UiT The Arctic University of Norway

Department of Electrical Engineering IET

Optimal attitude control and maneuver design for space debris detection in low earth orbit performed by a CubeSat

Andres Felipe Aldana Afanador

Master's thesis in Aerospace Control Engineering May-2024

Summary

Space debris has become an important topic, and several national and international space agencies are currently studying space debris detection and removal to increase the safety of space missions. UiT the Arctic University of Norway in Narvik is aiming to develop a 2U CubeSat that can perform in-situ detection of debris with the projects UNICube and QBDebris. The mission target is to detect mm-sized debris that cannot be detected from groundbased systems; detection will be done by radar technology. The main challenge of the mission relies on the small size and high velocity of the objects, for which the satellite must have the capacity for quick reaction in the pointing angle with a precise attitude controller without spending a high cost in energy.

The main goal of this work is to develop novel attitude maneuvers for debris detection. The satellite is programmed with three procedures. Debris detection scanning: The control system was designed with high relative importance on the state's performance and moderate input penalty. This allows the radar to scan while the CubeSat is pointing to a desired angle. However, due to the high speed of the objects, false positive detections exist. Validation process: The detected objects will be scanned inside the radar region, and it is possible to approximate the position in which the debris can be spotted after the first detection. The debris validation maneuver gives the satellite the ability to perform more aggressive maneuvers by lowering the penalty on the input changes. This maneuver is designed for angular trajectory tracking in which the CubeSat performs a controlled rotation that allows the radar to maintain the object inside of the detection volume for a longer period, obtaining more information regarding the debris. Charging process: This procedure is developed for solar panels to maximize energy acquisition using a high input penalty, minimizing energy costs. Achieving adequate attitude control significantly impacts the performance of the detection tasks. Furthermore, it is vital to consider the perturbations that the LEO environment can introduce to the satellite, such as drag, gravitational accelerations, and solar pressure.

A Model Predictive Controller will be implemented to obtain a cost-effective result with disturbance robustness, improve the transient response, and offset error. Additionally, a nonlinear cascade complementary filter is executed for attitude determination, performed with a gyroscope, sun-sensors, and 3-axis magnetometers. Simulations will be carried out to evaluate the performance of the control module, including reaction wheels and magnetorquers based on commercial models, and a comparison with different control strategies.

Preface

This master thesis has been developed in the Aerospace Control department as a contribution to the Arctic University of Norway UiT student satellite project with the goal of detecting millimetric size space debris. It describes the Attitude Determination Control System design for three different maneuvers to be performed by the CubeSat in Low Earth Orbit. This is included in a simulator to test the performances of optimal controllers.

I would like to thank my supervisor and professor Jose Juan Corona Sanchez for the support during the project development.

Contents

List of Figures

1 Introduction

The interest for satellites has been increasing over the last years as new needs and missions are to be launched, specially in low earth orbit (LEO) context. Because of this necessity, the development and improvement of both hardware and control systems in satellites is a topic of interest. One example is the space debris detection and removal as an international interest in order to increase the safety of satellites, space stations and astronauts. Monitoring space debris inside the range in which the objects are too small to be scanned from earth and too big to be a risk for astronauts, satellites and spaceships is one of the biggest concerns nowadays in space technology development. Two projects that are willing to provide a solution for this are UNICube and QBDebris developed by UiT the Arctic University of Norway in Narvik by students and professors to be applied in 2U and 3U cubesats respectively for debris detection in LEO. Another examples in the Norwegian context are the maritime surveillance and communications.

The mission target is to track the number of orbital debris that cannot be detected from ground-based systems, meaning objects in millimetric size. Consists of a 2U CubeSat (20cm sides) equipped with a small radar, orbiting around 450Km altitude. The satellite is equipped with three maneuvers. First, the debris detection pointing control system designed with a high relative importance of the states and a moderate input penalty, allowing the radar to scan while the satellite is pointing to a desired angle. However, due to the high speed of the objects and the possibility of false positive detections, it is necessary to include a validation process. Since the detected objects will be scanned inside the radar region, it is possible to approximate the position in which the debris can be spotted after the first detection given that the orbit period difference is less than once second relatively to the satellite. With the debris confirmation attitude control system, the cubesat can perform more aggressive maneuvers, lowering the penalty on the input changes, and is designed for angular trajectory tracking in which the satellite performs a controlled rotation that allows the radar to scan the object for a longer period, obtaining more information regarding the debris. Finally, thanks to the near sun-synchronous polar orbit, the third procedure is developed for the solar panel charging process with a high input penalty, minimizing the energy cost.

The main goal with the development of this Master Thesis is to design an attitude controller, taking into account the actuators (inertial wheel and magnetorques), the missions to be performed, the hardware and energy limitations, and costs. Moreover, it is important to consider the possible perturbations that the low orbit can provide to the satellite as such as drag, density variations, gravitational accelerations, solar pressure, among others.

Consequently, it is a challenging project for which is important to check the state of the art control techniques that are being implemented in satellites that try to provide solutions and achieve better performances for satellites on duty.

1.1 Literature review

The literature review is divided in two parts. First, it will be exposed the state of the art control techniques in a general context and afterwards, it will be presented a brief description of some of the projects that have been developed in the context of CubeSats control.

1.1.1 State of the Art Control Techniques

Process control is an essential field with challenging research problems an applications which motivates to the continuous work on the development of new methods based on hardware, computing strategies and optimization techniques [1] as can be seen in figure 1

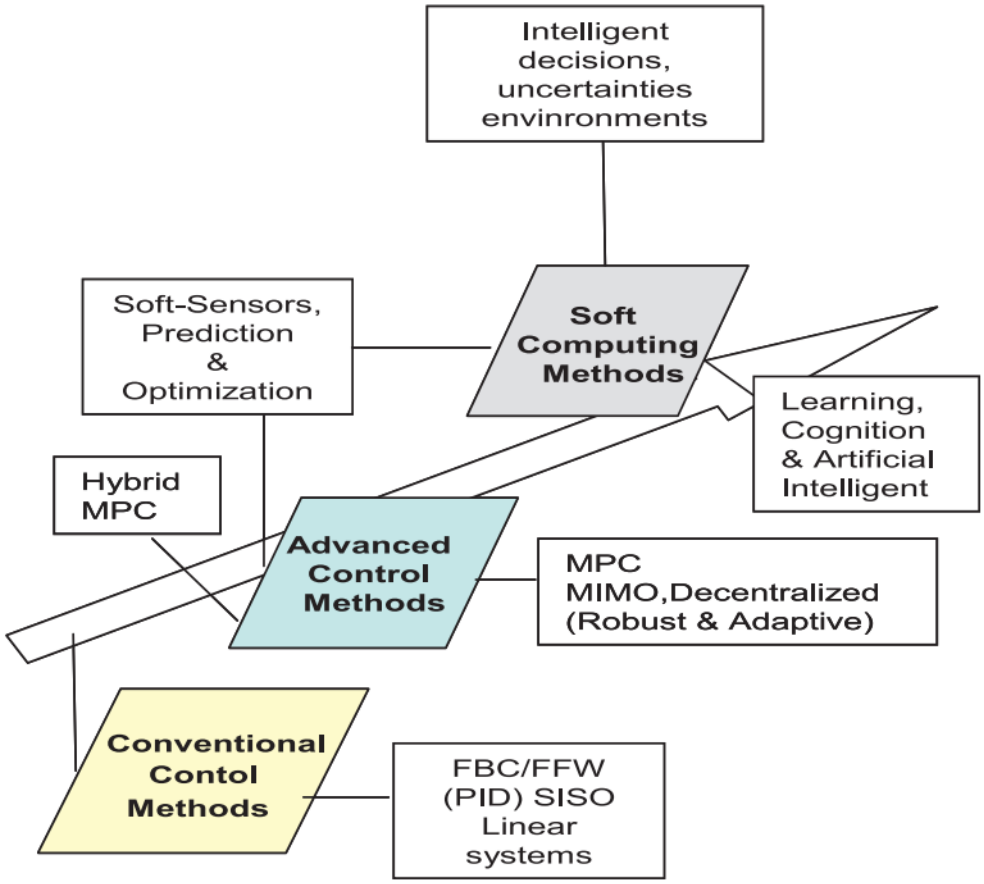

Figure 1: Trends in development of control methods [1].

Nowadays, controllers can be classified from conventional strategies which can be manually or automatically tuned, as such as PID, to more advanced control strategies that vary around techniques that are commercially available for industry as it is also found strategies that are being developed through soft computing methods (SCM) [1]. This can be observed in the figure 2

Some of this strategies are to be mentioned in the following sections. It is relevant to mention that the more advance the technique is, the more costly tends to be the implementation since it can carry larger computational requirements than the conventional controllers.

1.1.2 Conventional controllers based techniques

This section consists of the controllers from category A in figure 2, which are mostly PID based controllers. First project works with a 3U CubeSat and implements a PID controller for sun pointing mode and Nadir pointing mode, together with a B-dot controller for the detumbling mode and generates a control action with a torquer [6].

Furthermore, there is a CubeSat control design and implementation using Python to program the PID controller. The satellite is 1U size and is intended for low orbit performance. It works with reaction wheels and the results were achieved including also a simulation with a perturbance applied to the satellite [7].

A. Conventional PID Control	B. Advanced Control I	C. Advanced Control II	D. Advanced Control III
Manual Control	Adaptive and Selftuning Control	Optimal Control Methods (LQ and LQG)	Hybrid Predictive Control
Feedback Control (FB)	Gain Scheduling Method	Robust Control Methods (H_2, Hinf, IMC)	Fuzzy Control (PID, MPC FPGA)
Cascade Control (CC)	Multivariable Control Methods (State Space and Transfer Functions Models)	Model Predictive Control (MPC-DMC, MPC-GPC)	Neural Network Control (Optimal, MPC, FPGA)
Feedforward Control (FFW)	Multivariable Control Methods (Decoupling and Decentralized Control)	Decentralized Control (Time domain, Frequency domain)	Discrete Events Control (Hybrid with Petri nets)
Ratio Control (RC)	Pole Placement Methods (SISO, MIMO)	Algebraic Control Methods (Polynomial Synthesis)	Nonlinear Hybrid Soft Computing Control
Comb. Control Structures (FB + FFW + CC)	Nonlinear Control Methods (I/O linearization)	Robust QFT Control Methods	Expert Control Methods

Figure 2: Time development control engineering methods [1].

Finally, a CubeSat actuated by reaction wheels is controlled using a modified PI-D controller that includes a Genetic algorithm optimization technique applied in order to estimate the most suitable gains. This algorithm follows the Darwin's principle of reproduction and survival of the fittest using a Simulink toolbox for its tuning. The results were satisfactory even with included disturbances and white noise [8].

1.1.3 Advance Controllers group B

In this section the controllers of the group B in figure 2 are mentioned. The first project is a CubeSat attitude control in the presence of environmental and spacecraft uncertainties. It is intended for low orbit and includes a Drag Maneuvering Device DMD [2] as can be seen in figure 3:

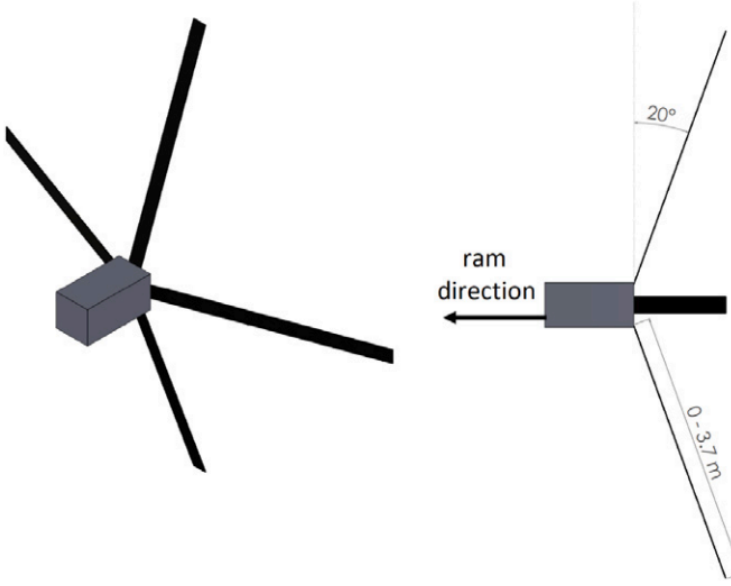

Figure 3: Drag Maneuvering Device [2]

which consist of a four repeatedly extendable/retractable surfaces offset 90 deg and inclined 20 deg with respect to the anti-ram face of the spacecraft. The control architecture is a predictive controller that aims to follow a time varying set point using only the influence of environmental torques on the spacecraft attitude dynamics. The uncertainties are the atmospheric density and

drag coefficient and the assumptions are the perfect knowledge of the inertia matrix and location of the center of gravity. The controller is evaluated with numerical simulation and provided a satisfactory performance being able to handle perturbations associated with the non-modeled behavior of the inertia matrix and compensates for the uncertainties [2].

Lastly, a Model Reference Adaptive Control strategy is developed for a 1U CubeSat actuated by magnetorquers. It is also connected through bluetooth to the Matlab system. The Model Reference Adaptive Control MRAC aims to include adjustable parameters with autoregulation and the uncertainties are purely related to the model parameters. Despite the good results of the simulation, the real implementation test presented inconveniences to achieve the given references and this is attributed to the delay caused by the hardware, leading to future work regarding testing improvements[9].

1.1.4 Advance Controllers group C

This section deals with the control architectures that are displayed on the third section C in figure 2. The first project deals with the attitude controller of an Uruguayan nano satellite 2u size CubeSat actuated by PWM commanded magnetorquers. The control law is divided in two parts. First a B-dot control law is implemented and secondly, the attitude control strategy is approached with with Model predictive Control techniques. whereas the controller results were achieved with this technique, it is said that this strategy is unsuitable for the satellite since it represents a big computational cost.[10].

Moreover, an attitude controller using model predictive and reallocation problem for a 3U CubeSat fault recovery is developed. It is developed as a simulation environment in Simulink. It includes three magnetic torquers and a single reaction wheel and it is designed to perform at low orbit. State-Space model is used and the control system finds future value predictions. It specifies that the C and D matrices are not implemented since there is a Kalman filter that handles the system outputs. The fault recovery algorithm is achieved by redistributing the commanded control torque via software by applying an unconstrained optimization problem. The results were satisfactory for magnetorquers failures and as future work it is proposed to find a solution for reaction wheels failures [11].

Furthermore, a Piece-wise affine MPC-based attitude control is implemented in a CubeSat during orbital manoeuvres in constellation deployment missions. The satellite possesses a hybrid actuator system composed of momentum wheels and a cold gas thruster. It implements a disturbance observer to obtain the real-time estimate of the eccentric torque. the Piece-wise affine model is included aiming to compensate for the strong nonlinearity, and the orbital manoeuvre thruster large eccentric torque requesting the implementation of an integrator. The controller shows positive results for the mission that was planned to be performed [12].

Another approach for a small Spacecraft is the fault tolerant attitude controller, designed using Robust Artificial Time-Delay Approach. The additions that this architecture contemplates are parametric uncertainties, surrounding disturbances, and time-varying actuator faults. For this type of control, the input–output information of the previous time instant is used to estimate the uncertain dynamical parts, integrating the time-delay approach with a feedback control technique. In addition, Lyapunov analysis is included to prove that the system states are uniformly ultimately bounded. The control architecture is shown in figure 4 [3]:

The model assumes that the uncertainties and the disturbances acting on the spacecraft are slowly time varying and bounded. The controller is tested through simulation achieving a control system able to deal with the uncertainties and also prevents the problem of unwinding.[3].

Finally, the last two projects are solved through optimal control. The first project tests two different controllers, comparing a MPC and a LQR system. It shows good performance for both of the designs, however, claims that the predictive controller has more robustness than the LQR controller[13]. Lastly, the second project designs a LQG controller by the combination of a LQR controller and a Kalman Filter. The results are positive, being able to control even after

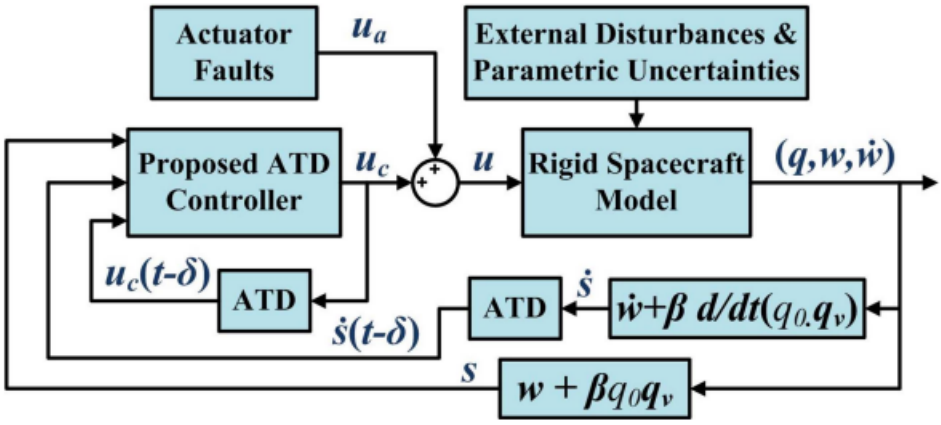

Figure 4: ATD control architecture [3].

variations of performance, actuation weightages, and the noise covariances [14].

1.1.5 Advance Controllers group D

This last section refers to the controller techniques located in the D section of 2. First, a 3U CubeSat is implemented in a space debris removal mission using the the net capturing concept. The satellite is actuated by a set of thrusters and is controlled using a Fuzzy controller. Additionally, the paper presents a simulation of failure in one of the thrusters including an algorithm for failure verification that works as feedback for the controller loop. Based on a set of rules, the fuzzy controller decides which thruster will be activated for the desired maneuver. A number of 3^6 rules is applied. The mission representation can be seen in figure 5 [4].

