POLITECNICO DI TORINO

MASTER's Degree in MECHATRONIC ENGINEERING

MASTER's Degree Thesis

NMPC with Physics-Informed Neural Network Residual Dynamic Compensation under Lyapunov-Based Safety Supervision for Underactuated Spacecraft Attitude Control

Supervisors

prof. Marcello CHIABERGE

PhD. Candidate Carlo CENA

Candidate

Amirhossein AYANMANESH MOTLAGHMOFRAD

March 2026

NMPC with Physics-Informed Neural Network Residual Dynamic Compensation under Lyapunov-Based Safety Supervision for Underactuated Spacecraft Attitude Control

Amirhossein AYANMANESH MOTLAGHMOFRAD

Abstract

This thesis addresses the attitude maneuvering and stabilization problem of underactuated spacecraft equipped with fewer than three independent internal actuators. Such systems exhibit fundamental controllability and stabilization limitations and are particularly sensitive to modeling errors and environmental disturbances. To tackle these challenges, a control architecture is proposed that combines a nonlinear model predictive controller (NMPC), a physics-informed neural network (PINN)–based residual torque compensator, and a Lyapunov-based supervisory safety mechanism.

The NMPC serves as the baseline controller and is responsible for constraint handling, underactuation-aware maneuver planning, and nominal closed-loop stability under practical actuator limits. To mitigate performance degradation caused by model mismatch—primarily inertia uncertainty and unmodeled dynamic couplings—a PINN is trained offline to estimate residual disturbance torques using simulation data. Physics-informed loss terms are incorporated to regularize learning and enforce consistency with rigid-body rotational dynamics. The learned correction is not applied directly; instead, a Lyapunov-based supervisory layer evaluates its admissibility in real time and limits or suppresses its influence to preserve safety and stability properties of the baseline controller.

The proposed architecture is implemented and validated in a high-fidelity simulation environment using the Basilisk framework, accounting for reaction wheel dynamics, actuator saturation, environmental disturbances, and computational constraints. Monte Carlo simulations demonstrate that the learning-augmented controller achieves statistically significant reductions in steady-state attitude error compared to standalone NMPC, while maintaining robust behavior under uncertainty. The supervisory mechanism ensures graceful degradation to purely model-based control when learning-based augmentation becomes unreliable. Overall, the results show that physics-informed learning can enhance predictive control performance in underactuated spacecraft without compromising safety, provided it is embedded within a structured, supervision-driven control architecture.

Acknowledgments

I am about to finish another chapter of my life, and here I take the opportunity to express my gratitude to the people who set me on this path or supported me along the way. In other words, I want to mention the treasures I have found so far.

First and foremost, I want to thank my parents, who did everything they could and supported me throughout my immigration process, which, to say the least, was far from easy. I am also deeply grateful for the mindset they passed on to me. Without such a lantern, finding a meaningful direction in today's world of appearances would have been far more difficult.

Special thanks go to the Bena family, who gave me the experience of having a family here in Italy. Since my very first months in the country, they always made me feel part of their family by welcoming me into their gatherings and ceremonies. Federico was the first friend I found here. He taught me Italian blasphemy, which I now use on a daily basis, and helped me understand the Italian language, culture, and history. He may not remember it, but I vividly remember the first coffee we had together at Politecnico di Torino. I have never been good at expressing gratitude, but he should know that he holds a special place in my life. Marinella, Fede's mother, always welcomed me into her home and treated me with the same attention and consideration reserved for family members. Valentina and Alessandro, as well as Nonna, Zia, and Zio, did the same. Through them, I experienced a sense of belonging that I will always associate with family.

Another family that holds a special place in my heart is the Faranna family. Federico is like a younger brother to me. Among all our memories, two stand out the most: the first is "Moleee, Moleee," and the second, which you may not remember, is when we were running back to your car after Beach Discotec while it was softly raining, in that moment, I felt truly free and happy. Finally, Claudio and Domenica, who always welcomed me with warmth and kindness.

Special thanks go to Reza Rahmani, an old friend of mine from my bachelor's days, who supported me greatly in tough times—especially during my first days in Turin. I share many memories with him from our undergraduate years, and I was fortunate to continue creating new ones together here in Italy.

I am deeply thankful for the friends I have found along the way. I want to thank Isacco Ceri, also known as Salame or Fino, a friend with a deep personality with whom I spent countless hours discussing profound topics; Gregorio Valenti, who motivated and pushed me to do better at university when I had little desire to do so; Simone Carletti, a friend with a remarkably balanced life; and Filippo, who was there for me during difficult moments. I am also grateful to Marco Chiriacò (also

known as Marco Omber), Rea, Fabio, Alice, Beni, Miele, and Leonardo Degl'Innocenti.

I would also like to thank the large group of informatics students who occasionally included me in their activities. In particular, special thanks go to Alessandra and Francesca Alta, who have been kind to me since the beginning of our acquaintance.

All the people I have mentioned, as well as those I may have neglected to mention, have been, in their own way, my Italian teachers, helping me speak the Italian I speak today, and I am thankful for their patience.

I would like to express my sincere gratitude to Professor Marcello Chiaberge for always being available and for his constant support throughout my master's program. His encouragement accompanied me not only during my studies but also continued beyond graduation, supporting me in the first steps of my post-graduation career. I am truly grateful for the time and attention he consistently dedicated to me.

Finally, I am equally thankful to my co-supervisor, Carlo Cena, who guided me throughout my thesis work. Thanks to his mentorship, I was able to learn tools and develop skills that I once believed would be impossible to master within the scope of a thesis project. He pushed me to grow both academically and professionally.

Oscar Wilde once wrote: *"Nowadays people know the price of everything and the value of nothing."* I, in my turn, appreciate more than anything the invaluable gestures they have made and the moments we have spent together. I know that time is the cure for every pain, yet it fades memories and friendships in the long run. I hope the meaningful friendships I have made endure and pass the test of time.

Table of Contents

List of Figures

List of Tables

Acronyms

CMG	Control Moment Gyro.
RW	Reaction Wheel.
STLC	Short-Time Locally Controllable.
LEO	Low Earth Orbit.
PD	Proportional–Derivative.
LQR	Linear Quadratic Regulation.
NMPC	Nonlinear Model Predictive Control.
PINN	Physics-Informed Neural Network.
EKF	Extended Kalman Filter.
RK4	Fourth-Order Runge–Kutta.
NARX	Nonlinear Autoregressive Model with Exogenous Inputs.
RMSE	Root-Mean-Square Error.
CLF–QP	Control Lyapunov Function–Quadratic Program.
DARE	Discrete Algebraic Riccati Equation.
NLP	Nonlinear Program.
IPOPT	Interior Point OPTimizer.

L-BFGS	Limited-memory Broyden–Fletcher–Goldfarb–Shanno.
CPU	Central Processing Unit.
FAP	Force-Application Point.

Chapter 1

Overview of Underactuated Spacecraft Attitude Control

1.1 Model-Based Control Theory Context

Spacecraft dynamic control is commonly divided into two largely independent problems: orbital control and attitude control. Orbital control concerns the translational motion of a spacecraft along its trajectory, whereas attitude control addresses its rotational motion. The attitude of a spacecraft, which is the concern of this work, describes the orientation of its body-fixed frame with respect to a chosen reference frame and is fundamental to almost all space missions. Typical applications include orienting thrusters for orbital maneuvers and corrections, Earth and deep-space observation, and communication alignment between satellites and ground stations, among others. An example of such applications is shown in Fig. 1.1.

In practical spacecraft systems, the number of available actuators may be insufficient to directly control all rotational degrees of freedom. Such systems are referred to as underactuated, a condition that can arise either by design or as a result of actuator limitations or failures. Underactuation introduces additional challenges in attitude control, as full authority over the rotational dynamics is not always available.

From a control-theoretic perspective, the analysis of underactuated spacecraft requires a clear characterization of the actuation system, as the nature of the actuators directly influences the attitude dynamics. In particular, actuators can be classified according to their ability to generate either internal or external control torques. The primary actuators employed in spacecraft attitude control systems include control moment gyros (CMGs), reaction wheels (RWs), and thruster pairs. Thrusters generate external torques through propellant expulsion, whereas CMGs and reaction wheels produce internal control torques by redistributing angular momentum within the spacecraft. In this work, the spacecraft under study is equipped with two reaction wheels, which are selected due to their capability to provide high-precision attitude control.

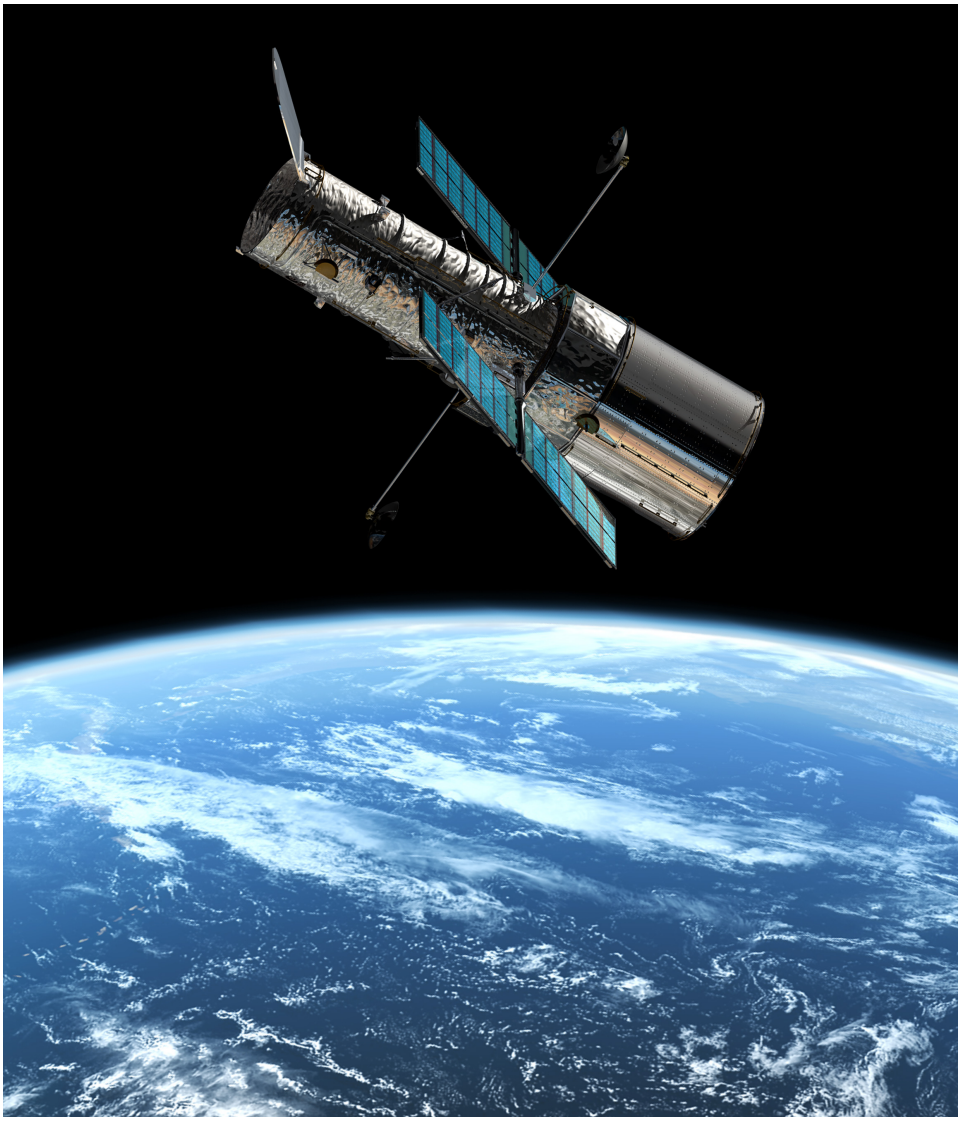

Figure 1.1: The Hubble Space Telescope, illustrating a spacecraft mission with stringent attitude pointing and stabilization requirements imposed by high-resolution imaging payloads. Image Source: [1].

The control of underactuated spacecraft is particularly challenging when only actuators generating internal torques are available. This difficulty stems from a fundamental conservation principle governing the rotational dynamics of rigid bodies, namely the conservation of angular momentum. In the absence of external torques, a rigid body tends to preserve its rotational state, which imposes inherent limitations on achievable attitude maneuvers. In particular, fundamental accessibility limitations arise in spacecraft attitude dynamics when fewer than three reaction wheels are available, as a consequence of angular momentum conservation [2]. From a control design standpoint, this limitation directly impacts the feasibility of feedback laws that rely on full-state authority and motivates alternative formulations based on reduced-order dynamics.

To circumvent this fundamental limitation, two main approaches have been adopted in the literature. The first assumes zero total angular momentum, while the second exploits an external torque along the underactuated direction, for instance through solar radiation pressure, to restore controllability. Under the zero–total–angular–momentum assumption, Krishnan *et al.* [3] showed that a spacecraft actuated by two reaction wheels is short-time locally controllable (STLC) for a reduced attitude dynamics, allowing rest-to-rest maneuvers between admissible attitude configurations. Alternatively, recent studies have demonstrated that actively utilizing environmental torques, such as solar radiation pressure, in combination with reaction wheels can recover three-axis attitude control in underactuated spacecraft without relying on angular momentum conservation assumptions [4].

Although not focus of this work, there is merit in exploring the use of external, predictable sources of disturbance for missions in which environmental disturbances cannot be neglected. As discussed in more detail in the following chapter, the uncontrollable state of an underactuated spacecraft—namely the component of the rotational angular momentum orthogonal to the plane spanned by the available reaction wheels—may drift over time due to external disturbances. Such disturbances include aerodynamic torques, impacts from small debris, and torques induced by solar radiation pressure. In the absence of a predictable and exploitable external torque, guaranteeing continued mission operation in disturbance-dominated scenarios would require the use of additional actuators, such as propulsive devices. However, the propellant allocated to such actuators is typically budgeted for orbital corrections rather than sustained attitude regulation, making this solution undesirable for long-duration missions.

While the zero–total–angular–momentum assumption has enabled important theoretical developments, its direct applicability to a broad class of practical missions—particularly those operating in Earth orbit—is limited. In realistic environments, external disturbances cannot be entirely eliminated, and over sufficiently long time horizons they may induce a gradual drift in the underactuated direction. As a result, control strategies based exclusively on the zero–total–angular–momentum assumption require additional considerations to remain applicable in the presence of disturbances. In this work, a problem setting of practical relevance is considered. The analysis is conducted under the following assumptions:

- The drift in the underactuated direction remains negligible over the time horizon of interest. This assumption is justified by the operational orbit considered in this study, as discussed in the following chapter.
- Only rest-to-rest maneuvers are addressed. For the underactuated direction, this implicitly assumes the existence of a mechanism—either passive or active—that mitigates accumulated drift when it becomes non-negligible. Given the first assumption, such drift is expected to evolve on a slow time scale.

Under these conditions, results derived under the zero–total–angular–momentum assumption can be meaningfully applied and interpreted within a realistic operational context.

Having established the control-theoretic context of the problem, the following section reviews learning-based approaches relevant to spacecraft attitude control.

1.2 Learning-Based Approaches to Spacecraft Attitude Control

In recent years, machine learning techniques, and artificial intelligence more broadly, have attracted significant attention. In several application domains, these methods have been shown to offer solutions to problems for which classical approaches are difficult or, in some cases, infeasible to implement. Motivated by this growing interest and by the demonstrated effectiveness of learning-based techniques, researchers have increasingly turned their attention to their potential application in space technology and spacecraft control. At present, such approaches are predominantly based on artificial neural networks and are employed to assist researchers and mission designers in computing optimal trajectories and in developing adaptive feedback control laws that may be suitable for onboard spacecraft implementation [5].

Among the applications of learning-based methods in spacecraft attitude control reported in the literature are [5]:

- Adaptive neural-network-based approaches for spacecraft attitude control. [6]
- Neural-network as an assistance to the main controller. [7]
- Neural-network-based optimal attitude control using four impulsive thrusters, shown to achieve lower total impulse consumption compared to logic-based and projective control strategies for comparable settling times [8].
- Physics-informed normalizing flow for learning attitude dynamics [9]

Physics-Informed Neural Networks (PINNs) incorporate governing physical laws directly into the learning process of neural networks. Classical PINN formulations are primarily designed for state prediction at fixed time instants, which limits their direct applicability to control and planning tasks that require short-horizon system evolution. To address this limitation, alternative hybrid strategies have been proposed, including approaches that couple physics-based models with data-driven learning components, as well as methods that augment data-driven losses with physics-based regularization terms enforcing consistency with the governing equations [9].

While learning-based approaches provide flexibility in pattern recognition and scalability, their application in safety-critical systems remains challenging [9]. A

significant body of work proposes neural-network-based controllers for spacecraft attitude control without providing accompanying closed-loop stability guarantees. In several cases, the adoption of neural networks as control-law approximators is motivated by their well-known theoretical properties—such as universal approximation capability, generalization, and robustness—yet these properties are often assumed rather than rigorously demonstrated within the specific control context [5]. Moreover, stability and reliability analyses are essential for space applications, including the assessment of controller behavior under potential neuron failures. Advancing the field requires moving beyond idealized benchmark problems toward realistic scenarios that explicitly account for onboard hardware and software constraints; nevertheless, such considerations remain largely underrepresented in the existing literature [5].

1.3 Contribution of this work

The aim of this work is to propose a hybrid model-based and learning-based approaches such that benefiting from the advantages of each method, and at the same time, it evades the weakpoints of them when implemented standalone.

In this work, a model-based baseline controller is implemented to provide the theoretical guarantees required for mission-critical and safety-critical operation, under the assumptions introduced at the end of the previous section. This baseline controller is subsequently augmented by a physics-informed neural network that acts as a residual torque compensator, with the objective of improving performance in the presence of underactuation and modeling uncertainties that cannot be fully addressed by model-based control alone. To mitigate risks associated with the learning-based component, a deterministic Lyapunov-based supervisory layer is introduced to constrain the influence of the neural network on the closed-loop system. This supervisory structure is consistent with the concept of a *safety-cage* architecture, as discussed in the ECSS Machine Learning Handbook [10].

Chapter 2

Problem Formulation and Control Architecture

2.1 Control Problem Definition

2.1.1 Mission Context and Operational Scenario

This thesis addresses the attitude maneuvering and stabilization problem of an underactuated spacecraft operating in Low Earth Orbit (LEO). The reference use-case is representative of missions requiring accurate pointing and rest-to-rest attitude maneuvers despite limited actuation authority and persistent environmental disturbances. Underactuation arises either by design (mass/power/cost constraints) or as a fault-tolerant operational mode following actuator degradation.

2.1.2 Attitude State, Reference Frames, and Notation

Let $\mathcal{N}$ denote an inertial reference frame and $\mathcal{B}$ the spacecraft body-fixed frame attached to the rigid hub. The attitude of $\mathcal{B}$ with respect to $\mathcal{N}$ is represented using a unit quaternion $q \in \mathbb{R}^4$, while the angular velocity of the body with respect to the inertial frame, expressed in the body frame, is denoted by $\omega \in \mathbb{R}^3$.

The overall control-oriented state is defined as

$$x \triangleq \begin{bmatrix} q \\ \omega \\ \omega_{RW} \end{bmatrix}, \tag{2.1}$$

where $\omega_{RW} \in \mathbb{R}^m$ denotes the reaction wheel angular velocities relative to the body-fixed frame, and m represents the number of reaction wheels.

The control input is defined as the vector of commanded reaction wheel motor torques

$$u \triangleq u_{RW} \in \mathbb{R}^m, \tag{2.2}$$

with $m < 3$ corresponding to the underactuated configuration.

The symbol τ is used for the residual disturbance that appear in the Euler equation. where

$$\tau \in \mathbb{R}^3$$

includes the residual torque along the bases of the body fixed frame $\mathcal{B}$.

In the following, additional notation is introduced for the control and learning layers employed in the proposed architecture.

The desired attitude trajectory is denoted by

$$q_{\text{ref}}(t) \in \mathbb{R}^4,$$

with corresponding reference angular velocity $\omega_{\text{ref}}(t)$. The attitude tracking error is defined using quaternion composition as

$$q_e \triangleq q^{-1} \otimes q_{\text{ref}} = \begin{bmatrix} q_{e,s} \\ q_{e,v} \end{bmatrix},$$

where $q_{e,s} \in \mathbb{R}$ and $q_{e,v} \in \mathbb{R}^3$ denote the scalar and vector parts of the error quaternion, respectively.

For nonlinear model predictive control, the continuous-time dynamics are written in compact form as

$$\dot{x} = f(x, u) + d(x, t),$$

where $f(\cdot)$ represents the nominal control-oriented model used for prediction, and $d(x, t)$ denotes a lumped disturbance and modeling error term. Time discretization with sampling period T_s yields the discrete-time prediction model

$$x_{k+1} = f_d(x_k, u_k),$$

where k denotes the discrete-time index along the prediction horizon.

The NMPC problem is formulated over a finite horizon of length N, with predicted state and input sequences denoted by

$$\{x_k\}_{k=0}^{N}, \qquad \{u_k\}_{k=0}^{N-1}.$$

The NMPC cost function employs weighting matrices

$$Q \succeq 0, \qquad R \succ 0, \qquad P \succeq 0,$$

corresponding to the stage cost on state tracking error, control effort, and terminal

state penalty, respectively.

The physics-informed neural network (PINN) is used to approximate the residual disturbance torque acting on the rigid-body rotational dynamics. Its output is denoted by

$$\hat{\tau}_{\text{PINN}} \in \mathbb{R}^2,$$

which represents an estimate of the true residual torque τ. The PINN is parameterized by a vector of trainable weights θ, and its input is constructed from the measured or estimated system state and control input.

The final control action applied to the plant is obtained by combining the nominal NMPC torque command with the learning-based correction through a supervisory mechanism. A scalar modulation factor

$$\alpha \in [0, 1]$$

is introduced to represent the action of the Lyapunov-based supervisor, such that

$$\tau_{\text{corr}} = \alpha\, \hat{\tau}_{\text{PINN}},$$

with $\alpha = 0$ corresponding to full rejection of the learning-based correction and $\alpha = 1$ corresponding to full acceptance.

Throughout the remainder of this work, vectors are assumed to be column vectors, $\|\cdot\|$ denotes the Euclidean norm, and $(\cdot)^\top$ denotes matrix transpose.

2.1.3 Underactuated Actuation Configuration

The spacecraft is actuated by a cluster of reaction wheels generating internal control torques. In the configuration of interest, only two independent wheel axes are available. For simplicity and without loss of generality, these axes are assumed to be aligned with the body-fixed directions $\hat{\mathbf{x}}_B$ and $\hat{\mathbf{y}}_B$, yielding an underactuated system with no direct torque authority about $\hat{\mathbf{z}}_B$.

This actuation structure implies that attitude control must be achieved either by eliminating the tracking error before reaching the reference plane perpendicular to the underactuated axis, or by exploiting coupled nonlinear dynamics and momentum exchange. The latter mechanism is, in practice, orders of magnitude weaker than the direct control torques available along the actuated axes and cannot be independently or arbitrarily commanded.

The actuator dynamics include wheel momentum build-up and saturation. The simulated wheel limits used as representative constraints are:

- maximum wheel speed $\omega_{RW,\max} = 6000$ rpm,
- maximum commanded motor torque $\|u_{RW}\|_\infty \leq 0.05$ N m,
- maximum stored wheel momentum on the order of 0.9 N m s (used to size wheel inertia).

These limits are treated as hard constraints in the control problem formulation. In the simulations, Coulomb friction is also considered; its modeling and implementation are discussed in the chapter dedicated to the simulation framework.

2.1.4 Control Objectives: Regulation, Maneuvering, and Pointing

The primary objective is to achieve rest-to-rest attitude maneuvers and/or attitude regulation to a desired reference attitude $q_{\text{ref}}(t)$ with low residual quaternion error and angular velocity. The target behavior is expressed as

$$q(t) \to q_{\text{ref}}(t), \qquad \omega(t) \to 0, \tag{2.3}$$

while respecting actuator and operational constraints.

In addition to nominal regulation, the architecture is designed to:

- maintain acceptable performance in the presence of disturbances and model mismatch;
- avoid unsafe behavior induced by learning-based components by enforcing a supervisory safety condition;
- degrade gracefully to a purely model-based closed loop when learning-based correction is unreliable.

