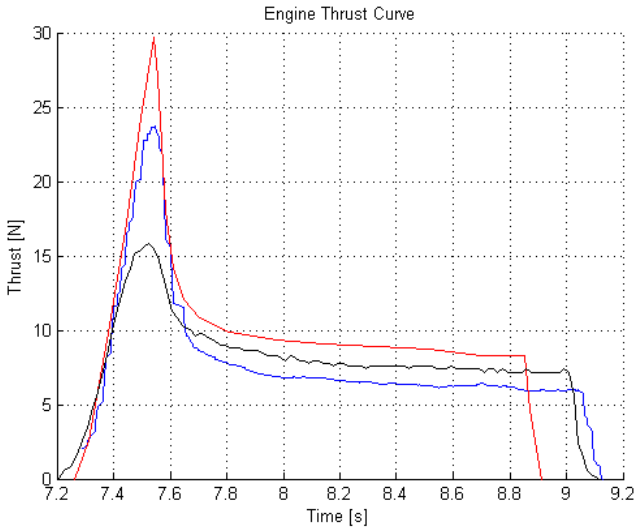

Figure 11.12: Launch #1 engine thrust (blue), manufacturer engine data (red) and static test (black).

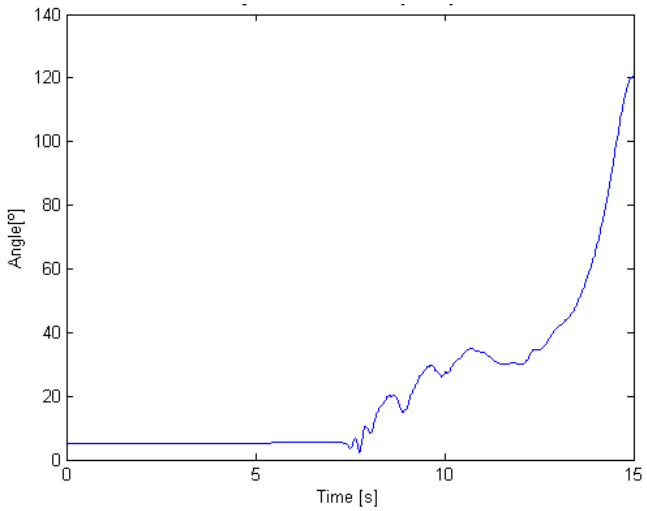

Figure 11.13: Launch #1 angle of attack during launch

11.1.7 Stability analysis

If the thrust curve and drag coefficient analysis exceeded the expectations, the analysis of the rocket stability did not result in such a great way. The attitude processing, with the fusion of the gyroscopes and the magnetometer is no as robust as expected. Despite the attitude estimation being presumably acceptable in the first second, during the rocket burn time, it becomes inaccurate in later stages. Also, the orientation estimation is very dependent on the complementary filter parameter. the angle of attack has been calculated as the angle between the z-axis in earth frame, and the longitudinal axis of the rocket, and the results are shown in Figure 11.13. If the magnetometer data is neglected, and the same calculations are done only with the gyroscope data, one can obtain the evolution on the angle of attack shown in Figure 11.14. This plot corresponds just to the immediate period right after ignition, and it can be clearly seen an oscillation that could be used to estimate some stability parameters. However, this situation is available only for a moment and it is not representative enough to consider it useful. This means it will be important in a future to develop a more robust strategy for estimating the attitude.

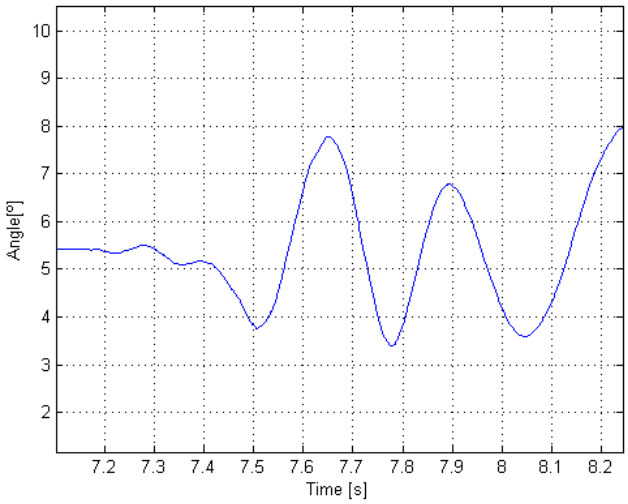

Figure 11.14: Launch #1 angle of attack right after ignition

11.2 Launch #2

Only relevant information and differences from Launch #1 will be shown, since the hypothesis and reasoning are equivalent.