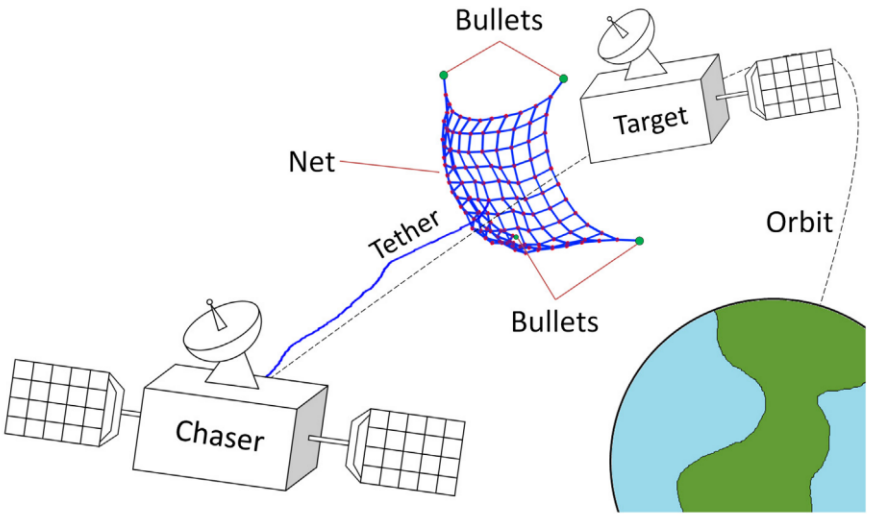

Figure 5: Mission representation for the CubeSat space debris removal[4]

Finally, the second project consists of a Neuro-fuzzy system based proportional derivative gain optimized attitude control of CubeSat under Low Earth Orbit perturbations. it is actuated by reaction wheels and considers external perturbations as such as gravity gradient torque, atmospheric drag torque, solar radiation torque, and residual magnetic dipole torque. Aiming to perform a 3 axis attitude controller, a closed-loop hybrid ANFIS-PD control methodology is implemented. It includes a neural network that trains the fuzzy membership functions using least

square and gradient descent method, which helps the fuzzy to find the desirable PD controller gains based on the perturbations. Figure 6 shows the structure of the controller. The simulation results were satisfactory and there is proposed future work, for example, the implementation of alternative fuzzifier/defuzzifier methods, real experimentation, and the implementation of Sliding Mode Control as an alternative for the PD controller[5].

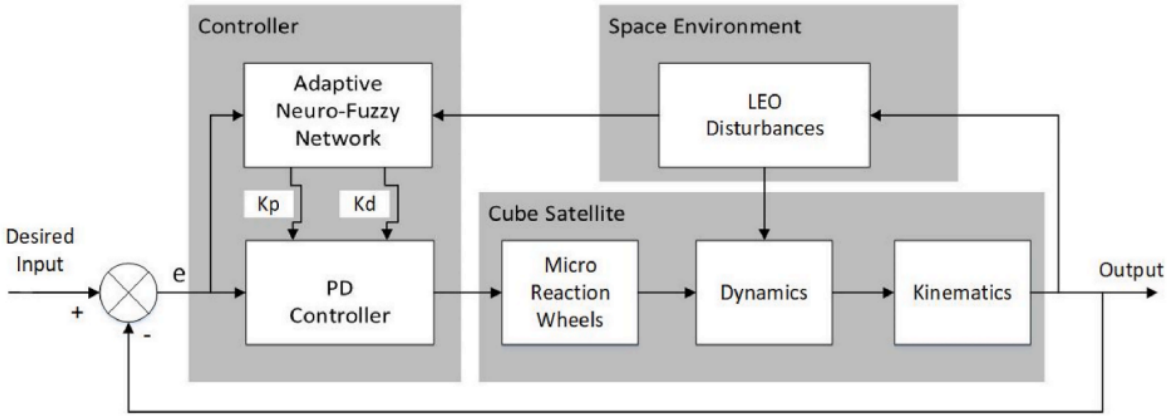

Figure 6: Control loop for the ANFIS-PD controller[5]

From the previous review it is possible to state that:

- There are several techniques that can be applied on CubeSat satellites for attitude controller with a designing that can be approached from different perspectives and levels, starting from PID approaches to more advanced model based strategies and SCM like Fuzzy controllers.
- The controller to be implemented has to provide a balance between performance, robustness and cost and should be designed based on the specific tasks and requirements that its mission demands.
- There is a large number of projects that are working in CubeSat satellites attitude control, and optimization both hardware and system oriented, as it is also targeted as an area of interest the space debris detection. These projects are being developed in relatively recent time as it can be seen in the reference section where most of the publications are from last 5 years, showing the relevance of the Thesis focus.
- Regarding real test implementations, it was possible to see that there can be improvements in the communicating protocols and other hardware choices that could lead to a more reliable testing bed experiment.

1.2 Contributions and scope of the thesis

Main research questions: How to simulate the space environment for a satellite with the properties of LEO? How to design an ADCS for the CubeSat that is able to achieve the maneuvers planned for space debris detection and equipped with reaction wheels, magnetorquers and sensors?

Main objective: Design an optimal attitude control solution for the ADC system of a 2U Cubesat supplied with three maneuvers for debris detection

Specific objectives:

- Create a simulator of the satellite in the LEO space environment that includes de ADC system, perturbations, and hardware

- Design three maneuvers that facilitate the detection of space debris taking into account that the cubesat will be equipped with a radar system
- Design the attitude determination control system interface that includes the optimal control strategies to ease the performance evaluation

In order to solve the research questions, the following project steps are chosen:

- Implement a Simulink model that includes the space environment for Low Earth Orbit
- Create a 3D model of the CubeSat to obtain more accurate physical properties to be included in the model
- Include the satellite model in the simulator
- Incorporate the sensors and actuators models in the Simulink
- Design an estimator for the attitude determination system
- Plan the maneuvers for debris detection
- Design optimal control strategies for the satellite to perform with the three maneuvers
- Simulate and compare the ADCS performances
- Create a simulator user interface that allows to select maneuver and controller

Delimitations:

- The solution must fit a 2U satellite
- The actuators are reaction wheels and magnetic torquers
- The maneuvers should be able to increase the chances of debris detection
- The ADCS must be implementable in a real prototype with realistic hardware requirements
- The controller should be commercially available and a state of the art solution
- The simulator has to include the LEO perturbations

1.3 Report outline

This thesis contributes as an ADCS design and simulation for a small satellite in LEO, implementing an optimal solution with models based on commercial hardware parameters. It also contains a detailed description of how to simulate an optimal control for a CubeSat performing three maneuvers for debris detection. It is divided into eight chapters:

- Chapter 1 – Introduction including the literature review, motivation, and research questions
- Chapter 2 – Preliminaries containing the reference frames, notation, and symbols implemented throughout the document
- Chapter 3 – Mathematical models presenting the space environment and perturbations, satellite model, the actuator models, the sensor models, and the attitude determination description
- Chapter 4 – Control design describing the detection maneuvers, optimal controllers implemented, and the hardware requirements for the solution

- Chapter 5 – Results showing the performance of the satellite ADCS during the three maneuvers
- Chapter 6 – Discussion comparing and analyzing the solution performances, their pros, cons and limitations for implementation
- Chapter 7 – Conclusion determining if the project goals were achieved with the results
- Chapter 8 – Future work presenting the steps that are to be completed in order to take the solution and implementing it in a real satellite, and improving its attributes

The appendix section includes additional information relevant to the project, containing:

- Appendix A Simulator Interface
- Appendix B Abstract for IAC 2024 congress

1.4 Tools

The simulation was carried in Matlab and Simulink 2023b version, including the model predictive control toolbox and the Aerospace blockset

2 Preliminaries

2.1 Reference Frames

Working with a spacecraft requires the use of several reference frames that help to understand the movement of the satellite around the space environment. These are:

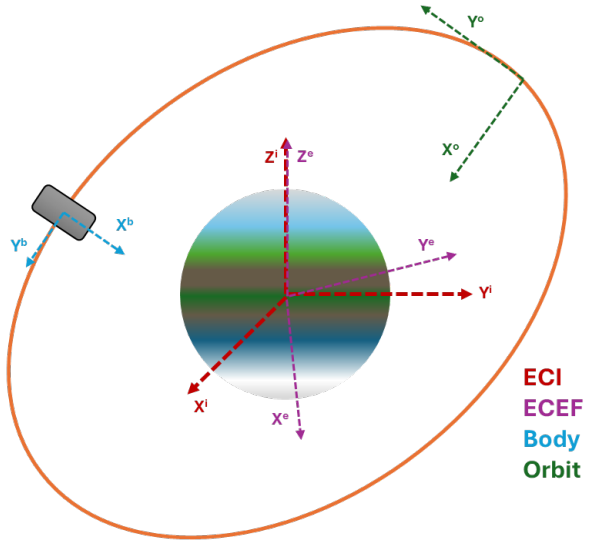

Figure 7: ECI, ECEF, Body, and Orbit frames

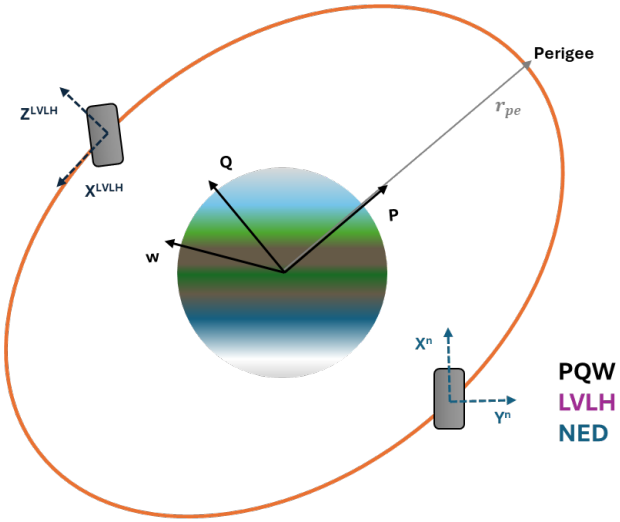

Figure 8: PQW, LVLH, and NED frames

- PQW Frame:

Where P points to the Perigee, q is the unit vector of the momentum and w a unit vector completing the right handed system. The origin is centered with the earth. Represented as: $P, Q, and\ W$

- ECI (Earth Centered Inertial) Frame:

  Where x points towards the vernal equinox, z points to the north pole and y is a unit vector completing the right handed system. The origin is centered with the earth. Represented as: x^i

- ECEF (Earth Centered Earth Fixed) Frame:

  Where x, y, and z are initially aligned with the ECI and rotates around z axis. The origin is centered with the earth. Represented as: x^e

- Orbit Frame:

  Where x is aligned with the radius vector, z is parallel to the momentum and y completes the right handed system. The origin is aligned with the orbit. Represented as: x^o

- Body Frame:

  Where x is aligned to the front of the satellite (being the sensing area), y points to one of the sides and z completes the right handed system. The origin is centered with the spacecraft. Represented as: x^b

- NED (North East Down) Frame:

  Where the x axis points north, y points to the east and z points down. The origin is centered with the spacecraft. Represented as: x^n

- LVLH (Local Vertical Local Horizontal) Frame:

  Where the x axis is the unit vector of the radius negative vector, y axis is aligned with the momentum vector and, z completes the right handed system. The origin is centered with the spacecraft. Represented as: x^{LVLH}

2.2 Notation

Additionally, it is important to state the mathematical notation that is implemented throughout the document:

- $T([]) \longrightarrow$ representing the multiplication between quaternions
- $\dot{[]} \longrightarrow$ representing the time derivative
- $S([]) \longrightarrow$ representing the skew-symmetric matrix used for cross product
- $[] \times [] \longrightarrow$ alternatively representing the cross product
- $[\boldsymbol{M}] \longrightarrow$ representing a matrix with bold letter
- $[0]_{n \times m} \longrightarrow$ representing a zeros matrix size $n \times m$
- $[I]_{n \times n} \longrightarrow$ representing the identity matrix size n
- $c([]) \longrightarrow$ representing the cosine abbreviation
- $s([]) \longrightarrow$ representing the sine abbreviation
- $INT([]) \longrightarrow$ representing the truncation function
- $[]^T \longrightarrow$ representing a transposed matrix

2.3 Subscripts / Superscripts

- $pqw \longrightarrow$ PQW frame
- $i \longrightarrow$ ECI frame
- $o \longrightarrow$ Orbit frame
- $b \longrightarrow$ Body frame
- $e \longrightarrow$ ECEF frame
- $n \longrightarrow$ NED frame
- $LVLH \longrightarrow$ LVLH frame
- $x \longrightarrow$ x axis
- $y \longrightarrow$ y axis
- $z \longrightarrow$ z axis
- $q_0 \longrightarrow$ component 0
- $q_1 \longrightarrow$ component 1
- $q_2 \longrightarrow$ component 2
- $q_3 \longrightarrow$ component 3
- $a \longrightarrow$ Actuator
- $p \longrightarrow$ Perturbation
- $ap \longrightarrow$ Apogee
- $pe \longrightarrow$ Perigee
- $n \longrightarrow$ Number of iteration
- $g_g \longrightarrow$ Gravity gradient
- $m_r \longrightarrow$ Magnetic residual field
- $rw \longrightarrow$ Reaction wheel

2.4 List of Symbols

- $t \longrightarrow$ Time
- $r \longrightarrow$ Radius vector
- $V \longrightarrow$ Velocity vector
- $a_c \longrightarrow$ Acceleration vector
- $q \longrightarrow$ Quaternion
- $\omega \longrightarrow$ Angular velocity
- $\boldsymbol{J} \longrightarrow$ Inertia matrix

- $j \longrightarrow$ Inertia component
- $\boldsymbol{R} \longrightarrow$ Rotation matrix
- $\boldsymbol{A} \longrightarrow$ A matrix ss
- $\boldsymbol{B} \longrightarrow$ B matrix ss
- $\boldsymbol{C} \longrightarrow$ C matrix ss
- $\boldsymbol{D} \longrightarrow$ D matrix ss
- $x \longrightarrow$ State ss vector
- $u \longrightarrow$ Input ss vector
- $y \longrightarrow$ Output ss vector
- $e \longrightarrow$ Eccentricity
- $n \longrightarrow$ Mean motion of the orbit
- $\mu \longrightarrow$ Product of gravitational constant and earth mass
- $T \longrightarrow$ Orbital period
- $a \longrightarrow$ Semimajor axis
- $\sigma \longrightarrow$ Augment of perigee
- $i \longrightarrow$ Inclination
- $\Omega \longrightarrow$ Right ascension of the ascending node
- $\theta \longrightarrow$ True anomaly
- $\psi \longrightarrow$ Eccentric anomaly
- $M \longrightarrow$ Mean anomaly
- $JD \longrightarrow$ Julian date
- $T_U T1 \longrightarrow$ Number of Julian centuries
- $\lambda_{M_\otimes} \longrightarrow$ Mean longitude of the sun
- $M_\otimes \longrightarrow$ Mean anomaly of the sun
- $\lambda_e cliptic \longrightarrow$ Ecliptic longitude
- $r_\otimes \longrightarrow$ Distance from earth to the sun
- $\epsilon \longrightarrow$ Obliquity of the earth
- $s \longrightarrow$ Sun vector
- $yr \longrightarrow$ Year
- $mo \longrightarrow$ Month
- $d \longrightarrow$ Day
- $ho \longrightarrow$ Hour

- $min \longrightarrow$ Minutes
- $sec \longrightarrow$ Seconds
- $l \longrightarrow$ Longitude LLA
- $\xi \longrightarrow$ Latitude LLA
- $h \longrightarrow$ Altitude LLA
- $b \longrightarrow$ Magnetic field vector
- $B \longrightarrow$ Earth's magnetic field vector
- $F \longrightarrow$ Force
- $\tau \longrightarrow$ Torque
- $P \longrightarrow$ Constant momentum flux by the sun
- $\eta \longrightarrow$ Reflexivity coefficient
- $A \longrightarrow$ Cross-sectional area of the satellite
- $m \longrightarrow$ Satellite mass
- $ecl \longrightarrow$ Eclipse factor
- $\rho \longrightarrow$ Mean atmospheric density
- $C_d \longrightarrow$ Satellite drag coefficient
- $h_\omega \longrightarrow$ Momentum
- $\tau_d \longrightarrow$ Desired torque
- $m_t \longrightarrow$ Magnetorquers magnetic field
- $\zeta \longrightarrow$ Earth's magnetic moment
- $o_b \longrightarrow$ TRIAD modeled vector
- $r_f \longrightarrow$ TRIAD sensor vector
- $T_q \longrightarrow$ TRIAD direction cosine matrix
- $\alpha \longrightarrow$ Design parameter for tuning
- J_c LQR cost function
- K_c Optimization K value of LQR controller
- s_c LQR s value for tracking
- Q_c State weights for LQR
- R_c Control signal penalty for LQR
- r_c Reference control signal
- H_c LQR Hamiltonian
- P_c LQR coestates

- k Discrete step
- A_{mpc} Augmented state-space model A
- B_{mpc} Augmented state-space model B
- C_{mpc} Augmented state-space model C
- N_{prd} Prediction horizon
- N_{ctr} Control horizon
- F_{mpc} Prediction model F matrix
- Phi_{mpc} Prediction model ϕ matrix
- Q_{mpc} States importance for MPC
- R_{mpc} Control signal penalty MPC
- Rs_mpc Reference signal matrix MPC
- H_{mpc} Hessian matrix for MPC
- f_{mpc} linear vector f for MPC
- U Incremental control inputs MPC
- Y Incremental Outputs MPC
- J_{mpc} Cost function for MPC

3 Mathematical models

This chapter describes all the elements that are included in the simulation environment that are not part of the control system, containing the space environment and perturbations models, the satellite model, the actuator models, the sensor models, and the attitude determination description.

