Alpha CubeSat Attitude Control System (ACS) Design, Analysis, and Integration

Nicholas R. Natsoulas (nnatsoulas@gmail.com) *
December 7, 2023
Cornell University Space Systems Design Studio, Ithaca, NY, 14850

This paper details the Attitude Control System (ACS) of the Alpha CubeSat. This low-cost 1U CubeSat mission aims to test a highly retroreflective material for light-sail propulsion, focusing on system validation and final preparations ahead of flight checks. The ACS shall detumble and spin the CubeSat about its maxima principal axis of inertia and point the spin-stabilized principle axis in a reference direction. In particular, detumbling, spin-stabilization, and pointing requirements are necessary for the satellite to stabilize its attitude once released from the International Space Station (ISS). The control laws, first chosen and designed by Carabellese [1], underwent rigorous tuning through Monte Carlo methods and edge-case simulations to meet subsystem requirements. Hardware checks ensure consistency between the gyroscope basis and ACS software definitions and testing for sensor bias or noise due to temperature variations. A significant aspect of Alpha CubeSat's ACS development involves its Extended Kalman Filter (EKF). Programmed in C++, this EKF is a high-performance algorithm for sensor fusion between the satellite's magnetometer and gyroscope. The remaining validations include temperature testing and an air-bearing test of the pointing control law. In aggregate, these efforts justify including the ACS's feedback control architecture in the final flight software for the mission.

Nomenclature

(All dimensional terms have SI units unless specified otherwise below.)

α = angular acceleration

A = state-transition matrix

B = external magnetic field vector

b = normalized external magnetic field vector

c = damping constant

e = error

*Undergraduate Researcher, 124 Hoy Rd Ithaca NY 14850 USA.

H = state-to-measurement matrix

H_{tot} = total angular momentum

h = angular momentum of the spinning disk

I_b = inertia matrix of the external body

I_d = inertia matrix of the damper

I_s = spin axis inertia component

J = jacobian matrix

K = Kalman gain

K_p = proportional gain

K_d = derivative gain

m = torque magnetic moment

P = state covariance matrix

Q = process noise covariance matrix

R = measurement covariance matrix

T = kinetic energy

u = control input

V = lyapunov function

x = state estimate

z = measurement

Z = attitude state

τ = torque

Ω = desired spin rate (radians per second)

ω = angular velocity (radians per second)

Subscripts and Superscripts

b = satellite body frame

d = damper

k = discrete step

N = inertial frame

P = prediction

x = first axis in the body frame

y = second axis in the body frame

z = third axis in the body frame

Contents

I. Introduction

ALPHA CubeSat's ACS is a technological demonstration of magnetorquer-only attitude control [1] for small satellites. Magnetorquers are low-cost and low-mass rods of coiled wire that generate a magnetic moment when subject to an applied current and a magnetic field. These magnetorquers do not generate large amounts of angular acceleration; rather, they provide a means of gentle convergence to a desired attitude and spin rate. The ACS's microcontroller, an Arduino teensy, sensors, a gyroscope, and an Inertial Measurement Unit (IMU) with a magnetometer and accelerometer are also low-cost and low-power. The trade-off between SWaP-C (Size, Weight, and Power,-Cost) and prolonged attitude convergence time is desirable for mission architectures that value low cost and have a long mission duration. For example, magnetorquers are a good candidate for attitude actuation for large-scale distributed space systems that require individual satellites to have low-Swap and coarse attitude control.

The requirements of spin stabilizing Alpha about the maximum principal axis of inertia and pointing this spin axis along the oscillating direction of the magnetic field due to the ISS-inclination orbit are inherently nonlinear problems. The nonlinear dynamics and the reliance of magnetorquers on the Earth's magnetic field is the root cause for Carabelllese's decision to derive a Lyapunov-stable attitude control law [1] for detumbling the satellite from its initial spin condition. Once detumbled and spin-stabilized by the Lyapunov-derived control law, the next step is the pointing maneuver to align with the magnetic field axis. This maneuver relies on a Proportional-Derivative (PD) control law. These maneuvers performed in succession result in an ideal attitude for light-sail deployment and testing. Given a proper tuning of both control laws, the controller will achieve the expected outputs if it receives accurate feedback signals.

Given the low SWaP-C sensors in tandem with the uncertainties of the space environment and the Earth's magnetic field, there can be a lot of noise for the magnetometer and the gyroscope. An Extended Kalman Filter, a near-optimal nonlinear state estimation algorithm, is implemented to provide filtering via sensor fusion of the magnetometer and gyroscope. The EKF is not a heuristic approach to filtering, like low or high pass filtering, and offers better state estimates than signal averaging. The EKF fits the needs of the controllers best, as averaging the state feedback from the sensors is insufficient.

The integration and validation process increases confidence in the ACS meeting its requirements during the mission. The gains of both the detumbling and pointing control laws are tuned using Monte-Carlo Simulations and Edge Case simulations. Through C++ software development, code optimization, and applied mathematics for numerical integration, the EKF implemented in the ACS software meets the filtering and performance standards necessary to provide reliable feedback to the controllers.

II. Design

This section defines the control laws, the EKF, and the different control modes that make up the ACS at a high level. The ACS design is crucial to the mission success criteria as spin stabilization makes communications between the satellite and ground stations more reliable and predictable than the scenario of a random tumble. Without reliable communications, light-sail deployment is immediately more challenging.

A. Attitude Parameterization

The parameterization implemented for Alpha's ACS uses quaternion and angular velocity:

$$Z = \begin{bmatrix} q_b \\ \omega_b \end{bmatrix} \tag{1}$$

Where the time derivative of the attitude state is:

$$\dot{Z} = \begin{bmatrix} \begin{bmatrix} {}^{B}\omega^{B/N} \\ 0 \end{bmatrix} \otimes q_b \\ \alpha_b \end{bmatrix} \tag{2}$$

This report references attitude differences in degrees and primarily utilizes angular velocity to represent attitude. However, the quaternion and angular velocity state parameterization is the basis of the simulation and embedded ACS software. Quaternions provide the advantage of being computationally efficient for run-time and memory compared to other representations.

B. Detumbling Controller

The detumbling controller is a Lyapunov-Stable nonlinear control law derived from the expression for the potential energy of the satellites's rotational motion. A control law with Lyapunov stability means a Lyapunov function exists for the system, and its time derivative is negative definite (or negative semi-definite). Negative-definite implies that the function is always decreasing except possibly at the equilibrium point.

Let ω represent the system's general attitude rate.

