UNIVERSITY OF CAMPINAS

INSTITUTE OF PHYSICS *"Gleb Wataghin"*

SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

Rocket Attitude Control System

Design of a Gain Scheduling Approach

By

Raone Guedes Souza Cruz

Campinas
2021

UNIVERSITY OF CAMPINAS

INSTITUTE OF PHYSICS *"Gleb Wataghin"*

SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

Rocket Attitude Control System

Design of a Gain Scheduling Approach

By

Raone Guedes Souza Cruz

This Undergraduate thesis is a requirement for obtaining the Bachelor's Degree in Engineering Physics at University of Campinas.

Advisor: Professor Dr Matheus Souza

Campinas
2021

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Física Gleb Wataghin
Lucimeire de Oliveira Silva da Rocha - CRB 8/9174

C889r

Cruz, Raone Guedes Souza, 1993-
Rocket attitude control system : design of a gain scheduling approach / Raone Guedes Souza Cruz. – Campinas, SP : [s.n.], 2021.

Orientador: Matheus Souza.
Trabalho de Conclusão de Curso (graduação) – Universidade Estadual de Campinas, Instituto de Física Gleb Wataghin.

1. Foguetes aéreos. 2. Veículos espaciais - Sistemas de controle de atitude. 3. SIMULINK (Software). 4. Arduino (Controlador programável). I. Souza, Matheus,1990-. II. Universidade Estadual de Campinas. Instituto de Física Gleb Wataghin. III. Título.

Informações adicionais, complementares

Título em outro idioma: Sistema de controle de atitude de foguete: projeto de uma abordagem em programação de ganho
Palavras-chave em inglês:
Rockets (Aeronautics)
Space vehicles - Attitude control systems
Simulink
Arduino (Programmable controller)
Titulação: Engenheiro Físico
Data de entrega do trabalho definitivo: 21-07-2021

I dedicate this thesis to my mother, Ailde Rosina Guedes. This project is proof that all her investment and dedication were worth it.

ACKNOWLEDGEMENTS

I would like to thank the following people for helping with this project. To my advisor, Prof. Dr. Souza Matheus, for his help with the theoretical understanding and development of the control model used. To Alves Igor, Senior Engineer at ICT Group - Netherlands, for helping with the circuit design. To Simões Daniel, Junior Engineer at Bosch, for his help with the launches. Also to the Ecocar Unicamp Team for access to infrastructure and to Bandeirantes company for sponsorship.

“Study hard what interests you the most
in the most undisciplined, irreverent and original manner possible.”

Richard Feynmann

CRUZ, Raone. Rocket Attitude Control System: Design of a Gain Scheduling Approach. Undergraduate thesis (Degree in Engineering Physics) – Institute of Physics ”Gleb Wataghin” University of Campinas, Campinas 2021.

ABSTRACT

Reducing rocket launch costs has a huge impact on our society, mainly by reducing the cost of satellites positioning and geolocation technologies. One way to reduce this cost is to have more efficient and robust launch control systems and more low-cost studies projects that can test different control approaches and help spread the knowledge needed to understand the industry challenges. To do that, this project studies a gain scheduling approach based on LQR technique to control a vertical launch model rocket.

To validate the system before the launch, several simulations were run using Matlab, Simulink, OpenRocket and Autocad CFD. After the system was validated, the control technology was incorporated into a rocket model and field tested.

The results in the simulated tests were very promising, reaching a reduction of the attitude angle error of 99.2%. For the real launch the results were not as promising, and technical difficulties in the ignition system prevented optimal testing of the control system. However, the test platform itself was validated, having a flight as expected for the case without on-board electronics, and for the case with the electronics assembled, the collection of data demonstrated the system's control capacity.

A technique that combines LQR with gain scheduling has shown to be very promising to improve the effectiveness of rocket launching, however it is worth mentioning that this type of technique requires a minimally in-depth study of the system dynamics. For the physical launch system some improvements need to be made, the system needs to be lighter and have more power to perform flights that can actually be significant for the control tests, in addition to improving the reliability of the ignition system with 3 engines in cluster.

Keywords: Gain Scheduling. Attitude Rocket Control System. Simulink. Arduino. Rocket Physical Modeling.

CONTENTS

1 Introduction

The aerospace industry has grown rapidly due private sector investments in the last decade and an important issue that arises in this scenario is to reduce spending on launching with controlling rockets, as shown by Jones (2018), the approximate launch cost in USD for a Nasa Space Shuttle (1981) was $61,700/Kg, for Ariane 5G Rocket (1996) was $13,100/kg and more recently for SpaceX Falcon 9 (2010) $2,700/Kg, revealing a trend of a exponential cost reduction in the sector. As Petrov (1977) shown, the control system has a fundamental impact on energy efficiency due to a large expenditure of energy to put the rocket at the right angle for the mission. Bearing this in mind and as a way to participate in this technical-scientific crash, the development of a robust control system to correct the rocket's trajectory has an enormous value.

2 Methodology

The dynamics involved in a rocket launch is highly complex, validating the effectiveness of a control system only with numerical simulation cannot be ideal. Therefore, a miniature prototype of a rocket was developed to be a physical test platform. The prototype has onboard electronics to change the positioning of the canards according to closed-loop control system that uses MPU5060 ,accelerometer and gyroscope sensor, to measure how this variation changes the rocket's attack angle.

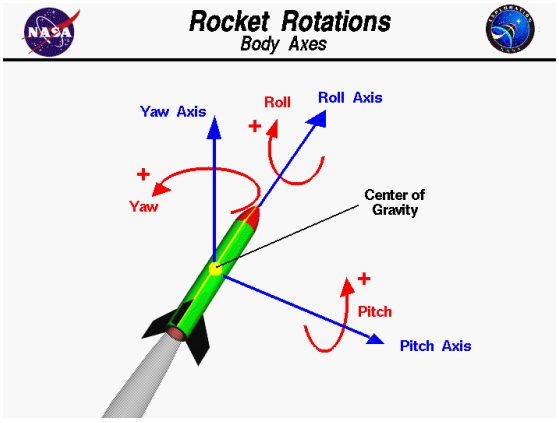

Figure 1. Nasa's definition of Attack angles.

Source: https://www.grc.nasa.gov/www/k-12/rocket/rotations.html (2021)

The angles that were analyzed was Pitch and Yaw, excluding Roll because it has little impact on the trajectory, see Figure 1. To design the control system, due the cost and risk involved in real launching, a simplified physical model that uses real values of the prototype previously measured or simulated with Autodesk CFD or OpenRocket was adopted. A dynamics simulation and system control was development using Matlab's Simulink tool.

Once the simulated results were promising, the system was taken for testing on the physical platform in the field as a way to validate the model and the control system.

3 Rocket Design Evolution

3.1 Proof of Concept

A primary proof of concept for the prototype rocket was used, together with a simple stabilizer platform shown below, in Figure 2. The Arduino-based system uses MPU5060 sensor to track the angles and 4 servo motors to try to correct the platform tilt. As a test, small disturbances were generated with pushes. With a simple PID control approach the system seemed coherent and the angle measure was realistic to indicate that electronic design was promising for the project.

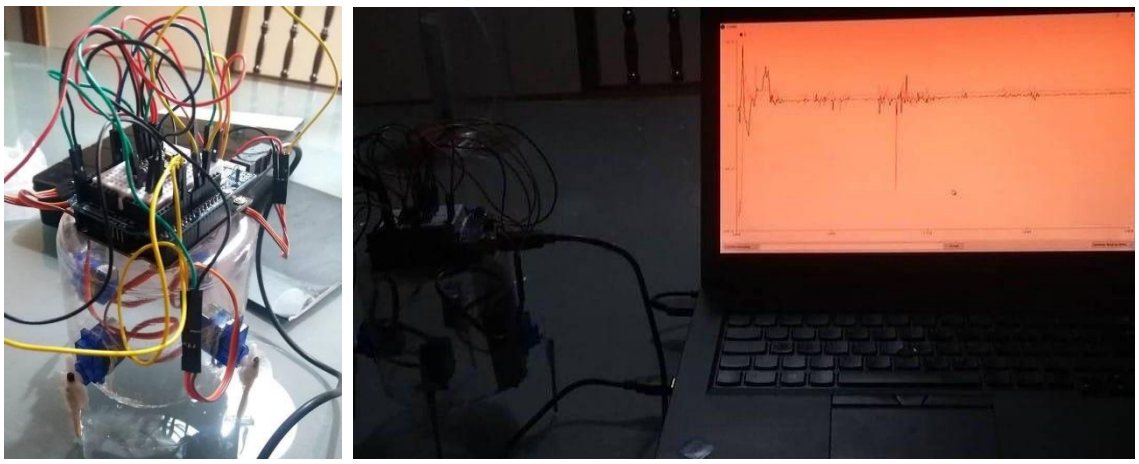

Figure 2. Stabilizer platform test.