3.1 Space models

The space models are divided into four sections. The first section shows the orbit rotations and parameters for the mission to be accomplished by the satellite. Afterwards, the sun vector model is explained followed by the magnetic field IGRF model. Finally, the four LEO perturbations included in the simulator are described.

3.1.1 Orbit

The orbit definition represents a great significance in the satellite performance, influencing not only the cubesat dynamics but also the sun and magnetic field models, perturbations, and the estimator. Furthermore, the orbit selection specifies the region in where the spacecraft will perform its mission. The circular orbit parameters established for the QBDebris and UNICube mission are:

- Apogee: 450 km
- Perigee: 450 km
- Inclination: 90 degrees
- Right ascension of the ascending node: 0 degrees
- Augment of perigee: 0 degrees

It is important to mention that for further calculations the Apogee and Perigee radius include the radius of the earth as well.

The eccentricity can be calculated as: [15]

$$e = \frac{r_{ap} - r_{pe}}{r_{ap} + r_{pe}} \tag{3.1}$$

And the mean motion is found as:

$$n = \frac{\sqrt{\mu}}{a^3} \tag{3.2}$$

Providing the orbital period as:

$$T = \frac{2\pi}{n} \tag{3.3}$$

These previous parameters are needed to calculate the true (the angle between the major axis and the position of the spacecraft) and eccentric anomalies.

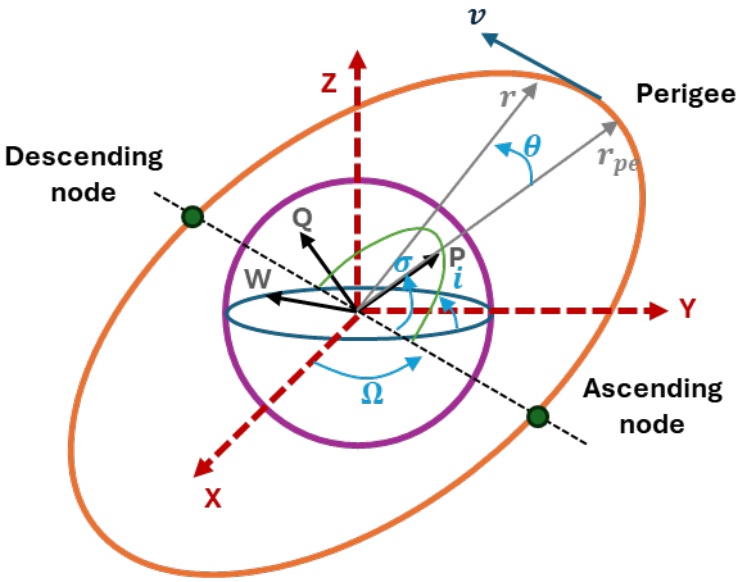

Figure 9: Orbit parameters illustration

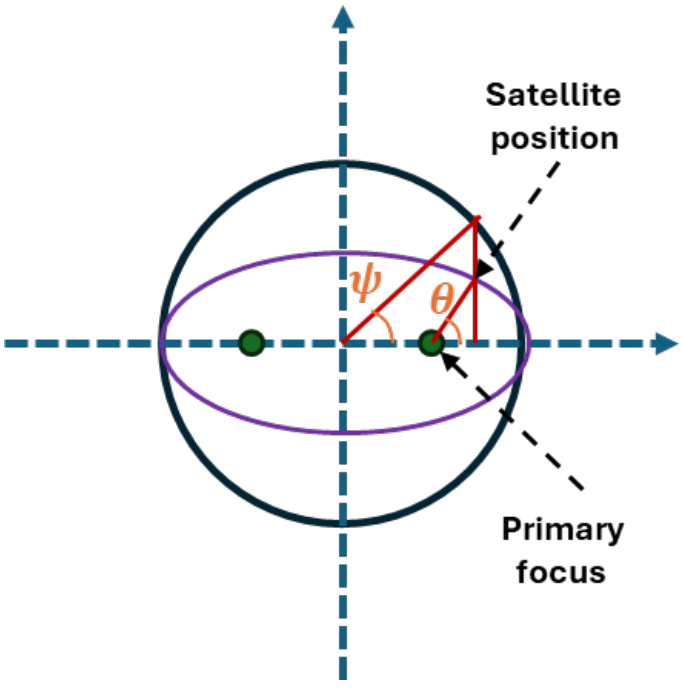

Figure 10: True anomaly and eccentric anomaly

The true anomaly is calculated as:

$$c(\theta) = \frac{c(\psi) - e}{1 - e\,c(\psi)} \tag{3.4}$$

However, because of the nature of this function the result will be restricted from -2π to 2π while the eccentric anomaly is not limited. For this purpose, the derivative of the true anomaly is founded and then integrated:

$$\dot{\theta} = \frac{n(1 + e\,c(\theta))^2}{(1 - e^2)^{\frac{3}{2}}} \tag{3.5}$$

For the eccentric anomaly, it is necessary to implement an iterative algorithm that uses the

mean anomaly as a function of time and then calculated as:

$$\psi_n = M + e\, s(\psi_{n-1}) \tag{3.6}$$

Until the error is less than a chosen factor. The semi-major axis is obtained as:

$$a = \frac{r_{ap} + r_{pe}}{2} \tag{3.7}$$

And is later used to calculate the satellite distance vector in pqw frame.

$$r^{pqw} = [a\, c(\psi) - a\, e, \quad a\, s(\psi)\sqrt{1-e^2}, \quad 0]^T \tag{3.8}$$

$$v^{pqw} = [-\frac{n\, a^2}{r} s(\psi), \quad \frac{n\, a^2}{r}\sqrt{1-e^2} c(\psi), \quad 0]^T \tag{3.9}$$

$$a_c^{pqw} = [-\frac{n^2\, a^3}{r^2} c(\psi), \quad -\frac{n^2\, a^3}{r^2}\sqrt{1-e^2} s(\psi), \quad 0]^T \tag{3.10}$$

Being r radius, v velocity and a_c acceleration. However they need to be rotated to ECI frame with the following rotation matrix:

$$\boldsymbol{R_i^{pqw}} = \begin{bmatrix} c(\sigma)c(\Omega) - c(i)s(\sigma)s(\Omega) & c(\sigma)s(\Omega) + s(\sigma)c(i)c(\Omega) & s(\sigma)s(i) \\ -s(\sigma)c(\Omega) - c(i)s(\Omega)c(\sigma) & -s(\sigma)s(\Omega) + c(\sigma)c(i)c(\Omega) & c(\sigma)s(i) \\ s(i)s(\Omega) & -s(i)c(\Omega) & c(i) \end{bmatrix} \tag{3.11}$$

On the other hand, the angular velocity of the orbit is found as:

$$\omega_{io}^i = \frac{r^i \times V^i}{(r^i)^T r^i} \tag{3.12}$$

and produces a constant output. Finally, the rotation matrix from ECI to orbit is calculated as:

$$\boldsymbol{R_i^o} = \begin{bmatrix} c(\sigma+\theta)c(\Omega) - c(i)s(\sigma+\theta)s(\Omega) & c(\sigma+\theta)s(\Omega) + s(\sigma+\theta)c(i)c(\Omega) & s(\sigma+\theta)s(i) \\ -s(\sigma+\theta)c(\Omega) - c(i)s(\Omega)c(\sigma+\theta) & -s(\sigma+\theta)s(\Omega) + c(\sigma+\theta)c(i)c(\Omega) & c(\sigma+\theta)s(i) \\ s(i)s(\Omega) & -s(i)c(\Omega) & c(i) \end{bmatrix} \tag{3.13}$$

3.1.2 Sun model

The sun vector is an important part of the simulation since it is used for perturbations, sensors, and the estimator. The model implemented in the space environment model is the one developed by Vallado [16] since it provides high accuracy. It only uses the Julian date to estimate the sun position vector in the ECI frame and then it is rotated to orbit frame. The following steps are implemented to get the result:

- Find the number of Julian centuries from the epoch $J2000.0$

$$T_{UT1} = \frac{JD - 2451545.0}{36525} \tag{3.14}$$

- Calculate the mean longitude of the Sun

$$\lambda_{M_\otimes} = 280.460 + 36000.771\, T_{UT1} \tag{3.15}$$

- Obtain the mean anomaly of the Sun

$$M_\otimes = 357.5277233 + 35999.05034\, T_{UT1} \tag{3.16}$$

- Approximate the ecliptic longitude

$$\lambda_{ecliptic} = \lambda_{M_\otimes} + 1.914666471^\circ s(M_\otimes) + 0.019994643 s(2M_\otimes) \tag{3.17}$$

- Find the distance from the Earth to the Sun

$$r_\otimes = 1.000140612 - 0.016708617 c(M_\otimes) - 0.000139589 c(2M_\otimes) \tag{3.18}$$

- Obtain the obliquity of the ecliptic

$$\epsilon = 23.439291^\circ - 0.0130042\, T_{UT1} \tag{3.19}$$

- Final step, calculate the sun vector in ECI frame

$$s^i = r_\otimes \begin{bmatrix} c(\lambda_{ecliptic}) \\ c(\epsilon)s(\lambda_{ecliptic}) \\ s(\epsilon)s(\lambda_{ecliptic}) \end{bmatrix} \tag{3.20}$$

Lastly, the Julian date can be calculated as: [16]

$$JD = 367(yr) - INT[\frac{7[yr + INT(\frac{mo+9}{12})]}{4}] + INT(\frac{275mo}{9}) + d + 1721013.5 + \frac{\frac{\frac{sec}{60*} + min}{60} + ho}{24} \tag{3.21}$$

where yr is the current year, mo is the current month, d is the current day, ho is the current hour, min is the current minute and sec is the current second. Also, the INT function provides the truncation, meaning that the output is the greatest integer less than or equal the input.

3.1.3 Magnetic field model

The magnetic field modeling for the space environment module is implemented using the IGRF which is a series of mathematical models that describes the earth's magnetic field between epochs 1900 A.D. and the present, producing a highly accurate result. [17] Due to its complexity, it is implemented with the Aerospace toolbox from Matlab Simulink software. It requires as an input the position vector of the satellite expressed in longitude, latitude, and altitude and results in the magnetic field vector output in NED frame. Additionally, Simulink includes a converter between ECEF frame and the LLA required by the IGRF block.

Since the cubesat distance is expressed in ECI frame from chapter 3.1.1, it is necessary to convert to ECEF frame before it is introduced in the Simulink toolbox. The rotation matrix is calculated as: [18]

$$\omega_{ie} = \frac{2\pi}{(24 \times 60 \times 60)} \tag{3.22}$$

$$\boldsymbol{R_e^i} = \begin{bmatrix} c(\omega_{ie}t) & -s(\omega_{ie}t) & 0 \\ s(\omega_{ie}t) & c(\omega_{ie}t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.23}$$

$$r^e = \boldsymbol{R_e^{iT}}(r^i) \tag{3.24}$$

Where the ω_{ie} represents the angular velocity of the earth and t is the time in seconds. Finally, to get the magnetic field expressed in orbit frame, the composite rotations are carried as NED to ECEF, ECEF to ECI and lastly, ECI to orbit frame.

$$\boldsymbol{R_n^e} = \begin{bmatrix} -c(l)s(\xi) & -s(l) & -c(l)c(\xi) \\ -s(l)s(\xi) & c(l) & -s(l)c(\xi) \\ c(\xi) & 0 & -s(\xi) \end{bmatrix} \tag{3.25}$$

$$b^o = \boldsymbol{R_i^o R_e^i R_n^e} b^n \tag{3.26}$$

Where the rotation from NED to ECEF takes the longitude, latitude and altitude values in radians.

3.1.4 LEO perturbations

Aiming to test the control robustness against disturbances, it is necessary to include the most common perturbation models, being:

- Gravity Gradient
  - Gravitational constant based model
  - Angular speed based model
- Magnetic residuals
- Solar Radiation
- Drag

These are frequently found in Low Earth Orbit environment.

The gravity gradient torque is produced due to the fluctuations in the gravitational pull, causing the spacecraft to have an unstable orientation [19]. It can be calculated as:

$$\tau_{g_g}^b = \frac{3\mu}{r^b 5} r^b \times J r^b \tag{3.27}$$

Where μ is the earth's mass product with the gravitational constant, r is the satellite distance, and J is the inertia. Alternatively, it can be also be approximated using the angular velocity of the orbit: [11] [13]

$$\tau_{g_g}^b = -3n^2 r^b \times J r^b \tag{3.28}$$

The next perturbation that can be found in LEO is the magnetic residuals torque. This disturbance is produced by the electronic elements equipped in the satellite, generating a magnetic field in the structure [11] [13] [5]. It is calculated as:

$$\tau_{m_r} = m_r \times B \tag{3.29}$$

Where B is the earth's magnetic field and m_r is the residual magnetic field generated by the residuals. Since the residuals magnitude has to be measured experimentally, the simulator implements the values expressed in the journal from [5].

On the other hand, the solar radiation pressure that the cubesat receives is caused because of the photons pressure produced by the sun that causes a momentum flux on the spacecraft. The solar force per unit of mass is obtained as:

$$F_{solar} = -P(1+\eta)\frac{A}{m} ecl \tag{3.30}$$

Where P is the constant of momentum flux caused by the sun, η is the reflexivity coefficient, and ecl the eclipse factor. This is multiplied by the satellite mass and the torque is calculated as the cross product between the geometric center and the center of gravity displacement and the force calculated above. The η and v parameters for the simulator are chosen as 1 in order to apply the maximum disturbance to the satellite [5] [20] [21] [22].

Finally, the drag produced by the friction between the cubesat and the atmosphere can be represented as a force per unit of mass:

$$F_{drag} = -\frac{1}{2}\rho(C_d \frac{A}{m})|V|V \tag{3.31}$$

where ρ is the mean atmospheric density at 450km of altitude, C_d is the drag coefficient from the satellite set as 2, A is the cross-sectional area of the vehicle, V is the linear velocity of the satellite calculated as the cross product between the satellite distance and the angular velocity of the orbit. The torque that is exerted in the cubesat is calculated as the cross product between the geometric center and the center of gravity displacement and the force multiplied by the satellite's mass. [5] [20] [22]

All of the previous torques are summed up in a resulting torque that is included in the nonlinear satellite model described in the following section.

3.2 Satellite model

3.2.1 Quaternion Kinematics:

The satellite orientation relative to the orbit is represented as:

$$\dot{q}_{ob} = \frac{1}{2} T(q_{ob}) \begin{bmatrix} 0 \\ w^b_{ob} \end{bmatrix} \tag{3.32}$$

Where

$$\dot{q}_{ob} = \frac{1}{2} \begin{bmatrix} q_{ob_0} & -q_{ob_1} & -q_{ob_2} & -q_{ob_3} \\ q_{ob_1} & q_{ob_0} & -q_{ob_3} & q_{ob_2} \\ q_{ob_2} & q_{ob_3} & q_{ob_0} & -q_{ob_1} \\ q_{ob_3} & -q_{ob_2} & q_{ob_1} & q_{ob_0} \end{bmatrix} \begin{bmatrix} 0 \\ w^b_{ob_x} \\ w^b_{ob_y} \\ w^b_{ob_z} \end{bmatrix} \tag{3.33}$$

Since the quaternion should be normalized, then:

$$q_{ob_0} = \sqrt{1 - q^2_{ob_1} - q^2_{ob_2} - q^2_{ob_3}} \tag{3.34}$$

Thus, the reduced quaternion kinematics is expressed as:

$$\dot{q}_{ob} = \frac{1}{2} \begin{bmatrix} \sqrt{1 - q^2_{ob_1} - q^2_{ob_2} - q^2_{ob_3}} & -q_{ob_3} & q_{ob_2} \\ q_{ob_3} & \sqrt{1 - q^2_{ob_1} - q^2_{ob_2} - q^2_{ob_3}} & -q_{ob_1} \\ -q_{ob_2} & q_{ob_1} & \sqrt{1 - q^2_{ob_1} - q^2_{ob_2} - q^2_{ob_3}} \end{bmatrix} \begin{bmatrix} w^b_{ob_x} \\ w^b_{ob_y} \\ w^b_{ob_z} \end{bmatrix} \tag{3.35}$$

3.2.2 Satellite Dynamics:

The rotation dynamics of the rigid body relative to the inertial frame ECI can be expressed as:

$$\dot{\boldsymbol{J}} w^b_{ib} = -w^b_{ib} \times \boldsymbol{J} \times w^b_{ib} + \tau^b_a + \tau^b_p \tag{3.36}$$

From Newton second law, where τ^b_a is the actuator torque and τ^b_p is the perturbation torque . Expressing the equation relative to the orbit frame we obtain:

$$J\dot{w}^b_{ob} = -S(w^b_{ib}) J w^b_{ib} + JS(w^b_{ib}) R^b_i w^i_{io} + JR^b_i \dot{w}^i_{io} + \tau^b_a + \tau^b_p \tag{3.37}$$

Where

$$w^b_{ib} = w^b_{ob} + \boldsymbol{R^b_i} w^i_{io} \tag{3.38}$$

$$w^i_{io} = \begin{bmatrix} 0 \\ w^i_{io_y} \\ 0 \end{bmatrix} \tag{3.39}$$

Where $w^i_{io_y}$ is constant so $\dot{w}^i_{io} = 0$ Then,

$$\dot{w}^b_{ob} = \boldsymbol{J^{-1}}(-S(w^b_{ob} + \boldsymbol{R^b_i} w^i_{io})J(w^b_{ob} + \boldsymbol{R^b_i} w^i_{io}) + \boldsymbol{J}S(w^b_{ob} + \boldsymbol{R^b_i} w^i_{io})\boldsymbol{R^b_i} w^i_{io} + \tau^b_a + \tau^b_p) \tag{3.40}$$