11.2.1 Geometry, conditions and mass

In order to proceed with the calculations, the following data has been used. Only different data from Launch #1 has been indicated.

- Air temperature, $T = 38.19^{o}\text{C} = 303.15\,K$
- Air pressure, $p = 9.69 \cdot 10^{4}\,Pa$
- Air density, $\rho = \frac{P}{287 \cdot T} = 1.0856\,\frac{kg}{m^3}$
- Air viscosity, $\mu = \frac{2.5393 \cdot 10^{-5} \cdot T}{273.15} \cdot \frac{1}{1+122/T} = 2.078 \cdot 10^{-5}\,Pa \cdot s$

11.2.2 Raw data

First, we will take an overview on the raw data logged by the board.

- The barometer and temperature data is shown in Figure 11.15. As seen in Figure 11.15, the apogee is approximately 200 m, almost the same as in the first launch. The air temperature of the day was higher than the one in the first launch. For this reason, constants such as density, viscosity have been recalculated according to the launch conditions.

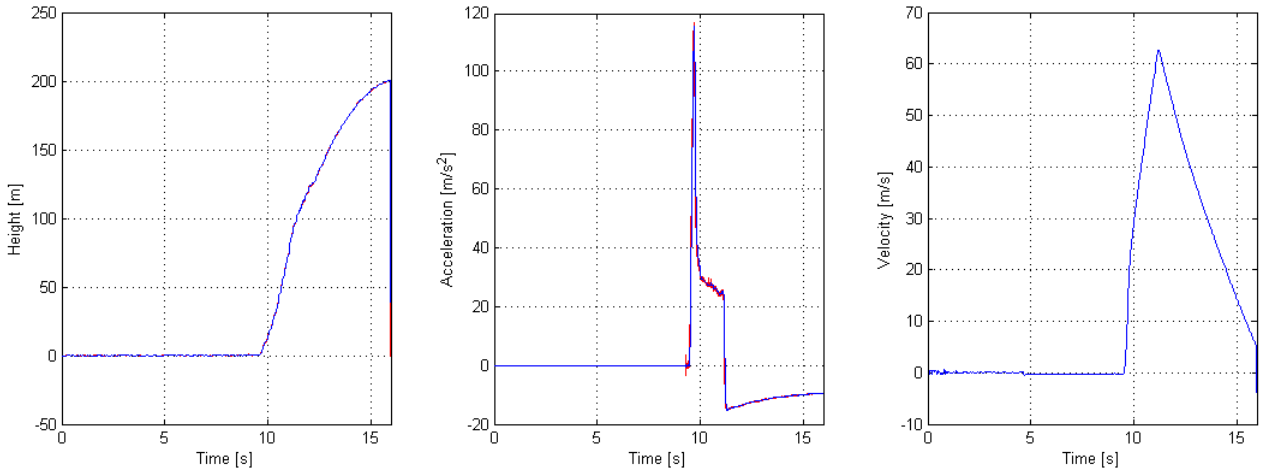

Figure 11.16: Launch #2 filtered height and acceleration, and velocity estimation (blue) and raw data (red)

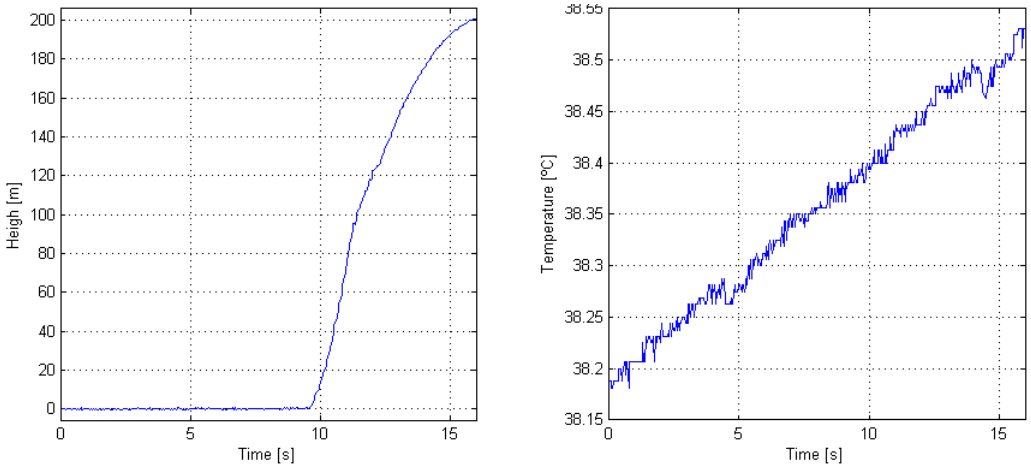

Figure 11.15: Launch #2 barometer and temperature raw data

11.2.3 Data fusion

Equivalently to the procedure done in the first launch, by fusioning data from the barometer and accelerometer, both can be filtered and the velocity can be estimated. The results are almost the same as in the first launch, considering for example peak velocity, as shown in Figure 11.16.