2.1.5 Constraints and Practical Operating Limits

The control problem is subject to constraints arising from actuator saturation, reaction wheel momentum storage, rate limitations, and computational requirements. These constraints can be summarized as follows:

$$\|u_{RW}(t)\|_\infty \leq u_{\max}, \tag{2.4}$$
$$|\omega_{RW,i}(t)| \leq \omega_{RW,\max}, \qquad i = 1, \ldots, m, \tag{2.5}$$
$$|h_{RW,i}(t)| \leq h_{RW,\max}, \qquad i = 1, \ldots, m, \tag{2.6}$$
$$\|\Delta u_{RW}(t)\|_\infty \leq \Delta u_{\max}, \tag{2.7}$$

where $u_{\max}$ denotes the maximum admissible reaction wheel motor torque, $\omega_{RW,\max}$ the maximum allowable wheel angular speed, and $h_{RW,i} = J_{RW,i}\,\omega_{RW,i}$ the angular momentum stored in the i-th reaction wheel, with $J_{RW,i}$ its rotational inertia. The quantity $\Delta u_{RW}(t) = u_{RW}(t) - u_{RW}(t - T_s)$ represents the discrete-time control input

variation, bounded to reflect actuator slew-rate limitations.

In addition to actuator-related constraints, attitude pointing performance is enforced by requiring the steady-state attitude error to remain within a prescribed tolerance. Using the quaternion error representation, the settling angle is defined as

$$\theta_s = 2 \arccos(q_s), \tag{2.8}$$

where q_s denotes the scalar part of the attitude error quaternion. The settling requirement is imposed as

$$\theta_s \leq 0.5^\circ, \tag{2.9}$$

ensuring convergence to a neighborhood of the reference attitude consistent with mission pointing accuracy requirements.

In addition to physical constraints, real-time feasibility imposes a limit on the computational burden of the controller. In particular, the maximum solver execution time is constrained not to exceed 80% of the controller sampling period,

$$t_{\text{CPU}} \leq 0.8\, T_{\text{controller}}, \tag{2.10}$$

ensuring that the control law can be computed reliably within the available onboard computational resources.

From a practical standpoint, these constraints are not treated as implementation details but as defining characteristics of the control design problem, motivating the adoption of a control strategy capable of systematic constraint satisfaction.

2.1.6 Uncertainties and Disturbance Model

In the considered scenario, the spacecraft attitude dynamics are affected by external disturbance torques of practical relevance in low Earth orbit (LEO). The plant model adopted throughout this work explicitly includes gravity-gradient torque and aerodynamic drag torque, which are treated as the dominant environmental disturbances at the considered altitude. Other effects, such as solar radiation pressure and magnetic torque, are regarded as secondary and are therefore neglected. This modeling choice is consistent with the order-of-magnitude comparison of environmental disturbance torques in LEO reported in the following figure, taken from [11].

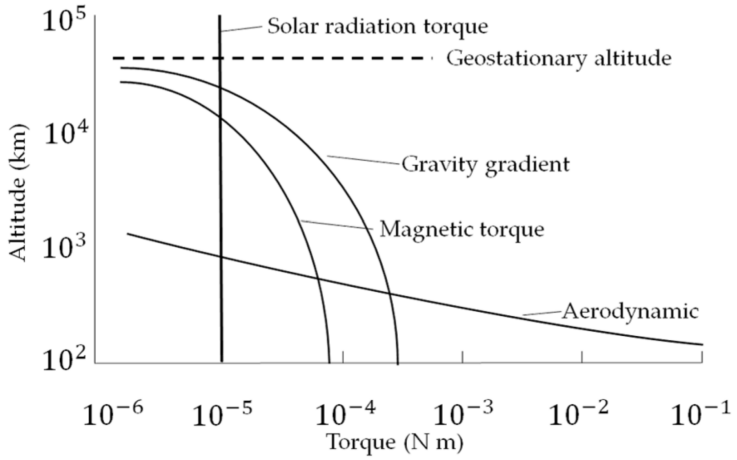

Figure 2.1: Order-of-magnitude comparison of external disturbance torque sources as a function of orbital altitude. Taken from [11].

In addition to these modeled disturbances, the spacecraft is subject to modeling uncertainties and unmodeled effects that are difficult to represent exactly in the nominal control model or are simplified to reduce computational burden. These include uncertainty in the spacecraft inertial matrix, actuator non-idealities, unmodeled dynamic couplings, as well as residual components of external disturbance torques.

For control design purposes, all such uncertainties and disturbances are grouped into a lumped residual term, leading to the control-oriented dynamics

$$\dot{x} = f(x, u) + d(x, t), \tag{2.11}$$

where $f(\cdot)$ denotes the nominal nonlinear dynamics used for prediction, and $d(x, t)$ represents the aggregate effect of modeling errors and external disturbances.

Within the proposed control architecture, a learning-based module is employed to estimate and compensate a component of this residual term, while a deterministic supervisory layer explicitly constrains its influence to preserve closed-loop stability and robustness.

2.1.7 Assumptions and Scope of Applicability

The scope of this work is defined by the following assumptions:

- **Momentum management capability:** it is assumed that a mechanism exists to regulate or dissipate the total angular momentum associated with the underactuated direction (e.g., through propulsors, magnetic torquers or equivalent means), preventing long-term momentum accumulation due to reaction wheel faults or drift.
- **Near-zero total angular momentum regime:** the spacecraft is assumed to operate in a near-zero total angular momentum regime. This assumption,

which is commonly adopted in the literature on underactuated spacecraft controllability and stabilization, is practically justified by the availability of the aforementioned momentum management mechanism.

- **Underactuated direction drift:** drift along the underactuated direction may be present but remains sufficiently small over the maneuver time scales of interest, such that stabilization and maneuver objectives remain meaningful.
- **Rest-to-rest maneuvers:** the primary focus is on maneuvering and regulation tasks in which the desired terminal angular velocity is close to zero.
- **Disturbance envelope:** gravity-gradient and aerodynamic drag torques are considered the dominant environmental disturbances under the operating conditions examined in this work, consistent with the orbital regime considered in the thesis.
- **Safety-first architecture:** learning-based compensation is treated as an auxiliary component and is not allowed to override stability or safety constraints; a supervisory mechanism is included to enforce a Lyapunov-based safety condition.

These assumptions define the intended applicability domain of the proposed architecture and guide both controller formulation and validation.

2.1.8 Formal Statement of the Control Problem

Given the nonlinear underactuated attitude dynamics

$$\dot{x}(t) = f\big(x(t), u(t)\big) + d\big(x(t), t\big), \tag{2.12}$$

with initial condition $x(t_0) = x_0$, the objective is to determine a control input $u(\cdot)$ such that the following requirements are satisfied:

1. **Attitude tracking and regulation:** the spacecraft attitude tracks a desired reference $q_{\text{ref}}(t)$ while achieving low terminal angular velocity, i.e.,

$$q(t) \to q_{\text{ref}}(t), \qquad \omega(t) \to 0. \tag{2.13}$$

The steady-state pointing error is required to remain within a prescribed settling-angle tolerance. Defining the attitude error quaternion as

$$q_e(t) \triangleq q^{-1}(t) \otimes q_{\text{ref}}(t) = \begin{bmatrix} q_{e,s}(t) \\ q_{e,v}(t) \end{bmatrix}, \tag{2.14}$$

the associated settling angle is given by

$$\theta_s(t) \triangleq 2 \arccos(q_{e,s}(t)), \tag{2.15}$$

and the pointing requirement is expressed as

$$\limsup_{t\to\infty} \theta_s(t) \leq 0.5^\circ. \tag{2.16}$$

2. **Constraint satisfaction:** actuator and operational constraints are satisfied for all $t \in [t_0, t_f]$, i.e.,
$$u(t) \in \mathcal{U}.$$

3. **Robust stability and safety:** the closed-loop system remains stable and safe in the presence of bounded disturbances $d(x, t)$ and modeling uncertainty.

The above formulation does not uniquely prescribe a specific control architecture. Rather, it emphasizes the simultaneous need to address nonlinear dynamics, explicit constraints, performance degradation due to model mismatch, and safety under uncertainty. In this work, these requirements are addressed through a hierarchical control architecture, adopted as a deliberate design choice rather than a direct consequence of the problem setting. The proposed structure consists of: (i) a nominal model-based controller responsible for constraint and underactuated-aware stabilization, (ii) a learning-based residual compensator intended to mitigate model mismatch effects, and (iii) a deterministic supervisory layer that constrains the influence of the learning-based component through a Lyapunov-based safety condition.

2.2 Proposed Control Architecture

2.2.1 High-Level Block Diagram and Signal Definitions

The proposed control architecture is organized as a hierarchical, modular closed-loop structure composed of four main functional blocks: a nominal nonlinear model predictive controller (NMPC), a learning-based residual torque compensator implemented via a physics-informed neural network (PINN), a Lyapunov-based supervisory layer, and a state estimation module. A high-level representation of the architecture and its signal interconnections is shown in following figure.

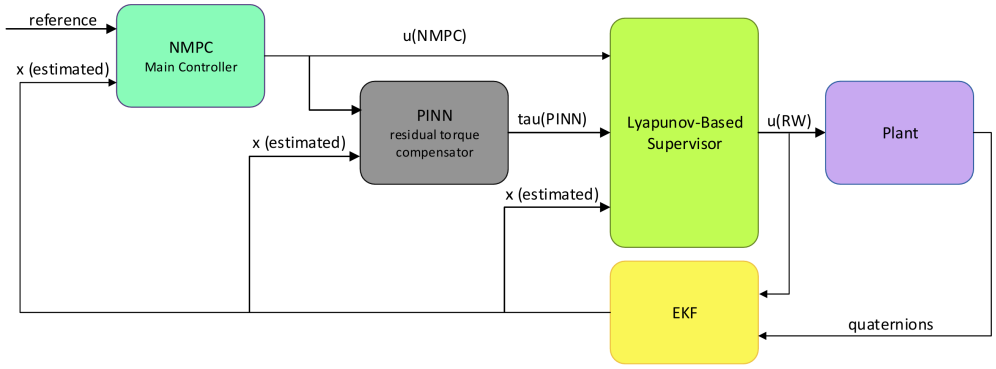

Figure 2.2: High-level block diagram representation of the proposed control system architecture for the underactuated spacecraft under study.

The plant represents the full nonlinear attitude and reaction wheel dynamics of the underactuated spacecraft, subject to external disturbances and modeling uncertainty. State estimates of attitude, angular velocity, and reaction wheel states are assumed available to all control modules. In simulation, estimation uncertainty is emulated by injecting measurement noise with statistics representative of EKF-level performance.

The NMPC block receives the estimated state and a reference attitude trajectory and computes a nominal reaction wheel torque command u_{NMPC}. This command is generated using a reduced-order, control-oriented prediction model and explicitly accounts for actuator and operational constraints.

In parallel, the learning-based module processes the same estimated state and outputs a corrective torque signal τ_{PINN}, interpreted as an estimate of the residual torque arising from modeling errors and unmodeled dynamics. This signal is not applied directly to the plant.

Instead, the nominal NMPC command and the candidate residual correction are combined through a Lyapunov-based supervisory layer. The supervisor evaluates

the admissibility of the learning-based correction in real time and produces the final applied reaction wheel torque command u_{RW}, limiting the influence of learning-based augmentation to ensure compliance with a prescribed safety condition.

2.2.2 Nominal Layer: NMPC as Baseline Policy

The nominal control layer is built around nonlinear model predictive control and serves as the primary authority within the proposed architecture. Its role is to generate a constraint-aware control policy based on a physics-based prediction model of the spacecraft dynamics.

The selection of the baseline controller is discussed in detail in the following chapter, where several candidate control strategies are analyzed and compared. NMPC is ultimately chosen as the baseline controller due to its ability to explicitly handle nonlinear dynamics, actuator saturation, reaction wheel momentum limits, and underactuation within a unified optimization framework. The adopted NMPC formulation is designed to ensure nominal closed-loop stability and constraint satisfaction in the absence of learning-based augmentation.

From an architectural standpoint, the NMPC layer is supposed to establish a well-defined baseline policy: if higher-level augmentation is disabled or rejected by the supervisory layer, the closed loop remains stable and constraint-compliant under the assumptions stated in the previous section.

2.2.3 Learning Layer: PINN as Residual Torque Compensator

The learning layer employs a physics-informed neural network to estimate and compensate for the mismatch between the predictive model used by the NMPC controller and the true spacecraft dynamics. This mismatch arises from modeling simplifications and parametric uncertainties and is treated as an additive residual torque acting on the rotational dynamics.

In the present formulation, the dominant source of model mismatch is associated with uncertainty in the spacecraft inertia properties. Specifically, the NMPC prediction model assumes a diagonal inertia matrix, whereas the true plant inertia may contain off-diagonal terms and parameter errors due to mass distribution uncertainty, unmodeled appendages, or payload variability. The resulting discrepancy manifests as an unmodeled torque contribution that degrades prediction accuracy and closed-loop performance.

The PINN is trained to approximate this residual torque using data generated from the difference between the NMPC predictive model and the true spacecraft behavior. A physics-informed loss term is incorporated during training to regularize the learning process using known properties of rigid-body rotational dynamics, thereby

discouraging non-physical corrections. The resulting PINN output is treated as a candidate residual correction and is evaluated by the supervisory layer prior to application.

2.2.4 Supervisory Layer: Lyapunov-Based Safety Filter

To constrain the influence of the learning-based component and preserve the stability properties of the baseline controller, a Lyapunov-based supervisory layer is introduced. This layer acts as a deterministic safety filter between the PINN output and the plant, regulating the contribution of the learned residual torque to the applied control input.

The supervisory layer does not generate control commands independently. Instead, it monitors the combined effect of the nominal NMPC input and the proposed learning-based correction on a candidate Lyapunov function associated with the baseline closed-loop system. A decrease condition on this Lyapunov function is enforced to ensure that the learning-based augmentation does not violate the stability and safety properties established by the nominal controller.

This mechanism provides a structured means of integrating learning-based performance enhancement while maintaining control-theoretic robustness and safeguarding against unreliable or non-physical neural network outputs.

2.2.5 Failure Modes and Graceful Degradation (Conceptual)

The proposed architecture is designed to degrade gracefully under failures or uncertainty in the learning layer. If the PINN produces unreliable outputs—due to poor generalization, numerical issues, or hardware-induced faults such as proton-induced memory corruption, to which neural network parameters tend to be particularly susceptible due to their large number compared to the nominal controller—the supervisory layer can suppress its influence entirely, reducing the control law to the nominal NMPC policy.

Similarly, if learning-based augmentation is disabled or unavailable, the closed-loop system remains well-defined and stable without requiring reconfiguration of the NMPC or state estimation modules. This property is particularly relevant for safety-critical aerospace applications, where learning-based components cannot be assumed reliable under all operating conditions. In the proposed architecture, performance improvements enabled by learning are optional, whereas stability and safety are mandatory.

2.3 Expected Advantages and Design Hypotheses (Pre-Validation)

As discussed previously, the ability of NMPC to address the underactuated attitude control problem stems from its capability to plan state trajectories over a finite horizon, coordinating angular-rate transients so as to reduce the attitude error component associated with the underactuated axis before convergence to the reference manifold. In this context, the accuracy of the internal prediction model plays a critical role: the predicted evolution of the state directly determines whether the optimizer can exploit nonlinear coupling and momentum exchange effectively along the horizon.

In the proposed formulation, the NMPC relies on a simplified control-oriented model that assumes a diagonal inertia matrix. While this approximation is sufficient to ensure nominal stability and constraint satisfaction, it is expected to introduce systematic prediction errors. These errors manifest as steady-state pointing offsets or residual oscillations, particularly along the underactuated direction, where convergence relies heavily on precise prediction of coupled dynamics.

The central design hypothesis of this work is that augmenting the NMPC with a learning-based residual torque predictor can reduce model-induced mismatch without increasing the complexity of the online optimization problem. By compensating a portion of the unmodeled dynamics at the torque level, the effective plant behavior is expected to align more closely with the prediction model employed by the NMPC. This improved model consistency is, in turn, expected to enhance trajectory prediction along the horizon and enable more effective exploitation of nonlinear coupling during maneuver planning.

2.3.1 Hypothesized Performance Gains from Residual Compensation

The introduction of a PINN-based residual torque compensator is hypothesized to improve closed-loop attitude regulation performance relative to the baseline NMPC controller. These hypotheses motivate the selection of attitude-error-based performance metrics and statistical comparison tools adopted later in this work.

- **Reduction of steady-state attitude error:** By compensating structured model mismatch, primarily associated with inertia uncertainty and unmodeled cross-couplings, the learned residual torque is expected to reduce systematic prediction bias in the NMPC model. This effect is hypothesized to manifest as a reduction in steady-state attitude error, quantified through median steady-state RMSE, median final attitude error evaluated over Monte Carlo simulations.
- **Systematic performance improvement across initial conditions:** If the residual compensation provides a meaningful correction rather than incidental

improvement for specific trajectories, its effect should be observable consistently across a population of initial conditions. This hypothesis motivates the use of paired statistical comparisons between controller variants, where nonparametric hypothesis testing (e.g., Wilcoxon signed-rank test) is employed to assess whether observed reductions in attitude error are statistically significant rather than attributable to random variation.

The learning module is not intended to fully cancel all disturbances or uncertainties. Instead, it is designed to compensate dominant, structured, and slowly varying components of model mismatch that degrade NMPC prediction accuracy. High-frequency or unstructured effects are intentionally left to be handled by the nominal NMPC feedback action, ensuring that stability and constraint satisfaction remain governed by the model-based controller.

Chapter 3

Control Formulation and Learning-Augmented Architecture

3.1 Baseline Controller Selection

In this section, a set of candidate attitude control strategies commonly employed in spacecraft applications is examined in the context of the underactuated attitude control problem considered in this work. The objective of this analysis is to systematically evaluate the suitability of both classical and nonlinear control approaches and to identify their limitations when applied to the present setting. Through this comparative assessment, the rationale for selecting nonlinear model predictive control (NMPC) as the baseline control strategy is established by excluding alternative approaches, including classical linear feedback methods (e.g., PD and LQR), robust and discontinuous controllers (e.g., sliding mode control), and linear or linear time-varying MPC formulations.

3.1.1 Limitations of Classical Smooth Time-invariant Feedback Control Laws (e.g., PD, LQR)

If the state is chosen as the small attitude error quaternion vector part and the body–rate error, i.e.

$$x \coloneqq \begin{bmatrix} \delta q_v \\ \delta\omega \end{bmatrix} \in \mathbb{R}^6, \qquad \delta q_v \approx \tfrac{1}{2}\,\delta\theta, \tag{3.1}$$

and the system is linearized around the equilibrium

$$q_v = 0_{3\times1}, \qquad \omega = 0_{3\times1}, \qquad u = 0_{2\times1},$$

with the rigid–body attitude dynamics expressed using the quaternion kinematic matrix $Q(q)$,

$$\dot{q} = \tfrac{1}{2}Q(q)\,\omega, \qquad J\dot{\omega} = u - \omega \times (J\omega), \tag{3.2}$$

where

$$q = \begin{bmatrix} qs \\ qx \\ qy \\ qz \end{bmatrix}$$

and

$$Q(q) = \begin{bmatrix} -q_x & -q_y & -q_z \\ q_s & -q_z & q_y \\ q_z & q_s & -q_x \\ -q_y & q_x & q_s \end{bmatrix}$$

the resulting linearized attitude error dynamics can be written in state–space form as

$$\dot{x} = Ax + Bu, \tag{3.3}$$

where

$$A = \begin{bmatrix} 0_{3\times3} & \frac{1}{2}I_3 \\ 0_{3\times3} & 0_{3\times3} \end{bmatrix}, \qquad B = \begin{bmatrix} 0_{3\times2} \\ J^{-1}B_{\text{act}} \end{bmatrix},$$

with the actuator mapping matrix

$$B_{\text{act}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}.$$

By examining the rank of the controllability matrix associated with the linearized system, it can be readily shown that the system is not fully controllable. In particular, due to the underactuated nature of the actuator configuration, control authority about the third body axis is absent at the linearized level. As a result, stabilization of the full attitude dynamics cannot be achieved using classical linear feedback controllers such as proportional–derivative (PD) control or linear quadratic regulation (LQR), regardless of the choice of feedback gains.

From a physical perspective, the controller attempts to drive the attitude error to zero only along the axes for which control authority is available. As a result, the feedback action actively reduces the error components associated with the actuated axes. Once the system reaches the reference manifold where the first two components of the quaternion error are driven to zero, the control input vanishes. Any residual error associated with the unactuated third axis is therefore preserved and cannot be further reduced. This remaining error persists due to the marginally stable nature of the unactuated rotational mode.

A Monte Carlo MATLAB simulation is used to illustrate this behavior. Considering a unit quaternion as the reference attitude, it can be observed that the quaternion vector component along the $\hat{z}$ direction does not converge to zero for almost all the simulations. It should be noted that reducing the error in the $\hat{z}$ direction would

require a specifically designed reference attitude that accounts for the cancellation of the third quaternion component upon reaching the reference manifold defined by the plane spanned by the actuated axes.

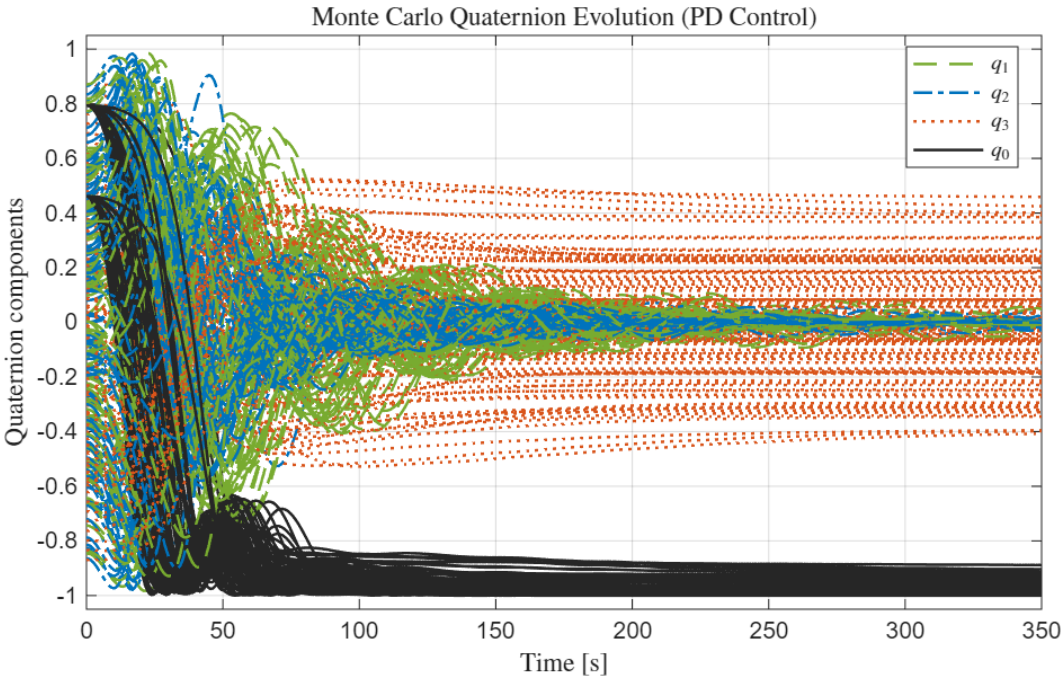

Figure 3.1: Monte Carlo quaternion evolution under PD control. One hundred initial conditions are considered: 50 directions uniformly distributed on a sphere, with rotations of 75° and 125° about each direction.

Furthermore, when the full nonlinear attitude dynamics are considered, the difficulty of the underactuated attitude control problem becomes more fundamental. Although the system is small–time locally controllable from rest equilibria when using thrusters, reaction wheels, or certain control moment gyroscope configurations, it is well known that the equations of motion cannot be asymptotically stabilized by any smooth, continuous, time–invariant state feedback law. This limitation arises because the spacecraft attitude dynamics violate Brockett's necessary condition for smooth stabilization [12].