Which can be expressed as

$$\dot{w}^b_{ob} = \begin{bmatrix} \frac{-(j_x w^i_{io_y} w^b_{ob_y} + j_y w^b_{ob_y}(w^i_{io_y} - w^b_{ob_z}) - j_z w^b_{ob_y}(w^i_{io_y} - w^b_{ob_z}))}{j_x} \\ \frac{(j_y w^i_{io_y} w^b_{ob_x} + j_x w^b_{ob_x}(w^i_{io_y} - w^b_{ob_z}) - j_z w^b_{ob_x}(w^i_{io_y} - w^b_{ob_z}))}{j_y} \\ \frac{(j_x w^b_{ob_x} w^b_{ob_y} - j_y w^b_{ob_x} w^b_{ob_y})}{j_z} \end{bmatrix} + \boldsymbol{J^{-1}}(\tau^b_a + \tau^b_p) \tag{3.41}$$

3.2.3 State Space:

Since the actuators have an internal control for desaturation, in this model the torque is taken as the value from τ^b_a. Additionally, the disturbance torques are not dependant on the states nor the model inputs, so they are not included in this model. The state space model is:

$$\dot{x} = \boldsymbol{A}x + \boldsymbol{B}u \tag{3.42}$$

$$y = \boldsymbol{C}x + \boldsymbol{D}u \tag{3.43}$$

Where,

$$x = \begin{bmatrix} w^b_{ob_x} \\ w^b_{ob_y} \\ w^b_{ob_z} \\ q_{ob_1} \\ q_{ob_2} \\ q_{ob_3} \end{bmatrix}$$

$$u = \begin{bmatrix} \tau^b_{a_1} \\ \tau^b_{a_2} \\ \tau^b_{a_3} \end{bmatrix}$$

$$\boldsymbol{A} = \begin{bmatrix} 0 & \frac{-(j_x w^i_{io_y} + j_y(w^i_{io_y} - w^b_{ob_z}) - j_z(w^i_{io_y} - w^b_{ob_z}))}{j_x} & 0 & [0]_{1\times 3} \\ \frac{(j_y w^i_{io_y} + j_x(w^i_{io_y} - w^b_{ob_z}) - j_z(w^i_{io_y} - w^b_{ob_z}))}{j_y} & 0 & 0 & [0]_{1\times 3} \\ 0 & \frac{(j_x - j_y) w^b_{ob_x}}{j_z} & 0 & [0]_{1\times 3} \\ \frac{1}{2} q_{ob_0} & -\frac{1}{2} q_{ob_3} & \frac{1}{2} q_{ob_2} & [0]_{1\times 3} \\ \frac{1}{2} q_{ob_3} & \frac{1}{2} q_{ob_0} & -\frac{1}{2} q_{ob_1} & [0]_{1\times 3} \\ -\frac{1}{2} q_{ob_2} & \frac{1}{2} q_{ob_1} & \frac{1}{2} q_{ob_0} & [0]_{1\times 3} \end{bmatrix} \tag{3.44}$$

$$\boldsymbol{B} = \begin{bmatrix} \frac{1}{j_x} & 0 & 0 \\ 0 & \frac{1}{j_y} & 0 \\ 0 & 0 & \frac{1}{j_z} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.45}$$

$$\boldsymbol{C} = [I]_{6\times 6} \tag{3.46}$$

$$\boldsymbol{D} = [0]_{6\times 3} \tag{3.47}$$

However, this is a nonlinear model containing state multiplications in 3.44, hence it needs to be linearized afterwards to be applied in the optimal controllers.

3.3 Sensor models

As a way to keep the simulator as realistic as possible, sensor models are applied providing the data to the estimator module. The three sensors that are modeled are a gyroscope, a magnetometer, and a sun sensor. First, the gyroscope is defined as the sum of the angular velocity true value from attitude dynamics, the bias, the sensor noise and the static error. The assumptions for this model are:

- Small effect on short time measurements
- The sensor is calibrated
- The gyroscope will operate at constant temperature

Based on this, the bias and the static error are assumed zero. The noise is interpretated as a white noise signal bounded by the magnitude provided by the datasheet of the commercial model MAX21000 [23]

Furthermore, the magnetometer has the magnetic field vector input from the satellite environment from the model described in chapter 3.1.3 and is rotated to body frame and summed up with a noise value represented as a white noise signal bounded by the values from the datasheet of the CubeMag Gen 2 produced by Cubespace satellite systems. Additionally, a rate transition is implemented to reduce the sensor block sample time to 0.2 seconds as it is stated in the datasheet. Similarly, the sun sensor is based on the CubeSense Sun Gen 2 by Cubespace satellite systems, and takes the input from the space environment model described in chapter 3.1.2 which is rotated to body frame and summed up with a noise value as in the magnetometer model. In this case the noise bound is defined by the angle error defined in the datasheet. As it is done in the magnetometer model, a rate transition is implemented to reduce the sensor block sample time to 0.5 seconds. [24]

3.4 Actuator models

As a means to translate the controller signal to an actual satellite movement, the use of actuators is required. The cubesat is equipped with a set of three reaction wheels and a set of magnetic torquers, both producing the torques needed in the spacecraft for attitude correction.

The reaction wheels are devices that use the principle of Newton's third law, creating a reaction that rotates the vehicle in the opposite direction as the wheel is spinning, meaning that the momentum of the satellite has the opposite sign of the momentum generated by the reaction wheels. From this, the reaction wheels torque can be calculated as the derivative of its momentum. [25]

$$\tau^b_{a_{rw}} = -\dot{h^b_\omega} - S(\omega^b_{i,b})h^b_\omega \tag{3.48}$$

Considering that the controller provides a desired torque as the output signal, it is possible to obtain the desired wheel momentum integrating the previous equation 3.48:

$$\dot{h}^b_\omega = -\tau^b_d - S(\omega^b_{i,b})h^b_\omega \tag{3.49}$$

Subsequently, this momentum is saturated by the device limits established by the manufacturer, resulting in the produce torque:

$$\tau^b_{a_{rw}} = -\dot{h}^b_\omega - S(\omega^b_{i,b})h^b_\omega \tag{3.50}$$

The values included in the simulator are based on the commercial model CubeWheels Gen 2 by Cubespace. One of the disadvantages of these actuators is saturation issues. Once the wheel reaches its highest speed it stops producing a torque. This is why a desaturation system is required.

Magnetic torquers are actuators that operate with local magnetic fields produced by a current that flows through a coil, which is aligned with the earth's magnetic field. These actuators are used mostly on nanosatellites due to their greatly limited torque capacity. This is why it is implemented as a desaturation system, complementing the reaction wheels. The magnetic torquers induced magnetic field can be obtained from the desired torque as: [26]

$$m_t^b = \frac{S(B^b)\tau_d^b}{||B^b||^2} \tag{3.51}$$

where B^b is the earths magnetic field in body frame. Afterwards, the induced magnetic field is saturated using the limit values from the manufacturer's datasheet, in this case the CubeTorquer Gen 2 from Cubespace Satellite Systems. The actuator torque can be calculated as: [15]

$$\tau_{m_t}^b = S(m_t^b)B^b \tag{3.52}$$

With the actuators described above, it is possible to implement a combined actuator system that is controlled by an internal actuator control designed as an On-Off system: The actuator

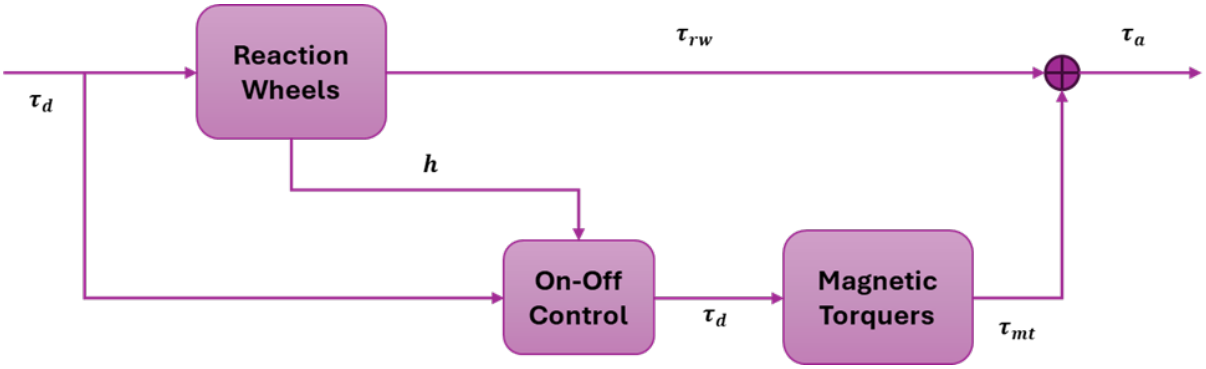

Figure 11: On-Off Actuator System Structure

controller is designed as an enabler for the magnetic torquers with the following algorithm:

- The controller reads the momentum from each wheel
- The momentum is converted into angular speed in RPMs
- If the angular speed exceeds the value of $\alpha \times \omega_{MAX}$, the magnetic torquers receive the desired torque for the respective axis
- If the angular speed is below the conditional speed the desired torque for the magnetic torquers is set as zero

It is important to mention that the actuators are assumed as aligned with the center of gravity of the spacecraft.

3.5 Attitude determination

One of the main components of the satellite's ADCS is the state estimator. In order to obtain a good performance from the control module, it is important that the system is able to have an accurate reading of the orientation and its angular speed. Nevertheless, the satellite is

equipped with a sun sensor, a magnetometer, and a gyroscope, meaning that there is no direct measurement of the orientation for which is necessary to estimate the cubesat angles. One of the most implemented estimators for small satellites is the TRIAD algorithm. This method uses unit vectors from attitude sensors, in this case, the magnetometer and the sun sensor, and the modeled unit vectors using a sun model and a magnetic field model, finding two orthonormal vector pairs which are used to calculate the direction cosine matrix that is converted to get the estimated quaternion. [27]

$$o_{b_1} = \frac{s_b}{|s_b|}$$

$$o_{b_2} = \frac{s_b \times B_b}{|s_b \times B_b|}$$

$$o_{b_3} = \frac{o_{b_1} \times o_{b_2}}{|o_{b_1} \times o_{b_2}|} \tag{3.53}$$

$$r_{f_1} = \frac{s_o}{|s_o|}$$

$$r_{f_2} = \frac{s_o \times B_o}{|s_o \times B_o|}$$

$$r_{f_3} = \frac{r_{f_1} \times r_{f_2}}{|r_{f_1} \times r_{f_2}|} \tag{3.54}$$

$$T_q = [o_{b_1} \quad o_{b_2} \quad o_{b_3}][r_{f_1} \quad r_{f_2} \quad r_{f_3}]^T \tag{3.55}$$

The modeled vectors can be obtained from the models described in chapter 3.1.2 in equation 3.20 and 3.1.3 in equation 3.26, however they can be hard to program for which two simpler models can be used to calculate the sun and magnetic field vectors. For the sun vector, it can be considered that the sun vector is aligned with the earth x axis in the first day of spring and rotates with an angular speed of:

$$\omega_{i,s} = \frac{2\pi}{365\ 24\ 60\ 60} \tag{3.56}$$

in the z axis.

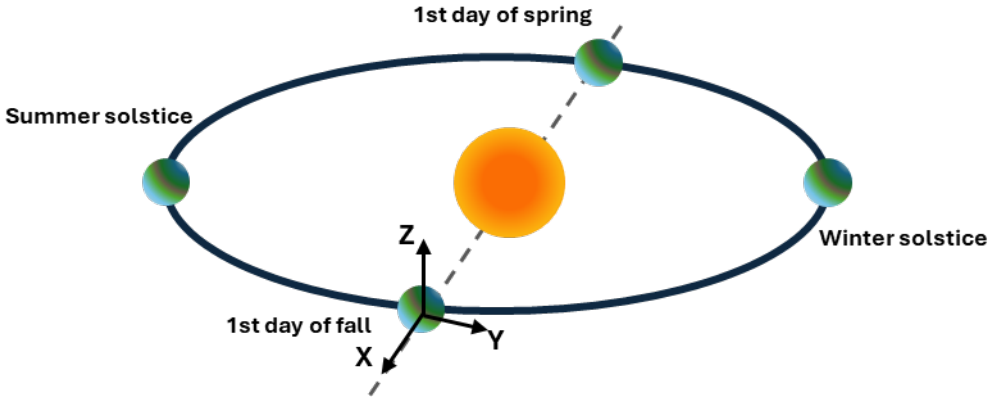

Figure 12: Earth position with respect to the sun

Additionally, the equatorial plane of the Earth is tilted by 23.5 degrees around x axis producing a composite rotation:

$$s^i = \boldsymbol{R_x R_z}(\omega_{i,s} t)[1 \quad 0 \quad 0]^T \tag{3.57}$$

where

$$\boldsymbol{R_x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(\epsilon) & -s(\epsilon) \\ 0 & s(\epsilon) & c(\epsilon) \end{bmatrix} \tag{3.58}$$

$$\boldsymbol{R_z} = \begin{bmatrix} c(\omega_{i,s}t) & -s(\omega_{i,s}t) & 0 \\ s(\omega_{i,s}t) & c(\omega_{i,s}t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.59}$$

On the other hand, the magnetic field vector can be found from the dipole model:

$$b_{LVLH} = \frac{\zeta}{r^3}[c(nt)s(i) \quad -c(i) \quad 2s(nt)s(i)] \tag{3.60}$$

$$n = \frac{2\pi}{T} \tag{3.61}$$

Including the constant ζ is the earth's magnetic moment. [28]
It is rotated to orbit frame with:

$$\boldsymbol{R^o_{LVLH}} = \begin{bmatrix} 0 & 0 & -1 \\ 1 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix} \tag{3.62}$$

Moreover, as a complement for the TRIAD estimator, it is possible to implement a nonlinear cascade complementary filter NCCF implementing the data from the gyroscope and quaternion dynamics to get an alternative quaternion estimation and complementing the values from the TRIAD. [29] The block diagram is:

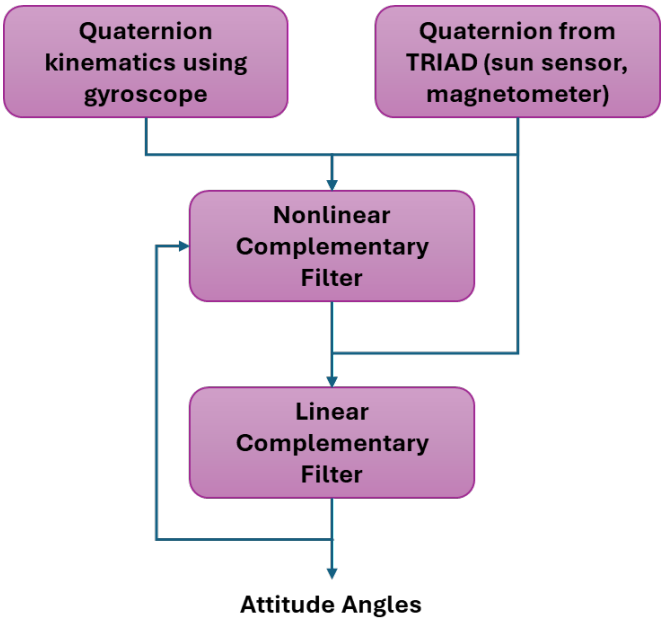

Figure 13: Nonlinear Complementary Cascade Filter

where the nonlinear complementary filter contains:
This sensor fusion method improves the estimation since the gyroscope performs better at higher frequencies while the magnetometer and sun sensors have better performance at lower frequencies, as well as avoiding the drifting and bias effects from the gyro sensor. Finally, the angular velocity is directly taken from the gyroscope measurement.

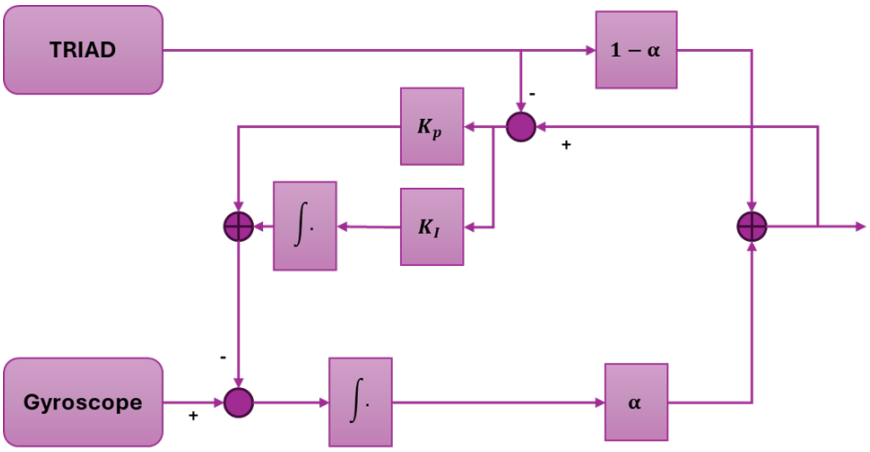

Figure 14: Nonlinear Complementary Filter

4 Control design

The main objective of this thesis is to design an optimal attitude control solution for the 2U Cubesat supplied with three maneuvers for debris detection. This section contains the development of the optimal controllers that are tested in the simulator ADCS module. First, the maneuvers that the satellite will perform are described, followed by the LQR and MPC design and structure. Subsequently, The MPC stability analysis is done providing the minimum memory requirements for hardware.

4.1 Maneuvers description

The Cubesat mission consists on debris detection, for which is important to develop a set of maneuvers that facilitate the objects recognition process. These are:

- Debris detection scanning: Where the control system was designed with high relative importance on the state's performance and moderate input penalty, allowing the radar to scan while the CubeSat is pointing to a desired angle spectrum. This is achieved by choosing the angle set point as a sine wave, making the spacecraft to perform an oscillatory movement enhancing the area of detection during the orbit.