The Lyapunov function, denoted as $V(\omega)$, is associated with the system, and its time derivative is defined as:

$$\dot{V}(\omega) = \frac{dV(\omega)}{dt} \tag{3}$$

For Lyapunov stability, the time derivative must be negative definite (or negative semi-definite), which can be expressed as:

$$\dot{V}(\omega) < 0 \quad \text{(Negative Definite)} \tag{4}$$

or

$$\dot{V}(\omega) \leq 0 \quad \text{(Negative Semidefinite)} \tag{5}$$

This condition ensures that the Lyapunov function decreases over time, indicating system stability. Now that Lyapunov stability is defined, the following reviews, at a high level, Carabellese's derivation for the detumbling control law as described in his and Umansky-Castro's paper on Magnetorquer-only attitude control for the Alpha CubeSat[1]. The expression for potential used as the candidate Lyapunov function encompasses the rotation of the CubeSat's body about its axes and a fictional dynamical system interacting with the satellite's body to dampen nutation and spin. This fictional element is referred to as the Kane Damper. The Kane Damper models the motion of a sphere that rotates within a viscous liquid. The viscous liquid sits between the sphere and interacts with the satellite's body. This interaction produces a reactionary torque that resists the difference between the rotation of the inner sphere and the spacecraft's body. Representing the reference attitude as that of the imaginary sphere means that this damper model will yield torque computations for the satellite body that result in a slow convergence to the damper(reference) attitude. These torque computations ($u = \tau$) are the feedback control signals the magnetorquers must impart. The fictional nature of the Kane Damper and satellite body interaction allows for tuning of the properties of the damper system, which includes the sphere's inertia and a property analogous to the viscosity of the intermediate fluid layer. These control gains are denoted I_d and c, respectively.

The following derivation starts with an expression for the total angular momentum of the CubeSat and Kane-Damper system.

$$H_{tot} = I_d \cdot \omega^{d/N} + I_b \cdot \omega^{b/N} \tag{6}$$

The left-most term on the right-hand side is equivalent to the damper's angular momentum, and the right-most term is the angular momentum of the external body. Without any external torque or disturbances, $\tau \rightarrow 0$. Recognizing that the net torque on a rigid body is equivalent to the time derivative of the body's total angular momentum, the time derivative of this expression must be set to zero to get the equations of motion for the system without disturbance. The N stacked over a variable, in the form of $\overset{\text{N}}{X}$, represents the time derivative of that variable with respect to the inertial

reference frame.

$$\overset{\mathrm{N}}{H} = (I_d \cdot \overset{\mathrm{N}}{\omega}_d + \omega^{b/N} \times I_d \cdot \omega^{d/N}) + (I_b \cdot \overset{\mathrm{N}}{\omega}_b + \omega^{b/N} \times I_b \cdot \omega^{b/N}) = 0 \tag{7}$$

By isolating the second term in (7) such that it exclusively contains the dynamics for the rigid spacecraft body and excludes that of the damper, the equations of motion for the satellite's rotation are formed. Since the damper parenthesis of (7) and the equations of motion of the satellite's body sum to zero, the first parenthesis term is essentially the control torque expression from the damper model[1].

$$(I_d \cdot \overset{\mathrm{N}}{\omega}_d + \ \omega^{b/N} \times I_d \cdot \omega^{d/N}) = \tau_d \tag{8}$$

$$(I_b \cdot \overset{\mathrm{N}}{\omega}_b + \ \omega^{b/N} \times I_b \cdot \omega^{b/N}) = -\tau_d \tag{9}$$

Note that the damper is a uniform sphere; therefore, its inertia matrix is equivalent to the identity matrix scaled by the controller gain I_d, which is just the magnitude of the sphere's inertia.

This control torque acting from the Kane damper onto the spacecraft body is also described as a resistance between the two elements' angular velocities.

$$\tau_d = c(\omega^{b/N} - \omega^{d/N}) \tag{10}$$

Where c is the resistance factor.

The angular velocity of the damper with respect to the inertial frame is the sum of the angular velocity of the damper with respect to the body frame and the angular velocity of the body frame and the inertial frame. Taking the time derivative with respect to the inertial frame yields:

$$\overset{\mathrm{N}}{\omega}_d = \overset{\mathrm{b}}{\omega}_d + \overset{\mathrm{N}}{\omega}_b \tag{11}$$

Combining the equations (8), (9), (10), and (11), the angular acceleration of the damper can be solved algebraically. This achieves the following expression:

$$\overset{\mathrm{b}}{\omega}_d = -\overset{\mathrm{N}}{\omega}_b - (\frac{c}{I_d} I + \omega^{\times}_{b/N}) \cdot \omega^{d/b} \tag{12}$$

*Note that I is the rank 3 identity matrix and $\omega^{\times}_{b/N}$ is the skew anti-symmetric matrix[1].

The commanded torque is then replicated as closely as possible by the magnetic dipole m generated by passing current through the magnetorquers.

$$\tau = m \times B \tag{13}$$

To solve for the magnetic dipole moment for the magnetorquers, we set (9) equal to $m \times B$. This yields the result below.

$$m = -\frac{\tau_d \times B}{\|B\|^2} \tag{14}$$

Magetorquers generate this moment via an input current i:

$$m = niA \tag{15}$$

n is the number of coil rounds
A is the cross-sectional area of the coil

Where this moment is equivalent to the feedback control command u for detumbling:

$$u = m \tag{16}$$

This orthogonality between the moment generated and Earth's magnetic field means the m produced by magnetorquers at any given moment is the vector projection of the vector opposite of the damper torque and the magnetic field. The control moment is not always precisely achievable due to orthogonality between field and moment, so achieving as much of it as is physically possible is desired.

Now that the dynamics and the control law are established as the output magnetic dipole moment for the magnetorquers, the Lyapunov stability of the control law is confirmed with the expression for the total kinetic energy of the system, including the Kane Damper, as the Lyapunov function.

$$V = T = \frac{1}{2}\omega_{b/N}^T \cdot I_b \cdot \omega^{b/N} + \frac{1}{2}\omega_{d/N}^T \cdot I_d \cdot \omega^{d/N} \tag{17}$$

The equilibrium point is established as:

$$\omega_{eq} = \omega_{eq}^{b/N} = \omega_{eq}^{d/N} = \Omega \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{18}$$

It is important to note that both the damper and satellite body have the same angular velocity at this equilibrium point, and the scalar magnitude Ω is the desired rate about the stable, maximum principal axis. The stability of this equilibrium point is proven by taking the time derivative of the Lyapunov function described by (17) and evaluating whether it is negative definite, semi-definite, or neither. Through rearranging (17) and substituting in previous expressions as well as evaluating a triple product as done in Carabellese's paper [1], the time derivative of the Lyapunov function is reduced to the form:

$$\dot{V} = (\omega^{d/N} - \omega^{b/N})^T c(\omega^{d/N} - \omega^{b/N}) \tag{19}$$

This formulation shows that this Lyapunov function is negative definite for $c < 0$, meaning there is a stable tuning domain at equilibrium. This confirms the stability of the Kane Damper. The analysis section further discusses the detumble algorithm's stability efforts and what defines successful spin stabilization.