3.2 Structural Design

The main part of the structural project is the nose cone, where the electronics and control fins were fixed. At first moment, a large diameter was designed for the nose cone as we can see in Figure 3. That is because the Arduino fixation is easier that way. As we will see in the sequel, mass is a big issue for this kind of system. So the structural design was optimized to be as lighter as possible. Our system uses an Arduino Uno like controller board called BlackBoard by Robocore. It has the same measure that original Arduino. Therefore the design was modified to have a diameter close to 53 mm.

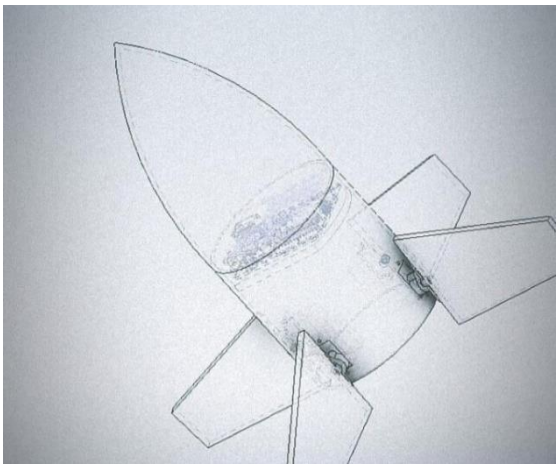

Figure 3. First nose cone design.

The first assembly, Figure 4, was made in Autodesk Inventor software. The idea was to use carbon-fiber for the structural material, because it is a widely used material in the aerospace industry, because it is light and also because of our previous experience with this material.

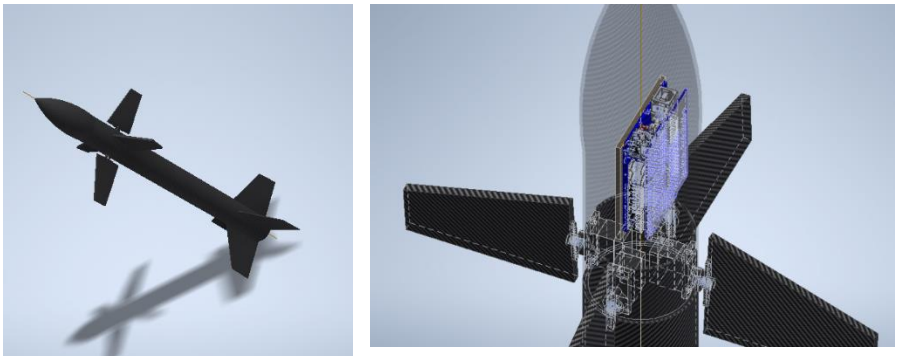

Figure 4. Carbon-Fiber based design.

As a test for the manufacturing process, the nose cone was made with carbon-fiber, Figure 5. The results were not promising, as the surface of the piece had many imperfections and weak spots, making it impossible to assemble the electronics. Moreover, the process was very expensive and laborious. So just the fins and canards, as these geometries are simpler, be kept with the material intended.

Figure 5. Nose cone carbon-fiber based result

For the body the material chosen was cardboard, more specifically a postal tube with dimensions of 60mm X 600mm, because it is cheap, accessible, easy to work with, it is light and the dimensions was compatible with whole project. For the nose Polyethylene terephthalate was chosen, as this kind of material is present in common plastic bottles that already have enough aerodynamic shape for the nose cone. Figure 6 shows the first design assembled. Some issues were identified, as the difficult to guarantee the perfect angulation of canards and onboard electronic fixation, considering the aerodynamic these elements have to be perfect symmetrical and easily aligned.

Figure 6. First version assembled system.

To solve these problems that emerged in the first assembly, 3D printing technology was employed, as it was easier to fabricate intricated and complex parts with a sufficient precision considering the equipment's limitations. Another fundamental part of the system is the engine fitting part. From our simulations we have an idea of the final mass, which is approximately 0.6 Kg. In order to use commercial engines for rocket models, we will need at less 3 engines to have thrust enough to make the rocket fly, so 3D print parts were used for the engine support too, Figure 7.

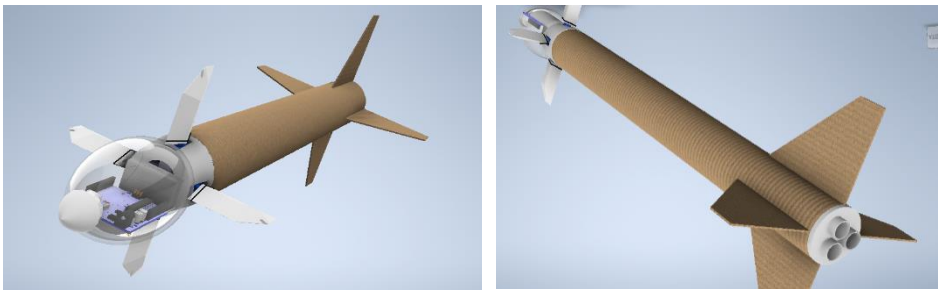

Figure 7. Assembly with 3D print parts.

3.3 Canards and Fins Design.

For the design of fins and canards, the proportions studied by ALLEN (2001) were used, where experimental data of aerodynamic coefficients are presented. As a first drawing, we considered the proportions studied for Mach = 0.6. However, for this project some changes were necessary, so to validate the adaptations were used simulations that will be shown in section 4.2. In Figure 8 we can see the final dimensions.

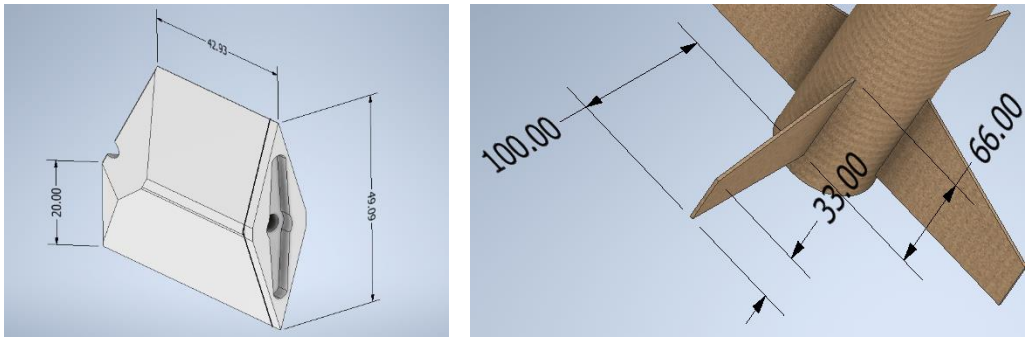

Figure 8. Canards and Fins final dimensions.

3.4 Onboard System

The core of the onboard system is a board based on Arduino called BlackBoard Uno - v1.0 Figure 9, that uses an NCP1117 regulator to supply 5V to the circuit together with the FTDI converter, instead of Arduino Uno's ATmega16U2 converter, the board achieved better performance for delivery current. With that performance the system does not need an external alimentation, or an external voltage regulator, that is normally needed to drive the servo motors, making the hole system light and simple.

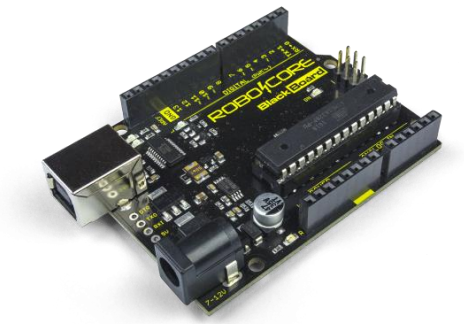

Figure 9. BlackBoard Uno - v1.0

The system needs a battery that can deliver enough peak current to drive the 4 servo motors at the same time. As the servo motor chosen was model SG90, the maximum current per motor is 250 mA. So, for 4 servo motor the battery need to delivery 4 x 250 mA, 1A of peck current. The battery chosen was a 7.4 V Li-Ion Battery, Figure 10, with maximum discharge current of 2.0 A, and total capacity of 2600 mAh or 2,6 h in operation with maximum current required, more than enough for flight tests of approximately 10 seconds.