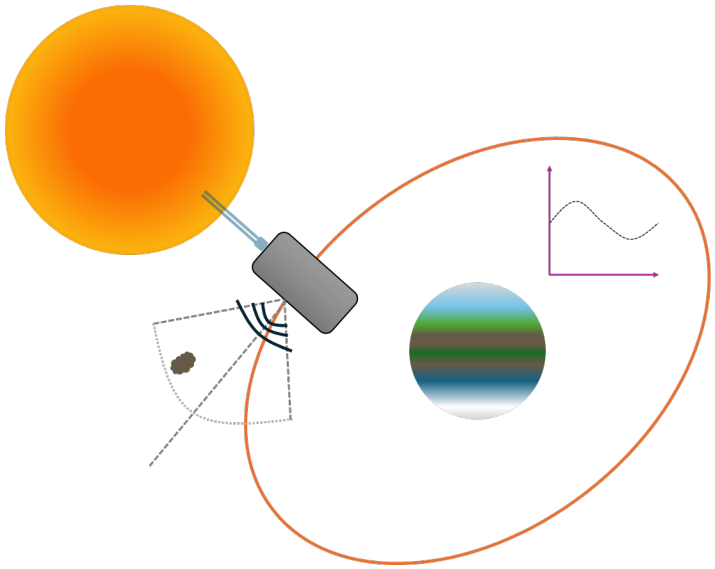

Figure 15: Debris Detection Scanning maneuver

- Validation process: The detected objects on the first maneuver will be scanned inside the radar region, making it is possible to approximate the position in which the debris can be spotted after the first detection. Because of the high speed of the objects, false positive detections exist requiring an extra step. The debris validation maneuver gives the satellite the ability to perform more aggressive maneuvers by lowering the penalty on the input

changes. This maneuver is designed for angular trajectory tracking in which the CubeSat performs a controlled rotation that allows the radar to maintain the object inside of the detection volume for a longer period, obtaining more information regarding the debris. This is accomplished by choosing the set point as a ramp function, making the satellite to rotate around a desired axis.

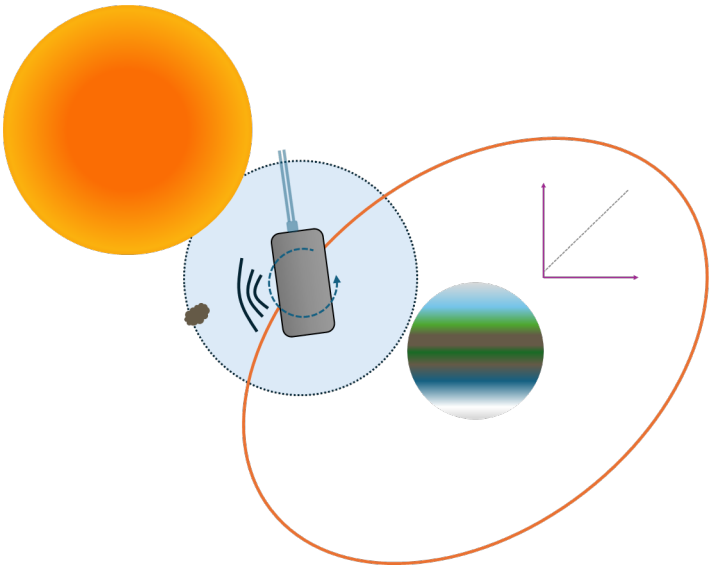

Figure 16: Validation process maneuver

- Charging process: This procedure is developed for solar panels to maximize energy acquisition using a high input penalty, minimizing energy costs. This maneuver uses a step reference, pointing the panels to the sun vector.

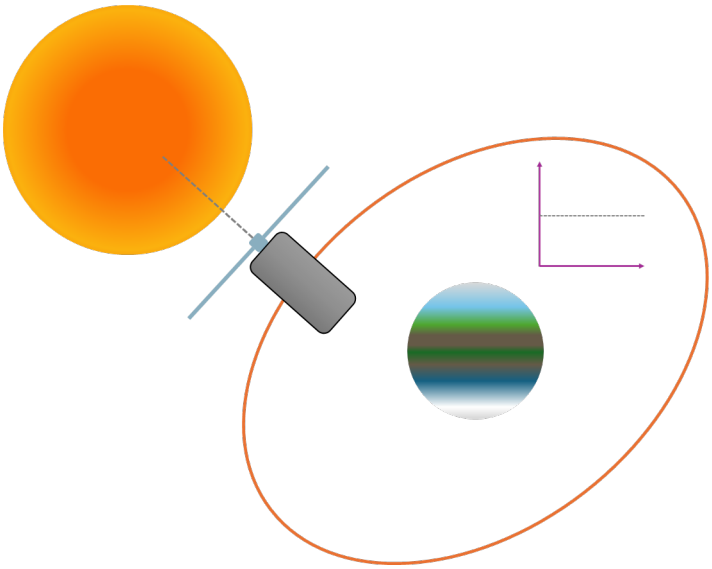

Figure 17: Charging process maneuver

4.2 Optimal control

Control design is a process where the goal is to achieve acceptable performances in terms of time, frequency, rise and settling time, peak overshoot, bandwidth and phase and gain margin. Due to the complexity of the MIMO systems, it is necessary to implement methods with modern

technology to meet the criteria. For this, Optimal Control theory has been developed with digital systems to accomplish a feasible approach.

The goal with Optimal Control is to find an admissible control input that causes the system to follow an admissible trajectory that minimizes the performance measure given by a cost function. This is reflected as a set of gains for a given time step. [30]

In order to evaluate the satellite performance, A Model Predictive Controller is designed, and an LQR controller is also developed for comparison purposes. Both of these control strategies are optimal controllers which are suitable architectures for spacecrafts since they perform specific missions, and the energy consumption is limited making it necessary to contemplate the control signal limits per task. This section is divided in two parts, first developing the LQR controller and lastly designing the MPC.

4.2.1 LQR

Linear Quadratic Regulators are optimal controllers that work based on the plant linear model and minimizing a cost function that provides the most suitable input for the system. There are three common methods used to solve the LQR problem, being:

- Dynamic Programming
- Calculus of Variations
- Iterative Numerical Techniques

By the method of calculus of variation, LQR controllers can perform tasks that require linear tracking by minimizing the following function: [30]

$$J_c = \int_{t_0}^{t_f} [x(t) - r_c(t)]^T \boldsymbol{Q}(t)[x(t) - r_c(t)] + u^T(t)\boldsymbol{R_c}(t)u(t)\ dt \tag{4.1}$$

Where it can be seen that the reference $r_c(t)$ is subtracted from the state $x(t)$. In order to find the optimal solution, the Hamiltonian and costate equations are found:

$$\boldsymbol{H_c} = \frac{1}{2}||x - r_c||^2_{Q_c} + \frac{1}{2}||u||^2_{R_c} + \boldsymbol{P_c}^T \boldsymbol{A}\ x + \boldsymbol{P_c}^T \boldsymbol{B}\ u \tag{4.2}$$

Providing the optimal control input as:

$$u = -\boldsymbol{R_c}^{-1}\boldsymbol{B}^T\boldsymbol{P_c} \tag{4.3}$$

When the Hamiltonian $\boldsymbol{H_c}$ is derived with respect to the control input u and equaled to zero, providing the minimum value. That can be expanded as:

$$u = -\boldsymbol{R_c}^{-1}\boldsymbol{B}^T\boldsymbol{K_c}\ x - \boldsymbol{R_c}^{-1}\boldsymbol{B}^T\ s \tag{4.4}$$

By differentiating the costates with respect of time:

$$\boldsymbol{P_c} = \boldsymbol{K_c}\ x + s_c \tag{4.5}$$

The equations implemented to find the values for $\boldsymbol{K_c}$ and s can be obtained:

$$\dot{\boldsymbol{K_c}} = -\boldsymbol{K_c}\boldsymbol{A} - \boldsymbol{A}^T\boldsymbol{K_c} - \boldsymbol{Q_c} + \boldsymbol{K_c}\boldsymbol{B}\boldsymbol{R_c}^{-1}\boldsymbol{B}^T\boldsymbol{K_c} \tag{4.6}$$

$$\dot{s} = -[\boldsymbol{A}^T - \boldsymbol{K_c}\boldsymbol{B}\boldsymbol{R_c}^{-1}\boldsymbol{B}^T]s_c + \boldsymbol{Q_c}\ r_c \tag{4.7}$$

These two equations 4.6 and 4.7 can be integrated to obtain their values and provide the necessary input to the LQR controller. The value for the regulation gains K_c are calculated offline while the values for s_c are calculated online since they vary depending on the reference r_c. The following diagram represents the process to implement the LQR controller on the Cubesat:

To summarize, the LQR controller implementation algorithm is:

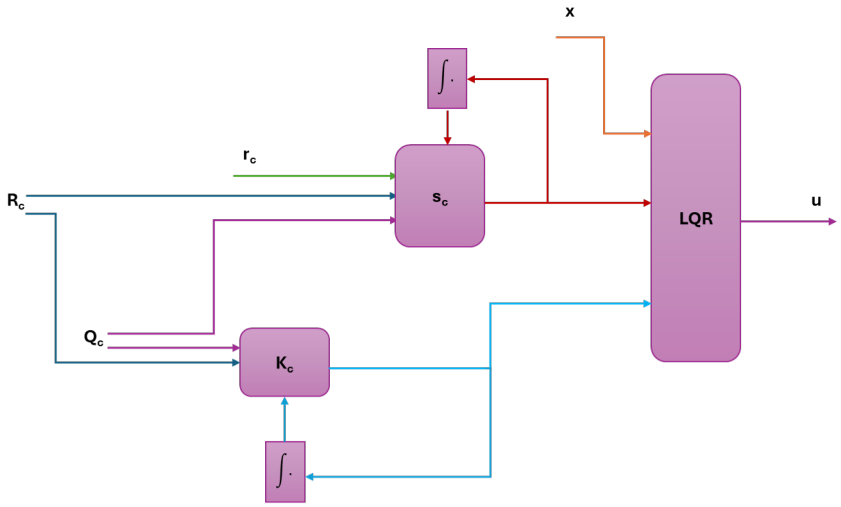

Figure 18: LQR architecture implemented

- Define $\boldsymbol{R_c}$ and $\boldsymbol{Q_c}$ values based on the desired input penalty and state importance respectively
- Obtain the value of $\boldsymbol{K_c}$ offline, integrating $\dot{\boldsymbol{K}}_{\boldsymbol{c}}$ for a defined period of time and choosing a convergence point using equation 4.6
- Obtain the value of s online, by integrating $\dot{s}$, using the chosen $\boldsymbol{K_c}$ from the previous step and the set point with equation 4.7
- Calculate the value of u with the current s at that time step, the chosen $\boldsymbol{K_c}$ from step 2 and the defined $\boldsymbol{R_c}$ in step 1 using equation 4.4

4.2.2 MPC design

The Model Predictive Controller is also an optimal controller that aims to find a trajectory of the future manipulated control input while optimizing the future behavior of the system output within a limited time window. It operates in a control and prediction horizon that allows the controller to look into the future responses of the satellite based on the plant model in order to optimize the current time step control signal. Each time step processes the prediction and optimization.

Compared to the LQR controller, even with the similarity of both being optimal controllers minimizing a cost function to calculate a control signal, it presents a big advantage by anticipating future events allowing the system to react appropriately with constant optimizations. Additionally, the controller runs with an augmented model that contains the prediction and an algorithm of Quadratic Programming that solves the optimization based on the specified constrains by finding a Hessian matrix and a linear vector that contains the predictive model and the tuning weights. [31]

The parameters that are included in the MPC design are:

- Prediction Horizon: Indicates the time period in which it is desired to look into the future for prediction purposes. It is given as the number of steps and it is dependent on the sample time.
- Receding Control Horizon: The control horizon is the set of control actions that generate the result for the predicted outputs. Even if the optimizer is able to find the control signal

for all the steps chosen, the Receding Control Horizon principle only uses the first sample and not the whole trajectory and instead calculates a new control input on the next time step.

- Q: It is the matrix that contains the weights for the states importance.
- R: It is the matrix that contains the weights for the control signal penalty.

Moreover, a linear model and a state estimator are required in order to predict and observe the system responses. It is possible to implement alternative versions of the MPC that improve the response on nonlinear systems as in the case of the CubeSat. These are:

- Nonlinear MPC: It directly implements the nonlinear model, nonetheless, to be able to solve the optimization part it is required to use nonlinear cost functions and constraints, leading to a nonconvex solution that is severely hard to implement in an online controller, requiring high computational effort.
- Gain-Scheduled MPC: This method is ideal for scenarios where the number of constrains and the model change during the mission. It implements an offline linearization for each case of interest, allowing the design of independent controllers that will switch depending on the current task of the spacecraft.
- Adaptive MPC: This structure works with a linear model that operates online as the mission conditions vary, updating the internal model on each time step. Although it cannot process constraints and states changes, it requires less computational effort than the methods mentioned above.

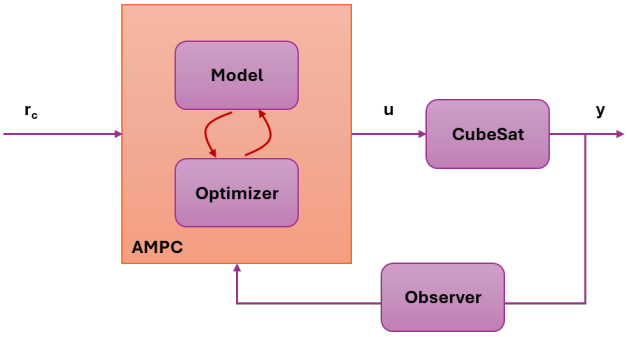

Figure 19: Adaptative MPC

In Matlab Simulink, it is possible to implement a toolbox for the AMPC. The first step for the implementation is to design an MPC controller. For this, the conventional parameters are chosen as:

- Prediction Horizon: 15
- Control Horizon: 3 (it is suggested to select a value between 15% to 20%)
- Q: An identity matrix with an escalation gain
- R: A diagonal matrix that contains zeros for velocity states and ones for angle states

Moreover, it is possible to include noise values during the design process, an α value that is a design parameter that escalates the Q and R depending on the wanted robustness or aggressiveness of the controller. Additionally, it has the option to select between an internal estimator

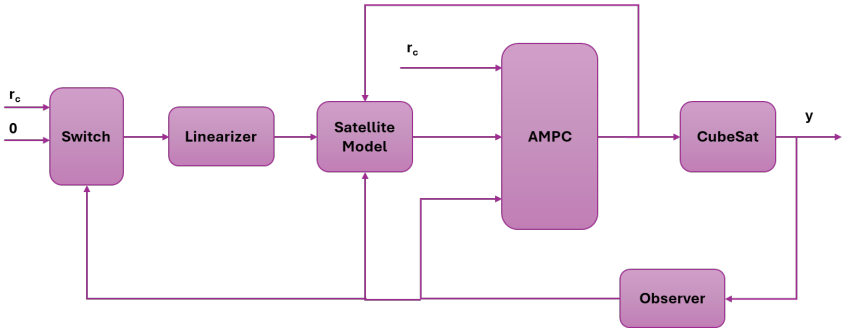

Figure 20: Adaptative MPC SIMULINK

based on a Kalman Filter or a custom estimator. Since the ADCS module already has an estimation segment, it is chosen as custom. The next diagram represents the structure of the AMPC implemented on Simulink:

Following this algorithm:

- The model is linearized around a chosen point between the set point, the current estimated states, and the equilibrium point at zero. Only the matrix A is affected since B and C are linear.
- The model is converted to digital around the controller sample time, not necessarily the same as the simulator.
- The Simulink AMPC block gets the input from the linearizer, the estimator and the reference signal.
- The control signal is feedback into the external model

On figure 21 the MPC Designer toolbox from Matlab is displayed:

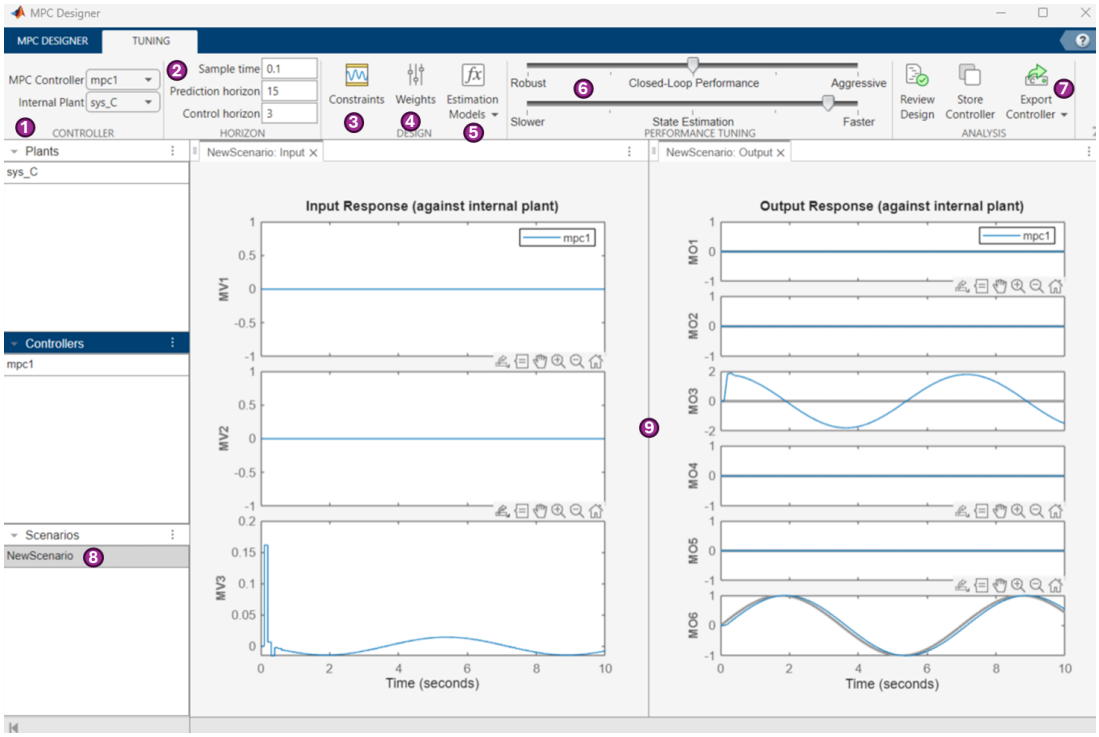

Figure 21: Matlab MPC Designer toolbox

1. Allows the user to select the linear model for the control design
2. The controller sample time, prediction horizon, and control horizon can be chosen
3. Includes the constraints menu to set the boundaries for both inputs and outputs
4. Here the states importance and control signal penalties are added
5. Helps to parameterize the internal estimator
6. Allows the user to balance the robustness and aggressiveness of the control signal
7. Exports the controller into the workspace
8. Here the scenarios to be tested can be added, for this example is a sine wave as reference on the z axis
9. The inputs and outputs responses are plotted here for tuning purposes

Whereas the controller can be implemented with the previous method, for real execution it is necessary to program the AMPC in a way that can be taken to a microcontroller. For this purpose, the prediction model is developed together with an optimization model.

