aerospace

MDPI

Article

Design and Simulation-Based Evaluation of the FuzzyBuzz Attitude Control Experiment on the Astrobee Platform

María Royo, Juan Carlos Crespo, Ali Arshadi, Cristian Flores, Karl Olfe * and José Miguel Ezquerro

E-USOC, Escuela Técnica Superior de Ingeniería Aeronáutica y del Espacio, Universidad Politécnica de Madrid, Plaza de Cardenal Cisneros 3, 28040 Madrid, Spain; mroyo@eusoc.upm.es (M.R.); jcrespo@eusoc.upm.es (J.C.C.); ali.arshadi@alumnos.upm.es (A.A.); cristian.flores@upm.es (C.F.); jm.ezquerro@upm.es (J.M.E.)

* Correspondence: kolfe@eusoc.upm.es

Abstract

Recent space missions demand higher pointing accuracy, smoother attitude transitions and lower energy consumption than those typically achievable with conventional control approaches. This motivates the exploration of intelligent and nonlinear control methods. The FuzzyBuzz experiment investigates the application of fuzzy logic for spacecraft attitude control using NASA's Astrobee robotic system aboard the International Space Station. Unlike traditional control methods, fuzzy logic introduces a rule-based approach capable of handling uncertainties and nonlinearities inherent in space environments, making it particularly suited for autonomous operations in microgravity. The objective of FuzzyBuzz is to evaluate the effectiveness of fuzzy controllers compared to traditional linear ones, such as Proportional–Integral–Derivative (PID) and H_∞ controllers. In addition, a comparison with a nonlinear controller based on a Model Predictive Control (MPC) strategy is considered. The controllers will be tested through predefined attitude maneuvers, evaluating precision, energy efficiency, and real-time adaptability. This work presents the design of the FuzzyBuzz experiment, including the software architecture, simulation environment, experiment protocol, and the development of a fuzzy logic-based attitude control system for Astrobee robots. The proposed fuzzy controller and a PID controller are optimized using a Multi-Objective Particle Swarm Optimization (MOPSO) method, providing a range of operational points with different trade-offs between two metrics, related to convergence time and energy consumption. Results show that the PID controller is better suited for scenarios demanding low convergence times, whereas the fuzzy controller provides smoother responses, reduced steady-state error, and maintains convergence under significant parametric uncertainties. Results from H_∞ and MPC controllers will be reported once the in-orbit experiment is performed.

Keywords: fuzzy controller; Astrobee; MOPSO; space robotics

Check for updates

Academic Editors: Andreas Strohmayer, Spiros Pantelakis and Gustavo Alonso-Rodrigo

Received: 26 February 2026
Revised: 21 March 2026
Accepted: 25 March 2026
Published: 28 March 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

The Attitude Determination and Control System (ADCS) is a key subsystem on any spacecraft, responsible for keeping the satellite aligned in the orientation needed to achieve its scientific, technological, or commercial goals [1]. This includes tasks such as stabilizing a communication antenna, directing a high-resolution camera, or positioning solar panels to maximize power generation.

Attitude control has historically relied on classical methods such as PID controllers [2] or Linear Quadratic Regulators (LQRs) [3]. These controllers are well understood, computationally lightweight, and widely used in numerous space systems. However, their

performance strongly depends on model accuracy and on the presence of nonlinearities. Even modern controllers based on linear system theory, such as robust H_∞ synthesis, may experience performance degradation in highly nonlinear systems or during abrupt maneuvers and configuration changes if these effects are not adequately captured during controller synthesis and weighting functions selection [4,5].

Recent space missions, including precise formation-flying and optical-communication systems, demand higher pointing accuracy and smoother attitude transitions than those typically achievable with conventional control approaches [6,7]. This motivates the exploration of intelligent and nonlinear control approaches.

Fuzzy logic control has gained attention due to its low computational burden and ability to approximate nonlinear control laws and handle uncertainty without requiring an exact analytical model [8–10]. Previous studies have demonstrated improved robustness and smoother responses in satellite attitude control applications [11–13]. Furthermore, advances in optimization techniques such as Multi-Objective Particle Swarm Optimization (MOPSO) have enabled the systematic tuning of fuzzy controller parameters, generating Pareto optimal trade-offs between tracking error and control effort [14].

Recent advances in flexible-structure control have also highlighted the importance of accurate time-delay estimation and decoupling strategies in systems subject to modal excitations [15]. The influence of such time-delay effects on the stability and performance of fuzzy attitude controllers has also been studied in nanosatellite platforms in [16].

These developments suggest that fuzzy control may be a strong candidate for autonomous microgravity robotics, yet experimental evidence on relevant platforms remains scarce, with most evidence still limited to ground-based test setups [17].

To address this gap, the FuzzyBuzz experiment focuses on the Astrobee robot attitude control problem and is designed to enable a future experimental comparison between four controllers under identical in-orbit conditions: (I) a fuzzy logic controller, (II) a classical PID controller, (III) a robust H_∞ and (IV) a Model Predictive Control (MPC). The present paper describes exclusively the design and performance of fuzzy and PID controllers, since they are the only ones for which cost-based and precision-error-based optimizations can be compared. For the other controllers, H_∞ and MPC, the comparison will be carried out in orbit at specific operating points. The current forecast is that the experiment will be carried out in orbit during the year 2027.

The NASA Astrobee free-flying robotic system has emerged as one of the most versatile platforms for orbit research and the development of autonomous spacecraft technology. Operating inside the International Space Station (ISS), Astrobee provides a unique combination of precise actuation, high-rate state estimation, and modular software architecture, making it an ideal testbed for investigating advanced Guidance, Navigation, and Control (GNC) algorithms in a relevant microgravity environment. Unlike traditional spacecraft simulators, Astrobee offers real hardware constraints. This feature has made it increasingly valuable for validating new control strategies and maturing technologies prior to deployment in future space missions [18,19]. As an example, it has been used to test sliding-mode and backstepping control [20] and On-orbit close proximity operations [21].

The paper is organized as follows: Section 2 describes the Astrobee platform, its dynamics, software architecture and the simulation environment. Section 3 describes the design and performance of the controllers and MOPSO optimization. Section 4 shows the main results and the Monte Carlo analysis performed. Section 5 describes the experiment protocol. Finally, Section 6 is devoted to a discussion of the main results.

2. Materials and Methods

The Astrobee platform is a 6 degrees of freedom free-flying robot designed by NASA to support autonomous operations, intravehicular experimentation, and technology validation aboard the ISS. Its architecture closely resembles that of a small spacecraft while offering a safe, flexible, and fully instrumented platform for conducting research in microgravity.

2.1. Mechanical Structure and Propulsion System of Astrobee

The Astrobee project contains three cooperative cubic robots (Honey, Bumble, and Queen), each of them equipped with several fan propulsion modules arranged to generate forces and torques along all degrees of freedom. Each module contains electrically driven impellers and movable vanes that vector the thrust, providing full 6 DoF actuation capability, including the ability to generate control torques about the roll, pitch, and yaw axes. An Astrobee image indicating its main components is shown in Figure 1.

The mapping of actuation between motor commands and torques produced is nonlinear. These characteristics make Astrobee particularly suitable for evaluating controllers that must operate under realistic constraints, model uncertainties, and actuator saturation [14].

Astrobee performs several operational roles within the ISS, including (I) assisting astronauts by transporting tools or sensors and (II) supporting human–robot interaction experiments, etc. These tasks require precise and smooth pose–motion control, particularly during inspection, close proximity operations, and collaborative tasks.

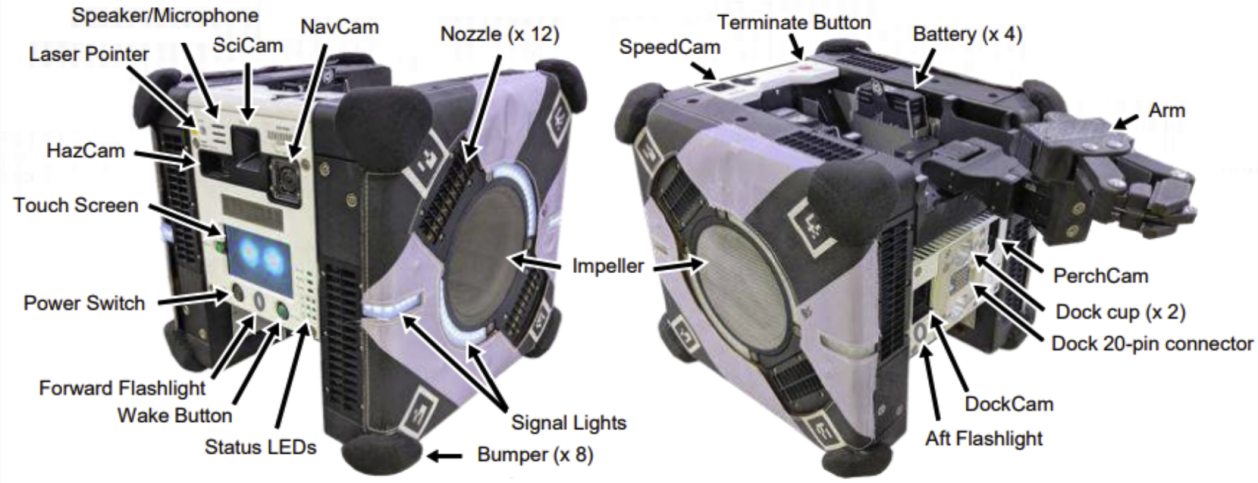


Figure 1. An Astrobee robot with its main external components highlighted.

2.2. Attitude Dynamics

As stated in the official Astrobee documentation [22], the Astrobee rotational motion is governed by Euler's rotational dynamics [3].

$$\mathbf{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times (\mathbf{I}\boldsymbol{\omega}) = \boldsymbol{\tau}, \tag{1}$$

where $\times$ denotes the cross product and

- $\mathbf{I} = \mathrm{diag}([I_{xx}, I_{yy}, I_{zz}])$ is the inertia tensor,
- $\boldsymbol{\omega} = [\omega_x, \omega_y, \omega_z]^\top$ is the angular velocity vector,
- $\boldsymbol{\tau} = [\tau_x, \tau_y, \tau_z]^\top$ is the torque vector generated by the propulsion system.