Figure 10. 7.4 V Li-Ion Battery

The schematic with all the compounds is presented in Figure 11. The sensor used was a MPU6050, the data was collect to a Micro SD card using a Micro SDcard Module, all the compounds was assembled in a Robocore shield, Figure 12.

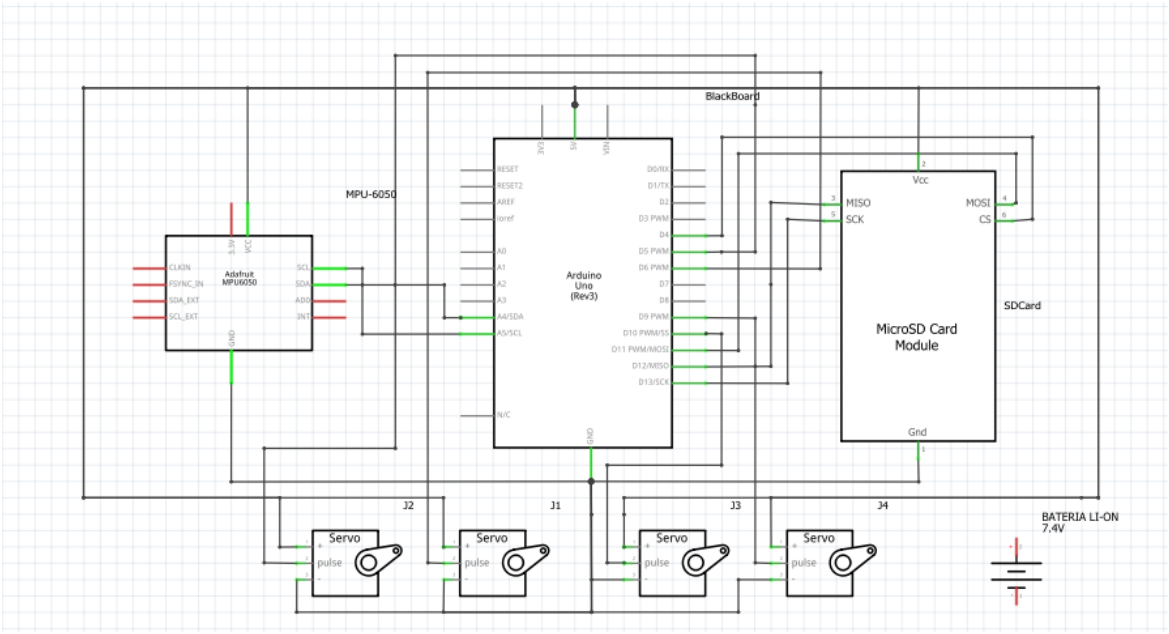

Figure 11. Schematic of the connections between the components of the circuit

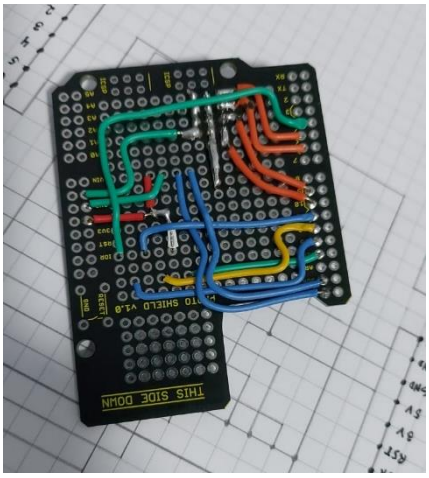

Figure 12. Routed shield for the compounds

The final onboard system assembled, Figure 13, uses the 3D print structure to position all the 4 servo motors and the sensor in the right angle for the project.

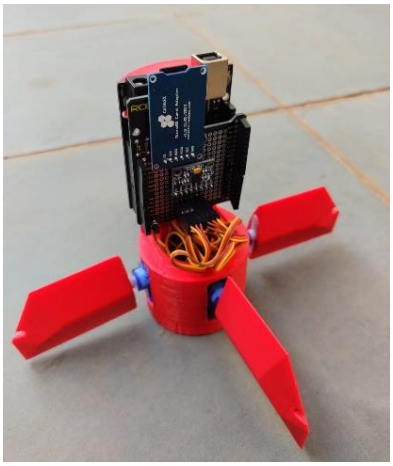

Figure 13. Final onboard system assembled

3.5 Engine Sizing Process

Once an estimate of the rocket's final mass was computed with OpenRocket software, it was possible to start sizing the engines. The two category chosen was B6-4, Figure 14, and C6-5. As Marchi, C. H. (2010) show, the first letter is thrust category, the fist number is the average thrust and the last number is the time do explode the parachute charge. The estimated mass is 0.6 kg, the B category give us approximately 12 N of maximum thrust and the C category 15 N, using 3 of them, we have for B6-4 36 N of maximum thrust with average 18 N, and for C6-5, 45 N of maximum thrust and average of 18 N. The Figure 15 shows the characteristic curve for a commercial model rocket engine. All engines were supplied by the project sponsor, Bandeirantes company.

Figure 14. B6-4 model rocket engines from BANDEIRANTES

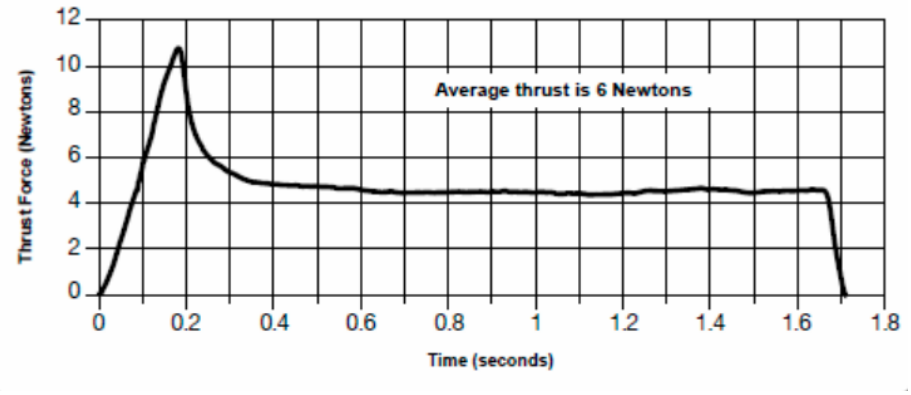

Figure 15. Thrust Force curve of a C6-5 model rocket engine.

3.6 Final Design

The final design of the rocket underwent one last change. Due to the excessive mass of the electronics, approximately 0.160 Kg, the center of mass was too far from the geometric center. To solve that, the electronics were repositioned in a new short cylinder now separate the nose cone for the control system, Figure 16. All the blueprints of the project are available in the attachments section.

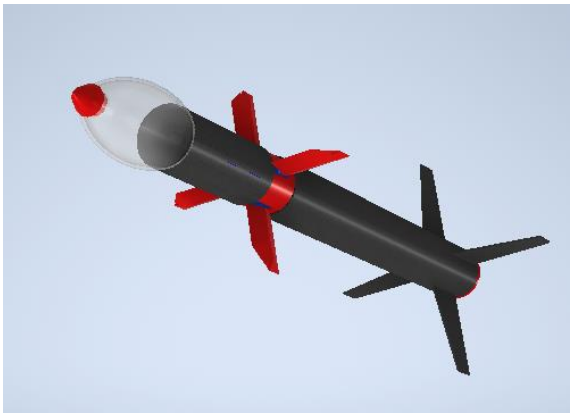

Figure 16. Final Inventor Simulated Assembly

To lauch this rocket, three more devices are necessary. First the parachute, made with a garbage bag, with 90 cm of diameter can carry approximately 332g, more than twice the mass of the onboard system. The parachute is opened when the remaining charge from the engines is activated, once airborne it uses the air drop resistance to stop the acceleration and thus reduce potential damage to the rocket's control systems.

Second the ignition system, Figure 17, that was designed to use for 9V common batteries, but as some launches failed the battery was replaced by a motorcycle battery with 12V and 1.5 Ah.