11.2.4 Drag estimation

Similarly to the procedure done in the first launch, the drag has been estimated considering two scenarios, constant drag and variable drag, as shown in Figure 11.17. However, the constant drag is considered the valid calculation as an assumption. The calculated drag coefficient is $C_D = 0.59$.

11.2.5 Thrust curve analysis

With the drag coefficient calculated in the section before, the thrust has been estimated as shown in Figure 11.18.

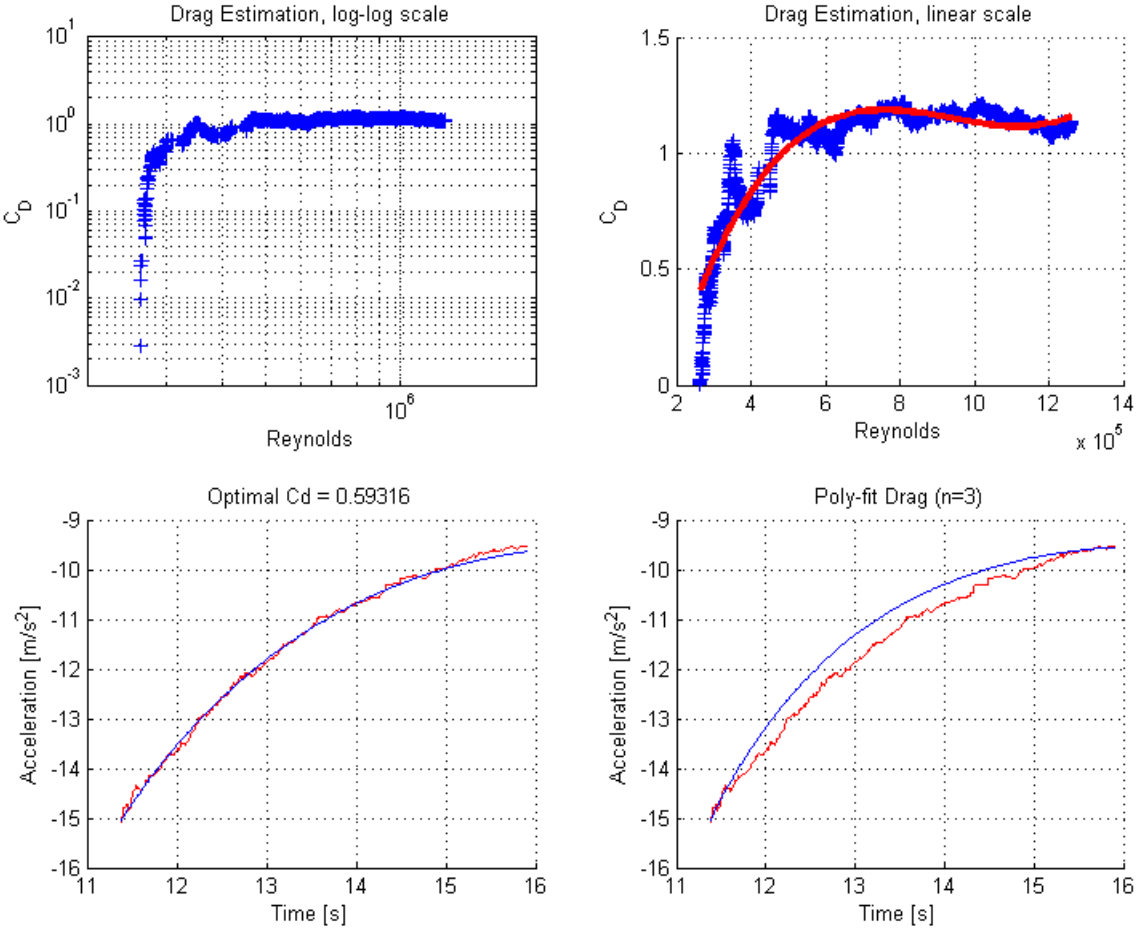

Figure 11.17: Launch #2 drag analysis

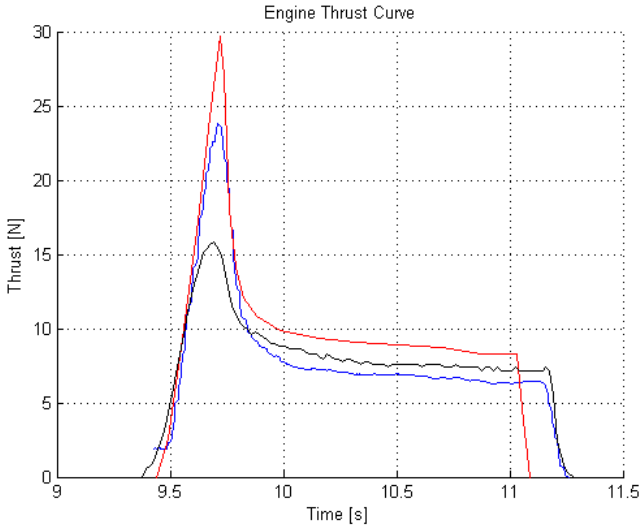

Figure 11.18: Launch #2 engine thrust (blue), manufacturer engine data (red) and static test (black).

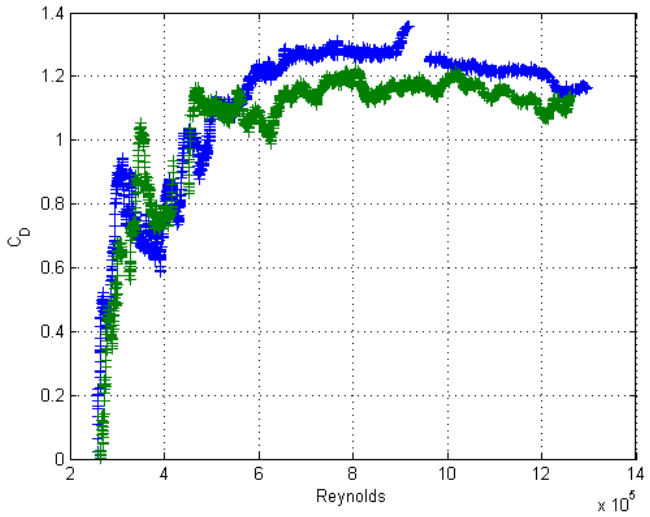

Figure 11.19: Drag coefficient comparison of launch #1 (blue) and launch #2 (green)

Table 11.4: Drag coefficient comparison

	Theoretical	Launch #1	Launch #2
Drag Coefficient	0.57	0.63	0.59

11.3 Comparison

The aim of this section is to compare the results from both launches, in order to see if results are similar. First, in Figure 11.19, the drag coefficient estimated in each point in the free fall trajectory is plotted. They have similar tendencies, however, it seems that the estimated drag coefficient from first launch is higher than the estimated in the second one. If we compare the drag coefficient obtained by the simulation method, as indicated in Table 11.4, we confirm that they are quite