Astrobee's kinematics is represented using unit quaternions,

$$\dot{q} = \frac{1}{2}\boldsymbol{\Omega}(\boldsymbol{\omega})q, \quad \boldsymbol{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -\boldsymbol{\omega}^\top \\ \boldsymbol{\omega} & -[\boldsymbol{\omega}]_\times \end{bmatrix}, \quad q = \begin{bmatrix} q_0 \\ \mathbf{q} \end{bmatrix}, \quad \|q\| = 1, \tag{2}$$

where $[\boldsymbol{\omega}]_\times$ denotes the skew-symmetric matrix associated with $\boldsymbol{\omega}$, $\|q\|$ denotes the quaternion norm, which satisfies $\|q\| = 1$ for unit quaternions, q_0 is the scalar part, and $\mathbf{q}$ is the vector part. This representation avoids singularities and is widely used in spacecraft attitude control.

For a commanded quaternion q_d and a current quaternion q, the attitude error quaternion is defined as

$$q_e = q_d^{-1} \otimes q, \tag{3}$$

where q_d^{-1}denotes the inverse of quaternion q_d, which for unit quaternions corresponds to its conjugate, and $\otimes$ denotes the quaternion product. The norm of the vector part $\mathbf{q}_e$ of the error quaternion is directly proportional, for small angles, to the angle attitude error.

2.3. Simulation Environment

A dedicated simulation framework was developed to support the design, optimization and evaluation of the proposed attitude controllers. Rather than relying on external tools, the full translational and rotational dynamics of the Astrobee free-flying robot were implemented directly in MATLAB R2025b. The model incorporates the equations of motion presented in (1), (2), and (3), including quaternion kinematics and the computation of the attitude error through quaternion inversion and composition. This direct implementation ensures full transparency of the dynamic model and allows for the precise manipulation of all states and control inputs.

State propagation is performed numerically using a fixed-step fourth-order Runge–Kutta method, selected to provide consistent accuracy with predictable computational cost—an essential requirement for multi-objective optimization. PID and fuzzy controllers have been simulated in this environment within the same integration loop, guaranteeing synchronous update of the control actions and avoiding discrepancies caused by solver-dependent timing.

NASA's official Astrobee simulator, based on the Gazebo–ROS (Robot Operating System) ecosystem, was also examined. However, preliminary experiments suggest that the publicly available Gazebo dynamic model may not fully capture Astrobee's true equations of motion. Consequently, the Gazebo–ROS simulator cannot be used as a validation environment for the controllers developed at this moment. Instead, the role of NASA's simulator in this project is strictly limited to integration testing. Specifically, it will be used to verify that the final software application can initialize the simulator, establish ROS communication, send motion commands, and receive telemetry. All quantitative evaluations of controller performance, stability, and robustness are conducted exclusively in the MATLAB simulation environment described above.

2.4. Software Architecture

The on-board software follows a hierarchical supervisory control architecture, as depicted in Figure 2.

2.4.1. External Interfaces with Astrobee (Astrobee/ROS Boundary)

The software is designed to integrate with the Astrobee GNC stack through a small set of ROS interfaces, using the same estimation and command pathways available to the onboard controller. Specifically, the software (i) consumes state estimates from the Astrobee EKF (Extended Kalman Filter). The software subscribes to `/gnc/ekf` (`ff_msgs/EkfState`) to obtain the state estimates required for control and logging. The minimum required fields are the attitude quaternion and the angular velocity; linear velocity and position may also be used when available. The software (ii) manages control authority by temporarily disabling and re-enabling the nominal Astrobee attitude controller using standard GNC services dur-

ing experiment runs. The software calls the service `/gnc/ctl/enable` (`std_srvs/SetBool`) to disable the nominal Astrobee attitude controller prior to executing an experiment run and to restore it after the run completes or is aborted. This mechanism provides a clear and auditable control handover between nominal GNC and experiment-driven commanding. The software publishes `ff_msgs/FamCommand` messages to `/gnc/ctl/command`. Commands are expressed in the Astrobee body-fixed frame (`header.frame_id = "body"`), include a defined `control_mode`, and provide the requested wrench (forces and torques) to the GNC Force Allocation Module. For attitude-only experiments, translational force components are set to zero, and only torque components are commanded.

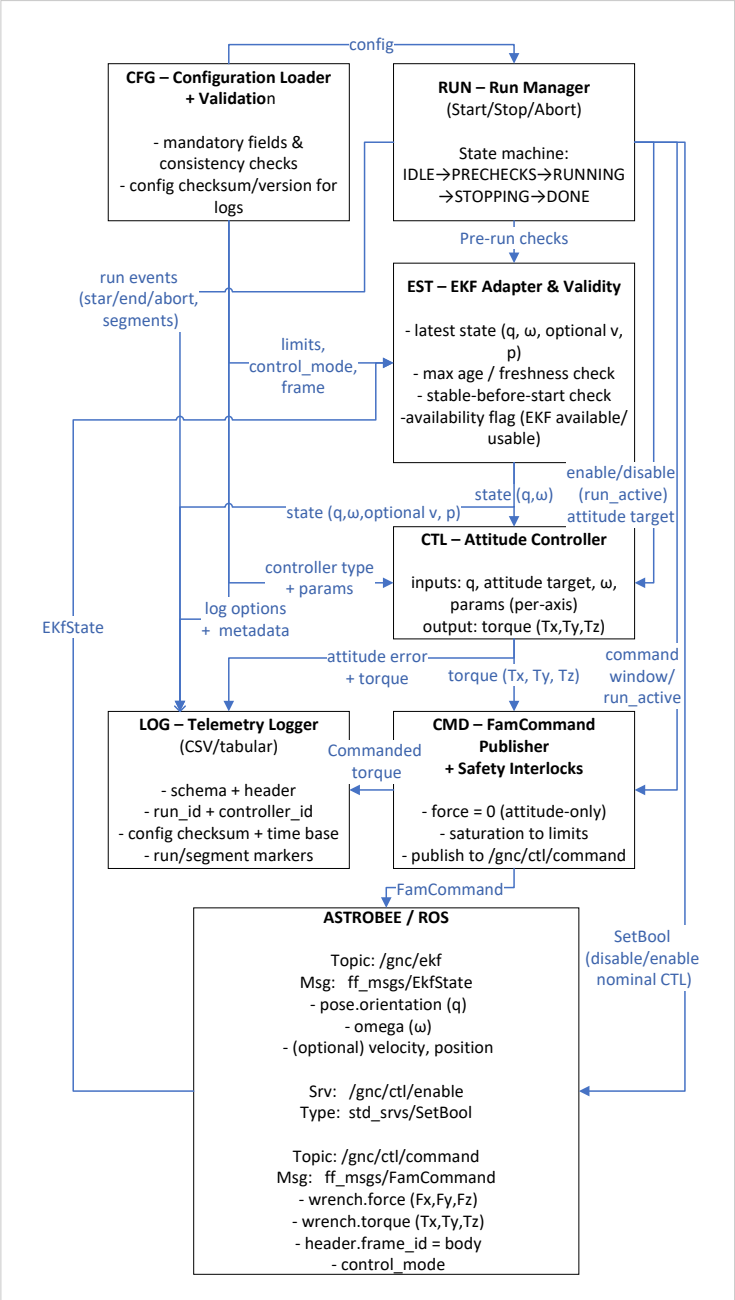


Figure 2. Software architecture schema. The right column shows the main components of a control feedback scheme—estimation, control, and command. On the left side, the configuration and logging capabilities are depicted. At the bottom, the plant to be controlled is shown.

2.4.2. Execution Flow (Nominal Run)

At startup, the software loads and validates the experiment configuration (CFG) and prepares the runtime context (run identifiers, limits, logging options, and time-base declaration). When an operator initiates an experiment run, the Run Manager (RUN) performs pre-run checks using the EKF adapter (EST) to ensure that valid and sufficiently fresh state estimates are available. RUN then disables the nominal Astrobee controller via `/gnc/ctl/enable` and opens a bounded command window for the experiment. During the active run, the controller component (CTL) computes attitude error and generates body-torque commands, which the command publisher (CMD) constrains (interlocks and saturation) and publishes as `ff_msgs/FamCommand` to `/gnc/ctl/command`. In parallel, the logger (LOG) records the time-stamped state, commands, control error, and run/segment markers together with configuration traceability metadata. At run completion (or termination condition), the software commands a safe stop (zero torque), restores the nominal controller via `/gnc/ctl/enable`, and finalizes the log products.

Next, the details of the internal software components and their respective responsibilities are given:

1. CFG—Configuration Loader and Validator. This component loads the experiment configuration from an external configuration file and validates that all mandatory fields are present and consistent (e.g., run identifiers, controller selection, limits, and logging options). It also provides configuration traceability by computing and exposing a configuration version or checksum to be recorded alongside the experiment logs. The validated configuration parameters are distributed to the remaining software components.
2. RUN—Run Manager (Start/Stop/Abort, State Machine). The Run Manager orchestrates the lifecycle of an experiment execution. It transitions through a well-defined set of run states (IDLE $\rightarrow$ PRECHECKS $\rightarrow$ RUNNING $\rightarrow$ STOPPING $\rightarrow$ DONE) and enforces operational sequencing: pre-run checks, controlled handover of control authority, bounded command-window execution, and safe termination. It issues `/gnc/ctl/enable` service calls to disable the nominal controller before the run and to restore it after run completion or abort, and it emits run and segment event markers for logging and post-processing.
3. EST—EKF Adapter and Validity Management. This component subscribes to `/gnc/ekf` and maintains the most recent available EKF state estimate for use by the controller and the logger. It enforces state validity constraints such as data availability and freshness (maximum acceptable age) and may apply a stability criterion before allowing a run to start. When EKF data are missing or become stale, EST signals an invalid-state condition so that downstream components can transition to a safe behavior.
4. CTL—Attitude Controller Component. This component implements the attitude control law. It takes as inputs the current attitude state (quaternion), angular velocity, the configured attitude reference/target, and controller parameters loaded from configuration. Then, it calculates the attitude control error using quaternion operations and produces commanded body torques (T_x, T_y, T_z) as output. The controller is designed to be selectable and configurable (e.g., fuzzy, PID, MPC, H_∞) through external configuration while maintaining a consistent input/output interface.
5. CMD—Command Publisher and Safety Interlocks. The CMD is responsible for translating the torque output of the controller into `ff_msgsCU3/FamCommand` messages and publishing them to `/gnc/ctl/command`. It enforces safety and interface constraints such as "attitude-only" commanding (translational forces set to zero), configured saturation limits, and command-window interlocks (non-zero torque commands are only permitted during an active run and only when nominal CTL is confirmed

disabled). This component ensures that the commands issued to Astrobee remain bounded and comply with the limits of the platform and the operational conditions.