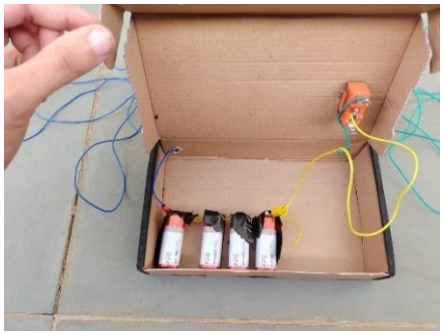

Figure 17. Ignition System

The other necessary device is the launch platform. A simple one was made using PVC tubes putting the rocket in an angle of 80 degrees. The rocket was named as Outlier, the final version with the launch platform is shown in Figure 18.

Figure 18. The Rocket Prototype named Outlier in its launch platform.

4 Physical Modeling

4.1 Differential Equations

To study the dynamics of the model rocket a physical model compatible with the launch vehicle need to be developed. Figure 29 shows a simple model of the forces

that are relevant for rocket, assuming a simple model with small angles based on Wie (2008) work, and not considering the wind velocity, as the difficult to measure this in real test.

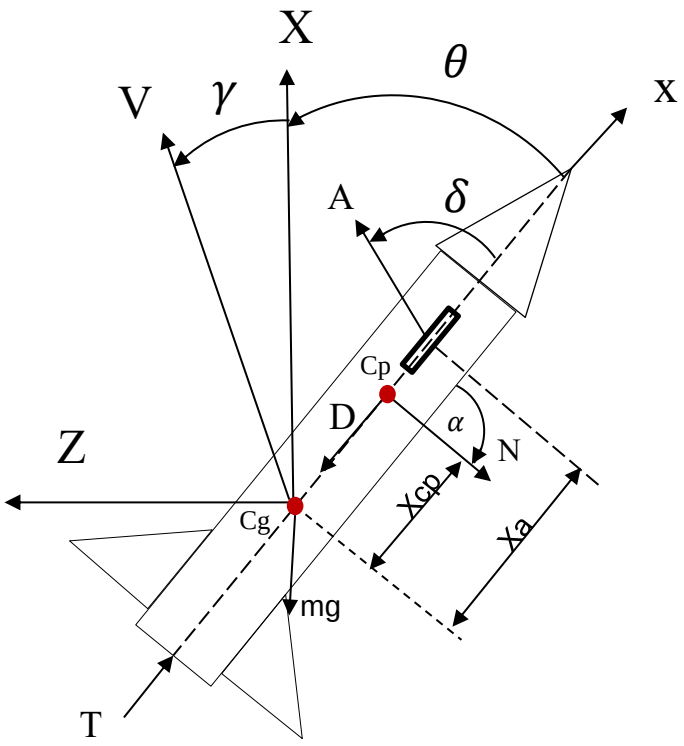

Figure 19. A one angle simplified model of a controlled canard configuration rocket.

Now equating the rocket forces and assuming that for small angles we have $sin\,\theta \approx \theta$ and $cos\,\theta \approx 1 - \frac{\theta}{2} \approx 1$ as a result of taking the first terms of the Taylor expansion about 0.

Thus for the forces, using newton's first law:

$$m\frac{dV}{dt} = (T - D) - mg \qquad (1)$$

$$m\frac{d^2Z}{dt^2} = -(T - D)\theta - N\alpha + A\delta \qquad (2)$$

$$\frac{d^2\theta}{dt^2} = M_\alpha\alpha + M_\delta\delta \qquad (3)$$

Where $\alpha = \theta + \gamma$ (4)

And $\gamma = \frac{\dot{Z}}{V}$ (5)

Where A = $\frac{1}{2}\rho V^2 A_c C_c$ and N = $\frac{1}{2}\rho V^2 A_f C_f$, V is the rocket velocity, that will change with temporal dependency, as it will become clearer in the section 4.2, m is the rocket mass 0.6 kg, g is gravitational acceleration approximately 9.81 m/s², T is average thrust, as was chosen for the system 3 C6-5 engines, approximately 18 N, D is the drag force, M_α and M_δ are the aerodynamics coefficients defined as below.

$$M_\alpha = \frac{x_{C_P}}{I_y}\frac{1}{2}\rho V^2 A_f C_f \quad (6)$$

$$and \;\; M_\delta = \frac{x_a}{I_y}\frac{1}{2}\rho V^2 A_c C_c \quad (7)$$

Where $A_f = 450x10^{-4}\, m^2$ is the superficial area, $C_f = 0.85$ is its axial drag coefficient, similarly $A_c = 30x10^{-4}\, m^2 \;\; and \; C_c = 0.33$ for the canards. I_y is longitudinal moment of inertia and assuming ρ ≈ 1,2 kg/m for air density.

4.2 Aerodynamics Simulation

To find the values shown above two simulations was necessary, first using OpenRocket, Niskanen (2013), was estimated the aerodynamics characteristics of a generic rocket with the same weight, dimensions and compounds as shown in Figure 21. With the simulation all the rocket trajectory can be observed, the program also returns very accurate estimative to Cp (pressure center), Cg (gravity center), I_y (longitudinal moment),velocity, apogees, average drag force axial and drag coefficient. To validated the simulation precision was measure in the real rocket Cg too, the simulated value is 42 cm, Figure 20, and in the real rocket the measure, finding a equilibrium spot, was 41.7 cm.

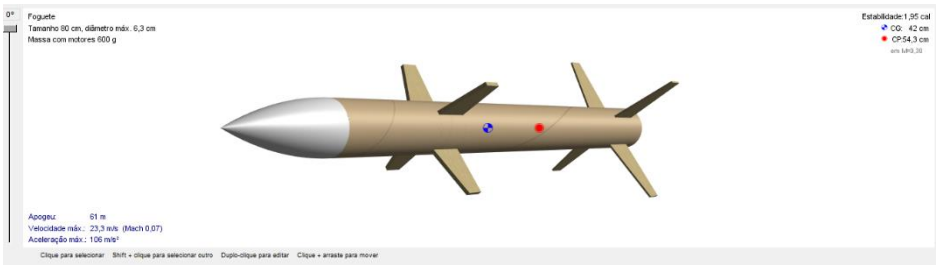

Figure 20. OpenRocket simulation Setup.

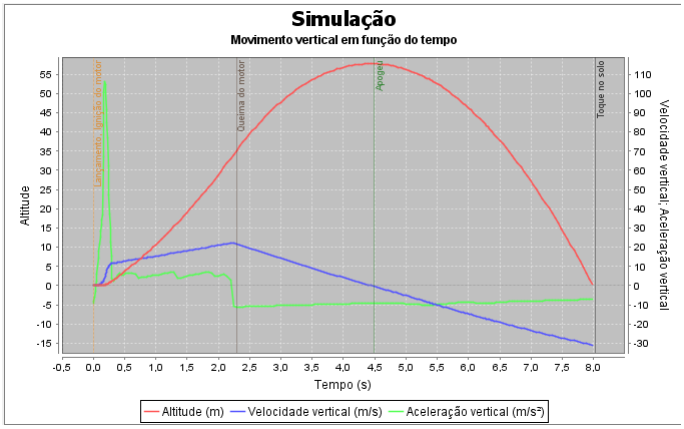

Figure 21. OpenRocket Altitude, velocity and acceleration curve .

However the core of the project it is not a common element, the canards have a unique design to assembly in the servo motors, so to estimate C_c, we used the project made in Autodesk Inventor and exported to Autodesk CFD. This software can simulate the effect of a flow of air in to the canard. To find C_c we considered a wind velocity 7 m/s, and with the force result of Ft = 0.015 N using the relation $Ft = \frac{1}{2}\rho V^2 A_c C_c$, C_c was estimated 0.33, Figure 22.