3.1.2 Limitations of Robust/Discontinuous Sliding Mode Control Law

From a theoretical standpoint, discontinuous control laws, such as sliding–mode control, are capable of circumventing stabilization obstructions that arise for smooth, continuous, time–invariant feedback laws. In the presence of underactuation, however, the nonlinear attitude dynamics exhibit a relative degree that does not match the dimension of the state vector. As a result, the imposition of a sliding manifold does not, in general, guarantee that the surface is attractive. This loss of attractivity prevents the system trajectories from reaching the sliding manifold, thereby undermining the reaching phase that is essential for the effective operation of sliding–mode control.

Following the sliding–mode attitude tracking formulation presented by Novara [13], a sliding surface is defined by combining the angular velocity tracking error and the quaternion tracking error as

$$s(q, \omega, t) \coloneqq \tilde{\omega} + k_2 \tilde{q}, \tag{3.4}$$

where $\tilde{\omega} = \omega_r - \omega$ denotes the angular velocity tracking error and $\tilde{q}$ is the vector part of the quaternion tracking error. When the system trajectories evolve on the manifold $s = 0$, the tracking error converges asymptotically to zero.

Taking the time derivative of the sliding surface yields

$$\dot{s} = \dot{\omega}_r + J^{-1}\omega \times J\omega - J^{-1}u + \frac{k_2}{2}\left(\tilde{q}_0\tilde{\omega} + \tilde{q} \times (\omega_r + \omega)\right), \tag{3.5}$$

where the expression for $\dot{\tilde{q}}$ follows from the quaternion kinematic equations (see Appendix in [13]).

Imposing the invariance condition $\dot{s} = 0$ and solving the above expression with respect to the control input leads to the equivalent control law

$$u_s = J\left(\dot{\omega}_r + \frac{k_2}{2}\left(\tilde{q}_0\tilde{\omega} + \tilde{q} \times (\omega_r + \omega)\right)\right) + \omega \times J\omega. \tag{3.6}$$

To render the sliding surface attractive, an additional stabilizing term is introduced. The complete sliding–mode control law is therefore given by

$$u = u_s + k_1 J \tanh(\eta s), \tag{3.7}$$

where $k_1 > 0$ and $\eta > 0$ are design parameters that regulate convergence and reduce chattering.

Considering the underactuated setting, the final control input that can be applied to the system is obtained by projecting the nominal commanded torque onto the subspace of achievable torques. Let $u \in \mathbb{R}^3$ denote the nominal control torque computed by the controller (e.g., the equivalent sliding–mode input). The applied torque is then defined as

$$u_{\text{act}} = P\, u_s, \tag{3.8}$$

where $P \in \mathbb{R}^{3\times3}$ is the orthogonal projector onto the column space of the actuator mapping matrix $B_{\text{act}} \in \mathbb{R}^{3\times2}$, i.e.

$$P \coloneqq B_{\text{act}} B_{\text{act}}^{\dagger}, \qquad B_{\text{act}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}. \tag{3.9}$$

For the present actuator configuration, the projector reduces to

$$P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{3.10}$$

and therefore the applied torque satisfies $u_{\text{act},3} = 0$ for all times.

Since the third control component is zero, no attractive behavior can be enforced on the sliding surface in the (ω_z, q_z) state space. This can be seen in the following figure, which represent a Monte Carlo simulation.

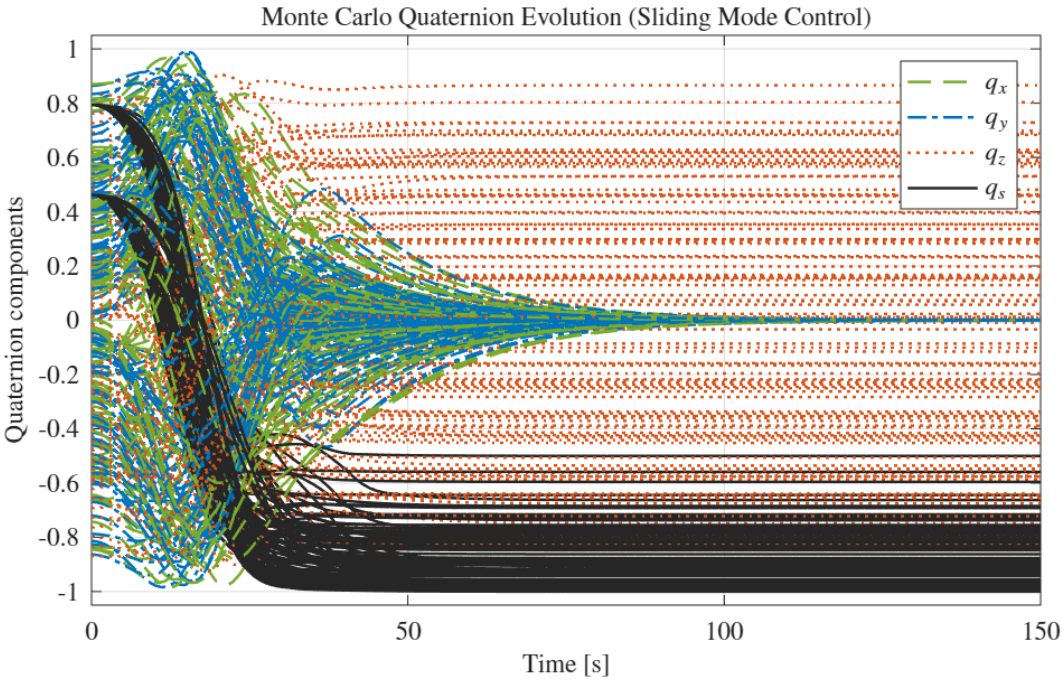

Figure 3.2: Monte Carlo quaternion evolution under SM control. One hundred initial conditions are considered: 50 directions uniformly distributed on a sphere, with rotations of 75° and 125° about each direction.

3.1.3 Rationale for Selecting Nonlinear Model Predictive Control

Nonlinear Model Predictive Control is selected as the baseline controller because it is the only examined strategy that can systematically handle nonlinear dynamics, hard constraints, and underactuation simultaneously, without relying on smooth stabilizing feedback or discontinuous control action.

Unlike classical linear controllers such as PD or LQR, which operate locally and rely on the controllability of the linearized dynamics, NMPC does not depend on local full controllability around the equilibrium. In the underactuated configuration considered here, a linear feedback law can only act on the components of the attitude and rate error that lie in the actuated subspace. Consequently, the controller drives the system toward a reference manifold defined by the actuated axes. Once the

trajectory reaches this manifold, the commanded torque naturally decays to zero and the closed loop loses the ability to further influence the remaining error component associated with the unactuated axis. As a result, any residual attitude error in the underactuated direction is preserved.

A similar limitation emerges when discontinuous control laws such as sliding-mode control are considered. Sliding-mode designs rely on enforcing a reaching condition by shaping the time derivative of the sliding variable, $\dot{s}$, so as to render the sliding manifold attractive. In the present underactuated setting, however, control authority is unavailable along one rotational axis, and therefore the corresponding component of $\dot{s}$ cannot be directly influenced. This prevents the enforcement of global attractivity of the sliding manifold, and the reaching phase cannot be guaranteed in the full state space. Once the projected error in the actuated subspace is eliminated, the control action collapses to zero along the available channels, leaving the residual dynamics associated with the unactuated direction weakly affected or unchanged.

NMPC overcomes these limitations by abandoning the notion of instantaneous manifold attractivity and instead planning the approach to the reference over a finite horizon. By optimizing the control sequence in time, NMPC can coordinate angular-rate transients and attitude evolution along the actuated axes so as to reduce—prior to arrival—the error component associated with the underactuated direction through nonlinear dynamic coupling and momentum exchange. Moreover, although the prediction model may be time-invariant, the receding-horizon implementation induces an implicitly time-varying, state-dependent feedback law, which can be interpreted as a locally linear time-varying (LTV) controller. This is a crucial property, as Brockett's necessary condition excludes global asymptotic stabilization of the spacecraft attitude dynamics by any smooth, continuous, time-invariant state feedback law in the presence of underactuation. Since NMPC does not belong to this class, the topological obstruction imposed by Brockett's condition does not apply, enabling stabilization and maneuvering through trajectory-dependent, time-varying control action.

Despite its advantages, nonlinear model predictive control is computationally demanding, as it requires the online solution of a nonlinear constrained optimization problem at each control update. In practice, however, this burden can be partially mitigated by exploiting warm-start strategies for the optimizer and by introducing auxiliary mechanisms, in this work PINN, that reduce model–plant mismatch, thereby keeping the system trajectory closer to the prediction model and improving numerical conditioning of the optimization problem.

3.2 Dynamics and Modeling for Control Design

In this section, the dynamic models used as the predictive model of the NMPC and, subsequently, the model including residual dynamics employed for PINN training are presented. The rigid-body dynamics formulation follows the modeling approach presented in [9].

3.2.1 Rigid-Body Attitude Kinematics

The spacecraft attitude is represented using a unit quaternion

$$\mathbf{q} \in \mathbb{S}^3 \subset \mathbb{R}^4, \qquad \mathbf{q} = \begin{bmatrix} q_s \\ q_x \\ q_y \\ q_z \end{bmatrix}, \qquad \mathbf{q}^\top \mathbf{q} = 1 \tag{3.11}$$

where q_s denotes the scalar part and $\mathbf{q}_v = [q_x, q_y, q_z]^\top$ the vector part. Let $\boldsymbol{\omega} = [\omega_x, \omega_y, \omega_z]^\top$ be the angular velocity of the spacecraft expressed in the body frame.

The quaternion kinematics are given by

$$\dot{\mathbf{q}} = \frac{1}{2}\,\boldsymbol{\Omega}(\boldsymbol{\omega})\,\mathbf{q}, \tag{3.12}$$

where

$$\boldsymbol{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & \omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_x & 0 \end{bmatrix}. \tag{3.13}$$

The unit-norm constraint

$$\|\mathbf{q}\| = 1 \tag{3.14}$$

is enforced numerically through explicit quaternion renormalization after each discrete-time integration step.

3.2.2 Rigid-Body Rotational Dynamics with Internal Actuation

The spacecraft is modeled as a rigid body equipped with n_u internal reaction wheels. Let $\mathbf{J} \in \mathbb{R}^{3\times 3}$ denote the spacecraft inertia matrix, $\mathbf{J}_{\text{RW}} = \text{diag}(J_{\text{RW},1}, \ldots, J_{\text{RW},n_u})$ the reaction wheel inertia matrix, and $\mathbf{B} \in \mathbb{R}^{3\times n_u}$ the wheel spin-axis matrix. The wheel angular velocities with respect to the spacecraft are collected in $\boldsymbol{\omega}_{\text{RW}} \in \mathbb{R}^{n_u}$, and the motor torques in $\mathbf{u}_{\text{RW}} \in \mathbb{R}^{n_u}$.

The total angular momentum of the reaction wheels expressed in the body frame is

$$\mathbf{h} = \mathbf{B}\,\mathbf{J}_{\text{RW}}\,\boldsymbol{\omega}_{\text{RW}}. \tag{3.15}$$

Assuming the absence of external torques, the coupled spacecraft–wheel dynamics are governed by

$$\begin{bmatrix} \mathbf{J} & \mathbf{B}\mathbf{J}_{\mathrm{RW}} \\ \mathbf{J}_{\mathrm{RW}}\mathbf{B}^{\top} & \mathbf{J}_{\mathrm{RW}} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{\omega}} \\ \dot{\boldsymbol{\omega}}_{\mathrm{RW}} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{\omega} \times (\mathbf{J}\boldsymbol{\omega} + \mathbf{h}) \\ \mathbf{u}_{\mathrm{RW}} \end{bmatrix}. \tag{3.16}$$

The spacecraft angular acceleration $\dot{\boldsymbol{\omega}}$ and the wheel accelerations $\dot{\boldsymbol{\omega}}_{\mathrm{RW}}$ are obtained by solving the above linear system at each time instant.

3.2.3 Discrete-Time Prediction Model (for NMPC Formulation)

For nonlinear model predictive control, the continuous-time dynamics

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \tag{3.17}$$

with state vector

$$\mathbf{x} = \begin{bmatrix} \mathbf{q}^{\top} & \boldsymbol{\omega}^{\top} & \boldsymbol{\omega}_r^{\top} \end{bmatrix}^{\top} \in \mathbb{R}^{7+n_u} \tag{3.18}$$

are discretized using a fixed-step fourth-order Runge–Kutta (RK4) scheme.

Given a sampling interval Δt, the one-step discrete-time prediction model is

$$\mathbf{x}_{k+1} = \Phi_{\mathrm{RK4}}(\mathbf{x}_k, \mathbf{u}_k), \tag{3.19}$$

where

$$\mathbf{k}_1 = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k), \tag{3.20}$$

$$\mathbf{k}_2 = \mathbf{f}\left(\mathbf{x}_k + \frac{\Delta t}{2}\mathbf{k}_1, \mathbf{u}_k\right), \tag{3.21}$$

$$\mathbf{k}_3 = \mathbf{f}\left(\mathbf{x}_k + \frac{\Delta t}{2}\mathbf{k}_2, \mathbf{u}_k\right), \tag{3.22}$$

$$\mathbf{k}_4 = \mathbf{f}(\mathbf{x}_k + \Delta t\, \mathbf{k}_3, \mathbf{u}_k), \tag{3.23}$$

and

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{\Delta t}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4). \tag{3.24}$$

To preserve a valid attitude representation, the quaternion component of $\mathbf{x}_{k+1}$ is renormalized after each integration step.

3.2.4 Disturbance and Uncertainty as a Lumped Residual Term

The nominal spacecraft attitude dynamics derived in the previous sections rely on exact knowledge of inertial properties, actuator models, and the absence of external torques. In practice, the true system dynamics are affected by multiple sources of uncertainty, including parametric mismatches in the spacecraft and reaction wheel inertias, unmodeled external disturbances, actuator imperfections, and numerical discretization errors.

Rather than modeling each uncertainty source explicitly, their combined effect is represented as a lumped residual disturbance torque acting on the rigid-body rotational dynamics. Accordingly, the coupled spacecraft–wheel dynamics including residual disturbances are expressed as

$$\begin{bmatrix} \mathbf{J} & \mathbf{B}\mathbf{J}_{\mathrm{RW}} \\ \mathbf{J}_{\mathrm{RW}}\mathbf{B}^{\top} & \mathbf{J}_{\mathrm{RW}} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{\omega}} \\ \dot{\boldsymbol{\omega}}_{\mathrm{RW}} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{\omega} \times (\mathbf{J}\boldsymbol{\omega} + \mathbf{h}) + \boldsymbol{\tau}_d \\ \mathbf{u}_{\mathrm{RW}} \end{bmatrix}, \tag{3.25}$$

where $\boldsymbol{\omega}$ denotes the spacecraft angular velocity, $\boldsymbol{\omega}_{\mathrm{RW}}$ the reaction wheel angular velocities, $\mathbf{u}_{\mathrm{RW}}$ the reaction wheel motor torques, and $\boldsymbol{\tau}_d \in \mathbb{R}^3$ is an unknown disturbance torque capturing all unmodeled effects.

The disturbance torque $\boldsymbol{\tau}_d$ acts directly on the rigid-body rotational dynamics and does not affect the internal wheel momentum balance. This formulation preserves the physical structure of the equations of motion while allowing the residual dynamics to be represented in a compact and model-agnostic manner.

In the present work, the dominant contribution to the residual disturbance torque $\boldsymbol{\tau}_d$ is attributed to inertial mismatches between the true spacecraft parameters and those employed in the predictive model, including uncertainties in the spacecraft inertia matrix and reaction wheel inertias. Additional contributions may arise from unmodeled external torques and numerical discretization effects; however, these are treated as secondary compared to inertial modeling errors. Accordingly, $\boldsymbol{\tau}_d$ is interpreted as the discrepancy between the true spacecraft dynamics and the nominal predictive dynamics used within the NMPC formulation.

The model including lumped residual dynamics serves as the physical backbone for training the Physics-Informed Neural Network (PINN), as discussed in subsequent sections. Specifically, the PINN is trained to approximate the unknown mapping

$$\boldsymbol{\tau}_d(k) = \mathcal{F}_\theta(\mathbf{x}(k),\, \mathbf{u}_{\mathrm{RW}}(k),\, \boldsymbol{\tau}_d(k-1))\,, \tag{3.26}$$

where $\mathcal{F}_\theta(\cdot)$ denotes a neural network parametrized by θ, and $\mathbf{x}$ and $\mathbf{u}_{\mathrm{RW}}$ denote the system state and control input, respectively. The learned residual torque is injected into the nominal dynamics as a corrective term, improving prediction accuracy without altering the underlying physical model.

This residual-based formulation enables a clear separation between first-principles modeling and data-driven correction and is particularly suited for integration with nonlinear model predictive control frameworks.

3.3 Nominal NMPC Formulation (Conceptual)

This section presents the nominal nonlinear model predictive control (NMPC) formulation used for spacecraft attitude regulation. The controller relies on the discrete-time rigid-body dynamics introduced in Section 3.2 and is designed to explicitly account for the underactuated nature of the spacecraft under study. The formulation emphasizes the structure of the optimal control problem, the cost function design, and the resulting closed-loop behavior.

3.3.1 Actuation Authority and Dynamical Time-Scale Considerations

The design of the NMPC formulation, including the choice of sampling period and prediction horizon, is closely linked to the available actuation authority, the characteristic time scales of the spacecraft rotational dynamics, and the computational effort required to solve the nonlinear program online.

The inertia matrix adopted for control design, which includes the diagonal elements of the nominal inertial matrix of the spacecraft is represented hereunder, which the z component include the inertia of the faulty reaction wheel.

$$\mathbf{J} = \mathrm{diag}(5.3,\ 3.1,\ 5.7)\ \mathrm{kg\,m^2}, \tag{3.27}$$

and the reaction wheel configuration is underactuated with $n_u = 2$ and actuator distribution matrix

$$\mathbf{B}_{\mathrm{act}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}. \tag{3.28}$$

Accordingly, direct control torques can be generated about the body x and y axes only.

With a maximum available motor torque of $u_{\max} = 0.05$ N m, the achievable angular accelerations in the actuated axes are bounded by

$$\dot{\omega}_x^{\max} \approx \frac{u_{\max}}{J_x} \approx 9.4 \times 10^{-3}\ \mathrm{rad/s^2}, \qquad \dot{\omega}_y^{\max} \approx \frac{u_{\max}}{J_y} \approx 1.6 \times 10^{-2}\ \mathrm{rad/s^2}, \tag{3.29}$$

neglecting gyroscopic coupling terms. These bounds indicate that the closed-loop attitude transients are inherently limited by actuator authority and therefore evolve on a comparatively slow time scale. This supports the use of a moderate NMPC update period

$$T_{NMPC} = 0.5\ \mathrm{s}, \tag{3.30}$$

which is sufficiently fast to react to the dominant rigid-body dynamics while avoiding unnecessary discretization of slow system behavior.

The prediction horizon length is set to

$$N_h = 24, \tag{3.31}$$

corresponding to a prediction window of $T_h = N_h.T_{NMPC} = 12$ s. This horizon duration is long enough to allow the optimizer to plan meaningful rest-to-rest attitude corrections under input saturation and torque-rate limits, and to anticipate the coupling effects induced by the rigid-body nonlinear dynamics in the presence of underactuation. At the same time, it keeps the size of the nonlinear program tractable for real-time operation, respecting $80\%.T_{controller}$ solving time constriant of the controller.

The selected combination $(T_{NMPC}, N_h) = (0.5\text{ s}, 24)$ was therefore chosen as a practical trade-off between control performance and computational feasibility, and was validated in tuning phase and through extensive numerical simulations, where reliable convergence of the NMPC solver was achieved within the available computation time at each control update.

Environmental disturbances such as gravity-gradient and aerodynamic drag torques act with comparatively small magnitudes and vary smoothly over time for the considered orbital conditions. As a result, they do not introduce high-frequency, large-magnitude rotational dynamics over the time horizon of interest and are treated as slowly varying, small-magnitude perturbations relative to the actuation-limited closed-loop response.

3.3.2 State Vector, Reference, Control Vector, and Constraints

The NMPC state vector is defined as

$$\mathbf{x} = \begin{bmatrix} \mathbf{q}^\top & \boldsymbol{\omega}^\top & \boldsymbol{\omega}_{\mathrm{RW}}^\top \end{bmatrix}^\top \in \mathbb{R}^{7+n_u}, \tag{3.32}$$

where $\mathbf{q} \in \mathbb{S}^3$ is the unit quaternion representing the spacecraft attitude, $\boldsymbol{\omega} \in \mathbb{R}^3$ is the angular velocity expressed in the body frame, and $\boldsymbol{\omega}_{\mathrm{RW}} \in \mathbb{R}^{n_u}$ denotes the reaction wheel angular velocities.

The control input is the vector of reaction wheel motor torques

$$\mathbf{u}_{\mathrm{RW}} \in \mathbb{R}^{n_u}. \tag{3.33}$$

The corresponding control torque applied to the spacecraft body is

$$\boldsymbol{\tau} = \mathbf{B}\,\mathbf{u}_{\mathrm{RW}}, \tag{3.34}$$

where $\mathbf{B} \in \mathbb{R}^{3\times n_u}$ is the wheel spin-axis (actuator distribution) matrix. In the considered configuration $n_u = 2$, hence the spacecraft is underactuated in torque

space, i.e., $\text{rank}(\mathbf{B}) = 2 < 3$.

Reference definition. The NMPC formulation uses a full-state reference

$$\mathbf{x}_{\text{ref}} = \begin{bmatrix} \mathbf{q}_{\text{ref}}^\top & \boldsymbol{\omega}_{\text{ref}}^\top & \boldsymbol{\omega}_{\text{RW,ref}}^\top \end{bmatrix}^\top, \tag{3.35}$$

where $\mathbf{q}_{\text{ref}} \in \mathbb{S}^3$ is the desired attitude, and $\boldsymbol{\omega}_{\text{ref}}, \boldsymbol{\omega}_{\text{RW,ref}}$ are reference angular rates for the rigid body and the reaction wheels, respectively. For the rest-to-rest regulation problem considered in this work and with zero-total-angular-momentum assumption, the reference is chosen as

$$\boldsymbol{\omega}_{\text{ref}} = \mathbf{0}, \qquad \boldsymbol{\omega}_{\text{RW,ref}} = \mathbf{0}, \tag{3.36}$$

while $\mathbf{q}_{\text{ref}}$ is regulation or tracking inertial-pointing attitude. In this work the quaternion reference is chosen to be the unitary quaternion, and the problem starts from different innitial conditions and tries to reach the unitary quaternion, correspondent to the inertial frame $\mathcal{N}$.

Constraints. Input and state constraints reflect reaction wheel limitations and are enforced as hard constraints within the NMPC optimization problem.

3.3.2.0.1 Reaction wheel speed bounds. Reaction wheel angular velocities are constrained component-wise as

$$-\omega_{\text{RW,max}} \leq \omega_{\text{RW},i}(k) \leq \omega_{\text{RW,max}}, \qquad i = 1, \ldots, n_u, \tag{3.37}$$

for all prediction steps k. In the present implementation, the maximum wheel speed is set to

$$\omega_{\text{RW,max}} = 6000 \text{ rpm} = 6000 \cdot \frac{2\pi}{60} \approx 628.3 \text{ rad/s}, \tag{3.38}$$

which corresponds to the nominal saturation limit of the reaction wheels used in the Basilisk simulation environment.

3.3.2.0.2 Motor torque saturation. The commanded reaction wheel motor torques satisfy

$$-u_{\text{max}} \leq u_{\text{RW},i}(k) \leq u_{\text{max}}, \qquad i = 1, \ldots, n_u, \tag{3.39}$$

for all prediction steps k. In this work, the torque bound is selected as

$$u_{\text{max}} = 0.05 \text{ N m}, \tag{3.40}$$

representing a realistic actuator torque limit for small spacecraft reaction wheels.

3.3.2.0.3 Torque-rate (increment) bounds. To avoid unrealistic changes in commanded motor torques and to reflect actuator slew-rate limitations, a bound is

imposed on the discrete-time input increment

$$\Delta \mathbf{u}_{\mathrm{RW}}(k) \triangleq \mathbf{u}_{\mathrm{RW}}(k) - \mathbf{u}_{\mathrm{RW}}(k-1), \tag{3.41}$$

with the convention that $\mathbf{u}_{\mathrm{RW}}(-1) = \mathbf{u}_{\mathrm{prev}}$.