6. LOG—Telemetry and Metadata Logger. The logging component produces persistent experiment records in a machine-readable tabular format (e.g., CSV) with a fixed schema and header. It records time-stamped telemetry needed for post-run analysis (state, commanded torques, control error) together with metadata that unambiguously identifies the executed scenario (run ID, controller/config identifiers, configuration checksum/version, time-base declaration). It also records run-level and maneuver-segmentation markers (start/end/abort, segment boundaries) to support segment-wise performance evaluation and reproducible analysis.

2.4.3. Design Safeguards (Operability and Safety)

The system incorporates several safeguards to ensure safe operation during experiment execution:

1. Control-authority separation: Non-zero torque commands are only permitted within an active run window and only after the nominal attitude controller has been successfully disabled.
2. State-dependent commanding: If EKF state estimates are missing or stale, the controller does not generate new non-zero commands, and the system transitions to a safe behavior by enabling the nominal attitude controller.
3. Command limiting: Torque commands are bounded and saturated to configured limits prior to publication to the Astrobee command interface.
4. Abort handling: An operator abort triggers an immediate safe stop (zero torque) and restoration of the nominal controller.
5. Traceability: Each run is uniquely identified, and logs include configuration version/checksum and time-base declaration to enable reproducible analysis.

3. Controller Design and Optimization

This section details the fuzzy and PID controllers developed for the Astrobee platform. Three different controllers are designed, one per axis X, Y, or Z, which correspond to the roll, pitch, and yaw Euler angles. The structure is the same for all three. The input variables are the attitude error quaternion ($\mathbf{q}_e = [e_x, e_y, e_z]^\top$) and the angular velocity. The output variable is the required control torque (τ_x, τ_y, τ_z).

The controller parameters are determined through a rigorous multi-objective tuning process, yielding Pareto-optimal solutions that quantify the inherent trade-off between two metrics.

3.1. Fuzzy Controller Design

To develop the fuzzy controller, the fundamental components, namely the definitions of the membership functions, the rule base, and the defuzzification strategy, must first be established. These elements are formulated in the following subsections.

3.1.1. Membership Functions

Figure 3 shows an example of the memberships for the Z axis. The membership functions were constructed following the structural criteria below. First, all functions are defined symmetrically with respect to the origin to ensure consistent behavior for positive and negative values. Second, triangular and trapezoidal functions were selected for central regions with symmetric 50% overlap, which guarantees unitary coverage and smooth transitions. The controller commands zero output only when the tracking error is exactly zero, which naturally reduces the steady-state bias. In contrast, the outer functions adopt

rectangular profiles. These shapes were not arbitrarily specified; rather, they arise from a previous optimization study [13] in which the optimal trapezoidal forms converged to rectangles, indicating that constant activation in the large-error domain improved the convergence properties of the controller [13]. The linguistic variables used are enumerated in Table 1.

Table 1. Linguistic labels used in the fuzzy rule base.

Labels	Linguistic Variable
Z	Zero
PS	Positive Small
P	Positive
PB	Positive Big
NS	Negative Small
N	Negative
NB	Negative Big

As illustrated in Figure 3, six parameters per axis: x_1, x_2, x_3 for the error and x_4, x_5, x_6 for the angular velocity define the shapes of the membership functions. So, a total of 18 parameters fully characterize the input membership functions of the fuzzy controllers.

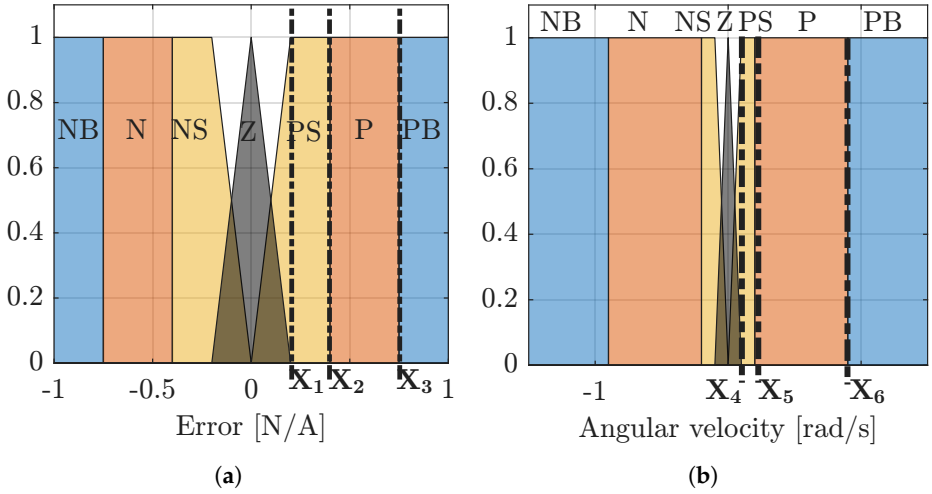


Figure 3. Example of membership functions used for the fuzzy controller: (**a**) Error membership functions. (**b**) Angular velocity membership functions.

3.1.2. Fuzzy Rules

The fuzzy rules for all axes follow the same structure, as summarized in Table 2. These rules determine the values of the output linguistic variables, which represent the control torque required along each axis, for every combination (AND operation) of the input variables: the error and the angular velocity on that axis. Examples of representative rules include

$$\text{If } (e_x = \text{NS}) \text{ and } (\omega_x = \text{PB}), \text{ then } (\tau_x = \text{NB}),$$
$$\text{If } (e_y = \text{NS}) \text{ and } (\omega_y = \text{Z}), \text{ then } (\tau_y = \text{PS}),$$
$$\text{If } (e_z = \text{PS}) \text{ and } (\omega_z = \text{NB}), \text{ then } (\tau_z = \text{PB}).$$

Table 2. Fuzzy rule base for each axis. The inputs are the error and the angular velocity, and the output corresponds to the control action.

Angular Velocity/Error	NB	N	NS	Z	PS	P	PB
NB	PB	PB	PB	PB	PB	PB	Z
N	PB	PB	PB	PB	PB	Z	NB
NS	PB	PB	PB	PS	Z	NB	NB
Z	PB	PB	PS	Z	NS	NB	NB
PS	PB	PB	Z	NS	NB	NB	NB
P	PB	Z	NB	NB	NB	NB	NB
PB	Z	NB	NB	NB	NB	NB	NB

3.1.3. Defuzzification

The Takagi–Sugeno method [8,23] has been used for defuzzification. In this approach, the output is represented by a set of discrete actuation levels expressed as fractions of the maximum control effort. These discrete output values, known as singletons, are listed in Table 3.

Table 3. Singleton output values used in the Takagi–Sugeno defuzzification method.

Output	NB	NS	Z	PS	PB
Maximum torque fraction	−1	−0.5	0	0.5	1

3.2. PID Controller Design

There are three PID controllers (one per axis); each of them obeys the following equation to calculate the output:

$$\tau_j = K_{Pj} \cdot e_j + K_{Ij} \cdot \int_0^\tau e_j dt + K_{Dj} \cdot \frac{de_j}{dt}, \qquad j = x, y, z. \tag{4}$$

where τ_j denotes the torque component along axis j, and e_j is the j-th component of the vector part of the quaternion error. The proportional, integral, and derivative gains (K_{Pj}, K_{Ij} and K_{Dj}), which result in a total of nine gains, are the tuning parameters of the PID controllers.

3.3. Tuning Maneuver

As will be described in Section 5, the experimental campaign involves the execution of step maneuvers of increasing amplitude. In the final stages of the experiment, the goal is to explore the performance limits of the controller with more complex maneuvers. For these reasons, an intermediate 20° step maneuver has been selected to calibrate both controllers. An example is shown in Figure 4. This maneuver is divided into three main phases:

- Initial phase: The satellite begins aligned with the target axes and must maintain the angular error approximately at zero. This phase is used to evaluate the initial stability metrics.
- Step phase: A 20° step is commanded around the body axis under calibration. The controller must respond accordingly to bring the satellite to the new target orientation. The duration of this phase depends on the dynamics of the specific axis and is used to evaluate the performance metrics targeted by the MOPSO optimization (see Section 3.4).
- Final phase: Once the target orientation is reached, the simulation continues for several seconds, during which the controller must again maintain the angular error close to zero. This phase is used to assess residual stability metrics.

The duration of the step phase varies depending on the dynamics of the system. In the case of Astrobee, the simulations show that a duration of 30 s is sufficient to reach steady state. Therefore, a total duration of 1 min is allocated for the complete maneuver to provide sufficient margins and to allow flexibility in finding diverse controller configurations.

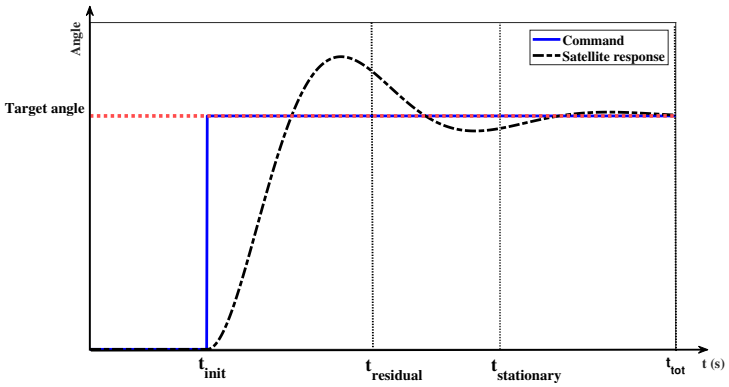


Figure 4. Example of the calibration maneuver.