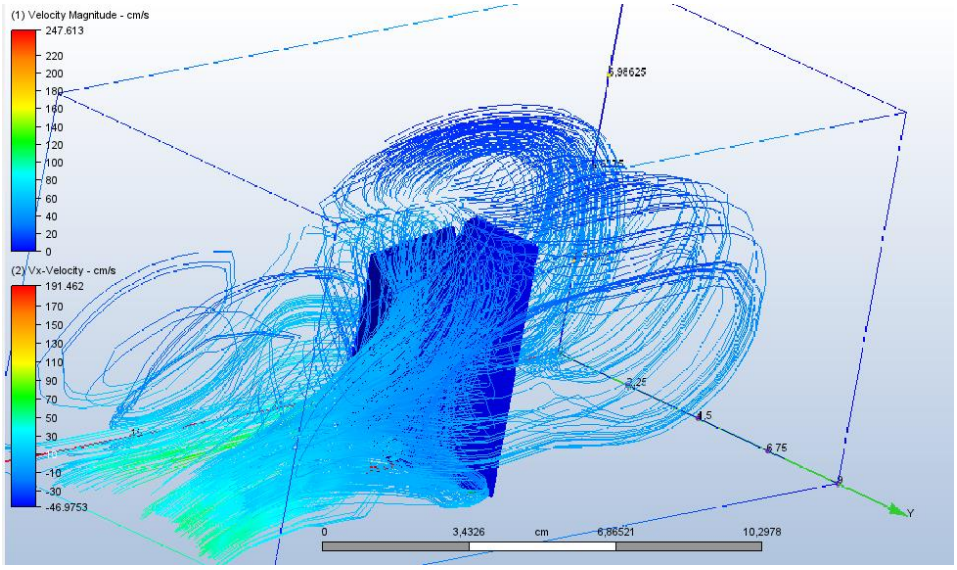

Figure 22. CFD Autodesk simulation for 7m/s wind into the canard

5 Control Model Development

5.1 Theoretical Development

To model the control systems, some aspects need to be considered. In classical control approaches, such as PID control design, the system uses only one input and one output (SISO). As the system is a multiple-input-multiple-output (MIMO) another method must be used. The Linear Quadratic Regulator (LQR) was chosen as an optimal controller for the project. It uses feedback of states and can determined the gains associated with states by solving an algebraic Riccati equation. The quadratic performance index of the regulator problem penalizes non-zero states and controls, usually yielding a state feedback control law that provides good closed-loop performance, as shown with more detailed explanation in Anderson (2007).

In general, a continuous linear system in time can be written as:

$$\dot{x} = Ax + Bu \qquad (8)$$

By LQR optimization, the functional cost is given by

$$J = \int_0^\infty (x^T Q x + u^T R u)\, du \qquad (9)$$

The feedback control law that minimizes the cost value is

$$u = -R^{-1} B^T P_x \qquad (10)$$

The Ricatti equation that needs to be solved is :

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \qquad (11)$$

The state-space model for the model rocket can be written combining equations (2), (3), and (4) as:

$$\frac{d}{dt}\begin{bmatrix}\theta\\ \dot{\theta}\\ \alpha\end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & M_\alpha \\ -g/v & 1 & -\left(\frac{\frac{1}{2}\rho V A_f C_f}{m} + \frac{\dot{V}}{V}\right)\end{bmatrix}\begin{bmatrix}\theta\\ \dot{\theta}\\ \alpha\end{bmatrix} + \begin{bmatrix} 0 \\ M_\delta \\ \frac{1}{2}\rho V A_c C_c / m\end{bmatrix}\delta \qquad (12)$$

The above equation has velocity dependency to solve, as we see in Figure 21 the velocity is not constant and just averaging does not seem very accurate. To solve this, another technique need to be used, Gain Scheduling, a common control technique for non-linear systems used when a single gain cannot provide the required performance and stability for the plant.

Basically equation (8) now is given by:

$$\dot{x} = A(V)x + B(V)u \qquad (13)$$

And we can transform that in N linearly independent systems

$$\begin{cases} \dot{x} = A(v_0)x + B(v_0)u \\ \dot{x} = A(v_1)x + B(v_1)u \\ \quad . \\ \quad . \\ \quad . \\ \dot{x} = A(v_N)x + B(v_N)u \end{cases} \quad \text{LQR treatment} \Rightarrow \quad \begin{cases} u = -k_0 x \\ u = -k_1 x \\ \quad . \\ \quad . \\ \quad . \\ u = -k_N x \end{cases} \qquad (14)$$

Where k_i are the gain for each velocity range. The range was chosen based on the simulations results in OpenRocket software previously mentioned. The intention is just to show the strategy used for a more general and more rigorous approach of gain scheduling see Khalil (2012).

5.2 MATLAB & Simulink Simulation

In Table 1, are shown the results of the MATLAB LQR solver, MathWorks (2021), for Q = [0.1 0 0] and R = 1, that was the best configuration tried. We considered velocity in range of 2.5 m/s from 0 to 25 m/s. With this information a block diagram environment was used, in Simulink, to simulate the control system behavior. Note that the K3 is very small and as α is an indirect measure i.e has a high experimental error, can be disregard its influences for the final control aproach.

i	V range [m/s]	V [m/s]	K1	K2	K3
1	0 - 2.5	1.25	0.1902	1.1734	0.0074
2	2.5 - 5	3.75	0.1998	1.2442	0.0225
3	5 - 7.5	6.25	0.2103	1.3111	0.0366
4	7.5 - 10	8.75	0.2205	1.3691	0.0488
5	10 - 12.5	11.25	0.2299	1.4165	0.0585
6	12.5 - 15	13.75	0.2383	1.4538	0.0656
7	15 - 17.5	16.25	0.2457	1.4825	0.0705
8	17.5 - 20	18.75	0.2523	1.5044	0.0735
9	20 - 22.5	21.23	0.2581	1.5209	0.075
10	22.5 - 25	23.75	0.2633	1.5334	0.0754

Table 1. LQR gain scheduling results

The general block diagram is shown in Figure 23, where the input is characteristic thrust curve of 3 C6-5 engines and a random disturbances signal and the dynamic of the rocket and reaction of the control system is the output.

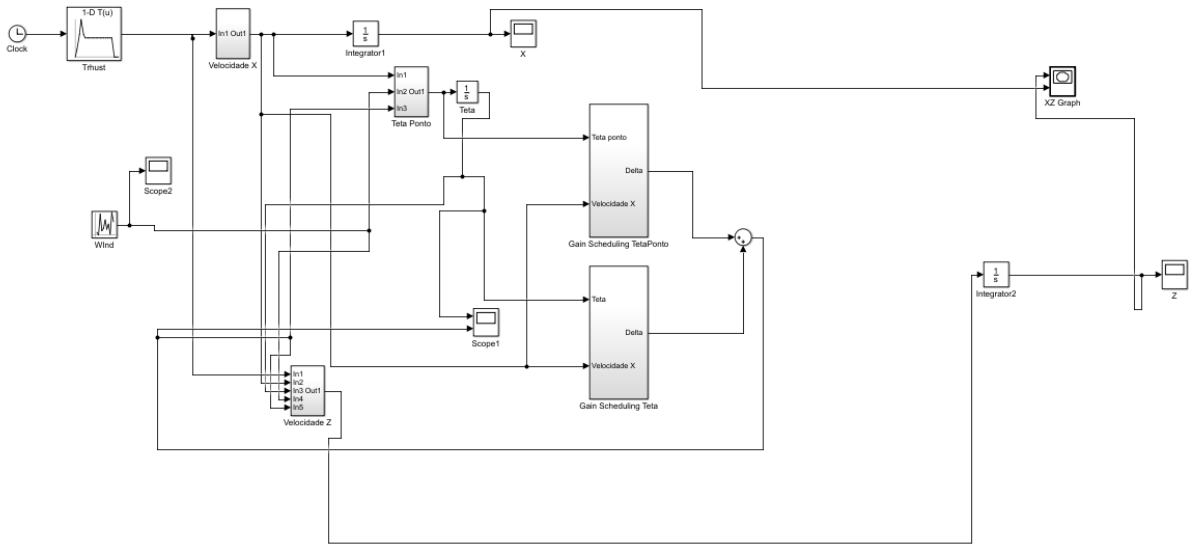

Figure 23. Simulink general block diagram

For x velocity was taken equation (1) and use constant, integrator and gain boxes to simulate the dynamic, Figure 24.