similar. The percent difference between them is 6.55%, and the error comparing the mean value of the measured drag coefficient with respect to the theoretical value is 7.02%. Since the theoretical drag coefficient calculated in the Open Rocket software is just a model, with its obvious differences with the real rocket, the calculations are surprisingly similar.

In second place, the thrust curve of both launches is compared, as shown in Figure 11.20. The exact data is shown in Table 11.3. The measures on the peak are very similar, with a percent difference of 0.42%. This is specially interesting since it is one of the most important parts of the curve. This is because it gives the rocket an initial acceleration that should be enough to achieve sufficient velocity so that the fins make the rocket stable. The measures on the average thrust and impulse are also very similar between launches. The biggest difference is in the burn time, with a percent difference of 4.25%. Despite this similarity, one must remember that if there was an error of calibration in the accelerometers, the thrust curve would not be accurate. The accelerometers are calibrated with an acceleration of 9.81 $\frac{m}{s^2}$, which represents approximately 13 times smaller than the peak acceleration. This is clearly on the weaknesses of the system.

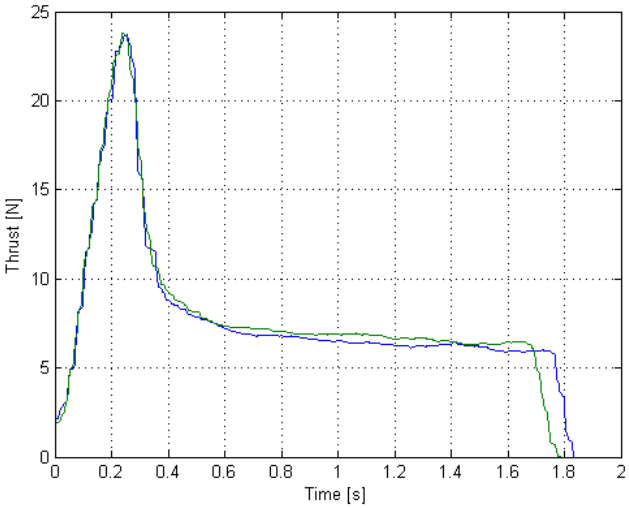

Figure 11.20: Thrust comparison of launch #1 (blue) and launch #2 (green)