3.4. Controller Optimization

A MOPSO [14,24] is used to minimize two performance metrics, hereafter referred to as Error and Cost. The Error metric is defined as the integral along the calibration maneuver of the absolute value of the attitude quaternion error on the axis being optimized. Lower error metrics lead to faster convergence of attitude error. The Cost metric quantifies the control effort and is defined as the integral of the absolute value of the angular acceleration on the same axis, see Equation (5). Since angular acceleration $\boldsymbol{\alpha}(t)$ is directly related to the applied control torque through the spacecraft inertia tensor, this metric effectively represents the total control input effort over time. Both metrics are normalized with respect to reference values; so, these metrics have the same order of magnitude.

$$\text{Error}_k = \frac{\int_{t_{\text{init}}}^{t_{\text{tot}}} |\mathbf{q}_{ek}(t)|\, dt}{\text{Error}_{\text{ref}}}, \quad \text{Cost}_k = \frac{\int_{t_{\text{init}}}^{t_{\text{tot}}} |\boldsymbol{\alpha}_k(t)|\, dt}{\text{Cost}_{\text{ref}}}, \tag{5}$$

where $|\mathbf{q}_{ek}|$ and $|\boldsymbol{\alpha}_k|$ represent the absolute value of the $k = 1, 2, 3$ component of $\mathbf{q}_e$ and $\boldsymbol{\alpha}$, respectively. The same Error and Cost performance metrics are used for the PID and the fuzzy controller per each axis optimization. This unified optimization framework ensures a consistent and fair basis for comparing the resulting Pareto fronts.

4. Results

Figure 5 presents the Pareto fronts obtained for the PID and fuzzy controllers along the three rotational axes. In all cases, a clear trade-off between Error and Cost metrics is observed, with similar Pareto shapes across the X, Y, and Z axes. For all axes, the Pareto front of the fuzzy controller dominates the PID Pareto front in the central area, that is, exhibits simultaneously lower Error and lower Cost metrics than the corresponding central solutions of the PID. In addition, the fuzzy controller achieves operating points with a low Cost metric that the PID is unable to achieve. On the other hand, the PID achieves operating points with a low Error metric that the fuzzy controller cannot achieve. It is important to note the asymptotic behavior for both low Error and low Cost areas; that is, a small decrease in Error entails a considerable increase in Cost and vice versa.

From each Pareto front, three representative solutions are selected—namely, a Low Cost point (LC), a Low Error point (LE), and a Central Point compromise solution (CP).

The Central Point is defined as the solution with intermediate values of Error and Cost on the Pareto front: it is the farthest point from all extremes.

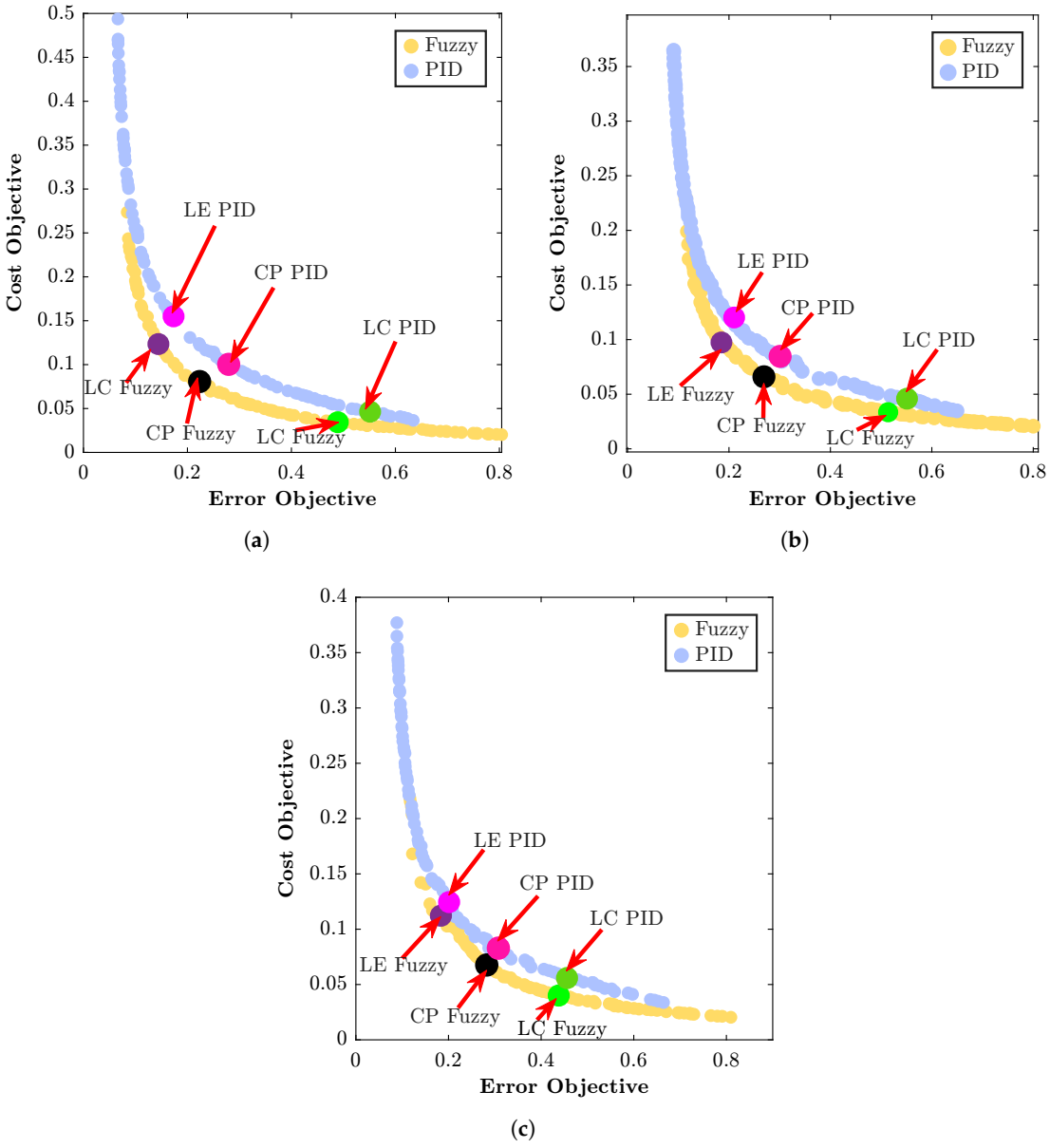


Figure 5. Pareto fronts for the PID and fuzzy controllers on the (**a**) X, (**b**) Y and (**c**) Z axis. From each Pareto front, a LC (Low Cost), a LE (Low Error) and a CP (Central Point) are selected.

To better understand the operation of the fuzzy controller, the nominal performance of the X-axis fuzzy and PID controllers for the central point are presented in Figure 6.

The fuzzy controller produces an initial high-magnitude control pulse to promptly initiate the system's motion toward the desired orientation. After this initial impulse, the controller allows the system to evolve at a nearly constant angular velocity, applying little to no torque, until it approaches the commanded angle, at which point it initiates braking. This control strategy results in smooth responses and contributes to the controller's efficiency in terms of energy consumption. No overshoot appears in the reference maneuver. In contrast, the PID controller produces a smaller initial impulse and continues applying control torque throughout the entire maneuver, which results in higher energy consumption. Consequently, no phase of constant angular velocity appears, and a slight overshoot can be observed.

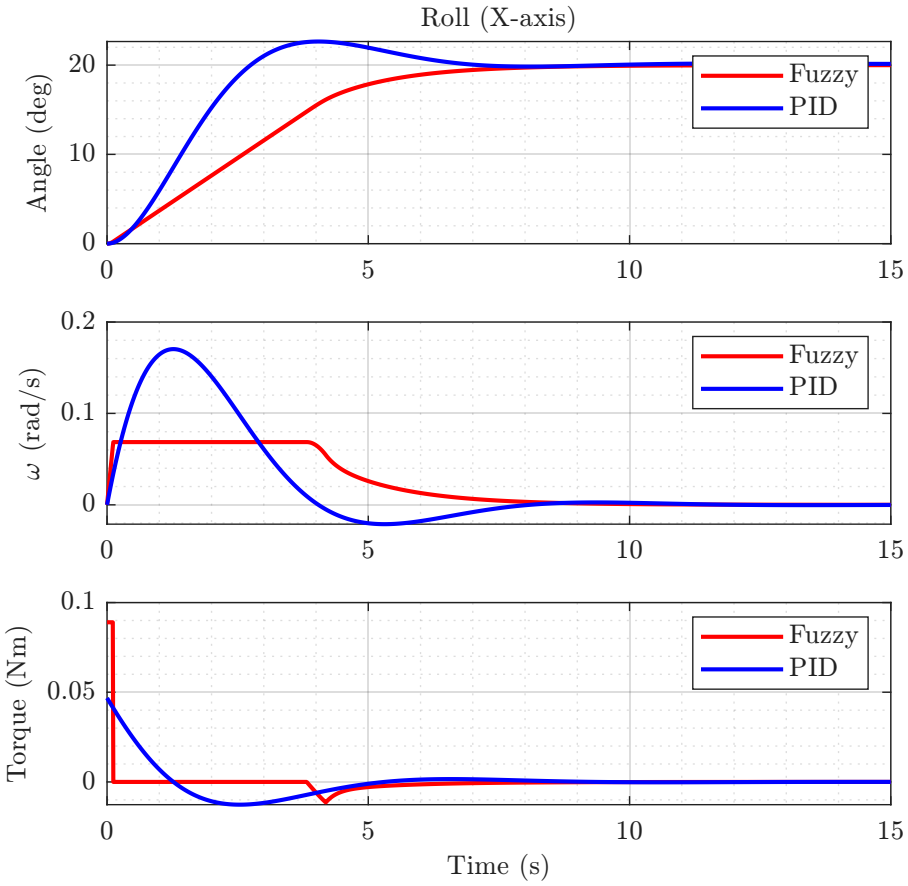


Figure 6. Roll angle, angular velocity and torque about roll axis for fuzzy and PID controller.