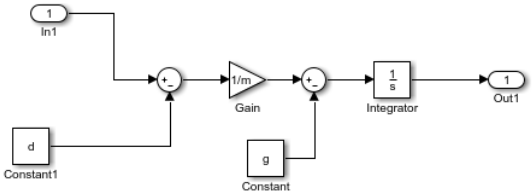

Figure 24. X Velocity subsystem block diagram

In Figure 25, is shown the block diagram of $\dot{\theta}$, using equation (3), (6) and (7), where $Aa = \frac{1}{2}\rho A_f C_f$, $Ad = \frac{1}{2}\rho A_c C_c$, $Ka = \frac{x_{C_P}}{I_y}$ and $Kd = \frac{x_{C_P}}{I_y}$, Figure 26 shown the z velocity behavior translated in a block diagram too.

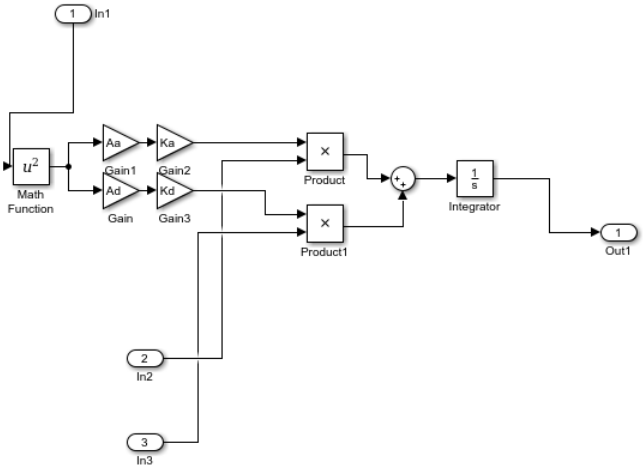

Figure 25. $\dot{\theta}$ subsystem block diagram

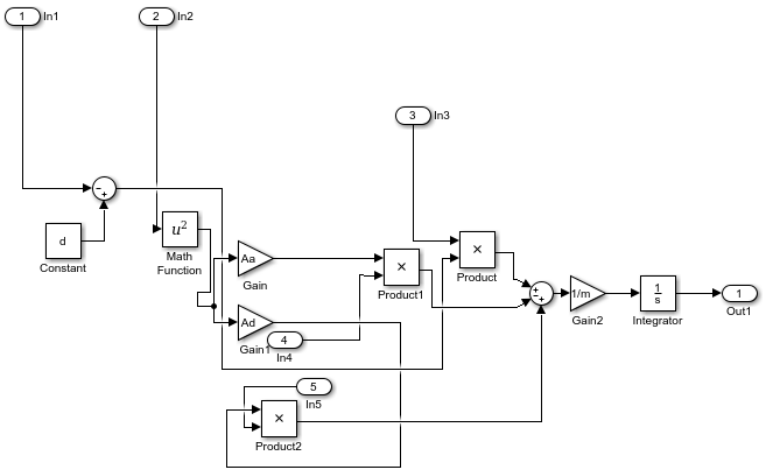

Figure 26. Z velocity subsystem block diagram

The gain scheduling block is shown in Figure 27, the $\dot{\theta}$ and θ input control blocks are analog based on switching K constants tracking the X velocity.

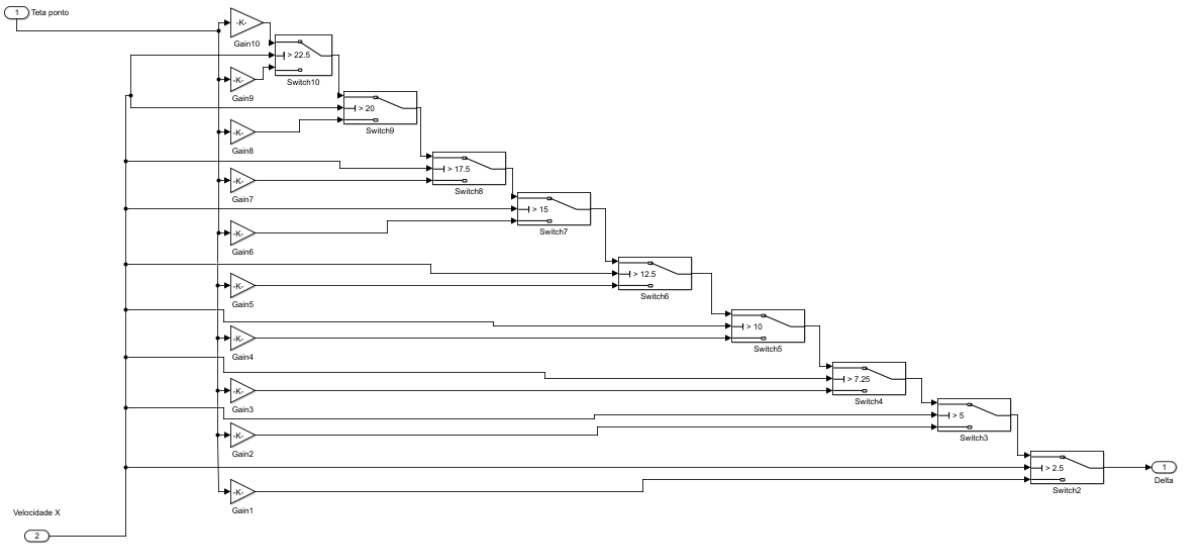

Figure 27. Gain scheduling control implementation subsystem

6 Results and Conclusions

6.1 Simulation Results

The previously commented model was simulated using a wind random disturbance between 0.5 to -0.5 radians, approximately 28º. With this disturbance the results with no control systems can be see below, Figures 28 and 29. The maximum angle in the trajectory was -2 radians, and the rocket deviated from the vertical trajectory by about 20 meters. As we can see in Figure 30, the Simulink simulations is very close to the OpenRocket simulations, Figure 21, the apogee occur in approximately 60 meters.