C. Pointing Controller

The Pointing Controller is designed based on the initial condition that the detumbling controller successfully spin stabilizes the satellite about its maximum principle axis of inertia. This assumption allows for simple Proportional-Derivative controller design and tuning for pointing the spin axis toward the Earth's magnetic field. The initial spin condition the PD controller inherits from the detumbling maneuver is approximately:

$$\omega_x \approx \omega_y \approx 0 \tag{20}$$

$$\omega_z \approx constant \tag{21}$$

This approximation permits simplified dynamics for the PD controller. The dynamics and control are, therefore, linearized as follows:

$$I \overset{\mathrm{N}}{\omega_b} = \tau \tag{22}$$

This means the magnetic dipole for this control law is expressed as:

$$m = -\frac{u}{\|B^b\|} \tag{23}$$

From this, the applied torque expression becomes:

$$\tau = \frac{u \times B}{\|B^b\|} = \frac{S(u)B^b}{\|B^b\|} \tag{24}$$

$S(u)$ is the anti-symmetric skew matrix and is approximately equivalent to the typical cross-product matrix of the u vector.

Finally, the PD control law is defined below:

$$u(t) = K_p e(t) + K_d \dot{e}(t) \tag{25}$$

e(t) is the attitude error defined as the difference between the spin axis and the Earth's magnetic field vector.

$$e(t) = sin^{-1}(\theta_{e_z B}(t)) \tag{26}$$

Where $\theta_{e_z B}(t)$ is the cross product of the spin axis with the magnetic field.

Lastly, the PD-controller gains K_p and K_d makes for reactive and predictive control without the nuisance of integral wind-up that is inherent with a PID controller. In the case of Alpha CubeSat, the lack of an integral gain is not problematic as the mission can afford the steady state attitude error in the pointing mode.

D. Attitude Control Algorithm Architecture

The overall architecture of Alpha CubeSat's Attitude Control Algorithm consists of three discrete modes. When first leaving the ISS, the satellite will tumble briefly while the electrical power systems (EPS) and the flight computer start. During this mission segment, the ACS remains in its "off" mode, where no power is being allocated to ACS, and so the algorithm nor the magnetorquers are active. Once these prerequisites are met, the satellite flight computer turns on the ACS's "detumble/spin-stabilize" mode, where the detumbling controller runs on the flight computer with state feedback from the IMU and gyroscope. This state feedback is filtered by the EKF a priori. Once the satellite is spin-stabilized about its maximum principle axis of inertia, the ACS mode is switched to "point," which utilizes the pointing controller to point the spin axis in the direction of the magnetic field. Note that these modes operate on the duty cycle of the ACS within the flight software. This means that although ACS may be in "point" or "detumble/spin-stabilize," it will alternate between its current mode and "off" as dictated by the frequency of the ACS duty cycle throughout the mission.

To summarize, the different control modes operate in the succession of "off" to "detumble/spin-stabilize" to "point" and remain at "point" for an arbitrarily necessary duration during the mission. Within the ACS software, the feedback

control loop starts with an initial attitude error. It utilizes filtered state feedback to minimize this error regardless of whether the ACS is in "detumble/spin-stabilize" or "point" mode. This control architecture is crucial for guaranteeing that the ACS meets its requirements of spin stabilization and pointing to provide a sufficient attitude state for the light-sail deployment and testing.

E. Extended Kalman Filter

The complexity of Alpha CubeSat's magnetorquer-only attitude control increases due to the inherent noise present in the IMU's magnetometer and the gyroscope sensor measurements. In response to this challenge, the EKF implements sensor-fusion of the gyroscope and magnetometer and uses a mathematical model to predict and correct the estimates for the attitude state feedback. State estimation outperforms heuristic filters and signal averaging and remains computationally lightweight.

Developed in C++ using the Eigen library for linear algebra, this EKF linearizes and discretizes the sensors' dynamics to find a near-optimal estimate provided state and state covariance of the system. With the EKF in C++ and not Matlab or Simulink, it is more easily integrated into flight software, tested, and simulated due to the performance boost of performance-optimized C++ software and the fact that all embedded flight software shares the same programming language.

Firstly, an overview of all the steps in the Kalman Filter algorithm:

1) Initialize System State Estimate and System State Error Covariance (first measurement step)
2) Predict System State and System State Error Covariance for next Measurement step
3) Compute the Kalman Gain based on Prediction and Measurement Uncertainty/Covariance
4) Use the Kalman Gain, State and Covariance Prediction, and State Measurement and Covariance to Estimate the System State and State Error Covariance for the following Measurement step
5) Repeat from step 2 and on for full control cycle duration.

Now that the high level of the algorithm steps are listed, the algorithm's mathematical equivalent is:

$$k = 0 \tag{27}$$

$$x_k \quad P_k \tag{28}$$

$$\begin{aligned} x_P &= Ax_k \\ P_P &= AP_kA^T + Q \end{aligned} \tag{29}$$

$$K_{k+1} = P_pH^T(HP_PH^T + R)^{-1} \tag{30}$$

$$\begin{aligned} x_{k+1} &= x_P + K_{k+1}(z_{k+1} - Hx_P) \\ P_{k+1} &= P_P - KHP_P \end{aligned} \tag{31}$$

$$k = k + 1 \tag{32}$$

repeat steps described by (28-32) until control duration finishes (33)

This algorithm outlines the inner workings of the linear Kalman filter. However, the EKF used in Alpha CubeSat diverges from this outline in two different ways. One is that it linearizes the dynamics by computing the Jacobian at each step, and the other is that it discretizes the linearized dynamics via a fixed-timestep Runga-Kutta fourth-order numerical integration method. This numerical integration only consists of a single step of the fourth-order Runga-Kutta method across the sampling time of each ACS step, which yields high performance and sufficient accuracy. Below are the steps for the EKF used for Alpha CubeSat's ACS. Notice the divergence from the linear Kalman Filter algorithm described above.

1) Initialize System State Estimate and System State Error Covariance (first measurement step)
2) Predict System State for the next Measurement step using the fourth-order Runga-Kutta numerical integration of the nonlinear state derivative expression provided the state and the sampling time. Predict the System State Error Covariance using the Jacobian of the system dynamics.
3) Compute the Kalman Gain based on Prediction and Measurement Uncertainty/Covariance
4) Use the Kalman Gain, State and Covariance Prediction, and State Measurement and Covariance to Estimate the System State and State Error Covariance for the following Measurement step
5) Repeat from step 2 and on for full control cycle duration.

In further detail, below are the system state and state time-derivative expressions.

$$x^T = [{}^B B^T \quad ({}^B\omega^{B/N})^T] \tag{34}$$

$$\overset{\text{B}}{x} = f(x,u) = \begin{bmatrix} -{}^B\omega^{B/N} \times {}^B B^T \\ I_b^{-1}({}^B\omega^{B/N} \times (I_b {}^B\omega^{B/N}) - {}^B\tau) \end{bmatrix} \tag{35}$$

The Jacobian for the magnetometer and gyroscope dynamics expressed in matrix form is below.

$$J = \begin{bmatrix} 0 & \omega_z & -\omega_y & 0 & -B_z & B_y \\ -\omega_z & 0 & \omega_x & B_z & 0 & -B_x \\ \omega_y & -\omega_x & 0 & -B_y & B_x & 0 \\ 0 & 0 & 0 & 0 & -\omega_z & -\omega_y \\ 0 & 0 & 0 & \omega_z & 0 & \omega_x \\ 0 & 0 & 0 & -\frac{\omega_x}{3} & -\frac{\omega_y}{3} & 0 \end{bmatrix} \tag{36}$$

Now that the expression for the state derivative is defined, the implementation of the fourth-order Runga-Kutta (RK4) integrates this nonlinear state derivative expression for every discrete step of the ACS control loop. This means the timestep used for integration is equivalent to the sampling time of the control loop. The results of this integration are used as the prediction in the EKF algorithm. Notice how this numerical integration technique only applies to the state prediction and not the state error covariance prediction. For the state covariance, the Jacobian matrix predicts each next step.