Figures 7 and 8 present the attitude error and normalized control torque corresponding to the Low Error, Low Cost, and Central Pareto points for PID and fuzzy controllers along the X, Y, and Z axes. As expected, Low Error solutions achieve the fastest error convergence (the dashed lines in Figure 7 decay to zero earlier than the others), at the expense of increased control effort (The dashed lines in Figure 8 show an initial peak corresponding to the maximum available torque (see the detail in subfigure (a)), followed by a second peak used to brake the satellite and complete the maneuver. This second peak appears earlier than in the other points and has a higher magnitude.). In contrast, the Low Cost solutions exhibit a slower decay of the attitude error (the dotted lines in Figure 7 take longer to converge than the others), reflecting a more conservative control action. The Central solutions provide a balanced compromise between the speed of convergence and the effort of control.

As can be seen in the detailed views of Figure 7, although the PID theoretically achieves a steady-state error of zero, it does so only in infinite time. Therefore, within reasonable response times, the steady-state error of the Fuzzy controller (red) is observed to be smaller than that of the PID (blue).

It can be clearly seen how the fuzzy controller consistently achieves lower control effort for the Central and Low Cost solutions, while maintaining smooth torque profiles with limited peak actuation. In the detailed view of Figure 8b, the same behavior as in Figure 6 can be observed in the commanded torque. While the PID controllers apply torque continuously, the fuzzy controllers apply no torque between the initial impulse and the braking impulse. As a result, the integral of this curve is smaller, meaning that the overall cost is lower. The Low Error solutions for both controllers exhibit higher initial torque peaks (see the detailed view of Figure 8a), as expected, due to the aggressive control action required for rapid error reduction. The PID controller generally produces

moderate torque levels with longer actuation durations and overshoot. Across all three axes, the fuzzy controller exhibits the fastest error convergence for all the operating points in the nominal case.

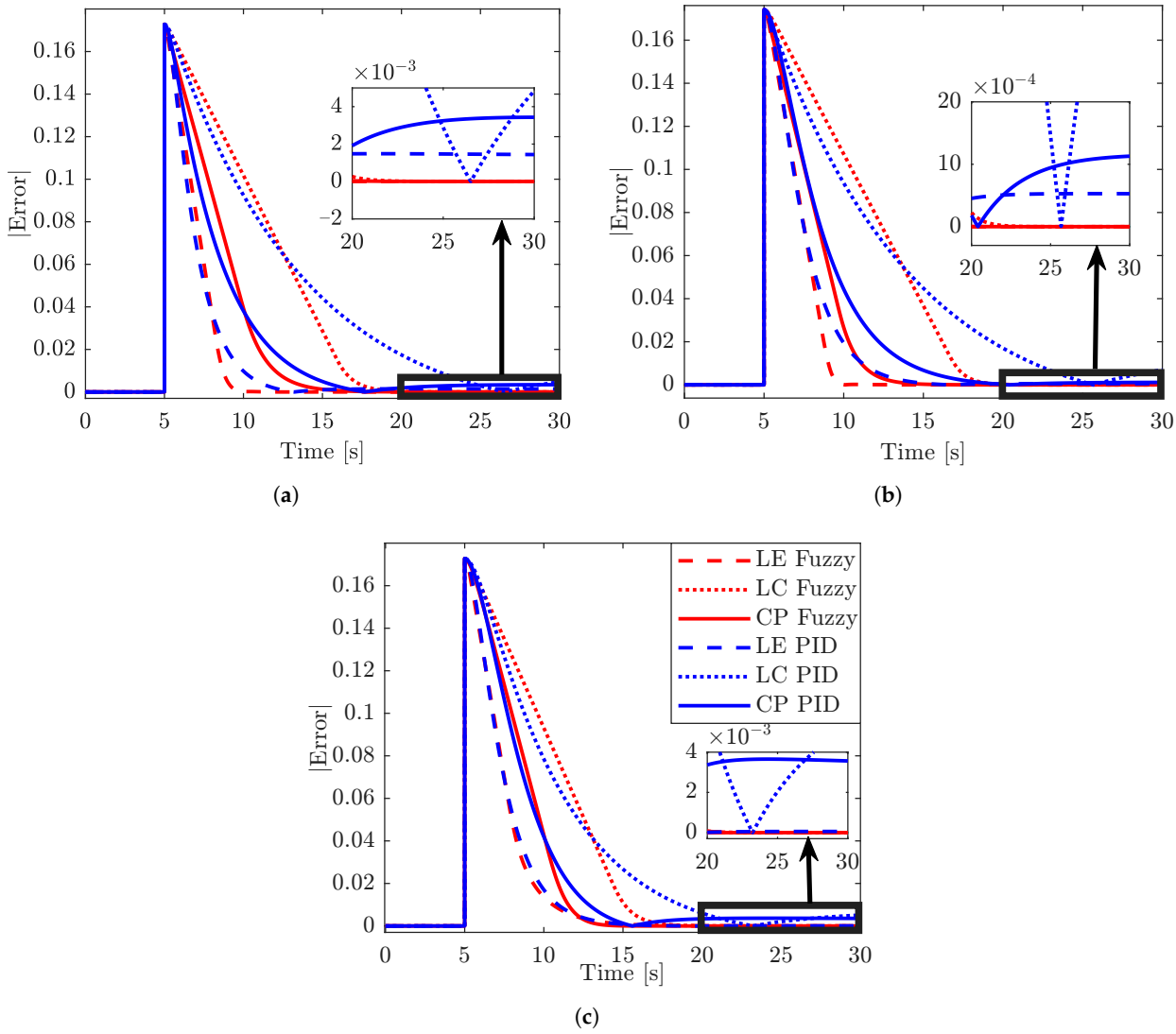


Figure 7. Error signals for the PID and fuzzy controllers on the (**a**) X axis, (**b**) Y axis, and (**c**) Z axis.

As we said before, the operating points selected from the Pareto front correspond to the 18 parameters of the membership functions of the fuzzy controller and the nine gains of the PID controller, having performed the optimization for the calibration maneuver of 20°. However, in the FuzzyBuzz experiment, we want Astrobee to be subjected to a wide range of step maneuvers. If a 90° maneuver is performed, the PID behavior is similar to that obtained for 20°, but the fuzzy controller performs much worse. Since all membership functions were calibrated for a 20° input, any maneuver with a larger amplitude is as "Big" (PB or NB), leading to an overly aggressive response and a degradation in performance. This led to repeating the optimization, this time choosing a 90° calibration maneuver. Figure 9 presents the Pareto fronts obtained for the PID and fuzzy controllers along the X axis. Similar results are obtained for the Y and Z axes.

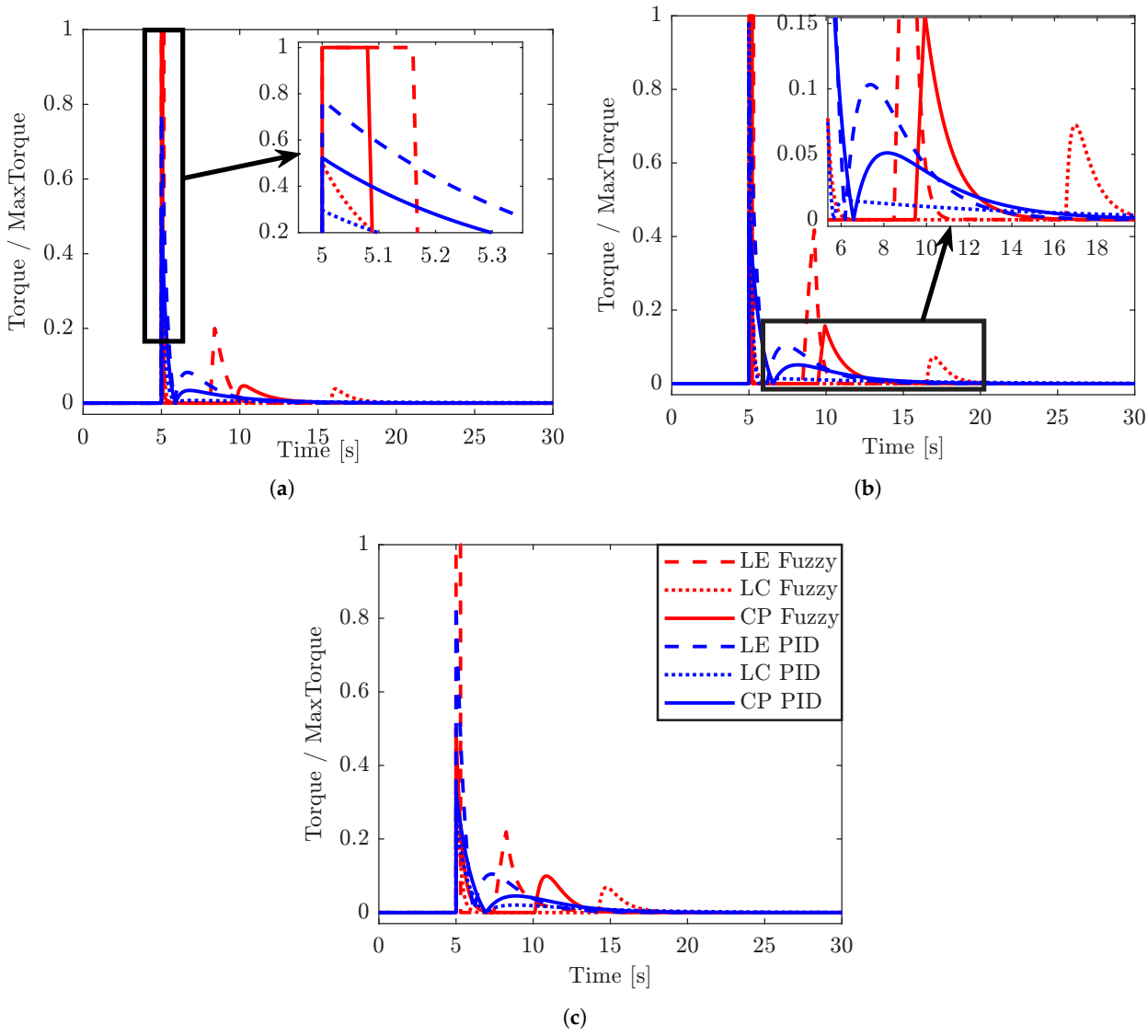


Figure 8. Cost signals for the PID and fuzzy controllers on the (**a**) X axis, (**b**) Y axis, and (**c**) Z axis.