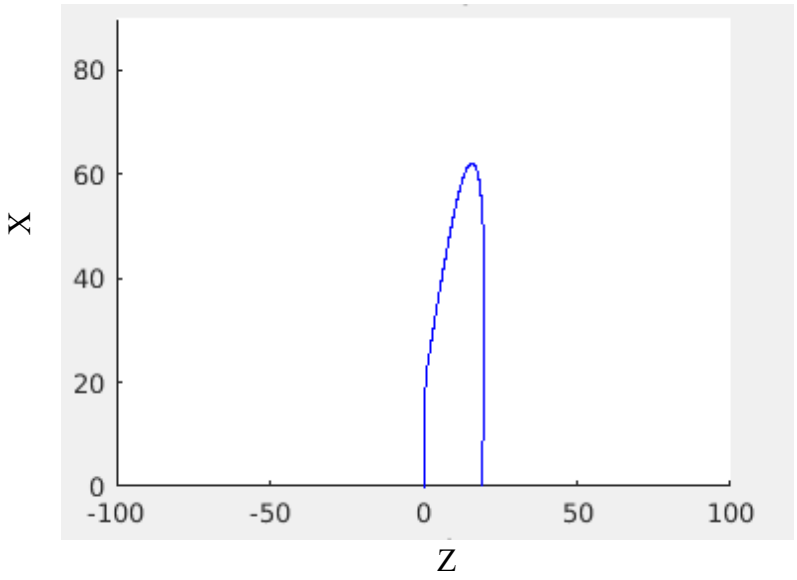

Figure 28. Rocket trajectory in Simulink simulation without control system

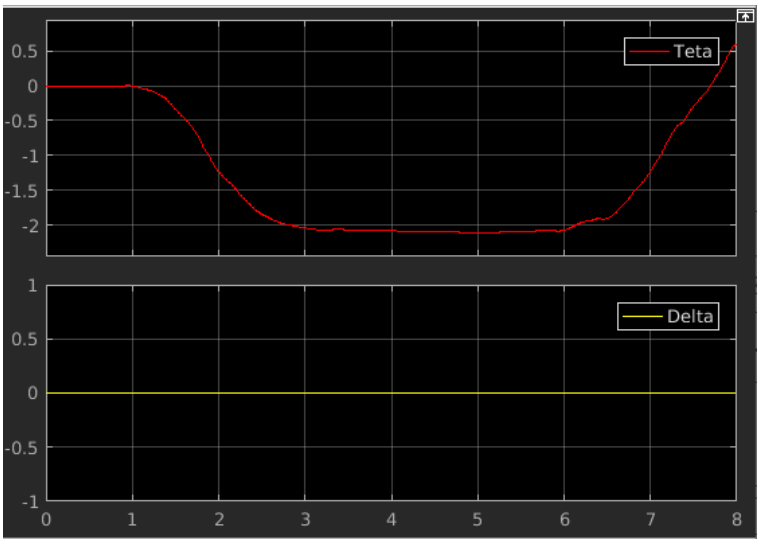

Figure 29. Rocket angle variation in Simulink simulation without control system

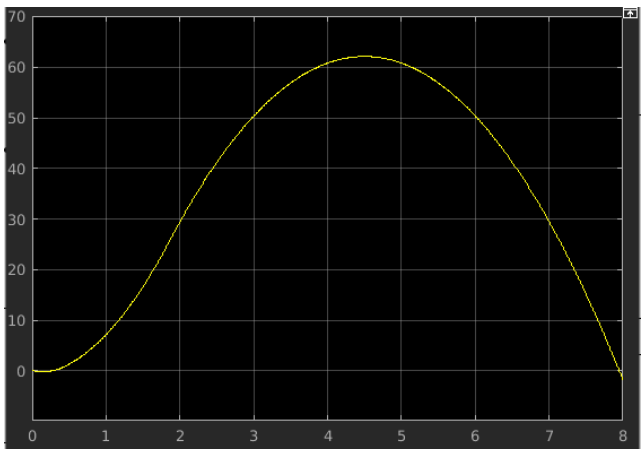

Figure 30. Rocket Simulink simulation X-axis behavior

With the system in closed-loop the results of LQR gain scheduling approach can be seen in Figure 31. The X-Z representation shows a huge improvement of the rocket trajectory , the maximum error for the vertical launching was only 1.6 meters, a 92% percent error in Z-exis improvement. For the angles the results are promising too, as shown in Figure 31 that the maximum angle is in order of 0.015 radians, or 0.57º that is an improvement about 99.2%. Due to the great results achieved, the control system is validated and can be tested in a real environment.

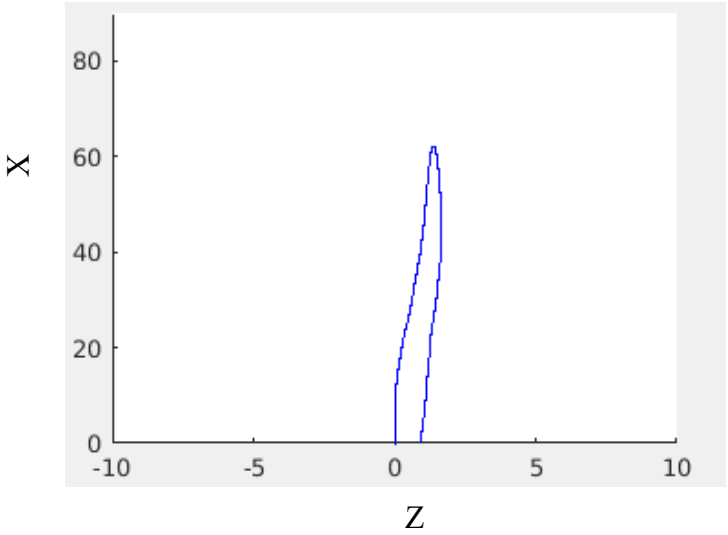

Figure 31. Rocket trajectory in Simulink simulation with control system in closed-loop

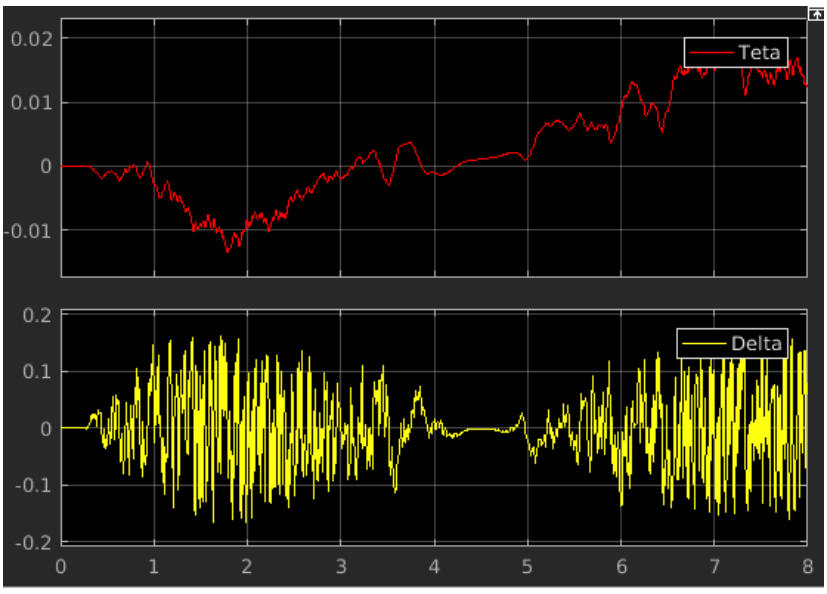

Figure 32. Rocket angle variation in Simulink simulation with control system in closed-loop

6.2 Experimental Results

First to test the system was launched the *Outlier* rocket without the onboard electronic, that because of the high risk to damage in a real launch. The ignition system and the engines had some troubles, because of the need to perform the ignition of 3 engines synchronized.

Figure 33 shows the first successful launch without the onboard electronic system. In the test all the systems work as expect. All 3 engines started, the rocket flew very well and the parachute system works in the right time protecting the nose cone. This test validated structural and aerodynamics design and the ignition system as well.

Figure 33. Rocket launch test without the onboard electronics.

After the success of the launch without electronics, the next step was test with the electronic and with the control system in closed-loop, for this the attached Arduino's C code was used. Some issues emerges of this configurations, with the electronics not only the total weight was increased but also the weight layout, the assembly was made thinking of minimizing the effect, but it was noticed a tendency of inclination much bigger comparing the assembly without the electronics.

Another important issue was the ignition system, as explained above the 3 engines cluster system presents technical difficulties in synchronizing, and unfortunately, in the tests with the electronics, this issue prevented the 3 engines from being activated at the same time, with only launches with 2/3 of the planned thrust capacity.

Even with the aforementioned problems, it was possible to collect data from the first few seconds of launch, Figure 34, where the control managed to act as planned. With this data we can validate the test platform and the potential of the control system.

Figure 35. Rocket launch test with the onboard electronics.

The final results can be seen in figures 35 and 36. Note that mostly for Yaw angle the control performed accurately, that can prove the potential of the system, even for small lack of time in the launch.

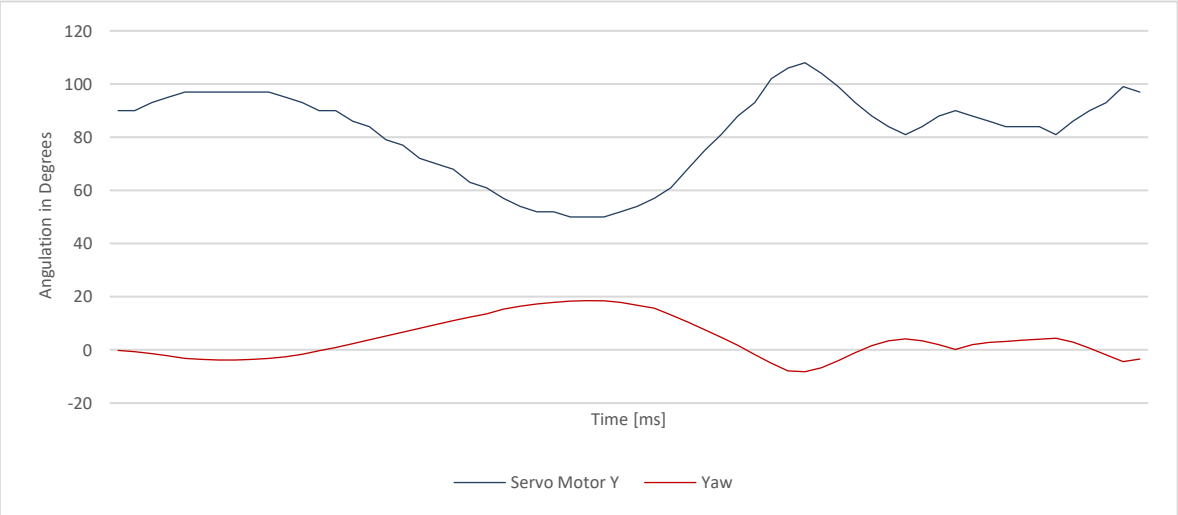

Figure 36. Yaw angle launch data for first 0.1 s.