The increment constraint is defined as

$$-\Delta u_{\max} \leq \Delta u_{\mathrm{RW},i}(k) \leq \Delta u_{\max}, \qquad i = 1, \ldots, n_u, \tag{3.42}$$

where

$$\Delta u_{\max} = \dot{u}_{\max} T_s. \tag{3.43}$$

In the implementation, the maximum torque slew rate is chosen as

$$\dot{u}_{\max} = 0.4 \ \mathrm{N\,m/s},$$

and the NMPC update period is

$$T_s = 0.5 \ \mathrm{s},$$

resulting in

$$\Delta u_{\max} = 0.2 \ \mathrm{N\,m}.$$

Accordingly, for the first predicted input, the constraint becomes

$$-\Delta u_{\max} \leq u_{\mathrm{RW},i}(0) - u_{\mathrm{prev},i} \leq \Delta u_{\max}, \tag{3.44}$$

while for subsequent prediction steps $k \geq 1$,

$$-\Delta u_{\max} \leq u_{\mathrm{RW},i}(k) - u_{\mathrm{RW},i}(k-1) \leq \Delta u_{\max}. \tag{3.45}$$

These constraints define the admissible input and state sets $\mathcal{U}$ and $\mathcal{X}$ used in the NMPC problem formulation and ensure that all computed control actions are compatible with physical actuator limitations.

3.3.3 Finite-Horizon Optimal Control Problem Statement

At each control update instant, the NMPC controller solves the following finite-horizon nonlinear optimal control problem:

$$\min_{\{\mathbf{u}_{\mathrm{RW},k}\}_{k=0}^{N_h-1}} \quad \sum_{k=0}^{N_h-1} \ell(\mathbf{x}_k, \mathbf{u}_{\mathrm{RW},k}) + V_f(\mathbf{x}_{N_h}) \tag{3.46a}$$

$$\text{s.t.} \quad \mathbf{x}_{k+1} = \Phi_{\mathrm{RK4}}(\mathbf{x}_k, \mathbf{u}_{\mathrm{RW},k}), \tag{3.46b}$$

$$\mathbf{x}_0 = \mathbf{x}(t), \tag{3.46c}$$

$$\mathbf{x}_k \in \mathcal{X}, \quad \mathbf{u}_{\mathrm{RW},k} \in \mathcal{U}, \tag{3.46d}$$

where N_h is the prediction horizon length and $\Phi_{\mathrm{RK4}}(\cdot)$ denotes the discrete-time prediction model obtained via fourth-order Runge–Kutta integration of the continuous-time dynamics presented in Section 3.2.3.

The objective function consists of a stage cost $\ell(\mathbf{x}_k, \mathbf{u}_{\mathrm{RW},k})$, accumulated over the prediction horizon, and a terminal cost $V_f(\mathbf{x}_{N_h})$. The stage cost penalizes attitude tracking error, angular velocity error, control effort, and control input variations, while the terminal cost provides a quadratic penalty on the terminal attitude and angular velocity error. The explicit structure of the cost function and the associated design parameters are detailed in Section 3.3.4.

3.3.4 Cost Function Structure and Design Parameters

The NMPC objective function is designed to achieve accurate attitude regulation while explicitly accounting for actuator limitations, underactuation, and the nonlinear nature of rigid-body rotational dynamics. The total cost is composed of a stage cost accumulated over the prediction horizon and a terminal cost penalizing the final state deviation.

3.3.4.1 Stage Cost

At each prediction step k, the stage cost is defined as

$$\ell(\mathbf{x}_k, \mathbf{u}_{\mathrm{RW},k}) = \ell_q(k) + \ell_\omega(k) + \ell_u(k) + \ell_{\Delta u}(k), \tag{3.47}$$

where each term is detailed below.

Attitude tracking term. The attitude error is expressed using the quaternion error

$$\mathbf{q}_e(k) = \mathbf{q}_k^{-1} \otimes \mathbf{q}_{\mathrm{ref}}, \tag{3.48}$$

where the quaternion sign ambiguity is resolved by enforcing a nonnegative scalar

part. Only the vector part $\mathbf{q}_{e,v}(k) \in \mathbb{R}^3$ is penalized. The corresponding cost term is

$$\ell_q(k) = \mathbf{q}_{e,v}^\top(k)\, \mathbf{Q}_q^{\text{eff}}\, \mathbf{q}_{e,v}(k). \tag{3.49}$$

To explicitly account for underactuation, the effective attitude weight matrix is constructed as

$$\mathbf{Q}_q^{\text{eff}} = \mathbf{P}_\parallel^\top \mathbf{Q}_q \mathbf{P}_\parallel + \alpha_N\, \mathbf{N}_\perp^\top \mathbf{Q}_q \mathbf{N}_\perp, \tag{3.50}$$

where $\mathbf{P}_\parallel$ projects onto the actuated torque subspace and $\mathbf{N}_\perp = \mathbf{I} - \mathbf{P}_\parallel$ onto its null space. The scalar parameter $\alpha_N > 0$ schedually increases the penalty on attitude error components that lie outside the directly actuated subspace.

Schedualling α_N weighting. The schedualling scalar parameter α_N is a heuristic method introduced to modulate the relative importance of attitude and angular rate error components that evolve in the underactuated subspace. Specifically, α_N is designed as a function of the Euclidean norm of the vector part of the quaternion error,

$$e_q \triangleq \|\mathbf{q}_{e,v}\|_2.$$

As the attitude error decreases, the value of α_N increases, thereby progressively emphasizing the reduction of error components that cannot be directly influenced by the available control torques. This design encourages the optimizer, near the target attitude, to exploit the nonlinear coupling of the rigid-body dynamics in order to indirectly regulate the underactuated direction.

The schedualling law for α_N is selected to be smooth, bounded, and monotonic with respect to e_q, in order to avoid abrupt changes in the cost function that could negatively affect numerical convergence of the NMPC solver. Several candidate functional forms were evaluated during controller development. The final choice was guided by the need to achieve high steady-state accuracy while preserving robust and reliable convergence of the nonlinear optimization problem.

The selected adaptive weighting function is defined as

$$\alpha_N(e_q) = \alpha_{\min} + (\alpha_{\max} - \alpha_{\min})\, \bar{s}(e_q), \qquad \bar{s}(e_q) \in [0, 1], \tag{3.51}$$

where $e_q = \|q_{ev}\|$ denotes the quaternion error norm. The scheduling variable $\bar{s}(e_q)$ is constructed as a normalized sigmoidal function

$$\bar{s}(e_q) = \frac{s(e_q) - s(1)}{s(0) - s(1)}, \qquad s(e_q) = \frac{1}{1 + \exp\big(k(e_q - c)\big)}, \tag{3.52}$$

with positive design parameters k and c controlling the steepness and transition point of the scheduling law, respectively. This normalization guarantees $\bar{s}(0) = 1$ and $\bar{s}(1) = 0$, ensuring a smooth, bounded, and monotonic decrease of $\alpha_N(e_q)$ over the admissible error range. As a result, high weighting is applied in the vicinity of

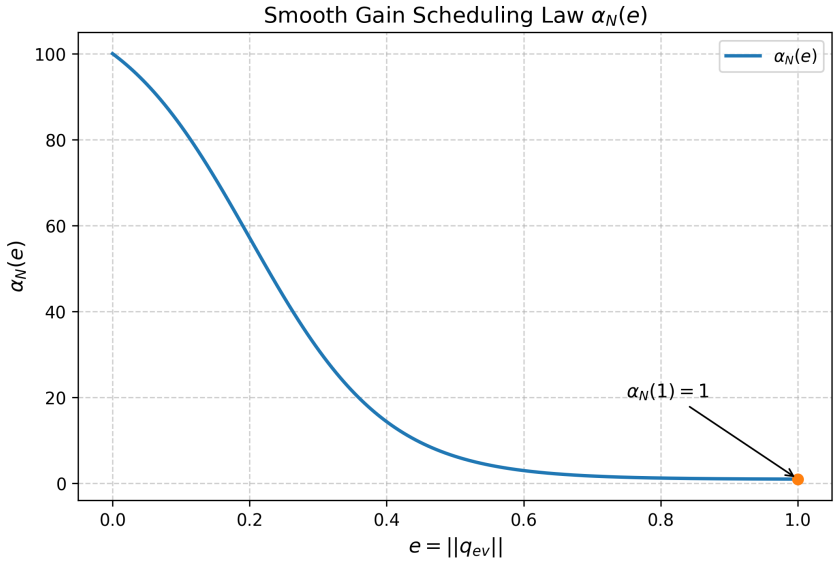

Figure 3.3: Smooth gain scheduling law $\alpha_N(e_q)$ as a function of the quaternion error norm $e_q = \|q_{ev}\|$.

the desired attitude, while reduced weighting is enforced for larger attitude errors; the weight is almost one for large errors. The figure3.3 represents the graph of the schedualling function.

This heuristic adjustment biases the optimizer toward minimizing residual errors in the underactuated axis and improves steady-state pointing accuracy without compromising numerical conditioning of the optimization problem. This gain schedualling mechanism was found to significantly improve steady-state pointing performance without compromising solver robustness or real-time feasibility.

Angular velocity tracking term. The angular velocity deviation $\Delta\boldsymbol{\omega}_k = \boldsymbol{\omega}_k - \boldsymbol{\omega}_{\text{ref}}$ is penalized in an analogous decomposed form:

$$\ell_\omega(k) = \Delta\boldsymbol{\omega}_{\|}^\top(k)\,\mathbf{Q}_\omega\,\Delta\boldsymbol{\omega}_{\|}(k) + \alpha_N\,\Delta\boldsymbol{\omega}_{\perp}^\top(k)\,\mathbf{Q}_\omega\,\Delta\boldsymbol{\omega}_{\perp}(k), \tag{3.53}$$

where $\Delta\boldsymbol{\omega}_{\|} = \mathbf{P}_{\|}\Delta\boldsymbol{\omega}$ and $\Delta\boldsymbol{\omega}_{\perp} = \mathbf{N}_{\perp}\Delta\boldsymbol{\omega}$. This formulation encourages rapid damping of angular rates while respecting the underactuated structure of the system.

Control effort term. Reaction wheel motor torques are penalized via

$$\ell_u(k) = \mathbf{u}_{\text{RW},k}^\top \mathbf{R}\,\mathbf{u}_{\text{RW},k}, \tag{3.54}$$

which discourages excessive actuator usage and contributes to smoother control profiles.

Control increment (slew-rate) term. To avoid aggressive torque variations and

to improve numerical conditioning of the optimization problem, a penalty on the discrete-time input increment is introduced:

$$\ell_{\Delta u}(k) = (\mathbf{u}_{\mathrm{RW},k} - \mathbf{u}_{\mathrm{RW},k-1})^\top \mathbf{R}_{\Delta u} (\mathbf{u}_{\mathrm{RW},k} - \mathbf{u}_{\mathrm{RW},k-1}), \tag{3.55}$$

where $\mathbf{u}_{\mathrm{RW},-1}$ is set to the previously applied control input.

3.3.4.2 Terminal Cost

To promote convergence and to improve the closed-loop behavior near the end of the prediction horizon, a quadratic terminal cost is introduced:

$$V_f(\mathbf{x}_{N_h}) = \mathbf{x}_{e,N_h}^\top \mathbf{P}\, \mathbf{x}_{e,N_h},$$

where the terminal error state is defined as

$$\mathbf{x}_{e,N_h} = \begin{bmatrix} \mathbf{q}_{e,v}(N_h) \\ \boldsymbol{\omega}_{N_h} - \boldsymbol{\omega}_{\mathrm{ref}} \end{bmatrix}.$$

The terminal weighting matrix $\mathbf{P} \succ 0$ is constructed based on a local linearization of the spacecraft attitude error dynamics around the equilibrium corresponding to the reference attitude $\mathbf{q}_{\mathrm{ref}} = [1,\, 0,\, 0,\, 0]^\top$ and zero angular velocity. Under the small-angle approximation, the linearized continuous-time error dynamics admit the block structure

$$\dot{\mathbf{x}}_e = \mathbf{A}\mathbf{x}_e + \mathbf{B}\mathbf{u}, \qquad \mathbf{A} = \begin{bmatrix} \mathbf{0}_{3\times3} & \frac{1}{2}\mathbf{I}_3 \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} \mathbf{0}_{3\times3} \\ \mathbf{J}^\dagger \end{bmatrix}, \tag{3.56}$$

where $\mathbf{J}^\dagger$ denotes the pseudo-inverse of the spacecraft inertia matrix. After discretization with sampling time T_s, the discrete-time model

$$\mathbf{x}_{k+1} = \mathbf{A}_d \mathbf{x}_k + \mathbf{B}_d \mathbf{u}_k \tag{3.57}$$

is obtained.

The matrix $\mathbf{P}$ is then computed as the stabilizing solution of the discrete algebraic Riccati equation (DARE) associated with the above linearized model and a discrete-time LQR design. The resulting matrix $\mathbf{P}$ serves as a Lyapunov matrix for the locally linearized closed-loop system and provides a principled terminal penalty for the nonlinear NMPC formulation.

It is important to note that the terminal cost is designed assuming a fully actuated spacecraft model. This choice is not claimed to provide mathematically regorous proof for the stability of the underactuated direction, since MPC assumes a fully

actuated system. However, it provides the guarantee for the actuation system. As discussed, NMPC provides precise-enough pointing for the underactuated system, but the stability of the underactuated direction relies on the existence of a system that benefits external source of torque, e.g. propulsors, to prevent drifting in the third axis due to sources of disturbance. To account for the underactuated nature of the actual system, the terminal weight matrix is further modified by increasing the penalty associated with the underactuated rotational direction by a factor of ten.

Overall, the terminal cost provides a locally stabilizing shaping of the NMPC objective near the equilibrium while remaining compatible with the nonlinear dynamics of the spacecraft.

3.3.4.3 Weight Selection

In the simulations presented in this work, the weighting matrices are chosen as

$$\mathbf{Q}_q = q_q \mathbf{I}_3, \qquad \mathbf{Q}_\omega = q_\omega \mathbf{I}_3, \qquad \mathbf{R} = r\mathbf{I}_{n_u}, \qquad \mathbf{R}_{\Delta u} = r_{\Delta u}\mathbf{I}_{n_u}, \tag{3.58}$$

with representative values

$$q_q = 10^3, \quad q_\omega = 1, \quad r = 5, \quad r_{\Delta u} = 10,$$

The matrix $\mathbf{P}$ corresponds to the stabilizing solution of the discrete algebraic Riccati equation associated with the linearized attitude error dynamics and is given numerically by

$$\mathbf{P} = 10^5 \begin{bmatrix} 0.00241 & 0.0000108 & 0.0000008 & 0.07210 & 0.000569 & 0.0000384 \\ 0.0000108 & 0.00183 & 0.0000068 & 0.000569 & 0.04174 & 0.000273 \\ 0.0000008 & 0.0000068 & 0.01402 & 0.0000384 & 0.000273 & 0.24400 \\ 0.07210 & 0.000569 & 0.0000384 & 4.3291 & 0.04542 & 0.002862 \\ 0.000569 & 0.04174 & 0.000273 & 0.04542 & 1.9071 & 0.01660 \\ 0.0000384 & 0.000273 & 0.24400 & 0.002862 & 0.01660 & 8.5226 \end{bmatrix} \tag{3.59}$$

This matrix was obtained offline and kept fixed throughout the simulations. Its magnitude relative to the stage weights reflects the emphasis on accurate convergence near the equilibrium, while preserving reliable numerical behavior of the NMPC solver.

3.4 PINN-Based Residual Torque Compensation (Conceptual)

3.4.1 Residual Definition: What Is Learned and Why

As introduced in Section 3.2.4, uncertainties affecting the spacecraft attitude dynamics are represented through a lumped residual disturbance torque $\boldsymbol{\tau}_d \in \mathbb{R}^3$ acting directly on the rigid-body rotational dynamics. This residual term captures the

discrepancy between the true spacecraft dynamics and the nominal predictive model employed within the NMPC formulation.

Rather than attempting to learn the full spacecraft dynamics or to replace the baseline controller, the Physics-Informed Neural Network (PINN) is designed to approximate the unknown residual torque appearing in the coupled spacecraft–reaction-wheel dynamics

$$\begin{bmatrix} \mathbf{J} & \mathbf{B}\mathbf{J}_{\mathrm{RW}} \\ \mathbf{J}_{\mathrm{RW}}\mathbf{B}^{\top} & \mathbf{J}_{\mathrm{RW}} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{\omega}} \\ \dot{\boldsymbol{\omega}}_{\mathrm{RW}} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{\omega} \times (\mathbf{J}\boldsymbol{\omega} + \mathbf{h}) + \boldsymbol{\tau}_d \\ \mathbf{u}_{\mathrm{RW}} \end{bmatrix}. \tag{3.60}$$

The residual torque aggregates the effects of inertial parameter mismatches, unmodeled coupling terms, numerical discretization errors, and slowly varying external disturbances. Accordingly, the learning objective is to approximate the unknown mapping

$$\boldsymbol{\tau}_d(k) = \mathcal{F}_\theta(\mathbf{x}(k),\, \mathbf{u}_{\mathrm{RW}}(k),\, \boldsymbol{\tau}_d(k-1))\,, \tag{3.61}$$

where $\mathcal{F}_\theta$ denotes a neural network parametrized by θ, and the dependence on the previous residual estimate introduces a nonlinear autoregressive structure.

This formulation preserves physical interpretability: the PINN does not generate control actions, but instead provides a corrective term that improves the fidelity of the predictive model used by the NMPC.

3.4.2 Physics-Informed Learning Principle

The PINN training procedure is physics-informed in the sense that no direct supervision on the residual torque $\boldsymbol{\tau}_d$ is required. Instead, learning is driven by enforcing consistency between the predicted state evolution of the augmented dynamics and the observed system trajectories over a finite prediction horizon.

During training, the network output $\boldsymbol{\tau}_d(k)$ is injected into the continuous-time spacecraft dynamics as an additive disturbance torque. The resulting augmented dynamics are discretized using the same fourth-order Runge–Kutta (RK4) scheme employed in the NMPC prediction model. This ensures full consistency between the learning model and the control-oriented model used in closed-loop operation.

By embedding the neural network inside the known physical dynamics, the learning problem is strongly constrained: only residual torques that produce physically plausible state trajectories can reduce the training loss.

3.4.3 Learning Objective and Loss Functional

Let the discrete-time augmented dynamics be written compactly as

$$\mathbf{x}_{k+1} = \Phi_{\mathrm{RK4}}(\mathbf{x}_k,\, \mathbf{u}_{\mathrm{RW},k},\, \boldsymbol{\tau}_d(k))\,, \tag{3.62}$$

where $\Phi_{\mathrm{RK4}}(\cdot)$ denotes the fourth-order Runge–Kutta discretization of the continuous dynamics including the residual disturbance torque $\boldsymbol{\tau}_d$.

In principle, the residual torque $\boldsymbol{\tau}_d(t)$ may vary continuously in time. However, in the proposed learning formulation, $\boldsymbol{\tau}_d$ is assumed to be constant over each NMPC sampling interval,

$$\boldsymbol{\tau}_d(t) \approx \boldsymbol{\tau}_d(k), \qquad t \in [kT_s, (k+1)T_s), \tag{3.63}$$

where T_s denotes the NMPC update period. This assumption is deliberately introduced as a structural regularization of the learning problem. It reflects the fact that the dominant sources of model mismatch—such as inertial parameter errors and slowly varying disturbances—evolve on time scales significantly slower than the NMPC update rate. Furthermore, this assumption is consistent with the way PINN is going to be adopted in the closed-loop control system, where the PINN correction remains unchanged throughout the NMPC period.

At each NMPC update instant k, the residual torque is generated by the PINN and held constant over the fast integration steps used to propagate the augmented dynamics. The sampling is done ten times faster than the NMPC update frequendcy. Therefore, ten step integration is required for evaluating the value of the prediction model at the end of the horizon. Let H denote the number of the integration steps corresponding to one NMPC sampling interval. Starting from $\mathbf{x}_k$, the augmented dynamics are propagated over H steps, yielding a predicted angular velocity $\hat{\boldsymbol{\omega}}_{k+H}$.

The training objective is defined as a finite-horizon prediction loss on the angular velocity:

$$\mathcal{L}(\theta) = \sum_{k \in \mathcal{K}} \|\hat{\boldsymbol{\omega}}_{k+H}(\theta) - \boldsymbol{\omega}_{k+H}\|_2^2\,, \tag{3.64}$$

where $\boldsymbol{\omega}_{k+H}$ denotes the measured angular velocity obtained from simulation data and $\mathcal{K}$ is the set of training time indices.

By construction, the loss depends on the network parameters θ only through the effect of the residual torque $\boldsymbol{\tau}_d(k)$ on the state evolution over the prediction horizon. As a result, the learned residual should be interpreted as an *equivalent horizon-wise disturbance* that reproduces the observed deviation from the nominal dynamics over one NMPC interval, rather than as a uniquely identifiable continuous-time disturbance torque.

This loss formulation therefore aligns the learning objective with the intended role of the PINN: improving short-horizon prediction accuracy of the rigid-body rotational dynamics in a manner that is physically consistent with the NMPC time discretization.

3.4.4 Gradient Propagation Through the Dynamics

The loss function depends on the network parameters θ only through the residual torque $\boldsymbol{\tau}_d$ and its effect on the predicted state trajectory. Using the chain rule, the gradient of the loss with respect to θ can be expressed as

$$\frac{\partial \mathcal{L}}{\partial \theta} = \sum_{k \in \mathcal{K}} \frac{\partial \mathcal{L}}{\partial \hat{\boldsymbol{\omega}}_{k+H}} \frac{\partial \hat{\boldsymbol{\omega}}_{k+H}}{\partial \boldsymbol{\tau}_d(k)} \frac{\partial \boldsymbol{\tau}_d(k)}{\partial \theta}. \tag{3.65}$$

The term $\partial \boldsymbol{\tau}_d(k)/\partial \theta$ is obtained through standard backpropagation through the neural network. The sensitivity term $\partial \hat{\boldsymbol{\omega}}_{k+H}/\partial \boldsymbol{\tau}_d(k)$ is implicitly computed by automatic differentiation through the RK4 integration steps.

As a result, the learning signal propagates through the physical model itself, forcing the network to account for how residual torques influence the rotational dynamics over time. This mechanism is fundamentally different from direct regression on disturbance labels and ensures that learning remains grounded in the governing equations of motion.

No explicit adjoint or analytical sensitivity model is required, as the entire computation graph—including numerical integration—is differentiable.

3.4.5 Network Inputs, Outputs, and Architectural Structure

The Physics-Informed Neural Network (PINN) adopts a nonlinear autoregressive model with exogenous inputs (NARX) structure, in which a one-step memory of the estimated residual torque is included to capture slowly varying or history-dependent components of the model mismatch. This autoregressive dependence is introduced as a structural regularization mechanism and does not imply the existence of an explicit dynamic model for the residual disturbance. At each NMPC update instant, the network receives as input a feature vector constructed from measured rotational states and applied control inputs, augmented with the residual torque estimate from the previous macro time step.

The network input is defined as

$$\mathbf{z}(k) = \begin{bmatrix} \boldsymbol{\phi}(\boldsymbol{\omega}(k),\, \mathbf{u}_{\mathrm{RW}}(k),\, \boldsymbol{\omega}(k-1)) \\ \boldsymbol{\tau}_d(k-1) \end{bmatrix} \in \mathbb{R}^{17}, \tag{3.66}$$

where $\boldsymbol{\phi}(\cdot) \in \mathbb{R}^{14}$ denotes a vector of engineered features, and $\boldsymbol{\tau}_d(k-1) \in \mathbb{R}^3$ is the previously estimated residual disturbance torque.