In order to have a fuzzy controller adaptable to step maneuvers over a wide range of values, it was decided to adapt the design of the membership functions so that in the low and medium error and angular velocity regions, that is, Z, PS and NS regions, the parameters of the Pareto front for 20° will be used, while in the more extreme regions, that is, P, PB, N and NB, the parameters of the Pareto front for 90° will be used, see Figure 10. In this figure, the values of parameters X1, X2, X4, and X5, which define the membership functions near zero, and therefore the steady-state behavior, correspond to the optimization results obtained for the 20° maneuver. The values of parameters X3 and X6, which are farther from zero and thus govern the behavior when the error and/or angular velocity are large, have been taken from the optimization performed for the 90° maneuver. This hybrid design achieves a much better response across the entire range of maneuvers, avoiding overly aggressive actions when the error or angular velocity are large, while still maintaining convergence, as can be seen in the Monte Carlo analysis below.

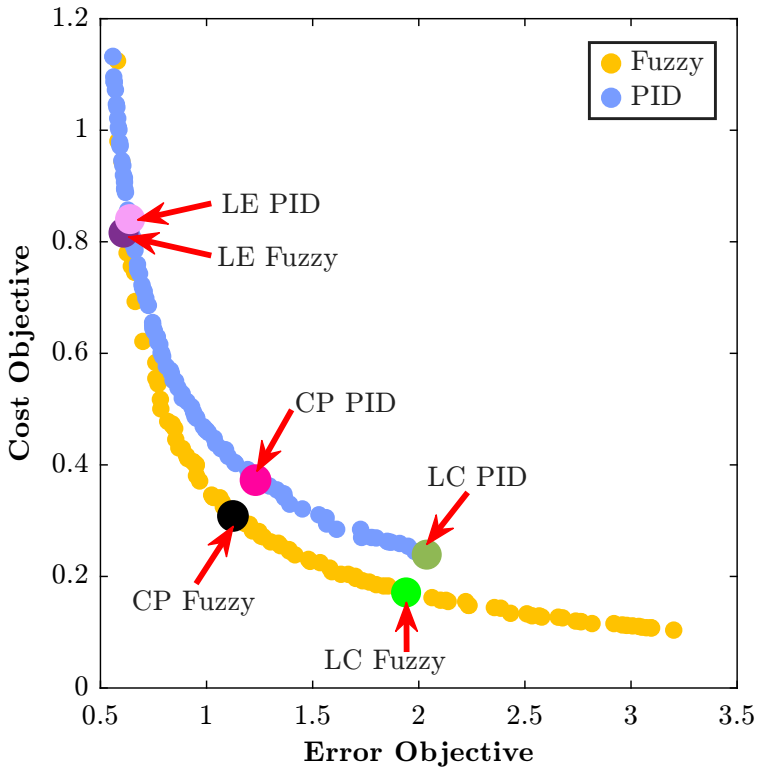


Figure 9. Pareto comparison of the Fuzzy and PID controllers on the X axis for a 90° maneuver. From each Pareto front, a Low Cost, a Low Error and a Central point are selected.

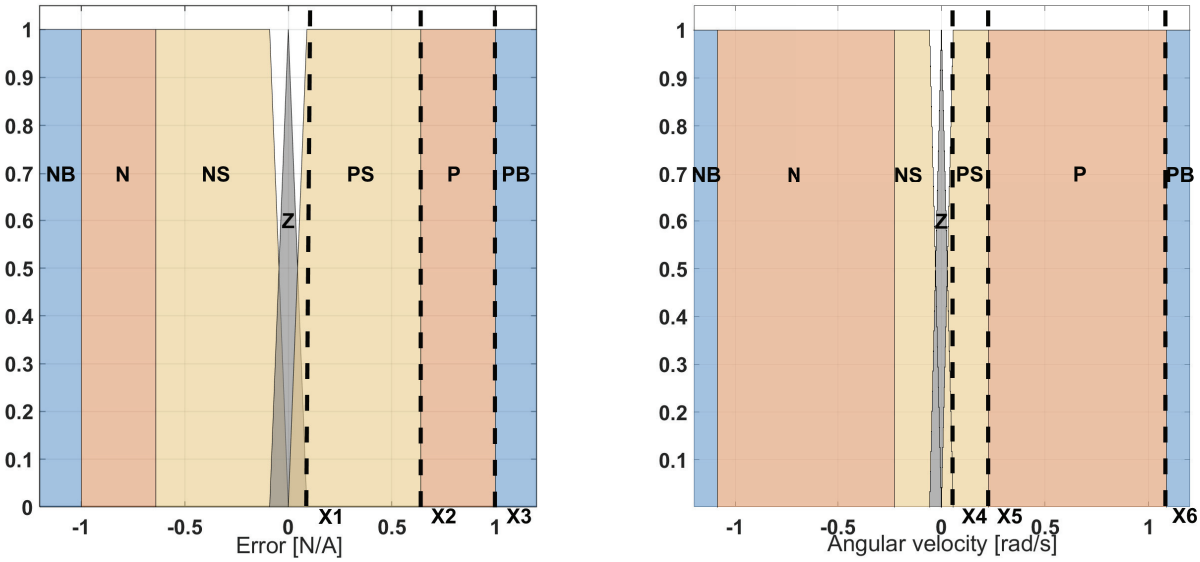


Figure 10. Combined memberships for the Z axis and central configuration.

Monte Carlo Analysis

To evaluate the robustness of the PID and fuzzy control strategies against parameter uncertainties, a Monte Carlo analysis was conducted. The variability is introduced in three main aspects of the Astrobee dynamics and mission profile:

1. Inertia Uncertainty: The principal moments of inertia (I_{xx}, I_{yy}, I_{zz}) are uniformly varied within a range of ±50% relative to the nominal Astrobee inertia tensor ($I \in [0.5, 1.5] \cdot I_{nom}$). This accounts for mass distribution changes due to possible variations in the final configuration with respect to the official documentation.

2. Actuator Degradation: The maximum torque available (τ_{max}) for each axis is randomly varied. The range extends from 50% to 110% of the nominal maximum torque.
3. Maneuver Geometry: To ensure that the controller is not tuned for a single specific trajectory, the step maneuver is varied between $\pm 90^\circ$ for each axis to cover a wide operational envelope.

In the Monte Carlo campaign, the control command is applied at t = 5 s. The initial interval $0 \leq t < 5$ s is intentionally included to verify that the system remains quiescent and free of spurious transients prior to command activation. A total of $N = 5000$ simulations per controller have been performed. All trials were executed using fixed fuzzy and PID controller parameters (no retuning).

For each simulation run, the overall performance is quantified using two global normalized metrics: the Attitude Error Index (J_e) and the Control Effort Index (J_u). The calculation is performed in a two-step process. First, the integral of the absolute error and the control action are computed for each axis during the maneuvering phase ($t \geq 5$ s), as defined in Equation (6):

$$J_{ek}^{(i)} = \frac{1}{J_{e,ref}} \int_{t_{init}}^{t_{tot}} |\mathbf{q}_{e,k}(t)|\, dt, \quad J_{uk}^{(i)} = \frac{1}{J_{u,ref}} \int_{t_{init}}^{t_{tot}} |\boldsymbol{\tau}_k(t)|\, dt, \tag{6}$$

where i indicates the run (for i = 1 to 5000), $t_{init} = 5$ s, $\boldsymbol{\tau}(t)$ is the applied control torque, and $J_{e,ref}$, $J_{u,ref}$ are normalization factors calculated based on a reference maneuvering time and acceptable error bounds as described in Section 3.3. Once individual metrics are obtained for each axis, they are aggregated to compute the global performance indices, as shown in Equation (7):

$$J_e^{(i)} = \sqrt{\left(J_{ex}^{(i)}\right)^2 + \left(J_{ey}^{(i)}\right)^2 + \left(J_{ez}^{(i)}\right)^2}, \quad J_u^{(i)} = \sqrt{\left(J_{ux}^{(i)}\right)^2 + \left(J_{uy}^{(i)}\right)^2 + \left(J_{uz}^{(i)}\right)^2}, \tag{7}$$

In addition to the integral performance indices, unified time-dependent signals are calculated to evaluate the transient response of the satellite throughout the simulation. To provide a comprehensive view of the maneuver, a single global attitude error signal $\theta_e(t)$ and a global control signal $u(t)$ are generated for each run. The global attitude error is defined by the principal rotation angle extracted from the error quaternion. To ensure mathematical robustness and avoid quadrant ambiguities, this angle is computed using the arctangent of the ratio between the Euclidean norm of the quaternion's vector part and the absolute value of its scalar part as defined in Equation (8):

$$\theta_e(t) = 2 \arctan\left(\frac{\|\mathbf{q}_e(t)\|}{|q_{0e}(t)|}\right), \quad u(t) = \|\boldsymbol{\tau}(t)\|, \tag{8}$$

The temporal evolution of the simulations for the central configuration is shown in Figure 11. As can be seen in the graphs, the fuzzy controller shows a much more variable transient response than the PID, due to its adaptive strategy. All the fuzzy controller simulations converge in less than 25 s from the moment the step maneuver begins, whereas in the PID a higher steady-state error is observed, which it is not able to eliminate.

In summary, for a platform such as Astrobee, the fuzzy controller shows robustness in the presence of uncertainty in design parameters, actuators or command. The PID controller performs adequately but yields a similar response for all maneuvers, scaling the generated torque according to the maneuver amplitude. The fuzzy controller, however, seeks to maintain intervals of constant velocity when no action is required, thereby minimizing energy consumption. In any case, it completes the maneuver before second 30, as intended, and maintains a smaller steady-state error.