Table 11.5: Engine performance and drag coefficient comparison

	Manufacturer	Launch #1	Launch #2
Max. Thrust $[N]$	29.73	23.71	23.81
Avg. Thrust $[N]$	10.21	7.60	7.86
Impulse $[Ns]$	16.84	14.51	14.67
Exhaust Speed $[m/s]$	-	580.31	586.67
Burn Time $[s]$	1.65	1.92	1.84

Part III

Project review

12 Environmental implications and safety

This technology must enable to get more profit from a rocket launch. At the moment, it is tested on a model rocket but it could be used in a sounding rocket for example. This are research vehicles which carry instruments to take measures and perform scientific experiments during its sub-orbital flight. This experiments are usually related to atmosphere analysis. One of them is shown in Figure 12.1. By analyzing the flight data, it is possible to get information about the engine, thus bringing the possibility to reduce the static tests of engines and reducing its emissions. Also, by studying the drag of the rocket it is possible to carefully optimize the rocket design, and with this maybe even having the opportunity to need for fewer fuel. The same happens with any other kind of vehicle, for example, if this system was implemented on UAV's.

During the realization of the project environmental implications has also been considered. It was not necessary to build a new rocket for the launches, and instead an existing one was used, and with this reducing the costs and used materials, the rocket is reusable. No debris were left in the launch camp and the number of engines that were used during the whole project was minimized.

Figure 12.1: Sounding rocket Black Brant XII.

12.1 Safety

This project requires handling with solid fuel engine, which can potentially be dangerous if not used properly. The engines are always stored under key which is only available for the Projects Coordinator of EUROAVIA Terrassa. During the launch, some important rules must be followed,

- Wind conditions must not endanger the launch, there must be few or not wind at all.
- Maintain a minimum safety distance from the launch of 15 m.
- It is required for the igniter to be inserted that everybody except for the operator is further than the minimum safety distance, and is not allowed to enter the safety perimeter.
- It is forbidden to connect the power supply to the launch control box until the igniter is inserted and everybody is outside the safety perimeter.

Despite the risks, if all the measures are followed it is highly unlikely of an accident to happen.

13 Planning and programming

13.1 Initial planning

In order to define the calendar, it is necessary to define the most important milestones of the project.These are indicated in the Gantt diagram in Figure 13.1. Also, there is those milestones related to the project itself, the different tasks that must be accomplished with the estimated time for each one are,

1. Hardware prototype manufacturing (1 week)
2. Electronics software programming (2 weeks). depends on 1
3. PCB design and order (3 weeks), depends on 1 and 2
4. Filtering and processing software development (6 weeks)
5. Experimental test (1 week), depends on 3
6. Data processing (2 weeks), depends on 5
7. Compilation of the results in the memory (4 weeks), depends on 6
8. Correction of the memory (4 weeks), depends on 7
9. Preparing the presentation (2 weeks).

This schedule has been arranged in a Gantt diagram, as shown in Figure 13.1.