For the MPC design, the state-space model is defined as: [31]

$$x(k+1) = \boldsymbol{A}x(k) + \boldsymbol{B}u(k) \tag{4.8}$$

$$y(k) = \boldsymbol{C}^* x(k) \tag{4.9}$$

Where it can be seen that the matrix $\boldsymbol{D}$ is omitted in equation 4.9, this because in the receding horizon control the input u cannot affect the output y simultaneously. Moreover, due to the controllability of the prediction model the matrix $\boldsymbol{C}^*$ is reduced to:

$$\boldsymbol{C}^* = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.10}$$

Meaning that the outputs are reduced to the three angle states.

Since the goal is to have a model that is able to determine the future state variables, control inputs, and outputs, it is taken to an incremental model defined as:

$$\Delta x(k+1) = \boldsymbol{A}\Delta x(k) + \boldsymbol{B}\Delta u(k) \tag{4.11}$$

Where the increment of the states and the inputs are:

$$\Delta x = x(k) - x(k-1) \tag{4.12}$$

$$\Delta u = u(k) - u(k-1) \tag{4.13}$$

Now, to relate the system output with the Δx from equation 4.11, the augmented state-space model in equations 4.14 and 4.15 is proposed:

$$\begin{bmatrix} \Delta x(k+1) \\ y(k+1) \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} & [0] \\ \boldsymbol{C}^*\boldsymbol{A} & [I]] \end{bmatrix} \begin{bmatrix} \Delta x(k) \\ y(k) \end{bmatrix} + \begin{bmatrix} \boldsymbol{B} \\ \boldsymbol{C}^*\boldsymbol{B} \end{bmatrix} \Delta u(k) \tag{4.14}$$

$$y(k) = \begin{bmatrix} [0]] & [I] \end{bmatrix} \begin{bmatrix} \Delta x(k) \\ y(k) \end{bmatrix} \tag{4.15}$$

If a vector ΔU containing the incremental future values of the input u within the receding control horizon, and a vector ΔY containing the future predicted values for the system output within the prediction horizon are defined in equations 4.16 and 4.17 respectively:

$$\Delta U = [\Delta u(k)^T \quad \Delta u(k+1)^T \quad ... \quad \Delta u(k+N_{ctr}-1)^T \quad]^T \tag{4.16}$$

$$Y = [y(k+1|k)^T \quad y(k+2|k)^T \quad ... \quad y(k+N_{prd}|k)^T \quad]^T \tag{4.17}$$

Then, based on the incremental model, the future predicted values for the system output Y can be obtained as:

$$Y = F_{mpc}\, x(k) + \phi_{mpc}\Delta U \tag{4.18}$$

Including:

$$F_{mpc} = \begin{bmatrix} C_{mpc}A_{mpc} \\ C_{mpc}A_{mpc}^2 \\ C_{mpc}A_{mpc}^3 \\ ... \\ C_{mpc}A_{mpc}^{N_{prd}} \end{bmatrix} \tag{4.19}$$

$$\phi_{mpc} = \begin{bmatrix} C_{mpc}B_{mpc} & 0 & ... & 0 \\ C_{mpc}A_{mpc}B_{mpc} & C_{mpc}B_{mpc} & ... & 0 \\ C_{mpc}A_{mpc}^2B_{mpc} & C_{mpc}A_{mpc}B_{mpc} & ... & 0 \\ ... & & & \\ C_{mpc}A_{mpc}^{N_{prd}-1}B_{mpc} & C_{mpc}A_{mpc}^{N_{prd}-2}B_{mpc} & ... & C_{mpc}A_{mpc}^{N_{prd}-N_{CTR}}B_{mpc} \end{bmatrix} \tag{4.20}$$

Where A_{mpc}, B_{mpc}, and C_{mpc} are the matrices formed in the augmented state-space model from equations 4.14 and 4.15

As a result of the equation 4.18 containing the prediction model which leads to the optimization part, it is now necessary to find a set of signals ΔU that cause the predictive output Y to be as close as possible as the reference signal. This is where the MPC acts, finding the optimal solution.

By defining a cost function:

$$J_{mpc} = \Delta U^T(\phi_{mpc}^T Q_{mpc}\phi_{mpc} + R_{mpc})\Delta U - 2\Delta U^T\phi_{mpc}^T Q_{mpc}(Rs_{mpc} - F_{mpc}x) \tag{4.21}$$

Where:

- Rs_{mpc} is the reference signal vector of size = $outputs \cdot N_{prd} \times 1$
- Q_{mpc} is the states importance diagonal matrix of size = $outputs \cdot N_{prd} \times outputs \cdot N_{prd}$
- R_{mpc} is the control signal penalty diagonal matrix of size = $inputs \cdot N_{ctr} \times inputs \cdot N_{ctr}$

There are two possible approaches from this point. It is possible to find a minimum value for the ΔU by deriving equation 4.21 with respect to ΔU and making it equal to zero. The second option is to implement a nonlinear programming solver for the optimization problem.

The most common way for solving optimization MPC problems is with Quadratic Programming, given by the general form:

$$U = MIN(\frac{1}{2}\Delta U^T H_{mpc}\Delta U + f_{mpc}^T\Delta U) \tag{4.22}$$

Equation 4.21 is algebraically treated to be on the form of equation 4.22, from where the Hessian matrix is given by:

$$H_{mpc} = (\phi_{mpc}^T Q_{mpc} \phi_{mpc} + R_{mpc})^{-1} \tag{4.23}$$

And the vector f_{mpc} being:

$$f_{mpc} = \phi_{mpc}^T Q_{mpc}(Rs_{mpc} - (F_{mpc}[\Delta x \quad Y])) \tag{4.24}$$

Consequently, the optimal control signal is obtained from:

$$u(k) = \Delta u + u(k-1) \tag{4.25}$$

Where Δu is the first element of the optimization result as it is stated by the receding control horizon. This value is provided by the Matlab function quadprog.

The diagram that represents this approach of the AMPC controller is presented in figure 22

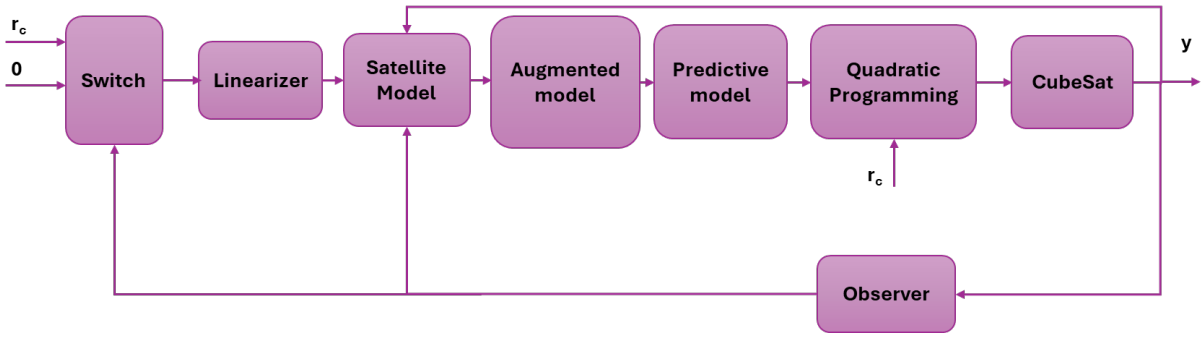

Figure 22: Adaptive MPC using Quadratic Programming

To sum up, the programmed AMPC can be executed by:

- The model is linearized around a chosen point between the set point, the current estimated states, and the equilibrium point at zero. Only the matrix A is affected since B and C are linear.
- The model is converted to digital around the controller sample time, not necessarily the same as the simulator.
- The incremental state-space model is calculated
- The augmented model is found providing the new $A_{mpc}, B_{mpc}, C_{mpc}$
- Find the matrices F_{mpc} and ϕ_{mpc} that produce the predicted output
- Expand the reference signal, Q_{mpc}, and R_{mpc} matrices into the proper dimensions for the prediction model
- Find the Hessian matrix and the vector f_{mpc} to provide an input to the QP solver
- Find the optimal value for u with the desired constrains, in this case, an upper and lower bound
- Last step, apply the Receding Control Horizon Principle by limiting the output to the first sample

4.2.3 Controller stability

Due to the constrains of the Model Predictive Controller, it is required to implement optimal predictive control properties that allow to establish stability of the closed loop system under terminal-state conditions.

Because of the receding control horizon defined in chapter 4.2.2, and the incremental control definition, where the general cost function 4.26 is minimized: [31]

$$J_{mpc} = \sum_{k=1}^{N_{prd}} x(k_i + k|k_i)^T Q_{mpc} x(k_i + k|k_i) + \sum_{k=0}^{N_{prd}-1} \Delta u(k_i + k)^T R_{mpc} \Delta u(k_i + k) \tag{4.26}$$

In every step, an equality constraint on the final state is necessary to establish stability. From this cost function, a Lyapunov function for the MPC is chosen as the cost function 4.26 minimum:

$$V(x(k), k) = \sum_{k=1}^{N_{prd}} x(k_i + k|k_i)^T Q_{mpc} x(k_i + k|k_i) + \sum_{k=0}^{N_{prd}-1} \Delta u(k_i + k)^T R_{mpc} \Delta u(k_i + k) = J_{min} \tag{4.27}$$

Where being a quadratic form make it positive definite, and moreover, the Lyapunov function stated tends to infinity when the state tends to infinity. When the function 4.27 is differentiated, it is obtained:

$$V(x(k+1), k+1) - V(x(k), k) \leq -x(k+1)^T Q_{mpc} x(k+1) - \Delta u(k)^T R_{mpc} \Delta u(k) < 0 \tag{4.28}$$

Which is a negative function, establishing that the system is asymptotically stable for the MPC controller.

Hence, the following assumptions are a requisite to achieve an asymptotically stable closed-loop system:

1. A constraint
$$x(k + N_{prd}|k) = 0 \tag{4.29}$$
is placed in the terminal state of the receding horizon

2. For each step of time there exist a solution where the cost function 4.26 is minimized taking into consideration the system constraints, including the one on equation 4.29

5 Results

This chapter presents the results achieved during this master thesis development. These include:

- An optimal model predictive control that is able to regulate the satellite attitude to perform space debris detection maneuvers, described in this chapter.
- A Satellite Simulator that provides the data necessary for the controller to perform, including space models with perturbations in LEO, plant model, hardware modeled based on commercial products, and an estimator. It is described in chapter 3 and complemented in Appendix A.
- A set of secondary controllers that provide an alternative perspective to the main solution proposed in the first result, described in this chapter.
- Three maneuvers that are potential to facilitate the space debris detection by the satellite, having as a reference the QBDebris and UNICube projects at UiT, described in chapter 4 and evaluated in this chapter.
- As a side goal, it is intended to publish this project as an article, thus an abstract that was presented to the 75th International Astronautical Congress is included in Appendix B.

Four control strategies were developed, including:

1. Adaptive MPC controller, implemented through a Matlab Simulink toolbox
2. Adaptive MPC controller, programmed manually on a Matlab function block based on the math presented in chapter 4.2
3. LQR controller, developed as well in chapter 4.2
4. PD controller

Moreover, the results shown throughout this chapter are evaluated including all the realistic parameters added in the simulator as such as perturbations, sensor noise, initial conditions for dynamics, and an actuator model.

5.1 Maneuver 1: PD, LQR, AMPC comparison

The first maneuver consist on a step signal as the reference for the satellite angles. IT is intended to emulate the spacecraft to point to an specific point, in this case, to face the solar panels to the sun.

The first strategy that is tested is the PD controller, which is able to regulate the satellite attitude and reach the reference signal provided. In addition, the error indexes known as IAE and ITAE, which evaluates the angles vs the reference signals weighting the transient response and the steady state error were calculated for this and the following controllers:

$$IAE = 1.16 \times 10^{-15}$$

$$ITAE = 7.486 \times 10^{-14}$$

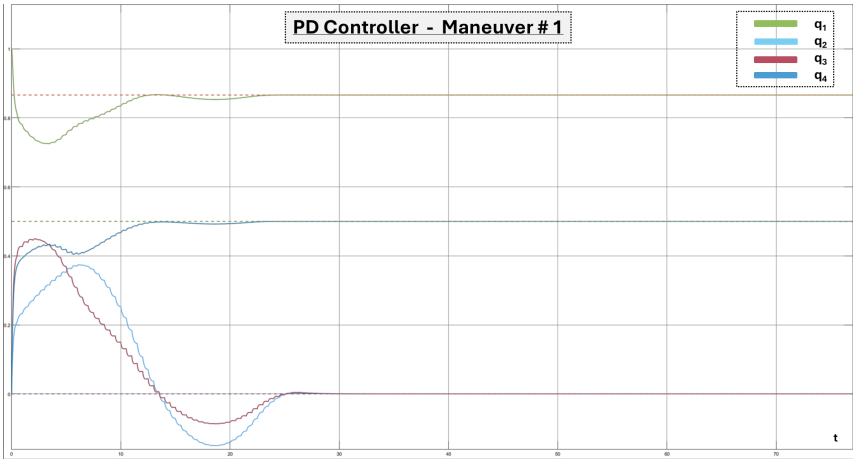

Figure 23: PD controller results for maneuver 1

Secondly, the LQR controller result is presented, which regulates the angles, however, not reaching the reference signal leaving an offset. The error indexes are:

$$IAE = 2.854 \times 10^{-15}$$

$$ITAE = 1.029 \times 10^{-13}$$

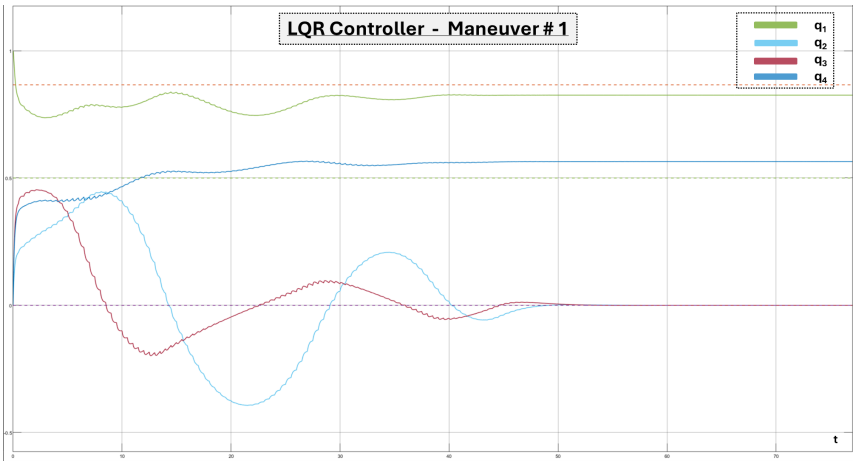

Figure 24: LQR results for maneuver 1

Afterwards, the AMPC developed with the Simulink Toolbox is shown, where it can be seen that the set point was reached but due to the aggressive tuning, it keeps a small oscillation in the zero angles.The error indexes are:

$$IAE = 2.532 \times 10^{-15}$$

$$ITAE = 9.55 \times 10^{-14}$$

Finally, the programmed AMPC results are presented, including the error indexes. This controller shows a good performance reaching the reference signal without hovering.