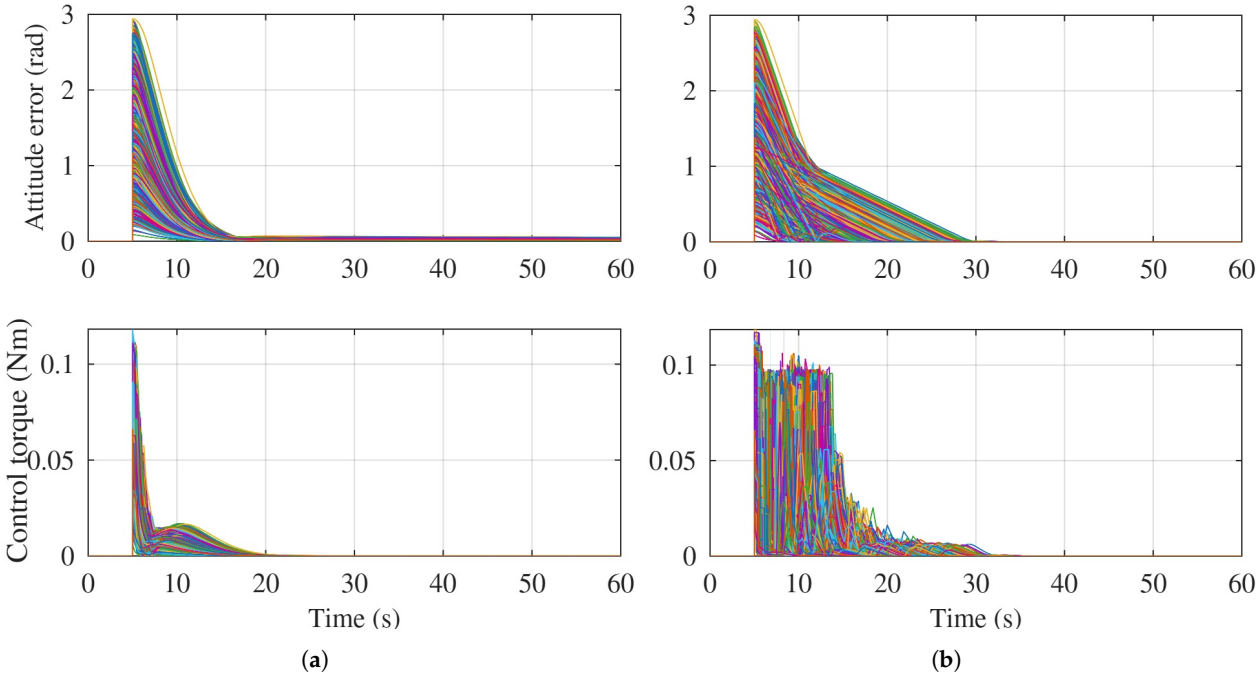


Figure 11. Time evolution of the attitude angle and control torque for the PID CP (**a**) and for the Fuzzy CP (**b**). Each line corresponds to one of the 5000 simulations.

In Figures 12 and 13, the correlation can be observed between the merit figure J_u, used to quantify cost in three axes, and the initial error expressed in Euler angles for the 5000 simulations per controller performed in the Monte Carlo analysis. The Pearson correlation coefficient, r, has been included, along with the statistical significance, p, which indicates the probability of obtaining such a correlation coefficient under the null hypothesis—that is, assuming no true correlation exists.

A correlation coefficient close to zero indicates the absence of a linear relationship, whereas a coefficient close to one indicates that one variable exhibits a linear dependence on the other. As can be observed, the fuzzy controller is less dependent on the maneuver amplitude. This is because it features a phase of constant angular velocity during which it does not act; actuation only occurs when the inputs transition from one membership function to another. Consequently, a maneuver of higher amplitude may simply involve a longer constant-velocity phase rather than an increase in energy consumption. This behavior is clearly visible in the points corresponding to lower consumption, where a noticeable jump appears that aligns with the boundary between the PS and P error regions.

In contrast, higher correlation coefficients are observed for the PID controller, which is consistent with the behavior expected from this type of linear control: since the controller acts throughout the entire maneuver, a larger maneuver amplitude entails a higher energy cost to complete it. This behavior can again be clearly observed in the points corresponding to the lowest cost metric, whose value increases linearly as the maneuver amplitude grows.

Regarding the statistical significance of these results, the large number of simulated maneuvers makes the probability that the computed coefficients are due to chance practically negligible.

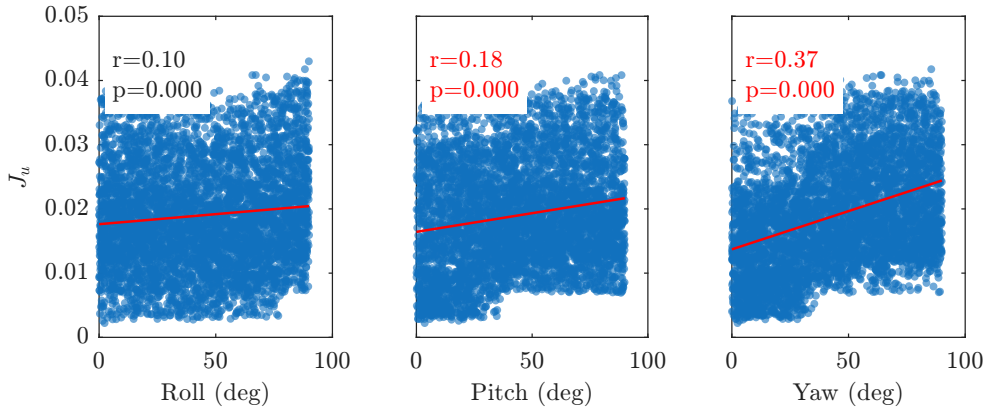


Figure 12. Correlations between the cost metric J_u and the initial error in Euler angles for the fuzzy CP controller.

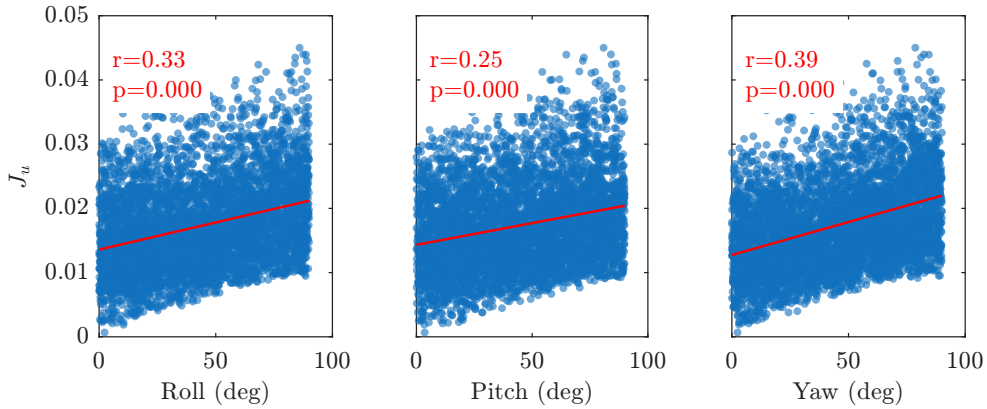


Figure 13. Correlations between the cost metric J_u and the initial error in Euler angles for the PID CP controller.

5. Operational Overview

As mentioned in the introduction, four controllers will be tested and compared in the FuzzyBuzz experiment: PID, H_∞, fuzzy and MPC. For H_∞ and MPC, operating points similar to those of the PID and fuzzy will be found; Low Error, Low Cost, and Central will be obtained for H_∞ and MPC with a different procedure.

The experiment consisted of a minimum of six runs; each run consists of a series of maneuvers which are performed via scripts uploaded to the Astrobee. The runs can be grouped in the operational slots enumerated in Table 4.

Table 4. Operational slots.

Maneuver	Description
Man #0	Default maneuver (required). This maneuver uses the controller already built into Astrobee to guarantee identical initial conditions across all experiments.
Man #1	Performing a step maneuver with a controller (required).
Man #2	Returning to the initial attitude (required).
Man #3–8	Repeating Man# 1 and Man# 2 with the other controllers (required).
Man #9–16	Repeating the sequence (Man# 1 to Man# 8) in the opposite direction. Only required for the first run; if the results are the same, the other runs are optional.

The maneuver sequence is shown in Figure 14.

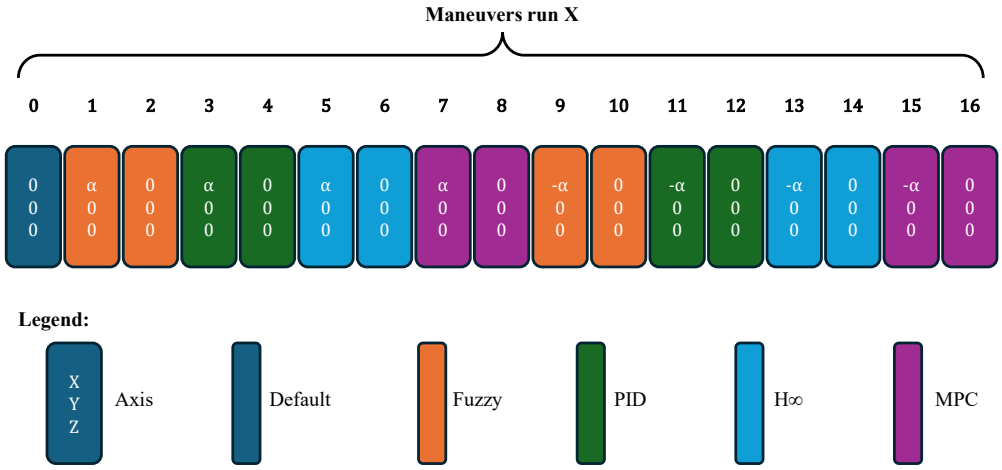


Figure 14. Maneuver sequence example for a run about X-axis.

The runs are grouped into six priorities:

1. Nine runs (Run# 1 to Run# 9) consisting of single-axis step maneuvers using the four controllers set at the central operating point parameters The maneuver of Run# 1 begins with a 10° step in one axis and increases in 10° increments for each axis until it reaches 30° for Run# 9.
2. Nine runs (Run# 10 to Run# 18) consisting of the same maneuvers of priority 1, now using the four controllers set at the Low Cost operating point parameters.
3. Nine runs (Run# 19 to Run# 27) consisting of the same maneuvers of priority 1, now using the four controllers set at the Low Error operating point parameters.
4. Nine runs (Run# 28 to Run# 36) consisting of three axis maneuvers using the four controllers set at all three characteristic operating points parameters.
5. Nine runs (Run# 37 to Run# 45) consisting of three axis maneuvers using the four controllers set at all three characteristic operating points parameters with higher maneuver amplitude.
6. Three runs (Run# 46 to Run# 48) consisting of complex maneuvers (e.g., sinusoidal, geodetic pointing).