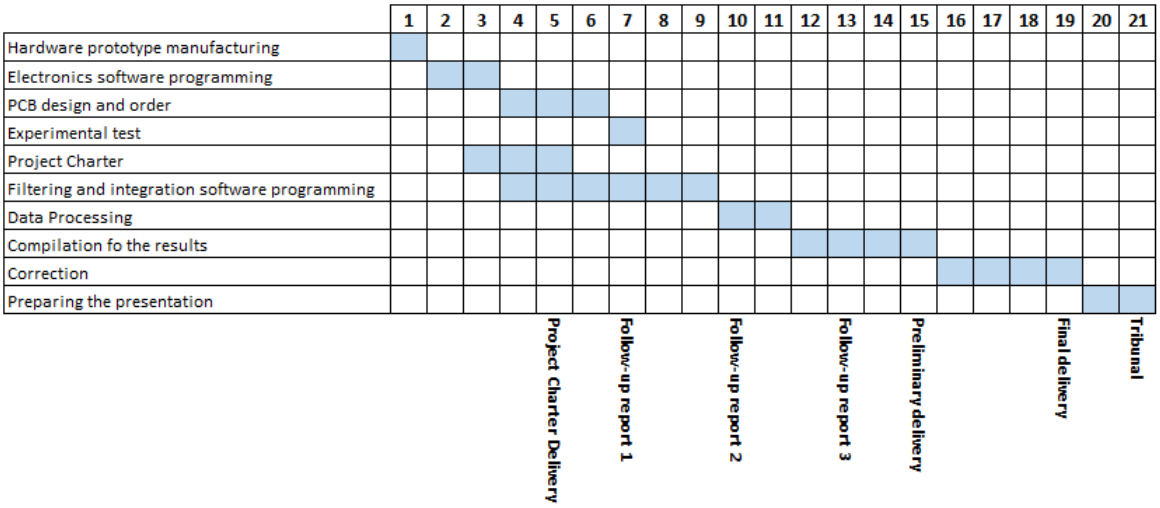

Figure 13.1: Gantt diagram of the project

13.2 Modifications

During the project there has been some problems that has modified the initial schedule of work. Fortunately, the project was designed from the beginning so that many parts of it could be done with the others not begin finished, for example, developing the electronic board, programming it or preparing the MATLAB software. The development of the electronic board, as it has been said, required three iterations, and this work took many more weeks than expected. The MATLAB software has been quite a difficult part of the project, but it was already expected to take a long time, so it did not cause major delays in the development.

13.3 Future planning

In order to be able to commercialize the board, further work would be necessary, specially considering testing and validation of the results. According to the recommendations that are indicated in next sections, it would be necessary to,

1. Set up a static test stand (2 week)
2. Test a least 10 engines (1 week)
3. Compare the results and evaluate the quality of the results obtained in the current project (1 week)
4. Improve the current project electronics and software in order to obtain similar results to the static test (2 weeks). It has been considered that it would be necessary to use static test facilities for 2 days, 16 hours.
5. Carry out at least 10 launches with the improvements(1 week)
6. Analyze the results and compare with previous data (1 week)
7. Develop a final version of the board, ready to be produced in the industry. (4 weeks)

In order to finish the project, it would be necessary to buy engines, igniters and some ready to fly rockets. Also, it is necessary to use static test facilities in the university. The cost of all this items have been indicated in the budget document.

14 Budget and economic viability

In order to complete the project, considering the pending tasks that are indicated in the conclusions, an amount of money is required. It has been considered which would be the total cost of developing the project, so that at the end of the it the board and the software could be commercialized. Next steps in the project are explained in the conclusions section. To sum up, the total budget for the project would be 23.650€. The cost for this project, which is detailed in the budget document, can be summarized as shown in Figure 14.1. It includes,

- Engineering cost
- Software licenses for one year. MATLAB licenses cost can be checked from its website.
- Production of 20 test units. The cost of each board is considered to be 25€. All the components required for the board and their price are explained in the budget document.
- Production of 20 boards.
- Testing costs. It included both the materials in necessary to proceed with the tests, engines, igniters and rockets. It has also been considered to cost 100€ per hour in the static test facilities, taking into account that there is need for an engineer to operate it.

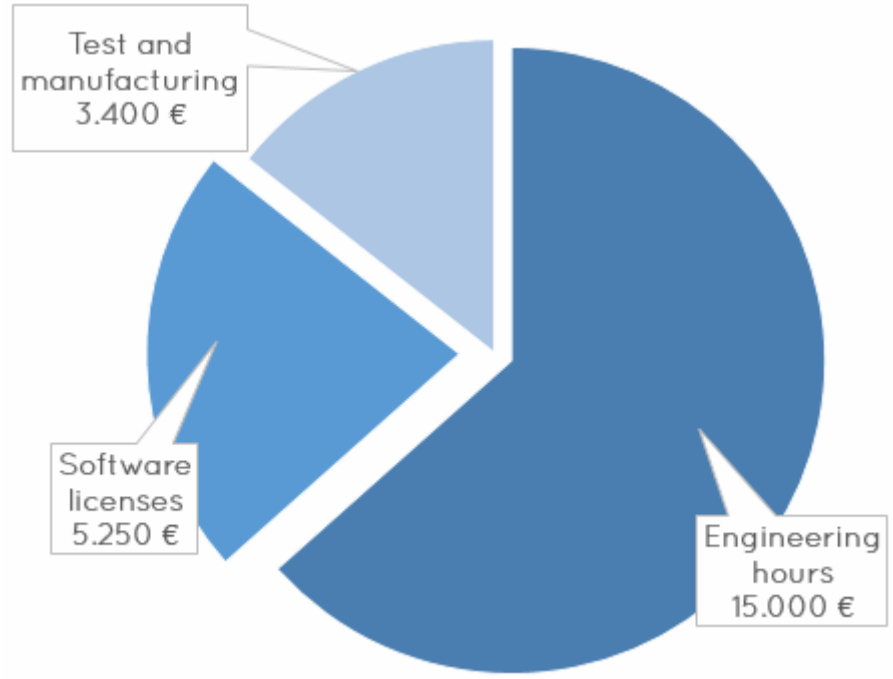

Figure 14.1: Distribution of the project's budget

15 Conclusions and recommendations

This chapter contains a summary of the research presented in this project, the conclusions drawn from the tests results, and some recommendations for future work in this field.