3.4.5.0.1 Feature construction The feature vector $\boldsymbol{\phi}(\cdot)$ is designed to reflect the dominant structure of the residual rotational dynamics induced by inertial mismatch between the true spacecraft and the nominal predictive model. To motivate this choice, consider the rigid-body rotational dynamics under an inertial perturbation $\Delta\mathbf{J}$. By subtracting the nominal dynamics from the true dynamics and retaining only inertia-mismatch-induced terms, the equivalent residual disturbance torque can be expressed as

$$\boldsymbol{\tau}_d^{(\Delta\mathbf{J})} = -\,\Delta\mathbf{J}\,\dot{\boldsymbol{\omega}} \;-\; \boldsymbol{\omega} \times (\Delta\mathbf{J}\boldsymbol{\omega}), \tag{3.67}$$

which shows that inertial uncertainty enters the rotational dynamics through linear terms in the angular acceleration and quadratic gyroscopic coupling terms in the angular velocity.

Based on this structure, the feature vector is constructed as

$$\boldsymbol{\phi}(k) = \begin{bmatrix} \boldsymbol{\omega}(k) \\ \boldsymbol{\omega}(k) \odot \boldsymbol{\omega}(k) \\ \omega_x(k)\omega_y(k) \\ \omega_y(k)\omega_z(k) \\ \omega_x(k)\omega_z(k) \\ \mathbf{u}_{\mathrm{RW}}(k) \\ \boldsymbol{\omega}(k-1) \end{bmatrix} \in \mathbb{R}^{14}, \tag{3.68}$$

where $\odot$ denotes the elementwise product. The feature set includes: (i) the angular velocity components, (ii) quadratic and cross-product terms capturing gyroscopic effects, (iii) the applied reaction wheel control torques, which influence angular acceleration through the coupled rigid-body–wheel dynamics, and (iv) delayed angular velocity measurements, providing a finite-difference proxy for angular acceleration. This construction enables the network to approximate inertia-mismatch-induced effects without explicitly estimating inertial parameters.

3.4.5.0.2 Network output and physical interpretation. The network output is the estimated residual disturbance torque

$$\hat{\boldsymbol{\tau}}_d(k) \in \mathbb{R}^2, \tag{3.69}$$

which is interpreted as an additive corrective torque acting directly on the rigid-body rotational dynamics, consistent with the lumped disturbance formulation introduced in Section 3.2.4.

3.4.5.0.3 Network architecture. The PINN is implemented as a fully connected feedforward neural network with three hidden layers, each consisting of 128 neurons. All layers are densely connected, yielding a mapping

$$\mathcal{F}_\theta : \mathbb{R}^{16} \to \mathbb{R}^{128} \to \mathbb{R}^{128} \to \mathbb{R}^{128} \to \mathbb{R}^2. \tag{3.70}$$

This architecture provides sufficient expressive capacity to approximate nonlinear model mismatch while maintaining a compact structure compatible with real-time NMPC integration.

Each hidden layer employs the hyperbolic tangent activation function,

$$\sigma(x) = \tanh(x), \tag{3.71}$$

which is smooth, bounded, and continuously differentiable. Bounded activations help prevent unrealistically large residual torque estimates, while smoothness ensures that the resulting network mapping is locally Lipschitz, not changing moving fast. These properties are essential for well-posed numerical integration of the augmented dynamics and for compatibility with the Lyapunov-based supervisory layer introduced in subsequent sections.

3.4.5.0.4 Input normalization and output scaling. Prior to network evaluation, the feature vector $\boldsymbol{\phi}(\cdot)$ is normalized using an affine transformation

$$\tilde{\boldsymbol{\phi}} = \operatorname{diag}(\boldsymbol{\sigma}_\phi)^{-1} \left(\boldsymbol{\phi} - \boldsymbol{\mu}_\phi\right), \tag{3.72}$$

where $\boldsymbol{\mu}_\phi$ and $\boldsymbol{\sigma}_\phi$ denote the mean and standard deviation of each feature, computed exclusively over the training set to avoid information leakage. Normalization parameters are fixed during validation and deployment.

The raw network output is subsequently scaled by a diagonal gain

$$\hat{\boldsymbol{\tau}}_d(k) = \mathbf{D}_\tau \, \mathcal{F}_\theta\left(\begin{bmatrix} \tilde{\boldsymbol{\phi}}(k) \\ \boldsymbol{\tau}_d(k-1) \end{bmatrix}\right), \qquad \mathbf{D}_\tau = \operatorname{diag}(\tau_{\max,x}, \tau_{\max,y}) \tag{3.73}$$

where $\mathbf{D}_\tau$ encodes conservative, axis-dependent bounds on the admissible residual torque magnitude. This output scaling introduces a physical prior on the disturbance magnitude, improves numerical conditioning during training, and limits the corrective action injected into the predictive model.

3.4.5.0.5 Temporal structure and memory handling. The inclusion of the previous residual estimate $\boldsymbol{\tau}_d(k-1)$ introduces a first-order temporal dependence at the NMPC sampling rate. The residual torque is evaluated once per NMPC update and held constant over the fast integration steps within the prediction horizon. The autoregressive state is reset at simulation boundaries during training and inference to prevent spurious temporal coupling across independent trajectories. This design enables the representation of slowly varying or history-dependent effects without resorting to recurrent architectures, improves numerical stability, and supports practical identifiability of the residual correction under finite-horizon rollout-based supervision.

3.4.6 Boundedness and Regularity of the Learned Residual

To ensure physical plausibility and numerical robustness, explicit bounds are imposed on the network output. The predicted residual torque is scaled by a-priori per-axis limits derived from conservative physical considerations related to inertial uncertainty and disturbance magnitudes.

Accordingly, the learned residual satisfies

$$\hat{\boldsymbol{\tau}}_d \in \mathcal{D} \subset \mathbb{R}^2, \tag{3.74}$$

where $\mathcal{D}$ is a compact set.

3.5 Learning Reliability and Safety Considerations (Qualitative)

A key concern in learning-augmented control architectures is the risk of overfitting to specific operating conditions or noise realizations, which may lead to degraded performance or unsafe behavior when the system operates outside the training distribution. Several architectural and modeling choices are made to mitigate this risk.

First, the PINN receives state estimates from an extended Kalman filter (EKF), which provides relatively low-noise, physically consistent signals. As a result, the learning module is not directly exposed to raw sensor noise, reducing the likelihood of fitting high-frequency measurement artifacts.

Second, neural networks are known to exhibit a spectral bias, whereby lower-frequency components of the target function are learned preferentially over high-frequency components [14]. In the present context, this bias is aligned with the physical structure of the problem: the dominant effects associated with inertia mismatch and rigid-body coupling are inherently low-frequency mechanical phenomena. Consequently, it is hypothesized that the network will naturally focus on learning physically meaningful residual dynamics rather than noise-driven or fast-varying effects.

Finally, the PINN output is not applied directly to the plant. The Lyapunov-based supervisory layer enforces a safety condition on the combined control input, effectively limiting the impact of erroneous or poorly generalized network predictions. In the worst case, the supervisor can suppress the learned correction entirely, reverting the control law to the nominal NMPC policy. This mechanism ensures that generalization errors in the learning layer cannot destabilize the closed-loop system.

3.6 Lyapunov-Based Supervisory Layer (Safety Cage)

3.6.1 Supervisory Objective and Role in the Architecture

The Lyapunov-based supervisory layer is introduced to regulate the interaction between the nominal nonlinear model predictive controller (NMPC) and the learning-based residual torque compensation provided by the Physics-Informed Neural Network (PINN). Its role is not to stabilize the spacecraft independently, nor to redesign the baseline controller, but to ensure that the learned correction cannot degrade the closed-loop stability properties already achieved by the nominal NMPC.

The supervisor operates as a safety filter that evaluates, at each control update, whether the injection of the learned correction is compatible with a non-increasing Lyapunov-like energy function associated with the attitude error dynamics. If this condition is violated, the learned correction is attenuated or rejected, while the nominal NMPC command remains unchanged.

This design realizes a safety cage around the learning module:

- when the learned correction is beneficial or neutral, it is fully admitted;
- when the learned correction is potentially destabilizing, its influence is reduced or suppressed;
- in the worst case, the closed-loop system reverts to the nominal NMPC policy.

As a result, the learning component can improve performance when accurate, but cannot destabilize the system when inaccurate or poorly generalized.

3.6.2 Candidate Lyapunov Function and Monotonicity Condition

To assess the effect of the learned correction on closed-loop stability, a Lyapunov-like function is defined in terms of the attitude and angular velocity tracking errors. Let the quaternion attitude error be

$$\mathbf{q}_e = \mathbf{q}^{-1} \otimes \mathbf{q}_{\text{ref}},$$

with scalar part $q_{e,0} \geq 0$ and vector part $\mathbf{q}_{e,v} \in \mathbb{R}^3$, and define the angular velocity error as

$$\boldsymbol{\omega}_e = \boldsymbol{\omega} - \boldsymbol{\omega}_{\text{ref}}.$$

The candidate Lyapunov function is chosen as

$$V(\mathbf{q}_e, \boldsymbol{\omega}_e) = \frac{1}{2}\, \boldsymbol{\omega}_e^\top \mathbf{J}\, \boldsymbol{\omega}_e + k_q \left(1 - q_{e,0}\right), \tag{3.75}$$

where $\mathbf{J}$ is the spacecraft inertia matrix and $k_q > 0$ is a design parameter.

The first term represents the rotational kinetic energy associated with the angular velocity error, while the second term is a standard quaternion-based attitude error measure that is locally quadratic in the small-angle regime. The function V is positive definite with respect to the reference

$$\mathbf{q}_e = [1, 0, 0, 0]^\top, \qquad \boldsymbol{\omega}_e = \mathbf{0}.$$

Taking the time derivative of V along the rigid-body rotational dynamics yields

$$\dot{V} = \boldsymbol{\omega}_e^\top \mathbf{J}\dot{\boldsymbol{\omega}}_e - \frac{k_q}{2}\, q_{e,0}\, \mathbf{q}_{e,v}^\top \boldsymbol{\omega}_e, \tag{3.76}$$

where the quaternion kinematics have been used to express $\dot{q}_{e,0}$.

The supervisory condition is formulated as a Lyapunov monotonicity requirement,

$$\dot{V} \leq 0, \tag{3.77}$$

which is used as a local, instantaneous safety criterion rather than as a global asymptotic stability proof.

3.6.3 Worst-Case Interpretation of the Learned Correction

The PINN is trained to approximate the unknown residual disturbance torque $\boldsymbol{\tau}_d$ acting on the spacecraft rotational dynamics. In nominal operation, the learned correction is intended to compensate this disturbance through internal actuation. However, due to limited training data, extrapolation, or operation outside the training distribution, the estimated correction may suffer from sign or magnitude errors. The supervisory layer is therefore designed to explicitly account for such failure modes.

To clarify the role of the learned correction, consider the true rigid-body rotational dynamics expressed in the body frame,

$$\mathbf{J}\dot{\boldsymbol{\omega}} = \boldsymbol{\tau} + \boldsymbol{\tau}_d - \boldsymbol{\omega} \times (\mathbf{J}\boldsymbol{\omega}), \tag{3.78}$$

where $\boldsymbol{\tau}_d$ denotes the unknown residual disturbance torque.

The total body torque applied to the spacecraft is decomposed as

$$\boldsymbol{\tau} = \boldsymbol{\tau}_{\text{nom}} + \boldsymbol{\tau}_{\text{corr}}, \tag{3.79}$$

where $\boldsymbol{\tau}_{\text{nom}}$ is the nominal torque generated by the NMPC controller and $\boldsymbol{\tau}_{\text{corr}}$ is the correction torque inferred by the PINN.

Both torques are generated through reaction wheel actuation. Due to conservation of angular momentum, a commanded wheel torque produces an equal and opposite

torque on the spacecraft body. Accordingly, the body-frame torques are given by

$$\boldsymbol{\tau}_{\text{nom}} = -\,\mathbf{B}_{\text{act}}\,\mathbf{u}_{\text{NMPC}}, \qquad \boldsymbol{\tau}_{\text{corr}} = -\,\mathbf{B}_{\text{act}}\,\mathbf{u}_{\text{PINN}}, \tag{3.80}$$

where $\mathbf{B}_{\text{act}}$ is the actuator distribution matrix.

Due to underactuation, the correction torque lies in the actuated subspace and takes the form

$$\boldsymbol{\tau}_{\text{corr}} = \begin{bmatrix} \tau_{\text{corr},x} & \tau_{\text{corr},y} & 0 \end{bmatrix}^{\top}. \tag{3.81}$$

If the disturbance torque were known exactly, an ideal corrective action would satisfy

$$\boldsymbol{\tau}_{\text{corr}} = -\,\boldsymbol{\tau}_{d}. \tag{3.82}$$

Substituting (3.82) into (3.78) yields

$$\mathbf{J}\dot{\boldsymbol{\omega}} = \boldsymbol{\tau}_{\text{nom}} - \boldsymbol{\omega} \times (\mathbf{J}\boldsymbol{\omega}), \tag{3.83}$$

which coincides with the nominal disturbance-free model assumed by the NMPC. In this ideal case, the learned correction cancels the residual disturbance torque, improving tracking performance without altering the closed-loop stability properties of the baseline controller.

In a conservative learning failure scenario, however, the PINN may predict a correction torque that is approximately equal in magnitude but opposite in sign to the ideal value (3.82), i.e.

$$\boldsymbol{\tau}_{\text{corr}} \approx -\boldsymbol{\tau}_{d}. \tag{3.84}$$

If such a correction were applied directly, the effective disturbance term in (3.78) would become

$$-\boldsymbol{\tau}_{d} + \boldsymbol{\tau}_{\text{corr}} \approx -2\,\boldsymbol{\tau}_{d}, \tag{3.85}$$

corresponding to a reinforcement rather than a cancellation of the disturbance. In this case, the learned correction would inject energy into the rotational dynamics and potentially destabilize the closed-loop system.

To explicitly guard against this possibility, the supervisory layer evaluates safety conditions under a conservative, near worst-case torque model in which the learned correction is assumed to act adversarially with respect to the true disturbance. By enforcing Lyapunov monotonicity under this pessimistic assumption, the supervisor ensures robustness to sign errors in disturbance estimation and prevents energy injection due to learning failure.

3.6.4 Supervisory Action: Scaling or Rejection of the Correction

The influence of the applied torque on closed-loop stability is assessed through the time derivative of the Lyapunov function $V(\mathbf{q}_e, \boldsymbol{\omega}_e)$. Using the rigid-body rotational dynamics,

$$\mathbf{J}\dot{\boldsymbol{\omega}} = \boldsymbol{\tau} - \boldsymbol{\omega} \times (\mathbf{J}\boldsymbol{\omega}), \tag{3.86}$$

the Lyapunov derivative can be written as

$$\dot{V} = \boldsymbol{\omega}_e^\top \left(\boldsymbol{\tau} - \boldsymbol{\omega} \times (\mathbf{J}\boldsymbol{\omega})\right) - \frac{k_q}{2}\, q_{e,0}\, \mathbf{q}_{e,v}^\top \boldsymbol{\omega}_e. \tag{3.87}$$

Consequently, any modification of the applied torque directly affects the instantaneous rate of change of the Lyapunov function through its projection onto the angular velocity error.

To regulate the influence of the learned correction, the supervisor introduces a scalar attenuation factor $\alpha \in [0, 1]$ and evaluates the Lyapunov derivative under the conservative test torque

$$\boldsymbol{\tau}_{\text{test}} = \boldsymbol{\tau}_{\text{nom}} - 2\,\alpha\,\boldsymbol{\tau}_{\text{corr}}. \tag{3.88}$$

The factor of two explicitly encodes the pessimistic assumption that the learned correction may reinforce the true disturbance rather than cancel it, leading to an effective disturbance of doubled magnitude.

The supervisory condition is defined as

$$\dot{V}(\boldsymbol{\tau}_{\text{test}}) \leq 0. \tag{3.89}$$

The largest value of $\alpha \in [0, 1]$ satisfying this inequality is selected. If no positive value of α satisfies the condition, the learned correction is rejected entirely by setting $\alpha = 0$.

This mechanism ensures that the learned correction is applied only when it is compatible with a non-increasing instantaneous Lyapunov function, even under adversarial disturbance assumptions. As a result, the supervisory layer mitigates the risk of energy injection into the closed-loop system while still allowing performance improvements when the learned correction is physically consistent with the underlying dynamics. No claim of global or asymptotic stability is made; the condition is employed strictly as a local, instantaneous energy-based safeguard against destabilizing learning-induced corrections.

3.6.5 Practical Considerations and Conservativeness

The proposed supervisory mechanism is intentionally conservative. By evaluating the Lyapunov condition under an almost worst-case disturbance-doubling assumption, the supervisor may reject corrections that would in practice be beneficial. However, this is aimed at ensuring safety under severe learning errors, noise, and modeling uncertainty.

Additional conservative elements are incorporated at the implementation level to further enhance robustness. Specifically, the magnitude of the learned correction torque is explicitly bounded, and the applied correction is saturated to 0.015 N m. Moreover, the total commanded body torque is limited to 0.05 N m, corresponding to the maximum torque capability assumed for the reaction wheels.

Finally, the learning-based correction is disabled in a neighborhood of the equilibrium attitude. When the attitude error falls below one degree, the PINN output is suppressed, preventing the injection of high-frequency or unnecessary corrective action in the vicinity of the target attitude. This design choice reduces jitter and numerical noise near convergence while preserving the nominal NMPC behavior in steady-state conditions.

While this approach provides strong safety guarantees, it may unnecessarily limit the influence of the learning module in benign operating conditions. Reducing this conservativeness—by incorporating probabilistic confidence measures, adaptive margins, or less pessimistic disturbance models—constitutes an important direction for future work.

3.7 Closed-Loop Summary and Interface to Implementation and Simulation

This chapter introduced a learning-augmented nonlinear model predictive control architecture for underactuated spacecraft attitude regulation. The closed-loop system consists of a nominal NMPC controller, a Physics-Informed Neural Network (PINN) providing residual torque estimates to improve prediction fidelity, and a Lyapunov-based supervisory layer that regulates the influence of the learned correction.

The PINN does not act directly on the plant but modifies the effective control torque through a safety-filtered correction term. The supervisory layer enforces a local, instantaneous Lyapunov monotonicity condition under conservative check, ensuring that learning-induced corrections cannot degrade the closed-loop stability properties achieved by the nominal NMPC. No claims of global or asymptotic stability are made; the supervisory mechanism is employed strictly as an energy-based safeguard.

The following chapters present the implementation of the proposed control architecture and its evaluation through high-fidelity simulations. First, the implementation details of the NMPC, PINN, and supervisory layer within the simulation framework are described. Subsequently, numerical simulation studies are used to assess the nominal NMPC performance under underactuation, the benefits of residual learning in the presence of inertial mismatch, and the robustness of the supervisory layer

under adverse and out-of-distribution operating conditions.

Chapter 4

Control Architecture Implementation and Simulation Framework

4.1 Introduction to the Simulation Framework (Basilisk)

This section introduces the simulation framework adopted throughout this thesis, namely the Basilisk astrodynamics simulation environment [15]. A concise overview of its design philosophy, internal architecture, advantages, limitations, and role within this work is provided to support the simulation and implementation chapters that follow.

4.1.1 Overview of the Basilisk Framework

In this part, a brief description of the logic of the Basilisk framework is presented. Basilisk is a modular astrodynamics simulation framework designed to support high-fidelity spacecraft simulations.

The Basilisk framework is an open-source Python package developed to simulate the complex dynamical behavior of spacecraft and their interactions with the environment. Its main advantages include modularity, a high-performance simulation engine implemented in C/C++, and a user-friendly interface enabled by a simplified Python wrapper, to mention but a few. This is the core simulation tool used consistently throughout the thesis.

4.1.2 Simulation Architecture and Scheduling

The Basilisk simulation engine is built around a **message-passing architecture**. Its design separates computation, communication, and scheduling into distinct layers, enabling flexible, multi-rate, and high-fidelity spacecraft simulations.

The simulation engine operates through a hierarchy of components:

```
SimBaseClass
 +- Process (container for tasks)
 |    +- Task (container for modules)
 |    |    +- Module (the building blocks)
```

The scheduling system is managed by the `SimulationBaseClass`, available through:

```
from Basilisk.utilities import SimulationBaseClass
```

This class creates the simulation engine and provides the core capabilities of:

- creating processes and tasks,
- adding modules to tasks,
- initializing and executing simulations.

A **process** groups related tasks (e.g., dynamics or flight software), and a **task** groups modules that are executed at a specific rate. Each process runs independently, allowing multi-rate execution.

4.1.3 Modules and Message-Passing Mechanism

Modules encapsulate the behaviors and components of the spacecraft. Each module may include several input and output **message** connections, allowing them to be assembled into a complete spacecraft simulation.

Every Basilisk module defines its internal behavior through three main routines:

- `__init__()` — connects output messages to their payloads (Python-based modules),
- `Reset()` — reinitializes internal states, checks configuration, and reads static inputs,
- `UpdateState()` — executes the main computational logic at each simulation step.

A message in Basilisk is a shared block of memory containing a specific data structure, referred to as a **payload**. Publisher modules write to this shared memory, while subscriber modules read from it. This enables fast, zero-copy communication between modules.

In this work, two custom Basilisk modules are developed and integrated into the simulation framework. The first implements the nonlinear model predictive control (NMPC)–based attitude reference generation and control logic, while the second encapsulates the Physics-Informed Neural Network (PINN) together with a Lyapunov-based supervisory mechanism for safe residual torque correction.

4.1.4 Rationale for Selecting Basilisk

There are several advantages to using Basilisk as the primary simulation tool compared to alternative platforms. In addition to being open-source, Basilisk provides a Python interface that enables seamless integration with widely used scientific and machine learning libraries such as PyTorch and CasADi. These capabilities are required for the neural-network–based disturbance estimation and the nonlinear model predictive control (NMPC) components developed in this work.

Moreover, Basilisk supports the coherent integration of modules implemented in Python and C/C++, allowing computationally intensive components to be executed efficiently while maintaining high-level flexibility at the system-integration level.

Furthermore, Basilisk offers a comprehensive set of ready-to-use simulation modules and building blocks, eliminating the need to re-implement standard spacecraft subsystems. This allows the user to focus on the development of custom modules and on system-level aspects of the spacecraft and simulation architecture.

4.1.5 Limitations of the Simulation Framework

Despite the aforementioned advantages, Basilisk also presents some limitations. First, effective use of the framework requires familiarity with Python, C, and C++, as well as with the internal building blocks needed to assemble a simulation. This initial learning curve may be non-negligible for new users.

Additionally, the creation of custom messages (defined as C-structure payloads) and modules implemented in C/C++ requires recompilation of the source code. This, however, is not required for modules implemented purely in Python, which offer greater flexibility during rapid prototyping.

4.1.6 Basilisk Utilization in This Work

In this work, Basilisk is adopted as the core high-fidelity simulation environment supporting all stages of analysis, design, and validation of the proposed spacecraft attitude control architecture. Rather than being used solely as a numerical propagator, Basilisk serves as a unifying platform in which spacecraft dynamics, environmental disturbances, control algorithms, and learning-based components are coherently integrated and evaluated under realistic operating conditions.

Specifically, Basilisk is employed for the following purposes:

1. **High-fidelity spacecraft and environment simulation**: The spacecraft rigid-body dynamics, orbital motion, and environmental disturbance sources (e.g., gravity-gradient, atmospheric drag, and external torques) are simulated using Basilisk's validated dynamics and environment modules.

2. **Dataset generation for learning-based disturbance modeling**: Basilisk simulations are used to generate time histories of states and control inputs,

forming the training and validation datasets for the Physics-Informed Neural Network (PINN) disturbance predictor.

3. **Model-in-the-Loop (MIL) controller development and tuning**: The baseline model predictive controller (MPC) is designed, tuned, and initially validated within a Model-in-the-Loop framework while preserving consistency with the nonlinear spacecraft dynamics.

4. **Integration and testing of hierarchical control architectures**: Basilisk enables the integration of the baseline MPC, the PINN disturbance predictor, and the Lyapunov-based supervisory module within a single simulation environment, allowing direct comparison under identical conditions.

Through this workflow, Basilisk acts as the common experimental backbone of the thesis, ensuring consistency across controller design, learning-based model development, and performance evaluation.

Having introduced the simulation framework and its architectural principles, the next chapter focuses on the physical modeling of the spacecraft and its interaction with the orbital environment.