The runs in priorities 1–3 are required, while those in priorities 4–6 are desired. Each run can be executed independently and can be organized in slots to adapt the execution pace to the operational constraints.

6. Discussion

The possibility of conducting an in-orbit experiment such as FuzzyBuzz aboard the ISS represents a significant opportunity for the validation of advanced attitude control strategies in a realistic microgravity environment. Unlike purely numerical studies, Astrobee operates under hardware-constrained actuation, real onboard computational limitations, and operational constraints that closely resemble those of small spacecraft. This makes it an exceptional intermediate platform between simulation and fully operational space missions, which directly contributes to the maturation of guidance, navigation, and control technologies.

This work has presented the design, multi-objective optimization, and simulation-based evaluation of a fuzzy logic attitude controller for the Astrobee platform, including a numerical comparison between the fuzzy controller and a PID controller for different scenarios. The MOPSO-based tuning procedure generates Pareto fronts that explicitly characterize the trade-off between two metrics, Error and Cost, related to the convergence time of the attitude error and energy consumption, allowing mission-driven selection of

operating points. The comparison has shown that an optimized fuzzy strategy constitutes a competitive alternative to classical PID control for spacecraft attitude control. The fuzzy controller dominates the PID performance in the central and Low Cost regions of the Pareto front, achieving the fastest error convergence and reduced energy consumption, while the PID controller proves more effective in scenarios where very fast convergence is required. Fuzzy torque profiles produce smooth transient responses with limited overshoot and reduced steady-state error. Under significant parametric uncertainty—including inertia variations, actuator degradation, and maneuver randomization—the fuzzy controller preserves convergence and eliminates residual steady-state errors that appear in Medium and Low Cost regions in PID configurations. Nevertheless, the results also reveal an important limitation: maneuver-calibration optimization reduces fuzzy robustness during maneuvers over large steps. Although the proposed hybrid membership strategy improves adaptability across different maneuver amplitudes, more research is recommended toward adaptive formulations.

The present study not only addresses control design but also provides a detailed description of the onboard software architecture and the experimental protocol. The modular ROS-based structure has been designed to ensure controlled execution, reproducibility, and compliance with flight operational requirements. In addition, the systematic definition of maneuver sequences, characteristic operating points, and normalized performance metrics establishes a rigorous benchmarking framework. This framework will enable a fair in-orbit comparison among fuzzy, PID, H_∞ and MPC controllers under identical operating conditions, reinforcing the experimental value of FuzzyBuzz beyond purely theoretical analysis.

In conclusion, this study reinforces the importance of conducting FuzzyBuzz as a flight-level benchmark for nonlinear and classical control strategies. The forthcoming in-orbit results will provide decisive validation of the performance trade-offs identified in simulation and contribute to the advancement of intelligent control methodologies for future spacecraft applications.

Finally, it should be noted that this work represents the first attitude control experiment designed by our team using the Astrobee platform, and its primary objective is the experimental validation and comparison of the controllers developed in this study. Consequently, the maneuvers implemented in the experiments are relatively simple, with only maneuvers #46–#48 introducing a higher level of complexity. More advanced scenarios, such as docking maneuvers or operations involving multiple spacecraft simultaneously, were not considered within the scope of the present study. These scenarios represent promising directions for future work.

Author Contributions: Conceptualization, K.O. and M.R.; methodology, K.O.; software, A.A. and C.F.; validation, J.C.C. and M.R.; formal analysis, A.A. and C.F.; investigation, J.M.E. and M.R.; resources, J.C.C. and M.R.; data curation, A.A., J.C.C. and M.R.; writing—original draft preparation, M.R. and J.M.E.; writing—review and editing, K.O. and J.M.E.; visualization, K.O.; supervision, K.O.; project administration, K.O. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this study are available in the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Orozco, M.L.; Giraldo, B.S. Attitude Determination and Control in Small Satellites: A Review. *IEEE J. Miniaturizat. Air Space Syst.* **2024**, *5*, 182–186. [CrossRef]
2. Åström, K.J.; Hägglund, T. *PID Controllers: Theory, Design, and Tuning*; ISA: Durham, NC, USA, 1995.
3. Yang, Y. *Spacecraft Modeling, Attitude Determination, and Control: Quaternion-Based Approach*; Taylor & Francis Group: Boca Raton, FL, USA, 2024.
4. Doyle, J.C.; Glover, K.; Khargonekar, P.P.; Francis, B. State-Space Solutions to Standard H-infinity Control Problems. *IEEE Trans. Autom. Control* **1989**, *34*, 831–847. [CrossRef]
5. Olfe, K.; Bello, A.; Fernández, J.; Rodríguez, J.; Lapuerta, V. Design and experimental verification of a robust control to avoid individual calibration of multiple reaction wheels in satellites. *Adv. Space Res.* **2025**, *75*, 767–778. [CrossRef]
6. Milaševičius, M.; Mačiulis, L. A Review of Mechanical Fine-Pointing Actuators for Free-Space Optical Communication. *Aerospace* **2024**, *11*, 5. [CrossRef]
7. Kim, G.-N.; Park, S.-Y.; Seong, S.; Choi, J.-Y.; Han, S.-K.; Kim, Y.-E.; Choi, S.; Lee, J.; Lee, S.; Ryu, H.-G.; et al. Design of Novel Laser Crosslink Systems Using Nanosatellites in Formation Flying: The VISION. *Aerospace* **2022**, *9*, 423. [CrossRef]
8. Zadeh, L.A. Fuzzy sets. *Inf. Control* **1965**, *8*, 338–353. [CrossRef]
9. Zhang, Y.; Guo, X.; Wu, Y.; Zhang, Y.; Lü, C. Active control of cables with piezoelectric actuation considering geometric and material nonlinearities. *Eng. Struct.* **2025**, *340*, 120773. [CrossRef]
10. Mazmanyan, L.; Ayoubi, M.A. Fuzzy Attitude Control of Spacecraft with Fuel Sloshing via Linear Matrix Inequalities. *IEEE Trans. Aerosp. Electron. Syst.* **2018**, *54*, 2526–2536. [CrossRef]
11. Calvo, D.; Avilés, T.; Lapuerta, V.; Laverón-Simavilla, A. Fuzzy attitude control for a nanosatellite in low Earth orbit. *Expert Syst. Appl.* **2016**, *58*, 102–118. [CrossRef]
12. Calvo, D.; Bello, Á.; Lapuerta, M.V.; Laverón-Simavilla, A. Comparison of Fuzzy and PID Controllers for the Attitude Control of Nanosatellites. In *Intelligent Systems and Applications*; Advances in Intelligent Systems and Computing; Springer: Cham, Switzerland, 2018; Volume 869, pp. 1062–1081. [CrossRef]
13. Bello, A.; Del Castañedo, A.; Olfe, K.S.; Rodríguez, J.; Lapuerta, V. Parameterized fuzzy-logic controllers for the attitude control of nanosatellites in low earth orbits: A comparative study with PID controllers. *Expert Syst. Appl.* **2021**, *174*, 114679. [CrossRef]
14. Albareda, R.; Olfe, K.S.; Bello, A.; Fernández, J.J.; Lapuerta, V. Comparison of Optimization Methods for the Attitude Control of Satellites. *Electronics* **2024**, *13*, 3363. [CrossRef]
15. Sun, X.G.; Chi, W.C.; Wang, Y.Q. Time-delay measurement and decoupling compensator control design for flexible beams under various modal excitations. *Measurement* **2026**, *257*, 118675. [CrossRef]
16. del Castañedo, Á.; Bello, Á.; Olfe, K.; Lapuerta, V. Effect of the Delay in Fuzzy Attitude Control for Nanosatellites. In *Intelligent Systems and Applications*; Springer International Publishing: Cham, Switzerland, 2020; pp. 966–981. [CrossRef]
17. Bello, Á.; Olfe, K.S.; Rodríguez, J.; Ezquerro, J.M.; Lapuerta, V. Experimental Verification and Comparison of Fuzzy and PID Controllers for Attitude Control of Nanosatellites. *Adv. Space Res.* **2023**, *71*, 3613–3630. [CrossRef]
18. NASA Ames Research Center. Astrobee Robot Software Repository and API Documentation. 2020. Available online: https://github.com/nasa/astrobee (accessed on 20 January 2026).
19. Bualat, M.A.; Barlow, J.; Fong, T.; Provencher, C.; Smith, T.; Zuniga, A. Astrobee: Developing a Free Flying Robot for the International Space Station. In Proceedings of the AIAA SPACE 2015 Conference and Exposition, Pasadena, CA, USA, 31 August–2 September 2015. [CrossRef]
20. Ruggiero, D. Robust Attitude Controller for NASA Astrobee Robots Operating in the ISS. *Aerotec. Missili Spaz.* **2021**, *100*, 185–193. [CrossRef]
21. Doerr, B.; Albee, K.; Ekal, M.; Ventura, R.; Linares, R. The ReSWARM microgravity flight experiments: Planning, control, and model estimation for on-orbit close proximity operations. *J. Field Robot.* **2024**, *41*, 1645–1679. [CrossRef]
22. Albee, K.; Ekal, M.; Oestreich, C.; Roque, P. A Brief Guide to Astrobee's Flight Software (Rev. 1.1). 2021. Available online: https://github.com/albee/a-brief-guide-to-astrobee/raw/master/a_brief_guide_to_astrobee_latest.pdf (accessed on 20 January 2026).
23. Jiang, B.; Gao, Z.; Shi, P.; Xu, Y. Adaptive Fault-Tolerant Tracking Control of Near-Space Vehicle Using Takagi–Sugeno Fuzzy Models. *IEEE Trans. Fuzzy Syst.* **2010**, *18*, 1000–1007. [CrossRef]
24. Coello, C.C.A.; Lechuga, M.S. MOPSO: A Proposal for Multiple Objective Particle Swarm Optimization. In Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600), Honolulu, HI, USA, 12–17 May 2002; Volume 2, pp. 1051–1056. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.