15.1 Summary

The work presented in this project dealt with the assessment of a low-cost INS + Barometer + Magnetometer integrated system for aerospace vehicle characterization. The chosen vehicle has been a model rocket. The motivation for using this system is obtaining a description of the performance in real flight conditions. The performance aspects that have been evaluated is drag coefficient, thrust curve of the engine and stability.

In order to reach this objective, a electronic board has been designed through three different iterations. Prototype in perfboard, and two design in PCB. Once the electronics have been available, two launches have been made, one of them successful. With all the information gathered, the data has been processed using the self-developed software. This software estimates the orientation of the vehicle by means of the gyroscopes and the magnetometer fusioned by a complementary filter. The vertical positions, velocity and acceleration is obtained by fusioning the accelerometer and barometer data by means of a Kalman filter. Finally, a more complete system for future development has been described.

15.2 Conclusions

The main objective of this project was to develop and test an INS system for low-cost aerospace vehicle characterization, and specially in the case of a model rocket. This goal has been met with the development of a hardware system and a software through which it has been possible to perform several types of simulations and tests. The following conclusions can be drawn from the results obtained,

- It is possible to get a complete description of the trajectory of a vehicle in the vertical axis, by means of low cost sensors and by using a barometer. However, only by means of an INS system is not possible to obtain a 3D trajectory.
- The case of a model rocket is acceptable to be used for the described system because of the almost vertical flight path. The nearest the analysis to the launch instant, the more accurate the results because there is fewer deviation from the vertical axis.
- Proper calibration is crucial in order to obtain valid results.
- It is possible to obtain the thrust curve of a rocket with data taken during the flight. The results seem to be reasonable, however it is uncertain if they differ from the manufacturer data because of a lack of accuracy or because the given data is not realistic.
- Results from different launches are similar.
- The drag coefficient can be assumed to be constant.
- The developed system for attitude estimation is very complex and need for a better design.
- Arduino is a comfortable and easy development board to work with, however when the program has to have demanding performance, such as high sample rate or multiple sensor logging it becomes unstable and difficult to debug.
- It is reasonably easy and cheap to develop a PCB for small sized projects like this.

15.3 Future work

The following recommendations can be made for future investigation on low cost INS systems for vehicle characterization, specially if the initial budget constraint was changed,

- Comparing the results obtained by means of the proposed strategy with measures from a static test. This would allow for validating the results obtained with the flight tests and confirm or discard the possibility of an scale error in the thrust curve due to calibration errors, for example. Also, if many tests were carried out, it would be possible to obtain the standard deviation of some engine figures such as the impulse, and therefore being able to differentiate spurious values from the usual variability of the results. This would allow for detecting board errors or unusual behaviors. Once the electronic systems would be fully validated, it would be possible to know if engine performance is dependent on the test conditions, static or flight. This would be specially interesting for the engine manufacturer in order to develop new engines.

- Use the technology in order to develop noozles for the solid fuel engines. At the moment, despite the studied engines having supersonic flow, they do not have a convergent-divergent nozzle, which reduces the engine thrust. If a noozle was designed, one would need a way to test its perfoemance on real flight conditions, which may differ from the ones in a static test.

- Develop a system to calibrate both the accelerometers and gyroscopes in the operation range.

- In order to have a three dimensional description of the flight path, including GPS in the board should be sufficient.

- Implementing the attitude estimation algorithm in a Kalman filter using quaternions instead of DCM.

- From the hardware point of view, the attitude estimation technology is the hardest challenge. It is quite difficult to have accurate information about the vehicle orientation without any external observer, camera,

References

(1979). *Guide to in-flight thrust measurement of turbojets and fan engines*. North Atlantic Treaty Organization, Advisory Group for Aerospace Research and Development. 2

Conners, T. R., Sims, R. L., and Facility., D. F. R. (1998). *Full flight envelope direct thrust measurement on a supersonic aircraft [microform] / Timothy R. Conners and Robert L. Sims*. National Aeronautics and Space Administration. 2

Covert, E. E. (1985). *Thrust and Drag: Its Prediction and Verification*. Progress in Astronautics and Aeronautics. 2

Eckert, E. R. G. and Drake, R. M. (1972). *Analysis of heat and mass transfer*. McGraw-Hill. 11.1.1

Gasior and Gardecki (2014). Estimation of altitude and vertical velocity for multirotor aerial vehicle using kalman filter. 267:377–385. 6.3.2