The RK4 method utilized is defined as the weighted summation below.

$$
\begin{aligned}
x(t) &= x_k \\
k_1 &= f(x(t) \cdot T_s \\
\frac{1}{2}k_1 &= \frac{1}{2} \cdot k_1 \\
k_2 &= f(x(t) + \frac{1}{2}k_1) \cdot T_s \\
\frac{1}{2}k_2 &= \frac{1}{2} \cdot k_2 \\
k_3 &= f(x(t) + \frac{1}{2}k_2) \cdot T_s \\
k_4 &= f(x(t) + k_3) \cdot T_s \\
x(t + T_s) &= x(t) + \frac{1}{6} \cdot T_s \cdot (k_1 + 2 \cdot k_2 + 2 \cdot k_3 + k_4) \\
x_{k+1} &= x(t + T_s)
\end{aligned}
\tag{37}
$$

Finally, now that all the elements of the EKF have been mathematically represented, the EKF algorithm can be summarized as follows.

$$k = 0 \tag{38}$$

$$x_k \quad P_k \tag{39}$$

$$
\begin{aligned}
x_P &= RK4(f(x), x_k, T_s) \\
P_P &= J_k P_k J_k^T + Q
\end{aligned}
\tag{40}
$$

$$K_{k+1} = P_p H^T (H P_P H^T + R)^{-1} \tag{41}$$

$$
\begin{aligned}
x_{k+1} &= x_P + K_{k+1}(z_{k+1} - H x_P) \\
P_{k+1} &= P_P - K H P_P
\end{aligned}
\tag{42}
$$

$$k = k + 1 \tag{43}$$

repeat steps described by (39-43) until control duration finishes (44)

This EKF design for the nonlinear first-order differential equation representing the sensor reading state combines the sensors' efforts in the state estimation algorithm, meaning inaccuracy/noise in one sensor can be compensated for by a

small covariance in the other sensor as well as correction from the mathematical model used in prediction. The Kalman gain weighs the uncertainty of the sensors against that of the model and factors this weight into the state estimation in the update step. With proper state estimation, the ACS experiences less noise in the state feedback, resulting in more robust control.

III. Analysis

For Alpha's mission design, among most other spacecraft, there is only one chance at achieving success. The Alpha CubeSat mission, albeit a low SWaP-C spacecraft, is still not inexpensive in terms of finance and labor. Accordingly, the behavior of each one of Alpha's subsystems must be closely scrutinized with thorough analysis. The objective of this analysis is to prove that the satellite will operate as close as possible to what the requirements dictate. Specifically for the ACS, this translates to evaluating the tuning and design of the EKF and each controller and ensuring high performance and stability despite the uncertainty in initial conditions, mass, the space environment, and the sensors' noise.

The general approach to ACS analysis includes many forms of simulation. Monte Carlo/Batch simulations account for all the aforementioned uncertainties. The main infrastructure for simulation that enables Monte Carlo and Edge-Case simulations is the personal computer simulation developed in C++. This simulation is denoted pcsim, and it is essentially the ACS flight software isolated from the rest of the flight software. A reliable and mutable C++ simulation encourages an efficient analysis process as C++ has an extremely high performance compared to alternative simulation software and object-oriented programming (OOP) languages. OOP provides structure to ACS software that promotes readable and adaptable control software. For instance, the short development timeline of the C++ EKF is due to the class definition of the filter, which modularizes all of its steps as class methods, making the EKF easy to integrate into the simulation and flight software as well as test.

The second major type of analysis for the ACS is Hardware-in-the-loop (HITL) analysis. HITL is necessary for ensuring the ACS hardware and software —the magnetorquers, IMU, gyroscope, and the microcontroller — both function as intended. The significant HITL testing for Alpha consists of temperature testing for sensor calibration, air-bearing tests to test the controllers, and general tests of the functionality of the gyroscope and magnetometer.

With simulation and HITL analysis, the confidence that the ACS performs as desired improves as the analysis uncovers many unexpected performance/operations. Identifying software and hardware issues leads to finding underlying root causes. Through an iterative process of root-cause analysis and subsequent fixes, Alpha's ACS is a rigorously studied system.

A. Monte Carlo Simulation Methods

This section provides an example of the Monte Carlo simulation setup for the Alpha CubeSat's ACS. In particular, this example simulates the performance of the Pointing Controller and provides the initial condition of a successful detumble and spin stabilization maneuver. The Monte Carlo for the Pointing Controller simulates the uncertainty in the controller gains, K_p and K_d. This is an effective method for tuning these gains as saving controller effort and state information for each Monte Carlo allows for data-driven evaluations proving which set of gains displays the best performance. Data-driven analysis for tuning the pointing controller based on a simulated magnetic field and three-dimensional attitude dynamics is the most viable method without prior flight heritage and/or experience. The Monte Carlo software referenced in this section can be seen in the Monte Carlo branch of the Alpha CubeSat ACS Simulator Github repository.

1. Simulation Parameters

The simulation includes the following parameters:

- **Number of Monte Carlo Iterations:** 1000
- **Percentage of Iterations used to determine the Average Pointing Error:** 20%
- **Run Time per Iteration:** 100 hours

2. Controller Parameters

The simulation randomly varies the proportional gain (K_p) and derivative gain (K_d) within specified ranges. The minimum and maximum values are set as follows:

- K_p range: 1×10^{-10} to 1×10^{-1}
- K_d range: 1×10^{-10} to 1×10^{-1}

3. Duty Cycle

The duty cycle for the simulation is set to 40 minutes with an active duty percentage of 25%.

4. Simulation Process

The simulation follows these steps for each Monte Carlo iteration:

1) Randomly generate K_p and K_d values within the specified ranges.
2) Initialize the plant(dynamics mode) and ACS objects with the given parameters.
3) Simulate the system for 100 hours with a time step of 0.2 seconds.
4) Calculate the average pointing error over the last 20% of the simulation time.
5) Record the K_p, K_d, and pointing error in an output file.

5. Output

The simulation outputs each Monte Carlo iteration's average pointing error, K_p, and K_d. The results are appended to the file `output/test.txt`. This text file saves data in CSV format. Python is used for parsing, processing, and visualizing the simulation data.

6. Generalization of Monte Carlo Methods

1) Randomly generate parameters of interest (high uncertainty) within specified ranges.
2) Initialize the plant and ACS objects with the given parameters.
3) Simulate the system for a set duration with a set time step.
4) Sample and statistically analyze some portion of the results.
5) Record the varied parameters and a statistical metric of interest.