4.2 Simulation Architecture Overview

4.2.1 Process–Task–Module Hierarchy in the simulation

The complete execution hierarchy of the simulation is summarized below. Each module is identified by its Basilisk class name and associated `ModelTag`, and is executed within a specific task at a predefined rate.

```
simProcess
|
+- simTask (1 kHz)
|  |
|  +- spacecraft.Spacecraft              [ModelTag: "sat"]
|  |   |
|  |   +- ReactionWheelStateEffector     [ModelTag: "RW_cluster"]
|  |   +- SpinningBodyOneDOFStateEffector[ModelTag: "panel1"]
|  |   +- SpinningBodyOneDOFStateEffector[ModelTag: "panel2"]
|  |   +- GravityGradientEffector        [ModelTag: "gravGrad"]
|  |
|  +- ExponentialAtmosphere              [ModelTag: "ExpAtmo"]
|
+- dragTask (1 kHz)
|  |
|  +- FacetDragDynamicEffector           [ModelTag: "DragEff"]
```

```
|
+- NMPC (2 Hz)                              [ModelTag: "nmpcTask"]
|
+- pinnFastTask (20 Hz)                     [ModelTag: "pinnFastTask"]
   |
   +- PINNLyapunovSupervisorNARX
```

This hierarchy reflects a clear separation between spacecraft dynamics, environmental modeling, control computation, and learning-based supervision, while enabling their interaction through Basilisk's message-passing infrastructure. All subsequent sections explicitly reference the tasks and modules introduced here.

As for the task execution frequencies, Table 4.1 summarizes the execution rates and primary responsibilities of each task defined in the simulation.

Table 4.1: Simulation tasks and execution rates

Task name	Execution rate	Primary function
`simTask`	1 kHz	Spacecraft dynamics and environmental propagation
`dragTask`	1 kHz	Aerodynamic drag computation
`nmpcTask`	2 Hz	Nonlinear model predictive control computation
`pinnFastTask`	20 Hz	Learning-based torque correction

4.2.2 Multi-Rate Execution Rationale

The adopted multi-rate organization reflects both physical modeling requirements and computational constraints. The rationale behind this structure is summarized as follows:

- High-frequency tasks (`simTask`, `dragTask`) propagate spacecraft dynamics and environmental effects with sufficient temporal resolution.
- The control task (`nmpcTask`) executes the nominal NMPC at a lower rate, dictated by the computational cost of solving the nonlinear optimization problem.
- The supervisory modul, which includes the PINN, is executed in a dedicated task (`pinnFastTask`) to remain consistent with the assumptions adopted during PINN training, as discussed in the section on PINN training.

This architecture allows the integration of model-based control and learning-based components within a single deterministic simulation framework.

4.3 Spacecraft Plant Modeling (simTask)

This section describes the high-fidelity spacecraft plant model implemented in Basilisk and used as the truth model for all closed-loop simulations. The plant is realized

through multiple simulation modules, each responsible for a specific subset of the physical modeling. In particular, the spacecraft rigid-body dynamics, actuator dynamics, articulated appendages, and gravity-gradient effects are executed within `simTask`, while aerodynamic drag effects are evaluated in a dedicated task, `dragTask`.

4.3.1 Spacecraft Dynamics and Internal Effectors (`simTask`)

The core spacecraft dynamics are propagated within `simTask`, which executes at 1 kHz. This task contains the `spacecraft.Spacecraft` module and all associated state and dynamic effectors that directly contribute to the rigid-body equations of motion.

The spacecraft is modeled using Basilisk's component-based paradigm, in which a rigid central hub defines the primary body dynamics, while additional state and dynamic effectors contribute inertia, momentum storage, and external torques. Within `simTask`, the spacecraft hub integrates the rotational equations of motion using the forces and torques provided by the attached effectors.

4.3.1.1 Rigid Hub and Inertia Properties

The spacecraft rigid body is instantiated using the `spacecraft.Spacecraft` class (ModelTag: `sat`). The body-fixed reference frame $\mathcal{B}$ is attached to the hub, with its origin coinciding with the hub center of mass. The inertia tensor is specified as a full, generally non-diagonal matrix expressed in the body frame, allowing asymmetric mass distributions and cross-axis coupling effects to be captured explicitly.

Table 4.2: Spacecraft hub inertia matrix expressed in the body frame

Inertia component	**Value** $[\mathrm{kg\,m^2}]$
J_{xx}	5.15
J_{yy}	2.56
J_{zz}	6.11
$J_{xy} = J_{yx}$	0.040
$J_{xz} = J_{zx}$	0.0024
$J_{yz} = J_{zy}$	0.008

The presence of non-zero products of inertia reflects a realistic spacecraft configuration rather than an idealized symmetric rigid body and plays a non-negligible role in the resulting attitude dynamics.

4.3.1.2 Reaction Wheel Cluster

Attitude control authority is provided by a cluster of two reaction wheels aligned with the x and y body axes, resulting in an underactuated configuration with no direct torque authority about the body z axis. The wheels are implemented using the `ReactionWheelStateEffector` module (ModelTag: `RW_cluster`) and are executed

within `simTask`.

As state effectors, the reaction wheels introduce additional dynamic states associated with rotor angular momentum. These states are explicitly propagated and participate in the conservation of total system angular momentum, yielding a physically consistent representation of hub–wheel coupling.

Table 4.3: Reaction wheel cluster parameters

Parameter	Value
Number of wheels	2
Wheel orientation	Body-aligned (x, y)
Maximum wheel speed	6000 rpm
Maximum stored momentum	0.9 N m s
Wheel rotor inertia	1.43×10^{-3} kg m^2
Maximum commanded torque	0.05 N m
Minimum realizable torque $u_{\min}$	8×10^{-6} N m
Coulomb friction torque f_{C}	3×10^{-5} N m

4.3.1.3 Articulated Appendages

Two articulated appendages representing deployable solar panels are modeled using `SpinningBodyOneDOFStateEffector` modules (ModelTags: `panel1` and `panel2`). These state effectors are executed within `simTask` and introduce additional rotational degrees of freedom relative to the hub.

Although not actively controlled, the appendages contribute inertia coupling effects that influence the spacecraft attitude dynamics and improve the physical realism of the plant model.

4.3.1.4 Gravity-Gradient Torque

Gravity-gradient effects are modeled using the `GravityGradientEffector` module (ModelTag: `gravGrad`), which is executed within `simTask`. This module computes the gravity-gradient torque acting on the spacecraft based on the hub inertia tensor and the instantaneous orbital position.

For a rigid spacecraft in a central gravity field, the gravity-gradient torque magnitude is bounded by

$$\|\boldsymbol{\tau}_{gg}\| \lesssim \frac{3\mu}{r^3}\Delta J, \tag{4.1}$$

where μ is the Earth gravitational parameter, r is the orbital radius, and $\Delta J = \max(\lambda_i(\mathbf{J})) - \min(\lambda_i(\mathbf{J}))$ is the difference between the principal moments of inertia.

The torque computed by the `GravityGradientEffector` is injected into the spacecraft hub as an external disturbance and is fully coupled with the rigid-body

dynamics propagated in `simTask`.

4.4 Operational Orbit Definition

All simulations are conducted in a representative low Earth orbit (LEO) regime. The spacecraft orbit is initialized using classical Keplerian orbital elements and propagated under a central-body point-mass gravity model, with Earth treated as the sole gravitating body.

The orbital elements used for initialization are summarized in the following table:

Table 4.4: Initial orbital elements of the operational orbit

Orbital element	**Value**
Semi-major axis a	7000 km
Eccentricity e	1×10^{-4}
Inclination i	33.3°
Right ascension of ascending node Ω	48.2°
Argument of perigee ω	347.8°
Initial true anomaly f_0	85.3°

The orbit is converted to inertial position and velocity vectors using standard Keplerian-to-Cartesian transformations and assigned as the initial translational state of the spacecraft hub. Orbital propagation is performed using Earth's gravitational parameter and neglects higher-order gravitational harmonics, consistent with the focus on attitude dynamics and disturbance modeling rather than precise orbit determination.

4.5 Aerodynamic Drag Modeling (`dragTask`)

Aerodynamic drag effects are computed in a dedicated task, `dragTask`, which executes at 1 kHz. This task contains the `FacetDragDynamicEffector`, responsible for evaluating aerodynamic forces and torques based on a faceted geometric representation of the spacecraft, the local atmospheric density, and the spacecraft velocity relative to the incoming flow.

4.5.0.1 Facet-Based Spacecraft Geometry

The spacecraft bus is modeled using a faceted geometric representation composed of six planar faces. Each face is characterized by its exposed area, outward surface normal, and force-application-point (FAP) location expressed in the body frame.

The separation of aerodynamic drag computation into `dragTask` allows the drag model to be evaluated independently from the spacecraft dynamics propagation, while the resulting force and torque contributions are injected into the spacecraft through Basilisk's message-passing and effector interfaces.

Table 4.5: Spacecraft body facet geometry used for aerodynamic drag modeling

Face	**Area** [m^2]	**Normal**	**FAP location** [m]
$+x$	0.256	[1, 0, 0]	[0.25, 0, 0]
$-x$	0.256	[−1, 0, 0]	[−0.25, 0, 0]
$+y$	0.320	[0, 1, 0]	[0, 0.20, 0]
$-y$	0.320	[0, −1, 0]	[0, −0.20, 0]
$+z$	0.200	[0, 0, 1]	[0, 0, 0.32]
$-z$	0.200	[0, 0, −1]	[0, 0, −0.32]

4.5.0.2 Solar Panel Facet Geometry

Each deployable solar panel is modeled using four planar facets to represent the upper and lower surfaces of the two panels. Although the panels are physically thin, modeling both surfaces ensures that aerodynamic forces are applied consistently regardless of panel orientation relative to the incoming flow.

Table 4.6: Solar panel facet geometry used for aerodynamic drag modeling

Facet	**Area** [m^2]	**Normal**	**FAP location** [m]
Panel 1 (lower)	0.22	[0, 0, −1]	[−0.25, −0.45, 0]
Panel 2 (lower)	0.22	[0, 0, −1]	[−0.25, 0.45, 0]
Panel 1 (upper)	0.22	[0, 0, 1]	[−0.25, −0.45, 0]
Panel 2 (upper)	0.22	[0, 0, 1]	[−0.25, 0.45, 0]

4.5.0.3 Aerodynamic Model Parameters

The aerodynamic drag model parameters used by the `FacetDragDynamicEffector` are summarized in Table 4.7. The total aerodynamic force and torque acting on the spacecraft are obtained by summing the contributions of all modeled facets.

Table 4.7: Aerodynamic model parameters

Parameter	**Value**
Atmospheric density model	Exponential atmosphere
Drag coefficient C_D	2.5
Total number of drag facets	10

The drag coefficient is conservatively assumed constant and set to $C_D = 2.5$. While a nominal value of $C_D \approx 2.2$ is often adopted for low Earth orbit spacecraft, reported drag coefficients span a broader range depending on surface properties, gas–surface interaction assumptions, and altitude. In particular, increased uncertainty and higher effective drag coefficients are expected at higher LEO altitudes due to reduced surface contamination and a greater contribution of quasi-specular reflection effects [16].

4.6 NMPC Implementation (nmpcTask)

The nonlinear model predictive controller is implemented as a *custom Basilisk module* in Python, hereafter referred to as `nmpcTask`. The purpose of this module is to provide a real-time executable realization of the NMPC formulation developed in Chapter 3, while satisfying the numerical robustness, timing, and safety requirements imposed by closed-loop spacecraft attitude control simulations.

The `nmpcTask` is executed within the Basilisk process–task–module architecture described in Section 4.2. It interfaces with the simulation environment exclusively through Basilisk message passing, receiving spacecraft and reaction wheel state estimates as well as reference commands, and producing either reaction wheel torque commands or reference trajectories depending on the selected execution mode.

The design of the module explicitly supports both standalone operation of the NMPC controller and its integration within the hierarchical control architecture introduced in Chapter 2. This dual-mode capability ensures that the NMPC controller remains a self-contained baseline policy, while allowing learning-based augmentation and supervisory filtering to be enabled without modification of the core NMPC logic.

4.6.1 Numerical Solver and Software Tools

The finite-horizon nonlinear optimal control problem solved by the NMPC controller is implemented using the *CasADi* symbolic optimization framework [17]. CasADi is used to construct the nonlinear program (NLP) by encoding the discrete-time prediction model, cost function, and constraints described in Chapter 3, while providing automatic differentiation required for efficient gradient-based optimization.

The resulting NLP is solved using the IPOPT interior-point solver [18]. A limited-memory quasi-Newton (L-BFGS) approximation of the Hessian matrix is employed to reduce memory requirements and to improve robustness under frequent warm starts. Full second-order derivative information is deliberately avoided, as it was found to offer limited benefit relative to its computational cost for the problem dimensions considered.

All symbolic expressions defining the NLP are generated once during module initialization. During runtime execution, the solver graph remains fixed and only numerical parameters—including the initial state, reference state, previous control input, and adaptive weighting parameters—are updated. This design avoids repeated symbolic expansion and ensures predictable and repeatable computational behavior.

4.6.2 Real-Time Execution and Computational Constraints

The `nmpcTask` is executed periodically with a fixed sampling interval T_{NMPC}, selected based on the time-scale considerations discussed in Section 3.3.1. To ensure compatibility with real-time execution constraints, the IPOPT solver is subject to a strict execution-time limit.

Specifically, the maximum allowable solver CPU time is constrained to

$$t_{\mathrm{CPU}}^{\max} = 0.8\, T_{\mathrm{NMPC}}, \tag{4.2}$$

leaving sufficient margin for message handling, numerical preprocessing, and task scheduling overhead within the Basilisk framework.

The controller does not assume guaranteed convergence of the nonlinear solver at every invocation. Instead, solver return status is explicitly monitored and incorporated into the control logic. This design choice reflects realistic onboard operation, where deterministic execution and bounded outputs are prioritized over strict optimality at each control update.

4.6.3 Warm-Start Strategy and Failure Handling

To improve convergence reliability and reduce average computation time, the NMPC solver is warm-started whenever possible using the solution obtained at the previous control update. The previously computed optimal state and control trajectories are shifted forward by one prediction step and used as the initial guess for the current optimization problem.

Prior to reuse, all quaternion states contained in the warm-start trajectory are explicitly renormalized to enforce the unit-norm constraint and to prevent accumulation of numerical drift across iterations. The warm-start solution is discarded if numerical pathologies such as NaN or infinite values are detected.

If a valid warm start is unavailable, the solver is initialized using a cold-start strategy based on the current measured state and a zero control sequence. In the event of repeated solver failures, a complete solver rebuild is triggered, reinitializing the optimization problem and clearing all internal solver memory.

When no valid solution can be obtained, the `nmpcTask` applies a deterministic fallback action. In this case, the next predicted state is taken from the most recent valid solution when available, otherwise from the current measured state, and the commanded reaction wheel torque is set to zero. This guarantees bounded control action and continuity of operation under adverse numerical conditions.

4.6.4 Execution Modes and Integration with the Hierarchical Architecture

The `nmpcTask` supports two execution modes, corresponding to whether learning-based augmentation and supervisory filtering are enabled in the closed-loop architecture.

When the learning-based components are *not* included in the closed loop, the NMPC module operates in standalone mode. In this configuration, the control input computed by the NMPC optimizer is written directly to the reaction wheel actuator interface. The NMPC controller therefore acts as the sole authority responsible for closed-loop attitude regulation, constraint enforcement, and actuator command generation.

When the Physics-Informed Neural Network (PINN) and the Lyapunov-based supervisory layer are enabled, the execution mode of the NMPC module is modified accordingly. In this configuration, the NMPC controller does *not* issue direct commands to the reaction wheels. Instead, it outputs a nominal reaction wheel torque command, interpreted as a baseline model-based control action. This nominal command is passed to the PINN–Supervisor module through a dedicated messaging interface. In this mode, the final command applied to the reaction wheels is determined exclusively by the supervisor, which may either pass the nominal NMPC command unchanged or apply a bounded correction. The NMPC module remains agnostic to this decision and does not receive feedback regarding whether its nominal command was modified.

This implementation is intentionally designed to avoid the duplication of control modules. Rather than defining separate NMPC modules for the standalone and learning-augmented configurations, a single NMPC module is employed whose execution mode is selected through a key–value configuration parameter.

4.6.5 Quaternion Handling and Numerical Robustness

Special care is devoted to quaternion handling to ensure numerical robustness of the NMPC implementation. Although unit-norm preservation is theoretically guaranteed by the continuous-time kinematics, discretization, finite precision arithmetic, and solver warm-start reuse introduce small violations that must be corrected explicitly. Quaternion normalization is therefore enforced after numerical integration within the prediction model, after warm-start trajectory shifting, and after solution extraction from the NLP solver.

To explicitly address underactuation at the implementation level, the attitude and angular-rate error vectors are decomposed into components parallel and orthogonal to the actuator subspace by means of orthogonal projection matrices derived from the

actuator distribution matrix. The weighting applied to the null-space component is modulated, as described in section 3.3.4, by a scheduling scalar parameter computed as a smooth function of the current attitude error magnitude. This parameter is evaluated once per NMPC sampling period and held constant over the prediction horizon, preserving the smoothness of the resulting nonlinear program while enabling selective penalization of the error component associated with the underactuated directions.

These measures collectively ensure that the `nmpcTask` remains numerically well-conditioned, robust to large initial attitude deviations, and suitable for long-horizon closed-loop simulations within the Basilisk framework.

4.7 Data Collection for Learning-Based Residual Modeling

This section describes the offline data generation procedure adopted for training the Physics-Informed Neural Network (PINN) used as a residual torque compensator. The objective of the data collection process is not full system identification, but rather the exposure of structured model–plant mismatch within the closed-loop operational regime enforced by the nominal NMPC controller. Particular emphasis is placed on time-scale separation, sampling consistency, and strict temporal causality, which are essential for reliable sequential learning and subsequent online deployment.

4.7.1 Simulation Campaign and Initial Condition Sampling

The offline training dataset is generated through a large-scale simulation campaign consisting of 750 independent nonlinear simulations. Each simulation is initialized from a distinct spacecraft attitude expressed in Modified Rodrigues Parameters (MRPs), which constitute the internal attitude representation adopted within the Basilisk simulation framework. The initial angular velocities and reaction wheel speeds are set close to zero, consistent with rest-to-rest attitude maneuvering scenarios.

The use of MRPs allows each initial condition to be interpreted geometrically in terms of a rotation axis and a rotation angle. Given an MRP vector $\boldsymbol{\sigma} \in \mathbb{R}^3$, the corresponding axis–angle representation is

$$\hat{\mathbf{e}} = \frac{\boldsymbol{\sigma}}{\|\boldsymbol{\sigma}\|} \in \mathbb{S}^2, \qquad \theta = 4 \arctan(\|\boldsymbol{\sigma}\|), \tag{4.3}$$

where $\hat{\mathbf{e}}$ denotes the rotation axis and $\theta \in (0, \pi)$ the principal rotation angle.

The initial condition set is constructed to provide broad directional coverage on the unit sphere, ensuring that attitude errors are distributed across all body-fixed axes. This isotropic distribution of rotation axes prevents directional bias in the

learning process and exposes the controller and learning module to a wide range of axis-dependent dynamic couplings.

In addition to directional coverage, the set deliberately includes large rotation angles. A significant portion of the initial conditions corresponds to rotations close to $\theta = \pi$. These configurations represent the most demanding maneuvering scenarios for the underactuated spacecraft, as they require the NMPC controller to exploit nonlinear coupling and momentum exchange to reduce the attitude error before convergence to the reference manifold.

A smaller subset of initial conditions covers moderate rotation angles, providing additional diversity in transient behavior while maintaining focus on the large-error regime. This mixed sampling strategy is designed to concentrate data collection in regions of the state space where model–plant mismatch and prediction errors are most pronounced, thereby improving the relevance of the collected data for residual torque learning.

Overall, the resulting set of 750 MRP initial conditions ensures both geometric diversity and sufficient excitation of challenging nonlinear dynamics.

4.7.2 Time-Scale Separation and Sampling Strategy

During the simulations, state and input trajectories are subsequently sampled at a lower rate

$$\Delta t_{\text{sample}} = 0.05 \text{ s},$$

corresponding to a sampling frequency of 20 Hz. These sampled signals constitute the interface between the simulated plant, the NMPC controller, and the learning module.

Consequently, within each NMPC period, ten sampled points are available. As described in the subsequent sections, during training, these samples are used to numerically roll out the nominal spacecraft model over one NMPC interval in order to evaluate the physics-informed loss. The rollout is performed by integrating the model dynamics over the H substeps, while the loss is computed only at the end of the NMPC period as the angular velocity prediction error $\omega(t + T_s) - \omega(t + T_s)_{\text{true}}$.

This choice of time-scale separation serves two purposes. First, the use of multiple integration substeps within each NMPC interval improves the numerical accuracy of the state propagation under nonlinear rotational dynamics and zero-order-held inputs. Second, since the loss is evaluated only at the macro-time scale and subsequently averaged over NMPC steps, high-frequency sampling noise does not enter the learning objective directly, but only through its accumulated effect over the NMPC period. Consequently, the dominant learning signal reflects coherent, low-frequency prediction errors relevant for control, while fast fluctuations are naturally attenuated.

In summary, the adopted sampling strategy ensures consistency between the NMPC update rate, the learning objective, and the numerical integration accuracy. The 20 Hz sampling rate provides sufficient temporal resolution for accurate model rollout within each NMPC period, while the 2 Hz controller update rate captures the slow time scale of reaction-wheel-driven attitude dynamics targeted by the learning-based residual compensation.

4.7.3 Dataset Construction and Preprocessing

The training dataset for the Physics-Informed Neural Network (PINN) is constructed entirely from simulation logs generated within the Basilisk framework, as described in Section 4.7. Each simulation run corresponds to a closed-loop execution of the nominal NMPC controller acting on the full nonlinear spacecraft model, including environmental disturbances and modeling uncertainties. The objective of the dataset construction procedure is to extract temporally consistent input–state–output sequences suitable for offline training of a NARX-type PINN that estimates the lumped residual disturbance torque introduced in Section 3.2.4.

4.7.3.0.1 Raw simulation data. For each 90-second simulation run, the following signals are recorded at a fixed integration time step $\Delta t = 0.05\,\mathrm{s}$:

- spacecraft attitude quaternion $q(k) \in \mathbb{R}^4$,
- body angular velocity $\omega(k) \in \mathbb{R}^3$,
- reaction wheel angular velocities $\omega_{\mathrm{RW}}(k) \in \mathbb{R}^2$,
- commanded reaction wheel motor torques $u_{\mathrm{RW}}(k) \in \mathbb{R}^2$.

Multiple simulation folders are loaded and concatenated into a single dataset. A simulation identifier is preserved for each sample in order to explicitly track temporal boundaries between runs during subsequent sequence extraction.

4.7.3.0.2 Feature vector construction. At each discrete time instant k, a nonlinear feature vector $\phi(k) \in \mathbb{R}^{14}$ is constructed using measured quantities only. The feature vector includes:

- the angular velocity components $\omega(k)$,
- quadratic and cross-product terms in $\omega(k)$ to expose nonlinear coupling effects,
- the commanded reaction wheel torques $u_{\mathrm{RW}}(k)$,
- a one-step memory of the angular velocity $\omega(k-1)$.

No quaternion components are provided as direct inputs to the neural network. Instead, the quaternion is treated exclusively as part of the physics state used during rollout of the dynamics inside the loss function, which will be discussed in the following section.

4.7.3.0.3 Physics state and target definition. In addition to the feature vectors, a physics state

$$x_{\text{dyn}}(k) = \begin{bmatrix} q(k)^\top & \omega(k)^\top & \omega_{\text{RW}}(k)^\top \end{bmatrix}^\top$$

is stored at each time step. During training, this state is propagated forward using the discrete-time rigid-body dynamics with reaction wheels and a candidate feedforward residual torque predicted by the PINN. In the training procedure, no explicit labels are used, Instead, learning is driven indirectly by minimizing the prediction error on the angular velocity after a fixed macro-step horizon, as explained in section 3.4.

4.7.3.0.4 Sequential dataset construction. To enable NARX-style learning and preserve temporal causality, the dataset is organized into sequences of fixed length $L = 50$ sampling steps. The NMPC update period is $T_s = 0.5\,\text{s}$, corresponding to a rollout horizon of $H = T_s/\Delta t = 10$ integration steps. Within each sequence, the loss is evaluated over $L - H = 40$ overlapping macro-steps, each consisting of an H-step physics rollout followed by a comparison with the measured angular velocity at time $k + H$.