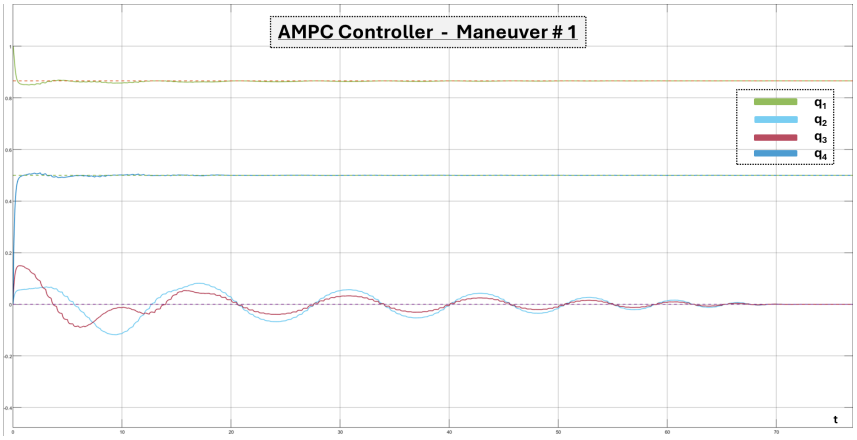

Figure 25: AMPC results for maneuver 1

$$IAE = 2.26 \times 10^{-15}$$

$$ITAE = 7.57 \times 10^{-14}$$

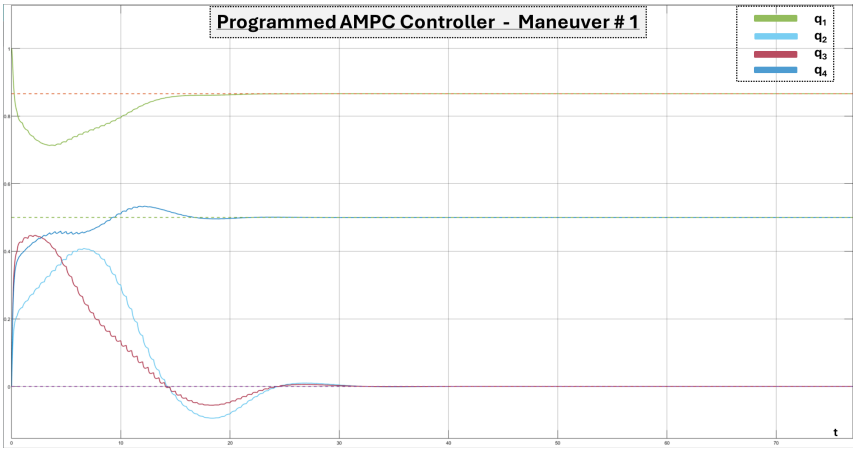

Figure 26: Programmed MPC results for maneuver 1

5.2 Maneuver 2: PID, LQR, AMPC comparison

The second maneuver consists of a sine wave that is the reference for the z axis rotation angle in body frame. It simulates a scanning movement where the CubeSat is oscillating around a chosen area.

The PD controller results show a good and fast performance, however, it is not able to reach the reference signal:

$$IAE = 3.033 \times 10^{-15}$$

$$ITAE = 1.19 \times 10^{-13}$$

Similarly, the LQR controller show an stable performance that is not able to reach the set point desired:

$$IAE = 2.7 \times 10^{-15}$$

$$ITAE = 1.04 \times 10^{-13}$$

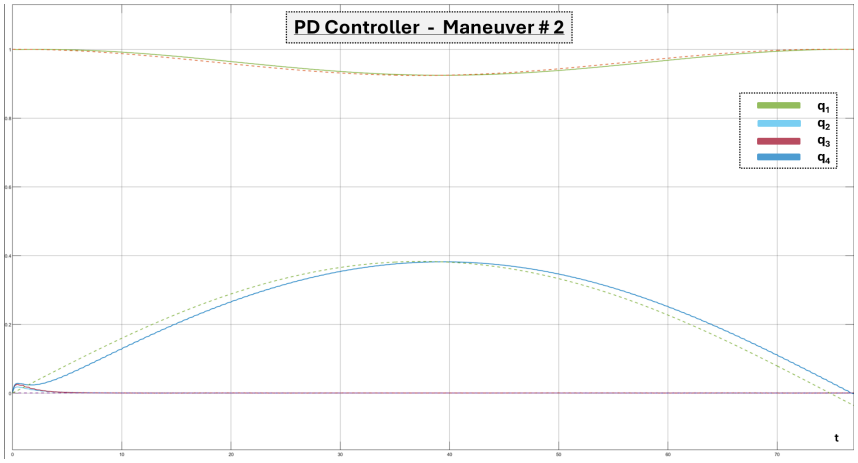

Figure 27: PD controller results for maneuver 2

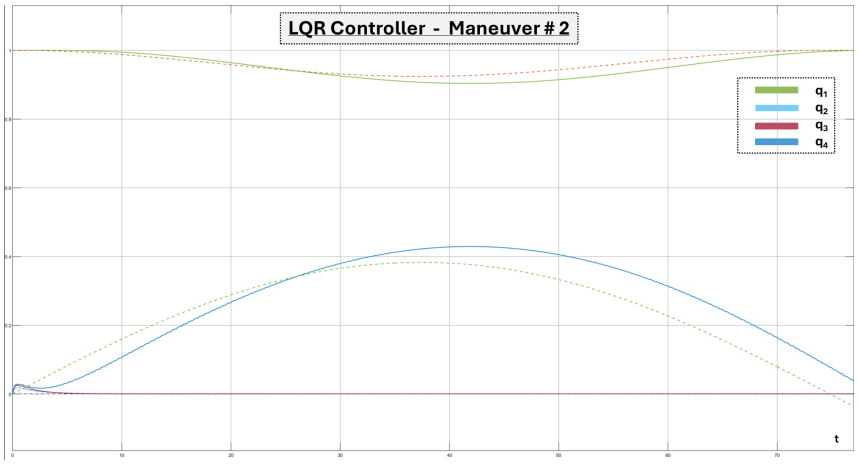

Figure 28: LQR results for maneuver 2

On the other hand, the AMPC from Matlab toolbox performs well, achieving the reference values in a relatively fast performance:

$$IAE = 2.98 \times 10^{-15}$$

$$ITAE = 1.15 \times 10^{-13}$$

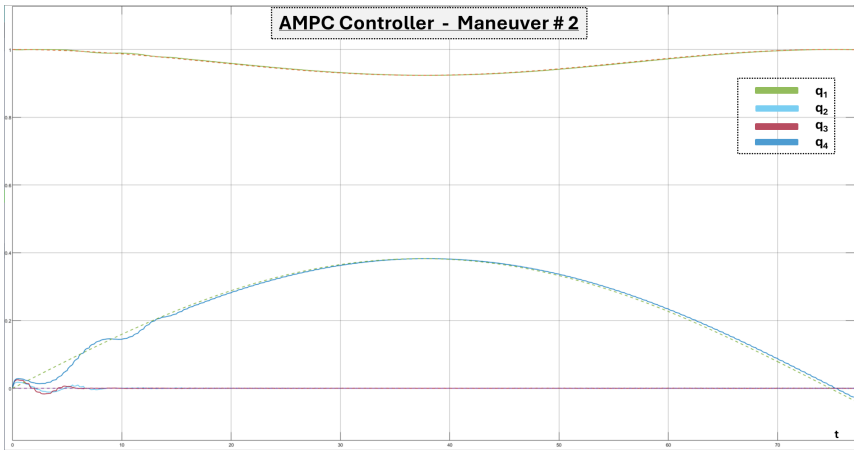

Figure 29: AMPC results for maneuver 2

Lastly, the programmed AMPC is able to regulate the angle, however, does not reach the desired set point.

$$IAE = 3.04 \times 10^{-15}$$
$$ITAE = 1.18 \times 10^{-13}$$

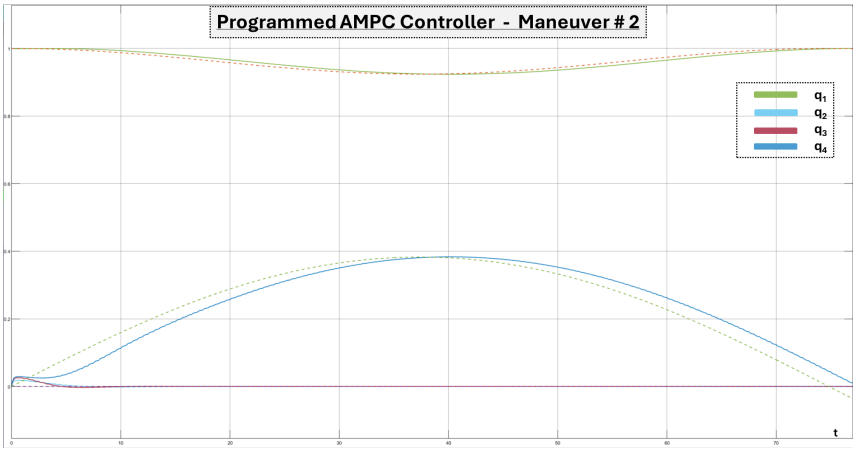

Figure 30: Programmed MPC results for maneuver 2

5.3 Maneuver 3: PD, LQR, AMPC comparison

The third and last maneuver is a ramp function that simulates a rotating movement with constant angular speed. This aiming to confirm the presence of space debris in areas of interest detected with the maneuver 2.

The PD controller performance is given by:

$$IAE = 3.111 \times 10^{-15}$$
$$ITAE = 1.37 \times 10^{-13}$$

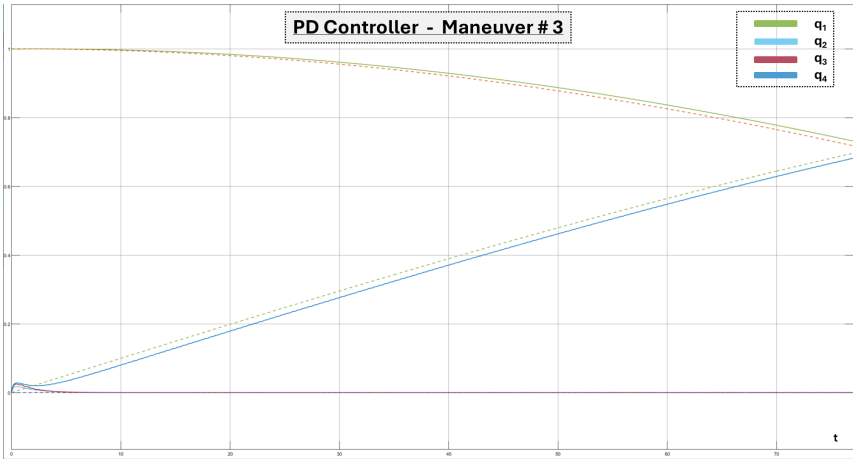

Figure 31: PD controller results for maneuver 3

Where the controller is not able again to reach the desired value for the angle. The LQR controller presents a similar result:

$$IAE = 3.21 \times 10^{-15}$$

$$ITAE = 1.4 \times 10^{-13}$$

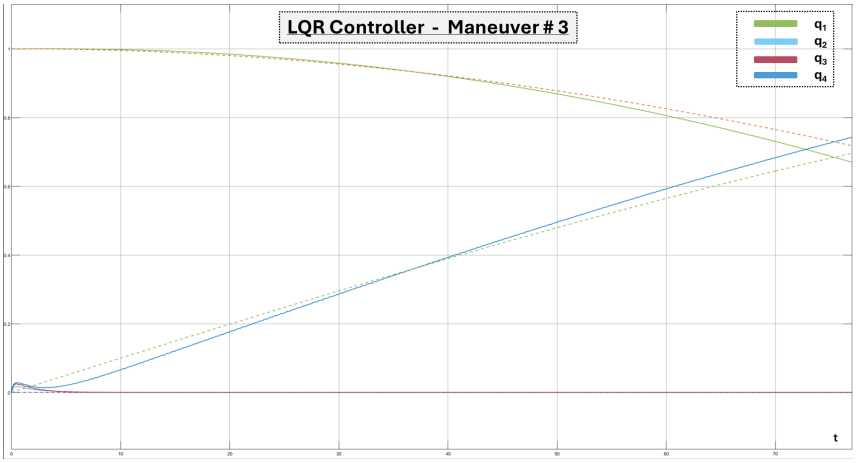

Figure 32: LQR results for maneuver 3

Unlike the previous controllers, the AMPC from the Simulink toolbox is able to control the attitude of the satellite to the given set point and maintain the regulation:

$$IAE = 3.4 \times 10^{-15}$$

$$ITAE = 1.52 \times 10^{-13}$$

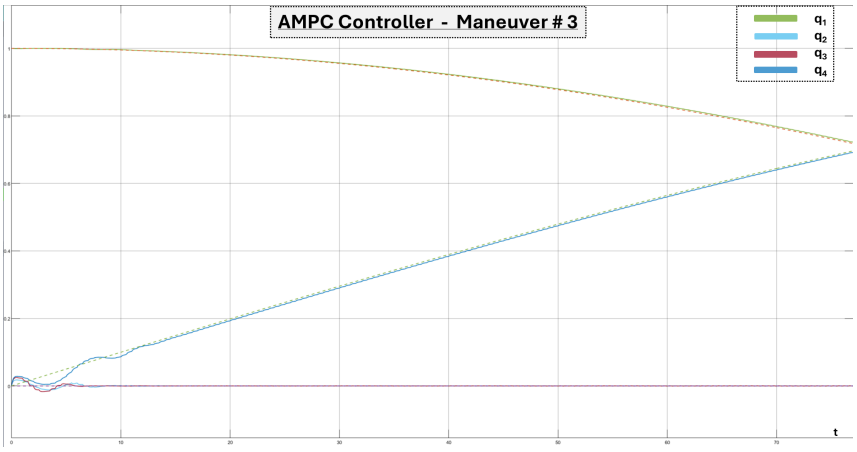

Figure 33: AMPC results for maneuver 3

Finally, the programmed AMPC regulates the signal but is not able to reach the desired value as the other AMPC controller is:

$$IAE = 3.41 \times 10^{-15}$$

$$ITAE = 1.5 \times 10^{-13}$$

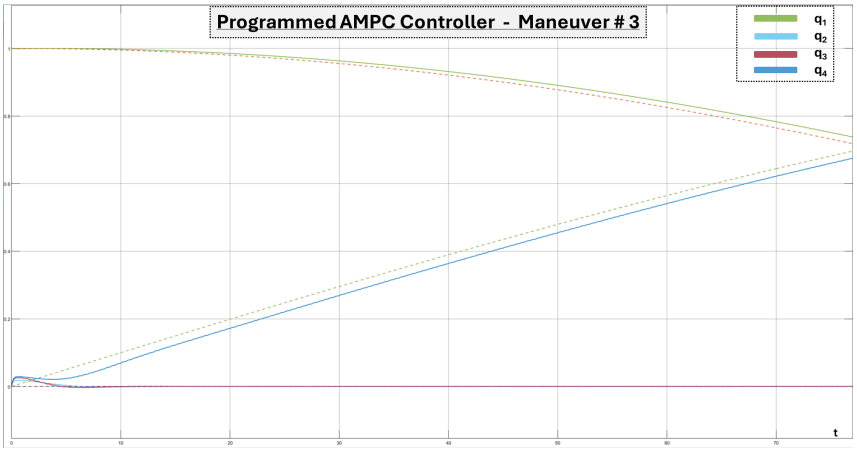

Figure 34: Programmed MPC results for maneuver 3

6 Discussion

This chapter reviews the results provided in the previous results section. It is structured per maneuver since it allows a better comparison of the performance of each controller relative to the given task. Additionally, the satellite performance is evaluated as a function of the actuators and estimator performances.

6.1 Maneuver 1: PD, LQR, AMPC comparison

- Pros: For this scenario, the AMPC controllers show a competitive performance with respect to the LQR and the PD controllers. Especially, the programmed AMPC shows the best response, achieving the reference signal with less overshoot and error.
- Cons: The Matlab Toolbox AMPC shows a very aggressive performance for which it might be required to implement a different tuning for this task.
- Challenges: Find a way to reduce the aggressiveness of the AMPC toolbox controller
- Possible improvements: Improve the tuning for the programmed AMPC to reduce the overshoot of the signal since it can cause saturation on the actuators.

6.2 Maneuver 2: PD, LQR, AMPC comparison

- Pros: For the second mission, the most accurate performance is given by the toolbox AMPC, being the only signal to reach the reference values.
- Cons: The programmed AMPC shows a delay between the angle and the reference signal.
- Challenges: Re-tune the AMPC programmed controller in order to remove the delay.
- Possible improvements: The programmed AMPC uses a constraint in the control signal that might be causing the delay with respect to the toolbox AMPC that does not include such constraint.

6.3 Maneuver 3: PD, LQR, AMPC comparison

- Pros: The last maneuver presents a similar case as the second one, being the toolbox AMPC the only controller to reach the reference signal.
- Cons: The programmed AMPC shows a delay as in the second maneuver.
- Challenges: Re-tune the AMPC programmed controller in order to remove the delay.
- Possible improvements: One way of improving the performance of the AMPC controller is by the usage of exponential weighted design parameters for Q and R, focusing more on the current values than those which are further in the control and prediction horizon. This because the model implemented does not include certain parameters as noise and perturbations causing the predicted model to not be as accurate as it is needed.

6.4 Actuator and estimator performances

- Pros: The actuators and the estimator implemented were consistent enough to allow the four control strategies to regulate the signals.
- Cons: However, the fact that the actuators have a limited torque and saturation issues causes the controllers to be more tricky to tune, as it was the case of the toolbox AMPC, that showed an initial good performance during the design that was not able to be replicated on the system once the actuator models were applied. Moreover, even tho the estimator showed an accurate performance, it is still needed to find a way to simplify the mathematical models required for its implementation, especially since the AMPC controllers require already a large usage of the microcontrollers memory.
- Challenges: Find a more effective way to deal with saturation and torque limits on the CubeSats. Look for an alternative estimator that is less dependant on the space models as the TRIAD is.
- Possible improvements: Implement a more advance internal control strategy for the actuator desaturation that contemplates the reaction wheels saturation before it actually occurs.

7 Conclusion

- The main objective of this master thesis is to design an optimal attitude control that is suitable for a 2U CubeSat and supplied with three debris detection maneuvers. This goal was accomplished, not only designing one optimal control strategy, but also by designing three alternative controllers that function as a reference point to evaluate the behaviour.
- Furthermore, the specific objectives were fulfilled by the implementation of the simulator developed, which includes also an attitude determination module with two alternatives.
- The system developed in this thesis takes into account the delimitation that was stated from the beginning of the thesis development, implicating the usage of the proposed hardware, being a controller able to be implemented with commercial solutions, and including the simulation parameters that help to create a more accurate environment for the performance test.
- The controllers that were designed as a point of reference for the AMPC confirm that the implementation of optimal model predictive architectures are valuable options to work with and not necessarily impossible to apply in real scenarios.
- Even though the solution provided achieve an acceptable performance, there is still room for improvement by the usage of more advanced models both for control and attitude determination, as it is also important to consider the real implementation leading to the need of focus on the programmed AMPC controller which is the most suitable to be tested in a microcontroller.