B. Detumbling Controller Analysis

Monte Carlo simulations are used to analyze the performance of the Detumbling Controller. The tuning parameters I_d and c are the parameters of interest that are varied across each simulation. Monte Carlo yields a controller tuning that results in the detumbling/spin-stabilization being executed as intended.

The Detumbling Controller's success metric is the convergence of the satellite's attitude to near zero angular velocity components in two axes and a constant angular rate in the maximum principal axis of inertia. The concept of Superspin is employed to quantify the spin criteria for stability. Superspin for stability is the concept of actively spinning up one axis to achieve passive stability by artificially increasing the inertia ratio of the satellite until it reaches a margin of $\sigma > 1.2$.

$$\sigma = \frac{I_s}{I_t} > 1.2 \tag{45}$$

Where the effective spin-axis inertia is described as:

$$I_{s_{eff}} = I_s + \frac{h_s}{\Omega} \tag{46}$$

$$\sigma_{eff} = \frac{I_{s_{eff}}}{I_t} \tag{47}$$

Note that I_t is the inertia of the transverse axis.

Given that the inertia of the spin axis is very similar to the transverse, spinning the maximum principal inertia axis up to one radian is sufficient to meet the margin for the inertia ratio.

The success criteria of the Detumbling Controller is thus refined to:

$$\omega_x \approx \omega_y \approx 0 \tag{48}$$

$$\omega_z \approx 1 \ \ rad/s \tag{49}$$

C. Pointing Controller Analysis

Once again, Monte Carlo simulations are used to analyze the Pointing Controller's performance. The tuning parameters I_d and c are the parameters of interest that are varied across each simulation. Monte Carlo yields a controller tuning, resulting in the pointing maneuver working as expected.

Again, the Pointing Controller's success metric is the convergence of the satellite's spin axis being aligned with the Earth's magnetic field vector. The Earth's magnetic field points North to South, and Alpha's orbit in LEO has ISS inclination (51°). This means that quantifying the success metric of the PD controller is more of an error tolerance as the magnetic field vector oscillates at a significant frequency in Alpha's body frame throughout the satellite's orbital period. Due to PD control, significant steady-state error is expected. Additionally, the pointing requirement is less high priority compared to spin-stabilization. This is significant as the success criteria are relaxed. The success of the pointing controller is thus defined as having a steady state error of lower than 40 degrees.

$$e_{ss} < 40^\circ \tag{50}$$

The results section further discusses the 40° tolerance.

D. ACS Control Mode Analysis

Validating Duty-Cycle consists of the majority of the Control Mode analysis. The Duty-Cycle is the portion of the satellite's power and computer dedicated to a particular subsystem/purpose. The ACS shall be able to meet its requirements with the provided duty cycle to meet EPS requirements. The duty cycle has been incorporated into the Monte Carlo simulations in all of the analyses above.

Analysis of potential control modes excluded from the flight software includes the edge-case simulations (singular predetermined flight conditions) of the "simple" mode. Simple mode is an ACS mode designed to output a constant current to one of the magnetorquers to replicate the effect of a small permanent magnet. This is an attempt at more passive spin stabilization. This mode was proposed as a backup if detumbling fails. Passive attitude control using permanent magnets is a retro ACS design concept used by the Australis-OSCAR 5 satellite launched in 1970. More attention is given to simple mode in the results section.

E. Extended Kalman Filter Analysis

The two main types of analysis for the EKF include the filter's performance smoothing the noise from the gyroscope and the magnetometer and the computational performance of the EKF software.

Analyzing filter smoothing involves collecting noisy IMU and gyroscope measurements and feeding that data through the filter. Success is determined by significant noise reduction. This makes analysis as simple as comparing the filter's output to the raw data and observing noise reduction and the filter output's similarity to a pre-established signal truth.

The control cycle for ACS must be relatively high frequency, meaning a short duration. The EKF shall meet an acceptable duration for the control cycle such that it is not a limiting factor for overall ACS and flight computer operation and performance. Simulation of the EKF is another form of analysis. Simulated noisy data can be filtered by the EKF just as raw collected data can be. However, EKF performance on real sensor noise provides a more reliable analysis as noise distribution is heavily sensor-dependent. For instance, ACS integrates into the flight software easily with a control cycle of around 100 milliseconds or less.

F. Miscellaneous Hardware Analysis

HITL analysis is used for the magnetorquers, IMU, gyroscope, and microcontroller. The analysis consists of temperature, magnetorquer, and pointing algorithm testing. Temperature testing is necessary for proper sensor calibration and, in particular, finding hard-iron offsets.

IV. Results

Results constitute visual representations of data collected via simulation and testing from the analysis described in the prior section. The controller tuning visualizations will consist of many data points that are either green or grey and one that is blue. The green represents tuning that meets convergence criteria, the blue represents the default tuning chosen for the flight software, and the grey represents failed tuning pairs.

A. Detumbling Controller Tuning

As a result of the dynamics simulation, the nominal performance of the Detumbling Controller is visualized in Figure 1.

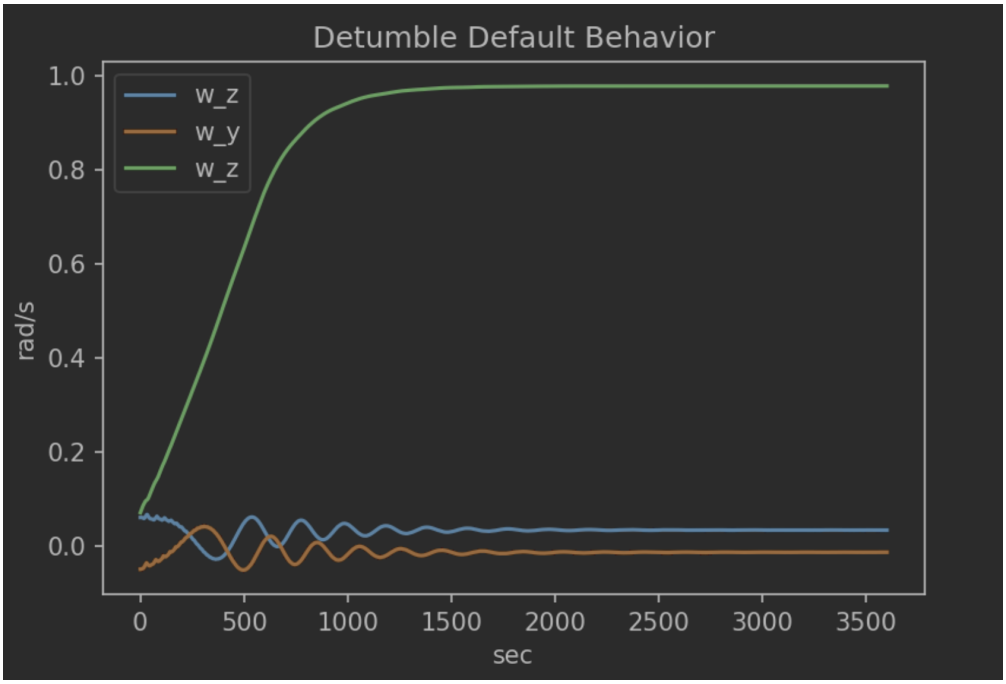

Fig. 1 Angular Velocity vs. time for nominal detumble maneuver

The Monte Carlo results for tuning parameters c and I_d for the Kane Damper algorithm yield the solution space with a significant safety margin shown in Figure 2.