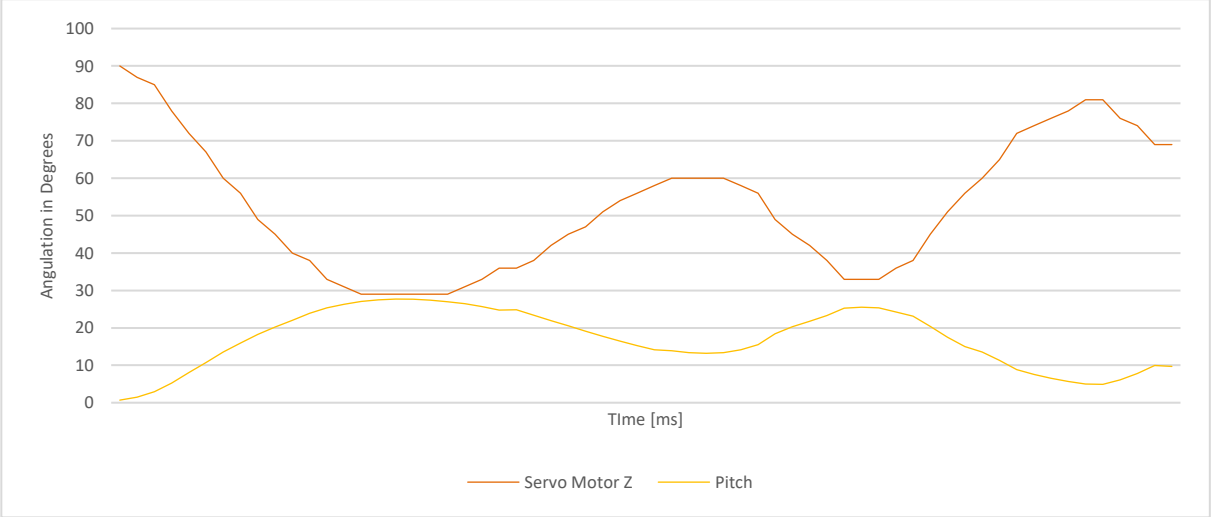

Figure 37. Pitch angle launch data for first 0.1 s.

6.3 Conclusion

This project had in its development several phases of the creating and designing process for a test platform used in a engineering control system. As the simulations and physical tests shows, the rocket design is statically and dynamically stable, the control strategy is promising and could be improved. As an initial development work, the project fulfilled the proposal, making clear the technical challenges faced when building a control system for launching rockets.

For future projects some issues need to be considered, the design needs to be lighter, using a tight control system that take up less space, like a Arduino nano or Raspberry Pi Pico based. The rocket's mass distribution needs to be better planned to prevent unwanted launch pitches. The ignition system needs to be better structured so that it does not fail for clustered engines. The engines need to have more power, this increases flight time and allows the system to act more efficiently.

REFERENCES

Petrov, B. N., Portnov-Sokolov, Y. P., & Andrienko, A. J. (1977). Control aspects of efficient rocket propulsion systems. *Acta Astronautica*, *4*(11-12), 1127-1136.

Jones, H. (2018, July). The recent large reduction in space launch cost. 48th International Conference on Environmental Systems.

Allen, J. M. (2001). Parametric fin-body and fin-plate database for a series of 12 missile fins. National Aeronautics and Space Administration, Langley Research Center.

Marchi, C. H. (2010). Testes estáticos de 27 Jul e 13 Set 2010 de motores-foguete do tipo BT de espaçomodelos. *Curitiba: Universidade Federal do Paraná*.

Wie, B., Du, W., & Whorton, M. (2008, August). Analysis and design of launch vehicle flight control systems. In *AIAA guidance, navigation and control conference and exhibit* (p. 6291).

Niskanen, S. (2013). OpenRocket technical documentation. *Development of an Open Source model rocket simulation software*, 11-13.

Anderson, B. D., & Moore, J. B. (2007). *Optimal control: linear quadratic methods*. Courier Corporation.

Khalil, H. K., & Grizzle, J. W. (2002). *Nonlinear systems* (Vol. 3). Upper Saddle River, NJ: Prentice hall.

Math Works, https://www.mathworks.com/help/control/ref/lqr.html Accessed on 06/24/2021

ATTACHMENTS

Arduino Code:

```
//used libraries

#include <Servo.h>
#include <SD.h>
#include "Wire.h"
#include <MPU6050_light.h>


//Initializing global variables
MPU6050 mpu(Wire);
Servo servoZ1;
Servo servoZ2;
Servo servoY1;
Servo servoY2;
const int chipSelect = 4;
double v_old = 0, pitch_old = 0, yaw_old = 0;


void setup() {
  Serial.begin(9600);
  Wire.begin();

  byte status = mpu.begin();
  while(status!=0){ } // stop everything if could not connect to MPU6050
  Serial.print("Initializing SD card...");

  // see if the card is present and can be initialized:
  if (!SD.begin(chipSelect)) {
    Serial.println("Card failed, or not present");
    // don't do anything more:
    while (1);

  }
  Serial.println("card initialized.");

  //Serial.println(F("Calculating offsets, do not move MPU6050"));
  delay(1000);
  mpu.calcOffsets(); // gyro and accelero
  //Serial.println("Done!\n");
  servoZ1.attach(9);
  servoZ2.attach(10);
  servoY1.attach(6);
  servoY2.attach(5);
}

void loop() {
  double k1=0,k2 = 0,  Z1, Z2, Y1, Y2, v = 0, dv = 0,  Yaw=0 , Pitch=0, YawPonto = 0, PitchPonto = 0, Delta_Yaw = 0, Delta_Pitch=0;
  mpu.update();
  Yaw = mpu.getAngleY();
  Pitch = mpu.getAngleZ();

  //calculating the derivative
  PitchPonto = (Pitch - pitch_old)*100;
  YawPonto = (Yaw - yaw_old)*100;
  pitch_old = Pitch;
  yaw_old = Yaw;
  //Calculating Velocity from Acceleration
  dv = (mpu.getAccX()* 9.81) * 0.01;
  v = v_old + dv;
  v_old = v;

  //Gain Scheduling
 if (abs(v) > 22.5){
    k1 = 0.2633;
```

```
    k2 = 1.5334;
  } else
    if (abs(v) > 20){
    k1 = 0.2581;
    k2 = 1.5209;
    }else
      if (abs(v) > 17.5){
      k1 = 0.2523;
      k2 = 1.5044;
      }else
        if(abs(v) > 15){
        k1 = 0.2457;
        k2 = 1.4825;
        }else
          if(abs(v) > 12.5){
          k1 = 0.2383;
          k2 = 1.4538;
          }else
            if(abs(v) > 10){
            k1 = 0.2299;
            k2 = 1.4165;
            }else
              if(abs(v) > 7.5){
              k1 = 0.2205;
              k2 = 1.3691;
              }else
                if (abs(v) > 5) {
                k1 = 0.2103;
                k2 = 1.3111;
                } else
                  if (abs(v) > 2.5) {
                  k1 = 0.1998;
                  k2 = 1.2442;
                  }else
                    k1 = 0.1902;
                    k2 = 1.1734;
Delta_Yaw = Yaw*(-k1) + YawPonto*(-k2);
Delta_Pitch = Pitch*(-k1)+ PitchPonto*(-k2);

//Control angle mapping and maximum values
Y2 = map(Delta_Yaw, -30, 30, 120, 60);
Y1 = map(Delta_Yaw, -30, 30, 60, 120);
Z2 = map(Delta_Pitch, -30, 30, 60, 120);
Z1 = map(Delta_Pitch, -30, 30, 120, 60);
servoZ1.write(Z1);
servoZ2.write(Z2);
servoY1.write(Y1);
servoY2.write(Y2);
//Data collect
String data = " " ;
data += String(Z1);
data += ",";
data += String(Y2);
data += ",";
data += String(Yaw);
data += ",";
data += String(Pitch);
data += ",";
data += String(v);
File dataFile = SD.open("data.txt", FILE_WRITE);
// if the file is available, write to it:
  if (dataFile) {
    dataFile.println(data);
    dataFile.close();

  }
// if the file isn't open, pop up an error:
  else {
    dataFile.close();
  }
delay(10);
}
```