8 Future work

In this chapter, the future goals with the project are presented. While the designed controller and the simulator showed to be working properly, there are certain activities that can be done in order to improve the whole performance. These are divided in the following categories:

8.1 Controller upgrade

There are some upgrades that can be implemented in the AMPC that can improve the performance of the controller from the design. This by implementing exponential weighted AMPC, consisting of the selection of the matrices Q_{mpc} and R_{mpc} not to be constant as the controller described in section 4.2.2, but with a factor α that causes an exponential increase or decrease on the values the further they are inside of the control and prediction horizons. This can be used to decrease the impact of the predicted responses in the current time step control signal. Additionally, as it is stated in the reference [31], asymptotic closed-loop stability is guaranteed even in large prediction horizons by using this approach.

Moreover, the weights of the programmed controller can be more accurately tuned in order to obtained a closer response to the Matlab toolbox, and also more advanced prediction models and optimizer constrains can be used to improve performance.

8.2 Prototype test

- Hardware parameters update based on elements included in real prototype:

  The simulator is currently working with actuator and sensor parameters that were taken from data-sheets and manufacturer manuals from commercial models. However, in real implementation it needs to be adjusted to the parameters of the elements that are bought for the prototype.

- C program

  From the programmed prediction-optimizer approach of the MPC it is possible to create a code in C language that can work in a microcontroller. This can be helpful for future prototype tests.

8.3 Launch test

Lastly, the UiT projects QBDebris and UNICube aim to launch satellite prototypes from the Andøya Space Center. This satellites can be used as the plant where the developed controller can be tested performing the maneuvers designed in the chapter 4.1.

References

[1] Štefan Kozák, "State-of-the-art in control engineering," *Journal of Electrical Systems and Information Technology*, vol. 1, no. 1, pp. 1–9, 2014.

[2] C. Riano-Rios, R. Sun, R. Bevilacqua, and W. E. Dixon, "Aerodynamic and gravity gradient based attitude control for cubesats in the presence of environmental and spacecraft uncertainties," *Acta Astronautica*, vol. 180, pp. 439–450, 2021.

[3] S. M. Amrr, A. Banerjee, and M. Nabi, "Fault-tolerant attitude control of small spacecraft using robust artificial time-delay approach," *IEEE Journal on Miniaturization for Air and Space Systems*, vol. 1, no. 3, pp. 179–187, 2020.

[4] M. A. Silva, M. Shan, A. Cervone, and E. Gill, "Fuzzy control allocation of microthrusters for space debris removal using cubesats," *Engineering Applications of Artificial Intelligence*, vol. 81, pp. 145–156, 2019.

[5] M. F. Shehzad, A. B. Asghar, M. H. Jaffery, K. Naveed, and Z. Čonka, "Neuro-fuzzy system based proportional derivative gain optimized attitude control of cubesat under leo perturbations," *Heliyon*, vol. 9, no. 10, p. e20434, 2023.

[6] K. Kamalaldin, M. Atallah, M. Okasha, and A.-H. Jallad, "Design and testing of attitude determination and control subsystem for alainsat-i: 3u cubesat," in *IGARSS 2023 - 2023 IEEE International Geoscience and Remote Sensing Symposium*, pp. 257–260, 2023.

[7] K. Madhana Mohan, U. Anitha, and K. Anbumani, "Cubesat attitude control by implementation of pid controller using python," in *2023 12th International Conference on Advanced Computing (ICoAC)*, pp. 1–5, 2023.

[8] M. Helmy, A. T. Hafez, and M. Ashry, "Cubesat reaction wheels attitude control via modified pi-d controller," in *2022 International Telecommunications Conference (ITC-Egypt)*, pp. 1–6, 2022.

[9] A. T. Santoso, M. R. Rosa, and Edwar, "Model reference adaptive control design for cubesat with magnetorquer," in *2023 International Conference on Computer Science, Information Technology and Engineering (ICCoSITE)*, pp. 117–122, 2023.

[10] M. Tassano, P. Monzon, and J. Pechiar, "Attitude determination and control system of the uruguayan cubesat, antelsat," in *2013 16th International Conference on Advanced Robotics (ICAR)*, pp. 1–6, 2013.

[11] L. Franchi, L. Feruglio, R. Mozzillo, and S. Corpino, "Model predictive and reallocation problem for cubesat fault recovery and attitude control," *Mechanical Systems and Signal Processing*, vol. 98, pp. 1034–1055, 2018.

[12] X. Zhang, K. Ling, Z. Lu, X. Zhang, W. Liao, and W. Lim, "Piece-wise affine mpc-based attitude control for a cubesat during orbital manoeuvres," *Aerospace Science and Technology*, vol. 118, p. 106997, 2021.

[13] R. Sutherland, I. Kolmanovsky, and A. R. Girard, "Attitude control of a 2u cubesat by magnetic and air drag torques," *IEEE Transactions on Control Systems Technology*, vol. 27, no. 3, pp. 1047–1059, 2019.

[14] H. Bano and B. Sajid, "Attitude estimation control of a cubesat using linear quadratic gaussian approach," in *2021 Seventh International Conference on Aerospace Science and Engineering (ICASE)*, pp. 1–5, 2021.

[15] M. J. Sidi, *Spacecraft Dynamics and Control: A Practical Engineering Approach*. Cambridge Aerospace Series, Cambridge University Press, 1997.

[16] D. Vallado and W. McClain, *Fundamentals of Astrodynamics and Applications*. Fundamentals of Astrodynamics and Applications, Microcosm Press, 2001.

[17] E. Thébault, C. C. Finlay, C. D. Beggan, P. Alken, J. Aubert, O. Barrois, F. Bertrand, T. Bondar, A. Boness, L. Brocco, E. Canet, A. Chambodut, A. Chulliat, P. Coïsson, F. Civet, A. Du, A. Fournier, I. Fratter, N. Gillet, B. Hamilton, M. Hamoudi, G. Hulot, T. Jager, M. Korte, W. Kuang, X. Lalanne, B. Langlais, J.-M. Léger, V. Lesur, F. J. Lowes, S. Macmillan, M. Mandea, C. Manoj, S. Maus, N. Olsen, V. Petrov, V. Ridley, M. Rother, T. J. Sabaka, D. Saturnino, R. Schachtschneider, O. Sirol, A. Tangborn, A. Thomson, L. Tøffner-Clausen, P. Vigneron, I. Wardinski, and T. Zvereva, "International Geomagnetic Reference Field: the 12th generation," *Earth, Planets and Space*, vol. 67, p. 79, May 2015.

[18] T. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control*. Wiley, 2021.

[19] H. Schaub and J. Junkins, *Analytical Mechanics of Space Systems*. AIAA education series, American Institute of Aeronautics and Astronautics, Incorporated, 2014.

[20] M. Eshagh and M. Najafi Alamdari, "Perturbations in orbital elements of a low earth orbiting satellite," *J. Earth Space Phys.*, vol. 33, pp. 1–12, 01 2007.

[21] H. J. Rim, S. Yoon, and B. E. Schultz, "The glas algorithm theoretical basis document for precision orbit determination (pod)," tech. rep., 2013.

[22] B. Tapley, B. Schutz, J. Ries, and C. Shum, "Precision orbit determination for topex," *Advances in Space Research*, vol. 10, no. 3, pp. 239–247, 1990.

[23] L. Larsson, "Design of spacecraft attitude determination system using mems sensors," 2016.

[24] S. T. E. M. Brembo, "Sensor modeling, attitude determination and control for microsatellite," *Norweg. Univer. of Scien. and Technol*, p. 80, 2005.

[25] E. Oland and R. Schlanbusch, "Reaction wheel design for cubesats," in *2009 4th International Conference on Recent Advances in Space Technologies*, pp. 778–783, 2009.

[26] R. Wisniewski, "Satellite attitude control using only electromagnetic actuation," 1997.

[27] R. de Carvalho, J. Estela, and M. Langer, *Nanosatellites: Space and Ground Technologies, Operations and Economics*. Wiley, 2020.

[28] P. A. Capo-Lugo, J. Rakoczy, and D. Sanders, "The b-dot earth average magnetic field," *Acta Astronautica*, vol. 95, pp. 92–100, 2014.

[29] P. Narkhede, S. Poddar, R. Walambe, G. Ghinea, and K. Kotecha, "Cascaded complementary filter architecture for sensor fusion in attitude estimation.," *Sensors (Basel, Switzerland)*, vol. 21, Mar 2021.

[30] D. Kirk, *Optimal Control Theory: An Introduction*. Dover Books on Electrical Engineering Series, Dover Publications, 2004.

[31] L. Wang, *Model Predictive Control System Design and Implementation Using MATLAB®*. Advances in Industrial Control, Springer London, 2009.

A Simulator Interface Guide

In order to test the optimal controllers designed for this master thesis, a Satellite Simulator was developed. It was programmed in Matlab Simulink software, and includes:

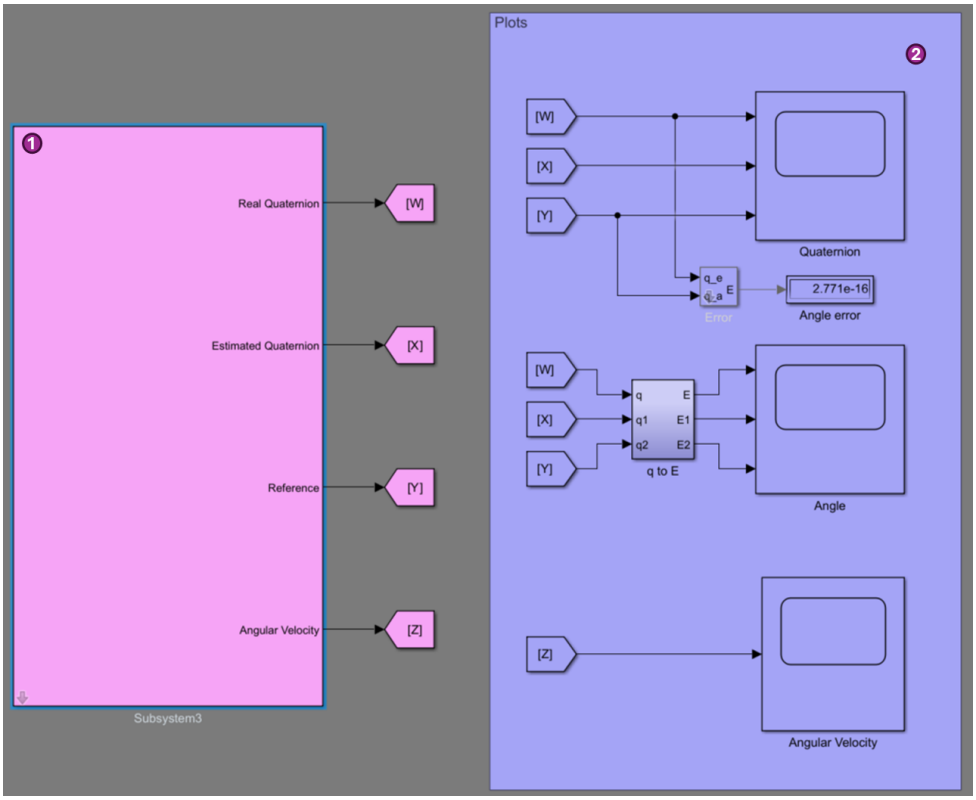

Figure 35: Satellite Simulator

And internally contains:
Including:

1. The system where the simulator is being processed, contains the mask to chose the simulation parameters.
2. A plotting area including a graph for quaternion, euler angles in radians, and the angular velocity. Additionally, it includes the block where the error indexes are calculated.
3. The space environment including the orbit equations, the perturbations, and the sun and magnetic models.
4. The satellite dynamics
5. The ADCS module where the actuator and sensor models is developed, the estimator and set points are produced and lastly, the controllers are processed.

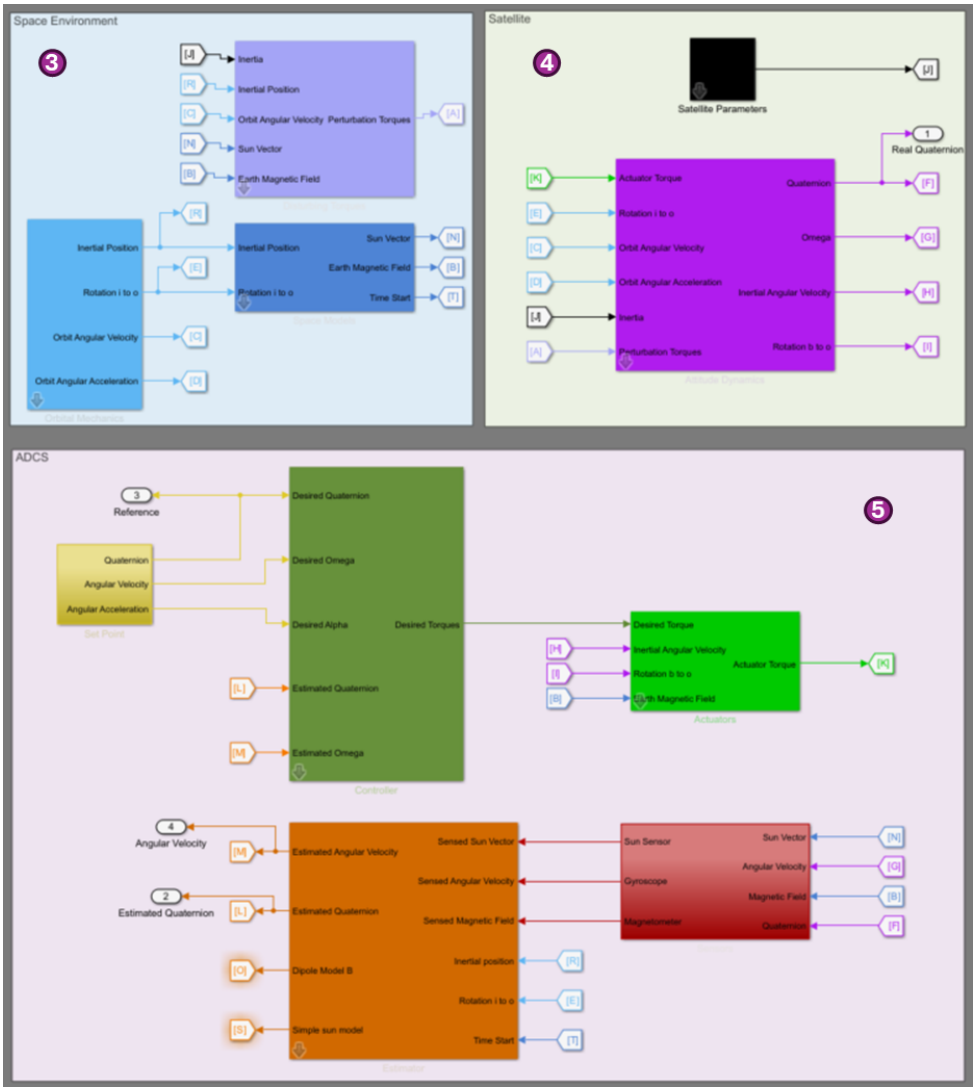

Figure 36: Satellite Simulator Subsystem

For more information regarding the simulator interface and its features, there is a YouTube video that describes its functionality on the following link:

https://youtu.be/AQcjjfraRTc

Finally, the simulation files are also attached containing the Simulink model and the initial script to run the simulator. The steps to run the simulation are:

- Run the Sim_init.m script
- Select the simulation parameters desired
- run the Simulink model

B Abstract for IAC 2024 congress

One of the side goals of the Masters Thesis development was to publish an article with the project results and progress. On February 2024, an abstract was submitted to the International Astronautical Federation for the 75th International Astronautical Congress to be hosted in Milan, Italy. The following paper was proposed for the Astrodynamics Symposium in the Attitude Dynamics session:

CubeSat attitude control and maneuver design for space debris detection in low earth orbit

Space debris has become an important topic, and several national and international space agencies are currently studying space debris detection and removal to increase the safety of space missions. UiT the Arctic University of Norway in Narvik is aiming to develop a 2U CubeSat that can perform in-situ detection of debris. The mission target is to detect mm-sized debris that cannot be detected from groundbased systems; detection will be done by radar technology. The main challenge of the mission relies on the small size and high velocity of the objects, for which the satellite must have the capacity for quick reaction in the pointing angle with a precise attitude controller without spending a high cost in energy. The main goal of this work is to develop novel attitude maneuvers for debris detection. The satellite is programmed with three procedures. Debris detection scanning: The control system was designed with high relative importance on the state's performance and moderate input penalty. This allows the radar to scan while the CubeSat is pointing to a desired angle. However, due to the high speed of the objects, false positive detections exist. Validation process: The detected objects will be scanned inside the radar region, and it is possible to approximate the position in which the debris can be spotted after the first detection. The debris validation maneuver gives the satellite the ability to perform more aggressive maneuvers by lowering the penalty on the input changes. This maneuver is designed for angular trajectory tracking in which the CubeSat performs a controlled rotation that allows the radar to maintain the object inside of the detection volume for a longer period, obtaining more information regarding the debris. Charging process: This procedure is developed for solar panels to maximize energy acquisition using a high input penalty, minimizing energy costs. Achieving adequate attitude control significantly impacts the performance of the detection tasks. Furthermore, it is vital to consider the perturbations that the LEO environment can introduce to the satellite, such as drag, gravitational accelerations, and solar pressure. A Model Predictive Controller will be implemented to obtain a cost-effective result with disturbance robustness, improve the transient response, and offset error. Additionally, a nonlinear cascade complementary filter is executed for attitude determination, performed with a gyroscope, sun-sensors, and 3-axis magnetometers. Simulations will be carried out to evaluate the performance of the control module, including reaction wheels and magnetorquers based on commercial models.