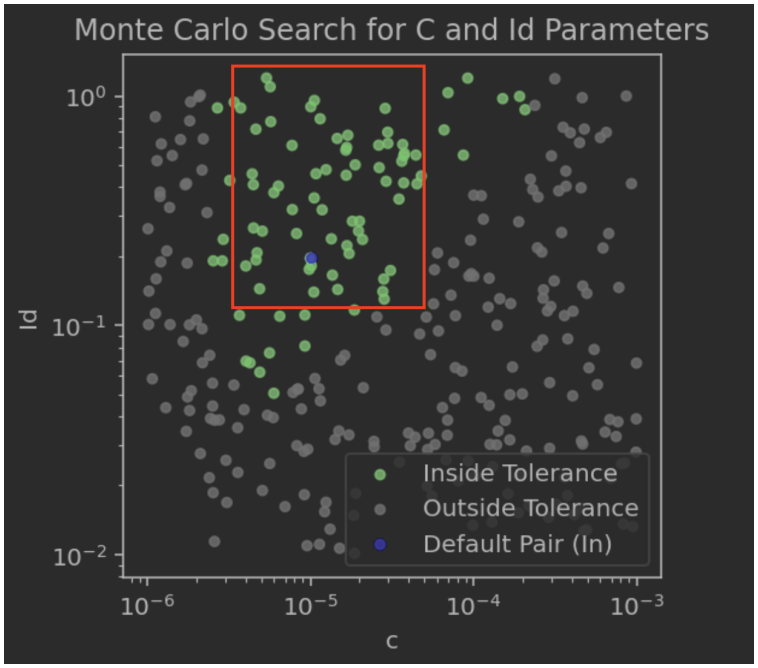

Fig. 2 Solution Space for Detumbling Controller Tuning

In Figure 2, the solution space chosen for the range of tuning hard-coded into the flight software is boxed in red. The tolerance for this plot is the final angular velocity of the spin axis being within 10 percent of the desired 1 radian per second.

Lastly, in Figure 3, the control effort, represented as the current supplied to the magnetorquers, is displayed. Note that the signals saturate in the beginning and then settle. This is expected as the magnetorquers do not produce a strong moment even at maximum current throughput of 250 mA.

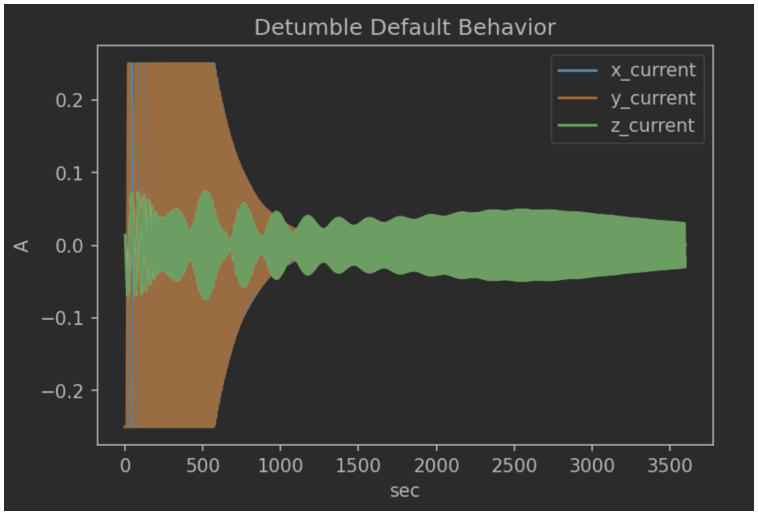

Fig. 3 Magnetorquer Current vs. time for nominal detumble maneuver

This nominal detumble maneuver behavior is consistent for initial conditions with a low angular rate below 10 degrees per second. Given that deployment from the ISS guarantees a minimal initial tumble condition (1-5 degrees per second), these results are sufficient for the maneuver's success.

B. Pointing Controller Tuning

As a result of the simulation, the nominal performance of the Pointing controller is as visualized in Figure 4.

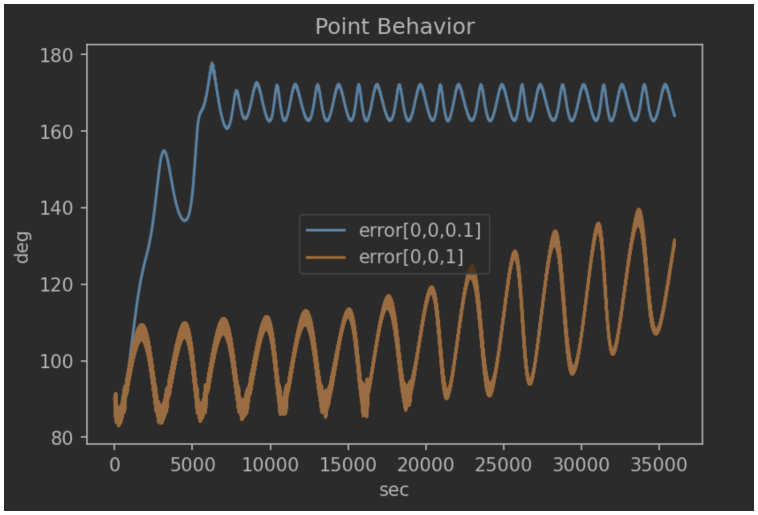

Fig. 4 Pointing Error (degrees) vs. time for nominal pointing maneuver

The key in Figure 4 assigns which initial angular velocities are simulated. The smaller the angular velocity, the quicker the Pointing Controller converges. The trade-off is slow convergence for stability. For the Alpha's mission design, spin-stabilization is the priority, so spin-stabilization at a high angular rate occurs before pointing despite the impact on the Pointing Controller's performance.

Figures 5 and 6 show the solution spaces of K_p and K_d tuning for different convergence/success criteria and ACS duty cycles. Note that all these plots result from Monte Carlo simulations where each data point is a single simulation of 100 hours in duration.