Sequences are extracted from each simulation run using a sliding window with a fixed stride of 10 samples. As a result, consecutive sequences share 40 time steps of raw data. This overlap occurs only at the data level. From a learning perspective, sequences are treated as independent training samples: the NARX memory state is reset at the beginning of each sequence, no temporal continuity is enforced across sequences, and no gradient information propagates between them.

Strict safety checks are enforced during sequence extraction. No sequence is allowed to cross simulation boundaries, and any sequence containing invalid (NaN or infinite) values is discarded. Simulation runs that do not yield at least one valid sequence are excluded entirely. These constraints guarantee that the NARX memory can be safely reset at sequence boundaries without introducing spurious temporal correlations.

After applying all validity checks and selection criteria, the resulting dataset contains a total of 130,650 training sequences, each of length $L = 50$.

4.7.3.0.5 Normalization and data partitioning. Feature normalization is performed using statistics computed exclusively on the training subset of the data, preventing information leakage into validation. The dataset is split at the sequence level, with approximately 80% of the sequences used for training and 20% reserved for validation. No learning-state continuity exists between training and validation sequences.

4.7.3.0.6 Physical scaling of network outputs. Although the PINN outputs a dimensionless quantity internally, the predicted residual torque is scaled by predefined per-axis bounds reflecting a-priori physical insight on disturbance magnitude. This scaling improves numerical conditioning during training, enforces output regularity, and facilitates safe integration with the Lyapunov-based supervisory layer.

Overall, the dataset construction and preprocessing pipeline ensures temporal consistency, physical plausibility, and learning stability, while aligning the training objective directly with the prediction accuracy required by the NMPC controller. This design is essential for reliable offline training and safe online deployment of the learning-based residual compensator.

4.8 Physics-Informed Neural Network Training and Deployment

This section describes the offline training procedure of the Physics-Informed Neural Network (PINN) introduced in Chapter 3, as well as its offline validation. The focus here is exclusively on the learning problem defined on the preconstructed dataset, without addressing closed-loop performance or controller interaction, which are instead analyzed in Chapter 5.

4.8.1 Network Interface and Architecture

The PINN implements a nonlinear mapping from measured features and internal memory to a residual disturbance torque acting on the spacecraft rotational dynamics. The network input is a 16-dimensional vector composed of a 14-dimensional feature vector, constructed as described in Section 4.7.3, and a 2-dimensional NARX memory term corresponding to the previously predicted residual torque.

The network output is a 2-dimensional residual torque expressed in the body frame, corresponding to the lumped disturbance acting along the actuated directions of the spacecraft. Internally, the network produces a normalized output, which is subsequently scaled by fixed per-axis bounds reflecting conservative physical limits on the disturbance magnitude.

The network architecture consists of a fully connected feedforward neural network with three hidden layers of 128 neurons each and hyperbolic tangent activation functions. Temporal dependencies are handled explicitly through the NARX formulation, avoiding the use of recurrent layers and ensuring that all memory effects are transparent and bounded.

4.8.2 Physics-Informed Loss Formulation

At each macro time step t, the physics-informed neural network outputs a residual disturbance torque $\tau_{\text{PINN}}(x_t;\theta) \in \mathbb{R}^2$, which is assumed constant over the prediction horizon and injected additively into the rigid-body rotational dynamics.

Given the current state x_t, the nominal control input sequence $\{u_{t+h}\}_{h=0}^{H-1}$, and the predicted residual torque $\tau_{\text{PINN}}(x_t;\theta)$, the system dynamics are rolled out forward for H discrete integration steps using a fixed-step numerical integrator. The predicted angular velocity at the end of the horizon is obtained as

$$\omega_{t+H}^{\text{pred}} = \Phi_{\omega}^{(H)}\Big(x_t,\ \{u_{t+h}\}_{h=0}^{H-1},\ \tau_{\text{PINN}}(x_t;\theta)\Big), \tag{4.4}$$

where $\Phi_{\omega}^{(H)}(\cdot)$ denotes the composition of H discrete-time integration steps of the rigid-body rotational dynamics.

The network parameters θ are trained by minimizing a terminal prediction loss defined on the angular velocity at the end of the horizon,

$$\mathcal{L}(\theta) = \frac{1}{2}\left\|\omega_{t+H}^{\text{pred}}(\theta) - \omega_{t+H}^{\text{meas}}\right\|_2^2, \tag{4.5}$$

where $\omega_{t+H}^{\text{meas}}$ denotes the angular velocity measured from the high-fidelity simulation at the corresponding macro time step.

This loss formulation enforces physical consistency by constraining the effect of the learned residual torque to act exclusively through forward propagation of the rotational dynamics, rather than by direct regression on the torque itself.

4.8.3 Training Procedure and Hyperparameters

Training is performed offline on the sequential dataset described in Section 4.7.3. The dataset is split at the sequence level into training and validation subsets, with approximately 80% of the sequences used for training and 20% reserved for validation. Feature normalization statistics are computed exclusively on the training subset to prevent information leakage.

The network is trained for a maximum of 9 epochs using the AdamW optimizer with a learning rate of 5×10^{-3} and a decoupled weight decay of 10^{-5}. AdamW is selected due to its ability to regularize network parameters without interfering with adaptive gradient scaling, which is particularly important in physics-informed training where gradient magnitudes may vary significantly across parameters associated with different physical effects.

Training is performed using sequence-wise mini-batches of 8 samples, resulting in

approximately 1.3×10^4 mini-batch updates per epoch and a total of about 1.21×10^5 optimization steps over the full training procedure. The NARX memory state is explicitly reset at the beginning of each sequence to avoid unintended temporal coupling across independent data segments. No early stopping criterion is employed; instead, training duration is fixed a priori. This choice reflects the fact that prolonged optimization does not necessarily improve physical consistency in the absence of explicit labels and may amplify sensitivity to measurement noise.

Validation loss is monitored throughout training. It is observed that the validation loss may be lower than the training loss, which is not indicative of overfitting in this context. Since the loss measures physical rollout consistency rather than pointwise prediction error.

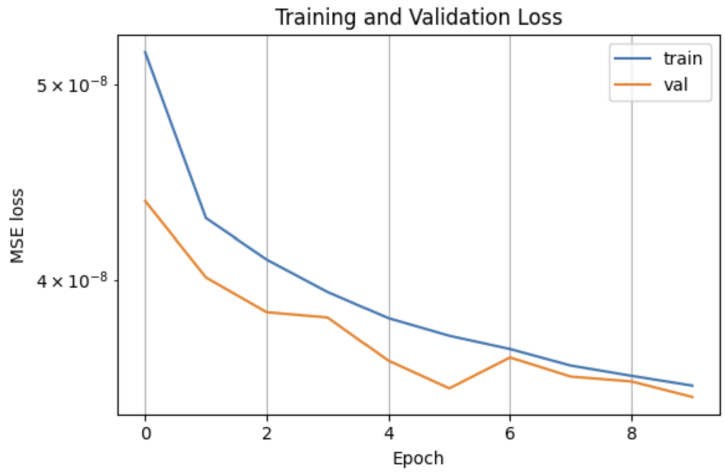

Figure 4.1: Training and validation loss as a function of epoch for the PINN.

4.8.4 Prediction Accuracy on the Loss Variable

This subsection presents a qualitative assessment of the prediction accuracy of the trained Physics-Informed Neural Network. The objective of this analysis is to visualize how closely the learned residual model reproduces the observed angular velocity evolution when embedded in the rigid-body dynamics.

The results shown in this subsection are obtained by evaluating the trained network offline on the validation dataset used during training loss monitoring. No additional validation or testing stage is introduced, and the data shown here do not influence training, hyperparameter selection, or model tuning. The effective impact of the learned residual model is instead assessed through closed-loop simulations of the full control architecture in Chapter 5.

For each validation sequence, the NARX memory state is reset at the beginning to prevent spurious temporal correlations across independent data segments. At each macro time step, the PINN predicts a residual disturbance torque, which is

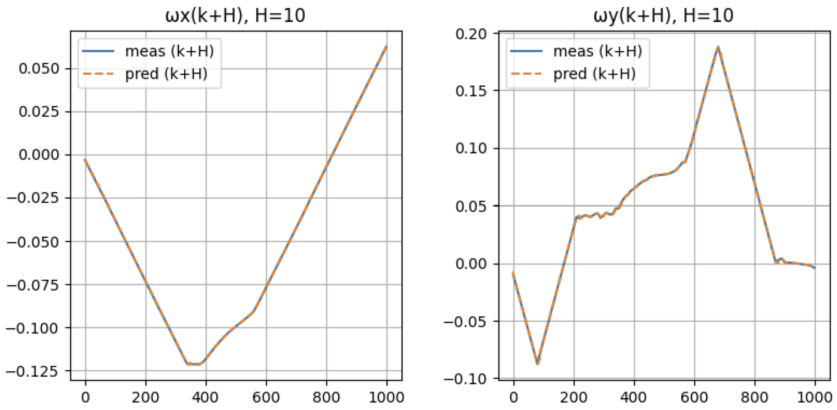

Figure 4.2: Prediction accuracy of angular velocity at the end of the NMPC horizon ($H = 10$). Measured and PINN-predicted angular velocities are shown for a representative subset of 1000 samples, plotted as a function of the discrete integration-step index.

held constant over the NMPC prediction horizon of $H = 10$ integration steps and injected into the rigid-body dynamics. The resulting angular velocity prediction at time $k + H$ is compared against the measured angular velocity from the dataset.

4.8.4.0.1 Angular velocity prediction at the end of the NMPC horizon. Figure 4.2 shows the predicted angular velocity components $\omega_x(k+H)$ and $\omega_y(k+H)$ compared to the corresponding measured values. The predictions are obtained by rolling out the nonlinear spacecraft dynamics from the measured state at time k, using the PINN-predicted residual torque held constant over the prediction horizon. The close agreement between predicted and measured angular velocities indicates that the learned residual torque provides dynamically consistent corrections to the nominal model along the actuated axes.

For clarity, only a representative subset of validation samples is shown. Although the evaluation is performed on the complete validation dataset, plotting all samples would result in severe overplotting and reduced interpretability. The horizontal axis corresponds to the discrete integration step index and can be mapped to physical time through the known sampling period Δt.

4.8.4.0.2 Predicted residual disturbance torque. Figure 4.3 illustrates the residual disturbance torque predicted by the PINN along the actuated body axes. These signals correspond to the network output evaluated sequentially on the validation data, with the NARX memory state updated using the previously predicted torque. The predicted profiles remain bounded within the predefined physical limits imposed during training and exhibit piecewise-smooth behavior, consistent with the assumed disturbance structure.

Overall, this analysis provides a direct visualization of the prediction accuracy achieved by the learned residual model on the loss variable used during training.

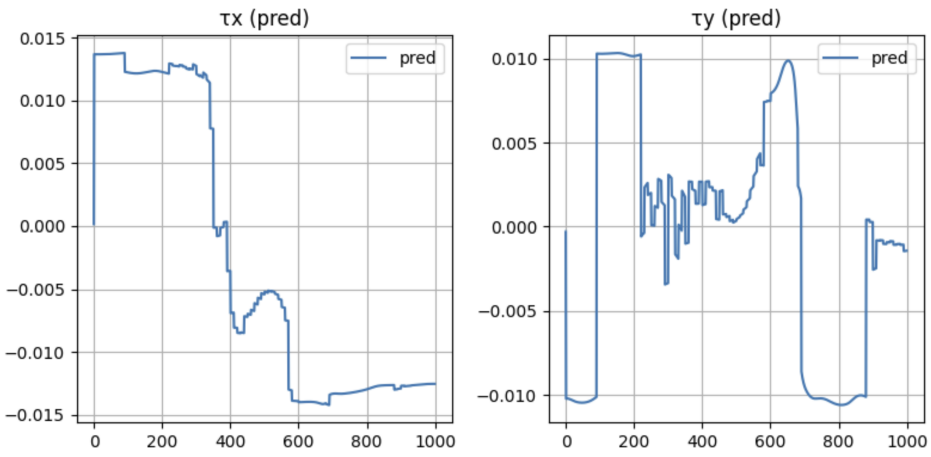

Figure 4.3: Residual disturbance torque predicted by the PINN on validation sequences for the actuated body axes. For clarity, only a representative subset of 1000 samples is shown.

These results serve as a consistency check prior to closed-loop evaluation of the full control architecture, which is presented in the subsequent validation chapter.

4.9 Supervisor-PINN Implementation

The Physics-Informed Neural Network (PINN) and the Lyapunov-based supervisory mechanism are implemented as a single custom Basilisk module executed within the dedicated task `pinnFastTask`. This module is responsible for evaluating learning-based residual torque corrections and for enforcing safety constraints before any modification of the nominal NMPC command is applied to the spacecraft actuators.

The module receives, through Basilisk message passing, the nominal reaction wheel torque command generated by the NMPC controller, the spacecraft attitude and angular velocity, the reaction wheel states, and the reference signals associated with the current NMPC solution. The output of the module is the final reaction wheel motor torque command transmitted to the spacecraft dynamics.

From an architectural standpoint, the supervisor does not act as an independent controller. Instead, it operates as a deterministic safety filter interposed between the nominal NMPC controller and the plant. The NMPC module remains fully agnostic to the presence of the learning-based augmentation and does not receive any feedback regarding whether its nominal command has been modified or left unchanged.

4.9.1 Fast–Slow Synchronization and PINN Evaluation

The `pinnFastTask` is executed at a higher rate (20 Hz) than the NMPC task (2 Hz), reflecting the need to evaluate supervisory logic at a frequency compatible with the simulation time step while preserving consistency with the assumptions adopted

during PINN training.

To reconcile this rate mismatch, a fast–slow synchronization mechanism is implemented. A discrete counter tracks the number of fast task executions elapsed since the last NMPC update. The PINN is evaluated only when the counter indicates that a new NMPC period has begun. At this instant, the network produces a candidate residual disturbance torque based on the current measured state and on its internal NARX memory. The resulting correction is then held constant over the subsequent fast execution steps until the next NMPC update.

This sample-and-hold strategy ensures that the learning-based correction is applied at the same effective rate used during offline training, while avoiding unnecessary neural network evaluations at intermediate simulation steps. During the fast-rate updates, the supervisor operates solely on the held correction, performing safety evaluation and scaling without recomputing the network output.

4.9.2 NARX Memory and Temporal Consistency

The PINN implements a nonlinear autoregressive structure with exogenous inputs (NARX), requiring access to a one-step memory of the system's recent behavior. To maintain strict temporal causality, the network input at each NMPC update includes the angular velocity measured one fast sampling interval earlier and the residual torque predicted at the previous NMPC step.

The internal memory is initialized at the beginning of the simulation using the current measured state and a zero residual torque. Subsequently, memory updates are performed exclusively at NMPC update instants. This design ensures that the network never accesses future information and that all predictions are based on quantities that would be available in a realistic onboard implementation.

By decoupling memory updates from the fast execution rate, the supervisor avoids spurious temporal correlations and preserves consistency between the training and deployment environments. The resulting behavior matches the discrete-time dynamics assumed during offline PINN training and validation.

4.9.3 Supervisor Execution Modes and Safety Enforcement

The Lyapunov-based supervisory module is implemented as an intermediate filter between the nominal NMPC command and the applied reaction wheel torque. Its primary function is to regulate the influence of the learning-based residual correction while preserving the stability and safety properties established by the baseline controller.

From an implementation standpoint, the supervisory module supports two execution modes, which differ only in the activation of the safety enforcement logic and do not alter signal interfaces, execution timing, or data flow within the control loop.

- **Active supervision mode:** the supervisory logic evaluates the candidate learning-based correction at each control update and enforces the Lyapunov-based safety condition by scaling or rejecting the residual torque contribution. This mode represents the nominal operational configuration of the proposed architecture.
- **Bypass mode:** the supervisory logic operates in a transparent pass-through configuration, allowing the PINN-generated residual torque to be applied directly in combination with the NMPC command. This mode is introduced exclusively for validation and ablation purposes, enabling isolation of the unconstrained performance contribution of the learning-based component. It does not represent a safety-certified or mission-ready configuration.

At each fast execution step, when active supervision is enabled, the supervisor evaluates whether the currently held learning-based correction can be safely combined with the nominal NMPC command. The candidate correction is first subjected to magnitude saturation, enforcing a conservative bound on the maximum admissible residual torque.

Additional gating logic suppresses the correction when the attitude error falls below a small threshold of 1 degree, preventing unnecessary learning-based intervention near the equilibrium and ensuring that steady-state behavior is governed by the model-based controller.

The core supervisory mechanism evaluates the effect of the correction on a Lyapunov-related quantity by explicitly computing its time derivative under the combined action of the nominal and corrected torques. A line-search procedure is used to scale the correction by a factor $\alpha \in [0, 1]$. Starting from the full correction, the supervisor progressively reduces α until a non-increase condition is satisfied. If no admissible scaling factor is found, the correction is rejected entirely.

When an admissible scaling factor exists, the accepted residual torque is mapped back to reaction wheel motor torques and added to the nominal NMPC command. The final command is saturated to respect actuator limits and transmitted to the spacecraft model. In the worst case, the supervisor outputs the unmodified NMPC command, reducing the architecture to a purely model-based closed loop without requiring mode switching or reconfiguration.

The execution mode of the supervisory module is selected through a configuration-level flag and can be modified without reinitializing or restructuring the control

architecture. In all modes, the supervisory module remains synchronized with the NMPC and PINN evaluation rates, ensuring consistent timing and comparability across validation scenarios.

4.9.4 Computational and Architectural Implications

The supervisory logic is computationally lightweight compared to the NMPC optimization and neural network evaluation. By restricting PINN execution to NMPC update instants and relying on simple algebraic operations during fast updates, the additional computational burden introduced by learning-based augmentation remains negligible.

Overall, the `pinnFastTask` provides a coherent runtime integration of learning-based residual compensation and deterministic safety enforcement, ensuring that performance improvements enabled by the PINN are realized only when they are compatible with the stability properties established by the nominal NMPC controller.

4.10 Closed-Loop Execution Summary

The proposed closed-loop control system operates as a hierarchical architecture integrating a model-based baseline controller, a learning-based residual compensation module, and a deterministic supervisory safety filter. All components execute within the Basilisk multi-task scheduling framework and communicate through fixed interfaces, ensuring consistent timing and signal flow across all execution configurations.

At each control update, the estimated spacecraft state is processed by the nonlinear model predictive controller (NMPC), which computes a constraint-aware nominal reaction wheel torque command based on a reduced-order predictive model. This command defines the baseline closed-loop behavior and guarantees nominal stability and constraint satisfaction under the assumptions introduced in the problem formulation.

In parallel, the physics-informed neural network (PINN) evaluates a candidate residual disturbance torque intended to compensate structured model mismatch and unmodeled dynamics. The PINN output is treated as an auxiliary correction signal and is not applied directly to the spacecraft model.

The nominal NMPC command and the learning-based correction are combined through a Lyapunov-based supervisory module, which regulates the influence of the PINN output according to a prescribed safety condition. The resulting reaction wheel torque command is saturated to respect actuator limits and applied to the spacecraft dynamics.

From an implementation standpoint, the closed-loop architecture supports multiple execution modes corresponding to different activation states of the learning-based component and the supervisory module. These modes correspond to the controller variants evaluated in the validation campaign: *NMPC*, *NMPC + PINN*, in which the supervisory action is bypassed to isolate the unconstrained contribution of the learned residual, and *NMPC + PINN + Supervisor*, which represents the nominal operational configuration of the proposed architecture.

The closed-loop execution flow and component interactions summarized in this section define the operating configurations considered in the validation study presented in the following chapter. Chapter 5 builds on this execution structure to assess the resulting closed-loop performance and safety properties under representative simulation scenarios.

Chapter 5

Validation

5.1 Validation Scenarios and Test Case Definition

5.1.1 Objectives of the Validation Campaign

The objective of the validation campaign is to quantitatively evaluate the design hypotheses formulated in the chapter 2 under controlled and repeatable simulation conditions. In particular, the validation aims to assess whether the introduction of PINN-based residual torque compensation leads to systematic improvements in attitude regulation performance relative to the baseline NMPC controller.

The validation is explicitly not intended for controller tuning. All controller parameters are fixed prior to the validation runs, and identical test scenarios are applied to all controller variants to enable fair, paired comparisons.

5.1.2 Initial Condition Sampling Strategy

A Monte Carlo simulation campaign is employed to evaluate closed-loop performance over a diverse set of initial attitudes. A total of 100 initial conditions are considered, constructed as follows.

Fifty unit vectors are generated with directions uniformly distributed on the unit sphere. For each direction vector, two initial attitude conditions are defined by applying rotations of 75° and 125° about the corresponding axis. This results in a set of 100 distinct initial attitude quaternions, covering both moderate and large-angle maneuver scenarios.

For all test cases, the initial angular velocity of the spacecraft bus and the reaction wheels is set to zero, corresponding to rest-to-rest maneuver conditions and an initial zero total angular momentum state.

5.1.3 Disturbance and Uncertainty Scenarios

All simulations include environmental disturbance torques representative of low Earth orbit operations, specifically gravity-gradient torque and aerodynamic drag torque, as defined in the plant model. These disturbances are applied consistently across all controller variants.

Model mismatch is introduced through uncertainty in the spacecraft inertia properties. In addition, while the true plant dynamics include inertial coupling effects, the NMPC prediction model assumes a simplified diagonal inertia matrix. These discrepancy induces structured prediction errors that motivate the use of residual torque compensation.

Disturbance and uncertainty realizations are held fixed across paired simulations to ensure that performance differences between controller variants are attributable to control strategy differences rather than stochastic variability.

5.1.4 Controller Variants Under Comparison

The following controller configurations are evaluated using identical test scenarios:

- **Baseline NMPC:** nonlinear model predictive controller using the nominal control-oriented model, without learning-based augmentation.
- **NMPC + PINN:** NMPC augmented with PINN-based residual torque compensation, without supervisory filtering.
- **NMPC + PINN + Supervisor:** NMPC augmented with PINN-based residual torque compensation, whose output is filtered through a Lyapunov-based supervisory layer before application to the plant.

These configurations are included to isolate the effects of residual compensation and supervisory filtering on closed-loop performance. All controller variants share identical state estimates, reference trajectories, actuator limits, and simulation parameters. No controller retuning is performed between configurations.

5.1.5 Simulation Horizon and Termination Criteria

Each simulation is executed over a fixed time one-hundred-second horizon sufficient to capture transient behavior and steady-state convergence. Convergence is defined in terms of both attitude error. Specifically, the simulation is considered converged when the attitude settling angle remains below 0.5°.

5.1.6 Performance Metrics and Statistical Evaluation Protocol

Closed-loop performance is evaluated using attitude-error-based metrics computed for each Monte Carlo run. The primary performance indicators considered in this study are:

- **Steady-state RMSE:** the root-mean-square value of the attitude error computed over bounded steady-state samples satisfying the prescribed 0.5° convergence criterion.
- **Final attitude error:** the attitude error evaluated at the final simulation time, used to characterize terminal pointing performance.

For each controller variant, these metrics are aggregated across the Monte Carlo campaign. Median values are reported to characterize typical closed-loop behavior and to reduce sensitivity to outliers arising from rare but severe transient responses. In addition, the standard deviation of each metric is computed to quantify performance dispersion across initial conditions and to assess the consistency of the closed-loop response.

To determine whether observed performance differences between controller variants are systematic rather than attributable to random variability across initial conditions, paired statistical comparisons are performed. For each initial condition, performance metrics obtained with learning-augmented controllers are directly compared against those obtained with the baseline NMPC controller. The Wilcoxon signed-rank test is employed as a nonparametric hypothesis test to evaluate whether the distribution of paired performance differences is centered away from zero, without assuming normality of the underlying error distributions.

Together, the combination of median performance metrics, dispersion measures, and nonparametric hypothesis testing enables a distinction between incidental improvements affecting isolated trajectories and consistent performance gains observed across the Monte Carlo population.

5.2 Qualitative Closed-Loop Behavior

The qualitative behavior of the closed-loop system is assessed through Monte Carlo simulations for the three controller configurations considered in this study: baseline NMPC, NMPC augmented with PINN-based residual compensation, and NMPC augmented with PINN-based compensation filtered by the Lyapunov-based supervisory layer.