Grewal, M. S., Andrews, A. P., and Bartone, C. G. (2013). *Global navigation satellite systems, inertial navigation, and integration*. John Wiley & Sons. 1.2, 4

Hall, J., Knoebel, N., and McLain, T. (2008). Quaternion attitude estimation for miniature air vehicles using a multiplicative extended kalman filter. *Position, Location and Navigation Symposium, 2008 IEEE/ION*, pages 1230–1237. 2

Higgins, W. (1975). A comparison of complementary and kalman filtering. *Transactions on Aerospace and Electronic Systems, IEEE*, 1(1):321–325. 6.2.1

Hill, P. G. P. G. and Peterson, C. R. (1965). *Mechanics and thermodynamics of propulsion*. Reading, Mass. : Addison-Wesley Pub. Co. Includes bibliographical references. 11.1.6

Horn, J. (2008). *Aircraft and Rotorcraft System Identification: Engineering Methods with Flight Test Examples (M.B. Tischler et al.; 2006) [Bookshelf]*, volume 28. 2

Howard, R. M. (1997). *Dynamics of Flight: Stability and Control*. 7.1, 7.1, 7.2

Hu, X., Li, Q., He, C., and Liu, Y. (2011). An adaptive kalman filter for three dimensional attitude tracking. *VR Innovation (ISVRI), 2011 IEEE International Symposium on*, pages 151–154. 2

Isermann, R. (1989). Fundamentals digital control systems. volume 1. Springer. 2nd edition. 1.3

John S. Orme, R. L. S. (1999). Selected performance measurements of the f-15 active axisymmetric thrust-vectoring nozzle. *NASA Dryden Flight Research Center*. 2

Liu, P., Meng, Z., and Wu, Z. (2011). Identification of lateral/directional model for a {UAV} helicopter in forward flight. *Procedia Engineering*, 16(0):137 – 143. International Workshop on Automobile, Power and Energy Engineering. 2

Lou, L., Xu, X., Cao, J., Chen, Z., and Xu, Y. (2011). Sensor fusion-based attitude estimation using low-cost mems-imu for mobile robot navigation. 2:465–468. 2

Niskanen, S. (2013). Openrocket technical documentation. Technical report. 11.1.5

Salmon, R. F. (1966). Evaluation of a thrustmeter for measurring in-flight thrust of turbojet engines. Technical report, Federal Aviation Agency - National Aviation Facilities Experimental Center. 2

Savage, P. G. (2008). Computational elements for strapdown systems. In *Low Cost Navigation Sensors and Integration Technology, NATO RTO-EN-SET-116, Section 3. 10 - 34 RTO-EN-SET-116(2009.* 4.1, 6.4.2

Schlichting, H. (1960). *Boundary Layer Theory.* MCGRAW HILL PUBLISHING COMPANY, 4th edition. 7.2

Taha, Z., Tang, Y., and Yap, K. (2011). Development of an onboard system for flight data collection of a small-scale {UAV} helicopter. *Mechatronics*, 21(1):132 – 144. 2

Tailanian, M., Paternain, S., Rosa, R., and Canetti, R. (2014). Design and implementation of sensor data fusion for an autonomous quadrotor. *Instrumentation and Measurement Technology Conference (I2MTC) Proceedings, 2014 IEEE International*, pages 1431–1436. 2

Warnasch, A. and Killen, A. (2002). Low cost, high g, micro electro-mechanical systems (mems), inertial measurements unit (imu) program. pages 299–305. 2

Welch, G. and Bishop, G. (2006). An introduction to the kalman filter. Technical report, Chapel Hill, NC, USA. 6.3.1, 6.6

Weston, J. (2005). *Strapdown inertial navigation technology*, volume 20. 2 edition. 4.1, 4.1.1

Wu, W. (2014). Identification method for helicopter flight dynamics modeling with rotor degrees of freedom. *Chinese Journal of Aeronautics*, 27(6):1363 – 1372. 2

Yongliang, W., Tianmiao, W., Jianhong, L., Chaolei, W., and Chen, Z. (2008). Attitude estimation for small helicopter using extended kalman filter. *Robotics, Automation and Mechatronics, 2008 IEEE Conference on*, pages 577–581. 2

Zhang, P., Gu, J., Milios, E., and Huynh, P. (2005). Navigation with imu/gps/digital compass with unscented kalman filter. *Mechatronics and Automation, 2005 IEEE International Conference*, 3:1497–1502 Vol. 3. 2

Zwirello, L., Li, X., Zwick, T., Ascher, C., Werling, S., and Trommer, G. F. (2013). Sensor data fusion in uwb-supported inertial navigation systems for indoor navigation. *2013 IEEE International Conference on Robotics and Automation (ICRA)*, vol., no.,:3154–3159. 2