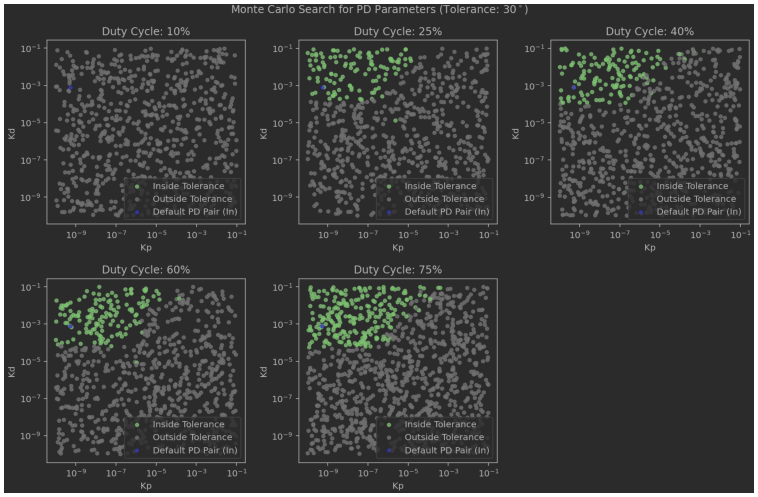

Fig. 5 PD tuning solution space with 30-degree tolerance

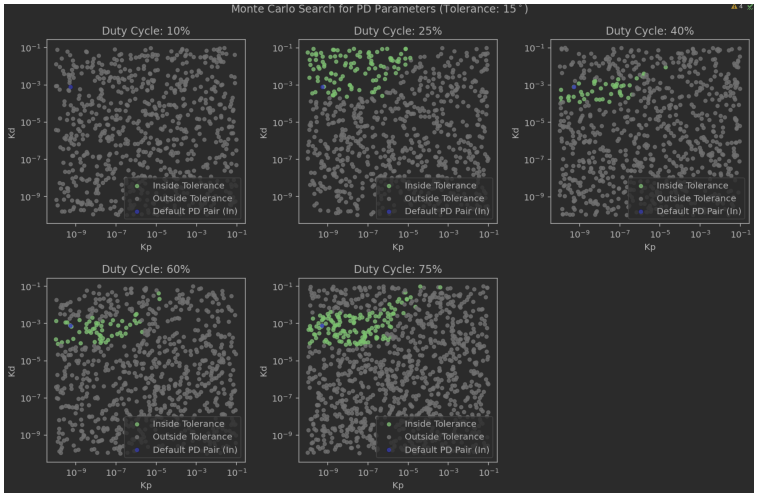

Fig. 6 PD tuning solution space with 15-degree tolerance

As the tolerance tightens, the solution space shrinks. Since the tolerance set for success is around 40 degrees, there should be plenty of margin for success with the default tuning.

C. Control Mode Investigation Conclusion

Simple mode is a candidate control mode mentioned in the analysis section. Ultimately, it is proven to introduce no benefit to the ACS. Figure 7 shows that a magnetorquer supplied with a constant maximum current of 250mA cannot establish passive stability for the satellite.

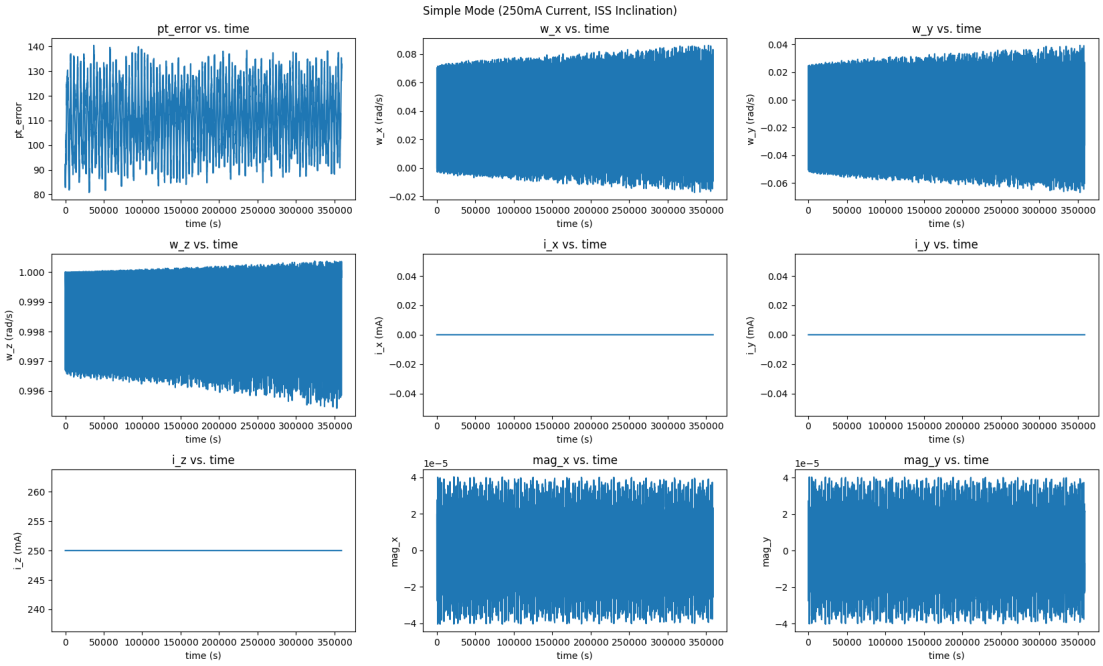

Fig. 7 Simple Mode Simulation Results

In Figure 7, the title denotes the current supplied, the inclination of the satellite's orbit, and the simulation duration. The most important thing to acknowledge is the pointing error, which is an extremely noisy signal. This indicates that a single one of Alpha's magnetorquers supplied a constant current cannot pseudo-passively align the CubeSat's attitude with the Earth's magnetic field. Simple mode shall not be used in the ACS flight software.

D. Extended Kalman Filter Performance

EKF performance is characterized by filter quality and computational efficiency. Figure 8 demonstrates the EKF's performance on IMU/magnetometer noise.

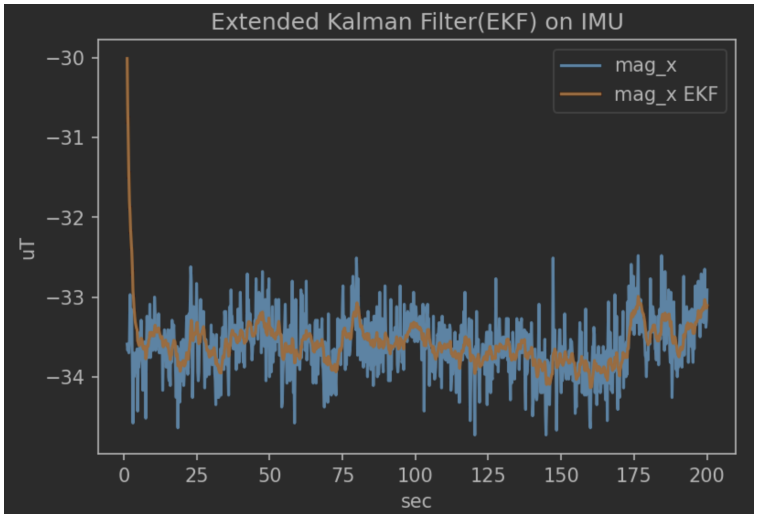

Fig. 8 EKF applied to IMU noise

From Figure 8, it is clear that the EKF nicely smoothes the sensor measurement signal. It is robust to significant spikes in measurement noise and is thus reliable for ensuring smooth controller performance.