Figure 5.1 illustrates the quaternion trajectories obtained under the baseline NMPC controller. From all considered initial conditions, the spacecraft attitude converges toward the reference equilibrium, with the scalar quaternion component q_0 approaching unity and the vector components q_1, q_2, q_3 converging toward zero.

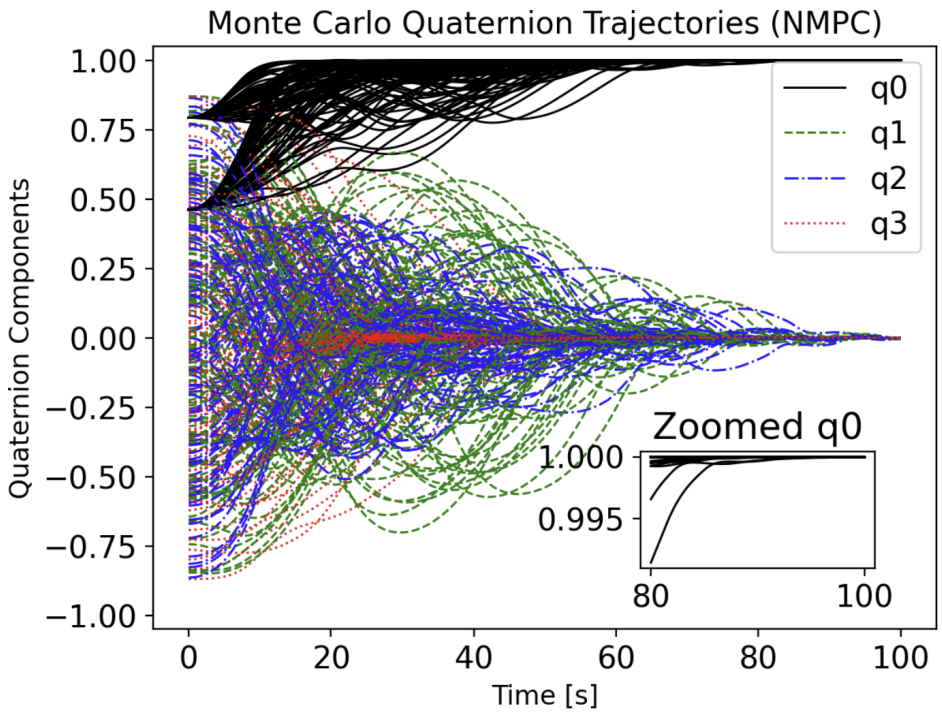

Figure 5.1: Monte Carlo quaternion trajectories under baseline NMPC control.

The corresponding trajectories obtained with NMPC augmented by PINN-based residual torque compensation are shown in Figure 5.2. As in the baseline case, all trajectories converge toward the reference attitude, indicating that the introduction of learning-based augmentation does not alter the fundamental stabilizing behavior of the closed loop. Differences are primarily observed in the transient and steady-state characteristics, where reduced dispersion is visible for a majority of initial conditions.

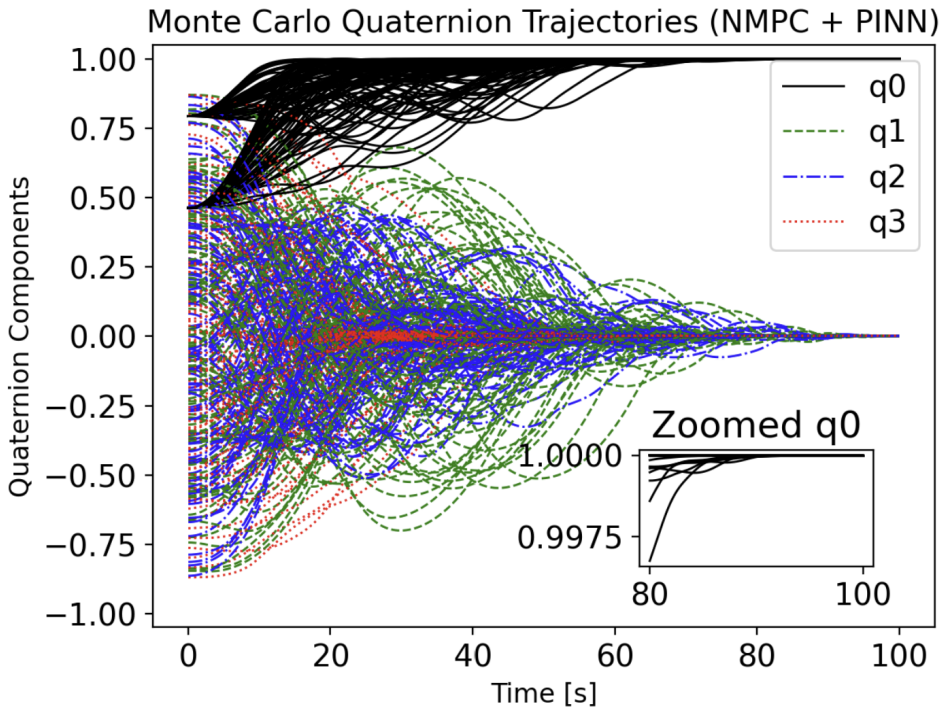

Figure 5.2: Monte Carlo quaternion trajectories under NMPC with PINN-based residual compensation (no supervisor).

Figure 5.3 reports the quaternion trajectories obtained when the PINN output is filtered through the Lyapunov-based supervisory layer. The supervised configuration preserves convergence toward the reference equilibrium for all initial conditions, while moderating the influence of aggressive learned corrections. No loss of stability is observed, and the overall behavior remains consistent with that of the baseline NMPC controller.

Overall, the qualitative results demonstrate that all three controller configurations are capable of regulating the spacecraft attitude from the considered set of initial conditions. The baseline NMPC controller establishes the core stabilizing behavior of the closed loop, while learning-based residual compensation and supervisory filtering act as secondary mechanisms that influence transient response characteristics and steady-state dispersion without modifying the nominal convergence properties.

5.3 Quantitative Performance Evaluation

5.3.1 Monte Carlo Aggregate Results

Aggregate performance metrics computed over the Monte Carlo campaign are summarized in Table 5.1. Median values are reported to characterize typical behavior, while standard deviations quantify dispersion across initial conditions.

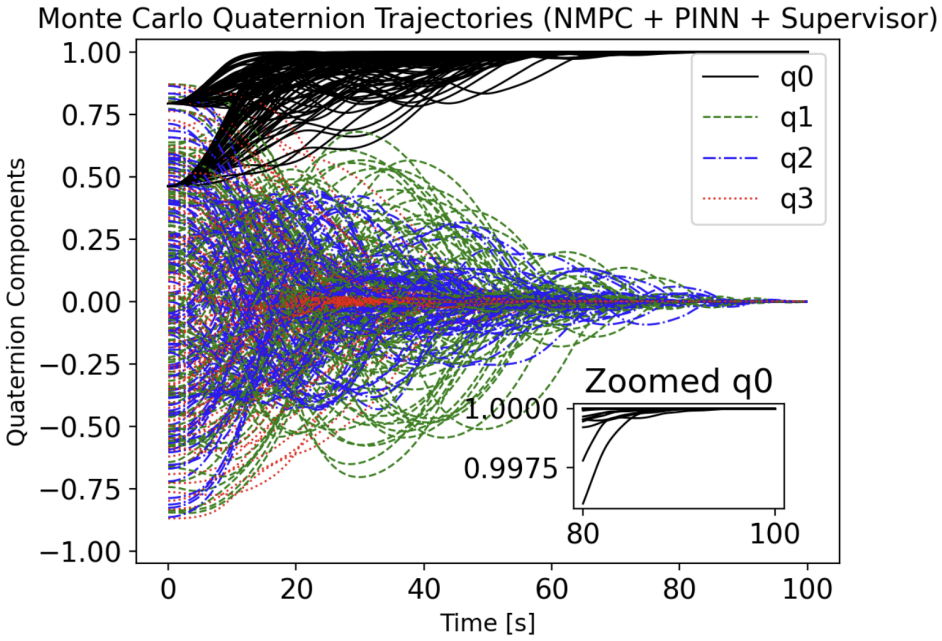

Figure 5.3: Monte Carlo quaternion trajectories under NMPC with PINN-based residual compensation and Lyapunov-based supervisory filtering.

Table 5.1: Monte Carlo aggregate attitude error metrics and relative changes with respect to NMPC.

Metric	**NMPC**	**NMPC+PINN**	**NMPC+PINN+Sup**
SS RMSE med. [deg]	0.1401	0.1362	0.1387
Δ w.r.t. NMPC [%]	–	−2.76	−0.99
SS RMSE std [deg]	0.0672	0.0437	0.0495
Δ w.r.t. NMPC [%]	–	−35.01	−26.32
Final error med. [deg]	0.0726	0.0638	0.0715
Δ w.r.t. NMPC [%]	–	−12.06	−1.45
Final error std [deg]	0.0704	0.0452	0.0497
Δ w.r.t. NMPC [%]	–	−35.77	−29.42

As can be seen, all reported steady-state RMSE values remain well below the 0.5° convergence threshold, confirming that the observed improvements occur within the prescribed regulation accuracy bound.

5.4 Statistical Significance Analysis

5.4.1 Paired Statistical Comparison Methodology

To assess whether observed performance differences are systematic rather than incidental, paired statistical comparisons are performed using the Wilcoxon signed-rank test. For each initial condition, the performance metrics obtained with learning-

augmented controllers are compared directly against those obtained with the baseline NMPC controller.

This nonparametric test does not assume normality of the error distributions and is therefore well-suited to the heterogeneous and nonlinear nature of the closed-loop responses observed in the Monte Carlo campaign.

5.4.2 Statistical Results

The statistical comparison results are summarized in Table 5.2, including median paired improvements, p-values, and the fraction of initial conditions exhibiting improved performance.

Table 5.2: Paired statistical comparison with respect to baseline NMPC.

Metric	**NMPC + PINN**	**NMPC + PINN + Sup**
Median improvement [deg]	7.08×10^{-3}	2.45×10^{-3}
Wilcoxon p-value	6.58×10^{-5}	6.72×10^{-3}
Fraction improved	0.71	0.57

For both learning-augmented configurations, the reported p-values indicate that the observed median improvements are unlikely to be attributable to random variability across initial conditions.

5.5 Interpretation of Validation Results

5.5.1 Performance Improvements Enabled by Residual Compensation

The quantitative results confirm the central design hypothesis formulated in Chapter 2: augmenting the baseline NMPC controller with a PINN-based residual torque compensator leads to systematic improvements in attitude regulation performance.

Across the Monte Carlo campaign, the NMPC + PINN configuration exhibits reduced median steady-state RMSE and reduced median final attitude error relative to the baseline NMPC controller. These improvements are not limited to isolated trajectories but are observed consistently across a majority of initial conditions, as confirmed by the paired statistical analysis. This behavior is consistent with the interpretation that the learned residual torque compensates structured model mismatch—primarily associated with inertia uncertainty and unmodeled coupling—thereby improving the effective prediction accuracy of the NMPC internal model.

These findings validate the conceptual motivation underlying the learning layer: improving model–plant consistency enhances the NMPC's ability to exploit nonlinear coupling effects during horizon-based planning, resulting in improved regulation accuracy.

5.5.2 Consistency Across Initial Conditions

Beyond median performance improvements, the dispersion metrics reported in Table 5.1 provide insight into the consistency of closed-loop behavior across the Monte Carlo population.

For both learning-augmented configurations, the standard deviation of steady-state RMSE and final attitude error is reduced relative to the baseline NMPC controller. This reduction indicates that the performance gains introduced by residual compensation are not confined to a small subset of favorable initial conditions but extend broadly across the sampled attitude space.

The Wilcoxon signed-rank test further supports this interpretation. The fraction of improved runs exceeds 55% for both learning-augmented controllers, with the unsupervised PINN configuration achieving improvement in approximately 71% of cases. These results align with the hypothesis that residual compensation provides a systematic correction of structured model mismatch rather than incidental trajectory-specific benefits.

Taken together, the reduction in dispersion and the statistical significance of paired improvements demonstrate that the learning-based augmentation enhances not only typical performance but also the reliability and uniformity of closed-loop behavior across diverse initial attitudes.

5.5.3 Performance–Safety Trade-Offs

While the supervised PINN configuration remains statistically superior to the baseline NMPC controller, its performance improvements are consistently smaller than those achieved by the unsupervised PINN configuration. The supervisor is designed as a conservative safety filter rather than a performance optimizer. Its primary objective is to prevent learning-induced destabilization by enforcing a non-increase condition on a Lyapunov-like energy function constructed using the nominal spacecraft inertia matrix. As a consequence, the supervisor evaluates the admissibility of the learned correction under pessimistic assumptions, including near-to-worst-case reinforcement of unmodeled disturbances.

Further, since the supervisory logic relies on the nominal inertia model rather than the true plant inertia, it may attenuate or reject residual corrections that are beneficial with respect to the true dynamics but appear unsafe under the nominal model. This conservativeness naturally limits the magnitude and frequency with

which the learned correction is applied, especially in regimes where the Lyapunov condition is marginal.

The resulting behavior represents a performance–safety trade-off: while some potential performance gains enabled by the PINN are suppressed, the closed-loop system retains the stability guarantees of the baseline NMPC controller. The observed reduction in performance relative to the unsupervised PINN configuration should therefore not be interpreted as a failure of the supervisory concept, but rather as a direct consequence of its safety-oriented design philosophy.

5.6 Architectural Implications

5.6.1 Benefits and Limitations of the Proposed Architecture

The learning-augmented architecture successfully balances performance improvement and safety preservation through a clear separation of responsibilities between the NMPC, the PINN, and the supervisory layer. The NMPC establishes nominal stability and constraint handling, the PINN provides performance-enhancing residual compensation, and the supervisor enforces safety guarantees.

At the same time, the validation results illustrate that conservative safety enforcement inevitably limits achievable performance gains. This limitation is not a defect of the proposed architecture, but a direct consequence of prioritizing closed-loop stability under learning-induced corrections. The results therefore underscore the importance of explicitly articulating performance–safety trade-offs when integrating learning-based components into safety-critical control systems.

5.7 Limitations of the Present Study

5.7.1 Simulation-Only Validation

All results presented in this chapter are obtained through high-fidelity numerical simulations. Although for the algorithm to be able to implemented, the process-in-the-loop and hardware-in-the-loop validations also are needed, which is beyond the scope of the present study.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

6.1.1 Summary of Contributions

This thesis addressed the attitude regulation problem of an underactuated spacecraft using a hierarchical control architecture that combines nonlinear model predictive control with learning-based residual compensation and deterministic safety supervision.

The primary contribution of this work is the development and validation of a modular learning-augmented NMPC architecture in which performance enhancement through machine learning is explicitly decoupled from stability and constraint enforcement. A nominal NMPC controller was designed as the baseline control authority, responsible for handling nonlinear dynamics, actuator constraints, and underactuation-aware maneuver planning. A Physics-Informed Neural Network was introduced as a residual torque estimator to compensate for structured model mismatch, primarily arising from inertia uncertainty and neglected inertial coupling effects. To ensure safety and robustness, a Lyapunov-based supervisory layer was interposed between the learning module and the plant, enforcing a conservative energy-based admissibility condition on the learned correction.

A key architectural feature of the proposed framework is that learning-based augmentation is optional rather than essential: when the learned correction is unreliable or rejected by the supervisor, the closed-loop system reverts seamlessly to the nominal NMPC policy without reconfiguration or mode switching. This design aligns with safety-first principles required in mission- and safety-critical aerospace applications.

The entire architecture was implemented in a high-fidelity simulation environment, with explicit consideration of real-time execution constraints, multi-rate scheduling, actuator limits, and numerical robustness. Particular attention was paid to ensuring

consistency between the assumptions made during offline PINN training and the conditions under which the learned model is deployed in closed loop.

6.1.2 Key Validation Outcomes

A comprehensive Monte Carlo validation campaign was conducted to assess the impact of learning-based residual compensation on attitude regulation performance. The results demonstrate that augmenting the baseline NMPC controller with a PINN-based residual torque estimator leads to statistically significant reductions in steady-state attitude error and reduced performance dispersion across a wide range of initial conditions.

When the learned correction is applied without supervision, the largest performance gains are observed, confirming that residual compensation effectively improves model–plant consistency and enhances the NMPC's ability to exploit nonlinear coupling effects during horizon-based planning. When the Lyapunov-based supervisory layer is enabled, performance improvements are reduced but remain statistically significant, illustrating a clear and interpretable performance–safety trade-off.

Importantly, the validation confirms that the learning-based augmentation does not alter the fundamental convergence properties of the closed-loop system. The supervisor successfully prevents learning-induced destabilization under all tested scenarios, including pessimistic disturbance assumptions, while allowing performance improvements when the learned correction is compatible with the imposed safety condition. The observed conservativeness of the supervisor is a design choice rather than a deficiency, reflecting the prioritization of stability and robustness over maximal performance.

Overall, the results validate the central hypothesis of this thesis: learning-based residual compensation can be safely and effectively integrated into a predictive control architecture for underactuated spacecraft, provided that its influence is explicitly constrained by control-theoretic safeguards.

6.2 Future Work

While the proposed architecture demonstrates promising performance and robustness in simulation, several extensions are necessary to further improve performance, reduce conservativeness, and increase operational relevance.

6.2.1 Optimization of Scheduling Mechanisms for Underactuation

In the present work, underactuation effects within the NMPC formulation are addressed through a heuristically designed, smooth gain scheduling mechanism that biases the cost function toward reducing errors in the underactuated direction near

convergence. While this approach was shown to be effective and numerically robust, the scheduling law is not optimized and may introduce suboptimal transients.

Future work could investigate systematic design and optimization of scheduling mechanisms, including co-design of NMPC weights and scheduling parameters based on performance metrics or robustness criteria. Adaptive scheduling strategies could further improve performance during transitions between large-angle maneuvers and fine pointing, without increasing online computational burden.

6.2.2 Less Conservative Safety Filtering via CLF–QP Formulations

The supervisory layer implemented in this thesis enforces stability through a conservative Lyapunov monotonicity condition evaluated using nominal model parameters and pessimistic disturbance assumptions. While this guarantees safety, it may reject or excessively attenuate learned corrections that are beneficial for the true plant dynamics.

An important extension would be to reformulate the supervisory logic using Control Lyapunov Function constraints embedded in a quadratic program (CLF–QP). Such formulations would allow the learned correction to be minimally modified rather than fully accepted or rejected, enabling a continuous trade-off between performance and safety. Integrating CLF–QP supervision with NMPC-based architectures could significantly reduce conservativeness while preserving formal stability guarantees.

6.2.3 Online and Continual Learning with Mode-Switching Architectures

In this work, the PINN is trained offline using prerecorded simulation data. While effective for compensating structured and slowly varying model mismatch, this approach does not exploit information available during nominal on-orbit operation.

Future research could explore online or continual learning strategies in which data is collected during normal reaction-wheel operation and used to incrementally refine the residual model. In such a framework, the learned model could remain dormant during nominal operation and be activated or reweighted in the event of actuator degradation or failure, enabling a smooth transition from nominal control to failure-aware modes.

6.2.4 Integration with Momentum Management and Detumbling Phases

The proposed architecture assumes near-zero total angular momentum and focuses on attitude regulation using reaction wheels. Extending the framework to include momentum management and detumbling phases would significantly enhance its

operational relevance.

Future work could investigate the integration of magnetic torquers or other auxiliary actuators for angular momentum unloading and detumbling. This would enable seamless transitions between detumbling, momentum management, and fine-pointing control within a unified hierarchical framework.

6.2.5 Hardware-in-the-Loop and Onboard Implementation

All results presented in this thesis are obtained through numerical simulation. A critical step toward practical deployment is the validation of the proposed architecture in process-in-the-loop and hardware-in-the-loop environments.

Future efforts should focus on assessing real-time feasibility, computational margins, sensitivity to sensor noise, and robustness to actuator nonlinearities under representative onboard hardware constraints. Such studies would provide essential insight into implementation challenges and further validate the suitability of learning-augmented predictive control architectures for safety-critical spacecraft applications.

Bibliography

[1] ESA. *The Hubble Space Telescope*. Accessed: 2025-10-01. 2024. URL: `https://www.esa.int/kids/en/learn/Technology/Spacecraft/The_Hubble_Space_Telescope` (cit. on p. 2).

[2] Peter Crouch. "Spacecraft attitude control and stabilization: Applications of geometric control theory to rigid body models". In: *IEEE Transactions on Automatic control* 29.4 (2003), pp. 321–331 (cit. on p. 2).

[3] Hariharan Krishnan, N Harris McClamroch, and Mahmut Reyhanoglu. "Attitude stabilization of a rigid spacecraft using two momentum wheel actuators". In: *Journal of Guidance, Control, and Dynamics* 18.2 (1995), pp. 256–263 (cit. on p. 3).

[4] Lei Jin and Yingjie Li. "Model predictive control-based attitude control of under-actuated spacecraft using solar radiation pressure". In: *Aerospace* 9.9 (2022), p. 498 (cit. on p. 3).

[5] Maksim Shirobokov, Sergey Trofimov, and Mikhail Ovchinnikov. "Survey of machine learning techniques in spacecraft control design". In: *Acta Astronautica* 186 (2021), pp. 87–97 (cit. on pp. 4, 5).

[6] Kalmanje KrishnaKumar, Susan Rickard, and Susan Bartholomew. "Adaptive neuro-control for spacecraft attitude control". In: *Neurocomputing* 9.2 (1995), pp. 131–148 (cit. on p. 4).

[7] Bruno Apolloni, Ferdinando Battini, and Costantino Lucisano. "A co-operating neural approach for spacecrafts attitude control". In: *Neurocomputing* 16.4 (1997), pp. 279–307 (cit. on p. 4).

[8] James D Biggs and Hugo Fournier. "Neural-network-based optimal attitude control using four impulsive thrusters". In: *Journal of Guidance, Control, and Dynamics* 43.2 (2020), pp. 299–309 (cit. on p. 4).

[9] Carlo Cena, Mauro Martini, and Marcello Chiaberge. *Learning Robust Satellite Attitude Dynamics with Physics-Informed Normalising Flow*. 2025 (cit. on pp. 4, 25).

[10] European Cooperation for Space Standardization (ECSS). *ECSS-E-HB-40-02A: Space engineering — Machine learning handbook*. Handbook. Noordwijk, The Netherlands: European Cooperation for Space Standardization (ECSS), ESA

Requirements & Standards Section, Nov. 2024. URL: https://ecss.nl/home/ecss-e-hb-40-02a-15-november-2024/ (cit. on p. 5).

[11] Karla Raigoza and Timothy Sands. "Autonomous trajectory generation comparison for de-orbiting with multiple collision avoidance". In: *Sensors* 22.18 (2022), p. 7066 (cit. on pp. 10, 11).

[12] Christopher Petersen. "Advances in Underactuated Spacecraft Control". Doctoral dissertation. Ann Arbor, MI, USA: University of Michigan, 2016 (cit. on p. 21).

[13] Carlo Novara. *Nonlinear Control and Aerospace Applications: Attitude Control*. Lecture notes. 2016 (cit. on p. 22).

[14] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. "On the spectral bias of neural networks". In: *International conference on machine learning*. PMLR. 2019, pp. 5301–5310 (cit. on p. 42).

[15] Patrick W Kenneally, Scott Piggott, and Hanspeter Schaub. "Basilisk: A flexible, scalable and modular astrodynamics simulation framework". In: vol. 17. 9. American Institute of Aeronautics and Astronautics, 2020, pp. 496–507 (cit. on p. 49).

[16] David Mostaza Prieto, Benjamin P Graziano, and Peter CE Roberts. "Spacecraft drag modelling". In: *Progress in Aerospace Sciences* 64 (2014), pp. 56–65 (cit. on p. 57).

[17] Joel AE Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl. "CasADi: a software framework for nonlinear optimization and optimal control". In: *Mathematical Programming Computation* 11.1 (2019), pp. 1–36 (cit. on p. 58).

[18] Andreas Wächter and Lorenz T Biegler. "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming". In: *Mathematical programming* 106.1 (2006), pp. 25–57 (cit. on p. 58).