Regarding computational speed/efficiency, performance in simulation is visualized in Figure 9. This figure is for the final version of the EKF that performs one step of RK4 and has no matrix exponential operation. RK4 discretizes the dynamics, so there is no use case for matrix exponential in the EKF algorithm.

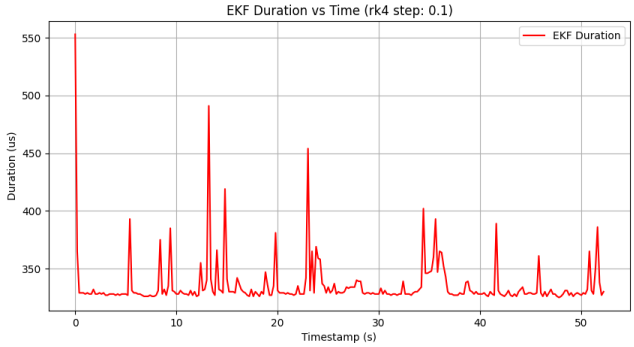

Fig. 9 Duration of EKF step in microseconds vs EKF step number

The duration is in microseconds, meaning the EKF is well below its allocated control cycle duration by many orders of magnitude. Note that these timing results are from simulation on a personal computer.

The final validation of the EKF is the time performance on the actual microcontroller. Since microcontrollers are less powerful than personal computers by many orders of magnitude, there is an equivalent increase in EKF duration. Measured from the teensy microcontroller, one step of the EKF lasts around 15 milliseconds. For context, this outperforms Simulink's automatically generated EKF code by approximately 100 times.

E. Miscellaneous Hardware Performance

One issue HITL addresses is the correction of the gyroscope's sign convention for angular velocity readings. Alpha's gyroscope does not read like a typical right-hand basis. This correction involves flipping the sign of the angular x and y components of angular velocity in the main ACS control loop. The performance of the ACS remains unaffected by this change.

System-level HITL results include successful pointing in a two-dimensional scenario. From testing the controller and a single magnetorquer in an air-bearing apparatus, the magnetorquer slewed the breadboard setup 180 degrees. Figure 10 shows the following behavior.

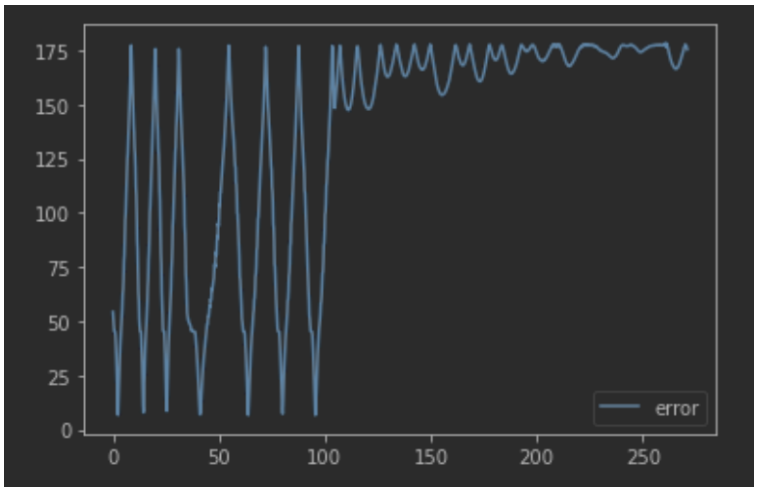

Fig. 10 Airbearing Slew Test behavior

Note that the y-axis of Figure 10 is degrees, and the x-axis is seconds.

Additionally, Figure 11 shows the complementary PWM and current signals.

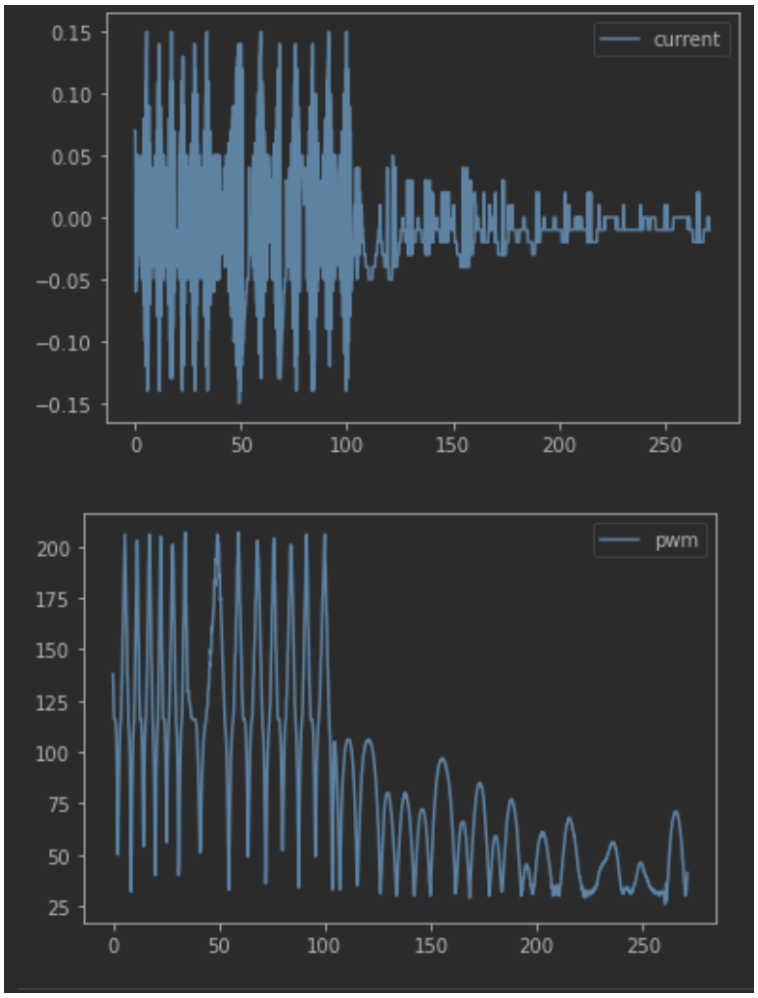

Fig. 11 Air-bearing test current and PWM signals

V. Conclusion

There is sufficient evidence that Alpha CubeSat's Attitude Control subsystem shall meet its design requirements. Through simulation, the performance of the controllers and state estimation have proven robust for the mission design. The integration of the ACS software and hardware is established and reliable per the HITL testing results. The ACS software is in a finalized state and is ready for integration with the overall flight software.

Acknowledgments

Joshua Umansky-Castro, Davide Carabellese, Deemo Chen (responsible for the majority of the fantastic plots), Lucas De Venecia, and Soumaryup Lahiri have contributed immensely to developing and integrating Alpha CubeSat's Attitude Control System.

References

[1] Carabellese, D., Umansky-Castro, J., and Peck, M., "Magnetorquer-only Nonlinear Attitude Control for CubeSats," *AIAA SciTech 2021 Forum*, Vol. 1, No. 1, 2021, pp. 11–15 19–21.