Closed Loop Analysis of Space Systems (CLASS)

Marc Akiki, Michael Lembeck
University of Illinois at Urbana-Champaign
Aerospace Engineering Department

34th Annual Small Satellite Conference
August 2020

Outline

1. Problem Statement and Research Motivation
2. CLASS Specifications and Design
3. Simulation Algorithms
4. Closed-Loop Test on CAPSat
5. Conclusion and Future Work

Section 1:
Problem Statement and Research Motivation

Low Mission Success Rate

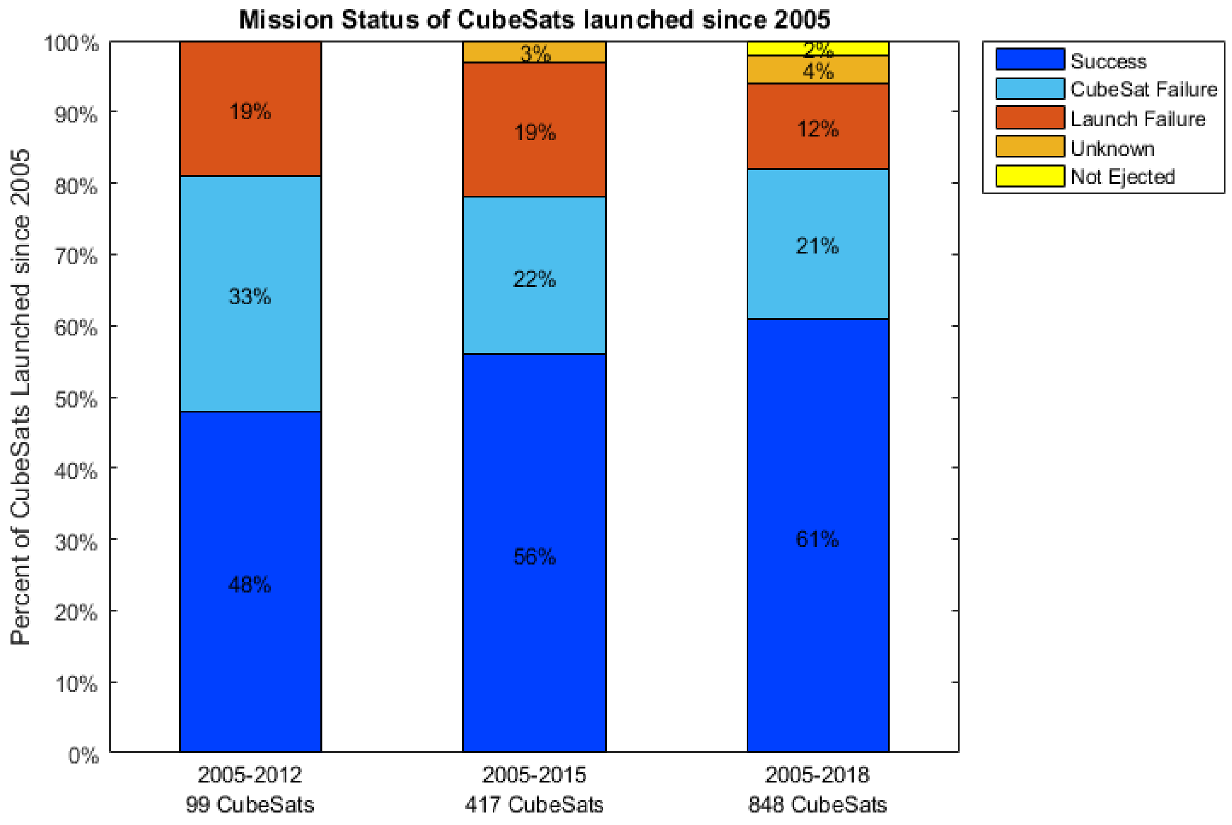


Figure 1: Mission status of CubeSats launched since the year 2005. Data extracted from T. Villela, C. A. Costa, A. M. Brandao, F. T. Bueno and R. Leonardi, "Towards the Thousandth CubeSat: A Statistical Overview".

- As of 2018, the international CubeSat success rate remains at 61%
- From 2015 to 2018, 431 CubeSats were launched
  - In a period of 3 years only, the number of CubeSats doubled
  - However, the success rate slightly improved from 56% to 61%

Low Mission Success Rate

Academic CubeSat Missions (2017)

Commercial CubeSat Missions (2017)

Figure 2: Mission status of academic CubeSats (left) and commercial CubeSats (right). Data extracted from C. C. Venturini, M. Tolmasoff and R. Delos Santos, "Improving Mission Success of CubeSats," U.S. SPACE PROGRAM MISSION ASSURANCE IMPROVEMENT WORKSHOP, El Segundo, 2017.

Source of Failure

According to M. Langer and J. Bouwmeester:

- Top 3 identifiable systems that experience a mission critical failure right after deployment:
  - Electrical Power System
  - Onboard Computer
  - Communication System

- However, 33% of CubeSat developers have reported that they have no means of identifying the subsystem that caused the mission failure

Data from M. Langer and J. Bouwmeester, "Reliability of CubeSats - Statistical Data, Developers' Beliefs and the Way Forward," in AIAA/USU Conference on Small Satellites, Logan, 2016.

Importance of System Testing

- Published by C. C. Venturini et. al., small satellite developers were particularly concerned about:
  - Lack of time and technology to perform testing
  - Interface between the payload and bus subsystems
  - Reliability of their testing procedures

- While some mission failures are unavoidable, the low testing reliability and rate during CubeSat development demonstrates that there are many failures that could have been avoidable

Data from C. C. Venturini, M. Tolmasoff and R. Delos Santos, "Improving Mission Success of CubeSats," U.S. SPACE PROGRAM MISSION ASSURANCE IMPROVEMENT WORKSHOP, El Segundo, 2017.

Section 2:
CLASS Specifications and Design

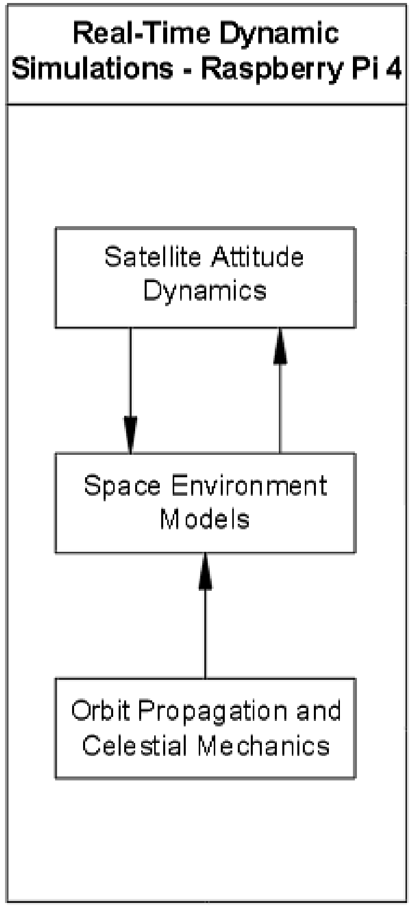
LASSI Project: CLASS

- Laboratory of Advanced Space Systems at Illinois (LASSI): we are a team of engineering students at the University of Illinois Urbana Champaign designing and developing multiple CubeSat missions
- CLASS: the project to building a modular closed-loop test system to increase the reliability and accuracy of space system testing and improve mission success rates

Test System Requirements

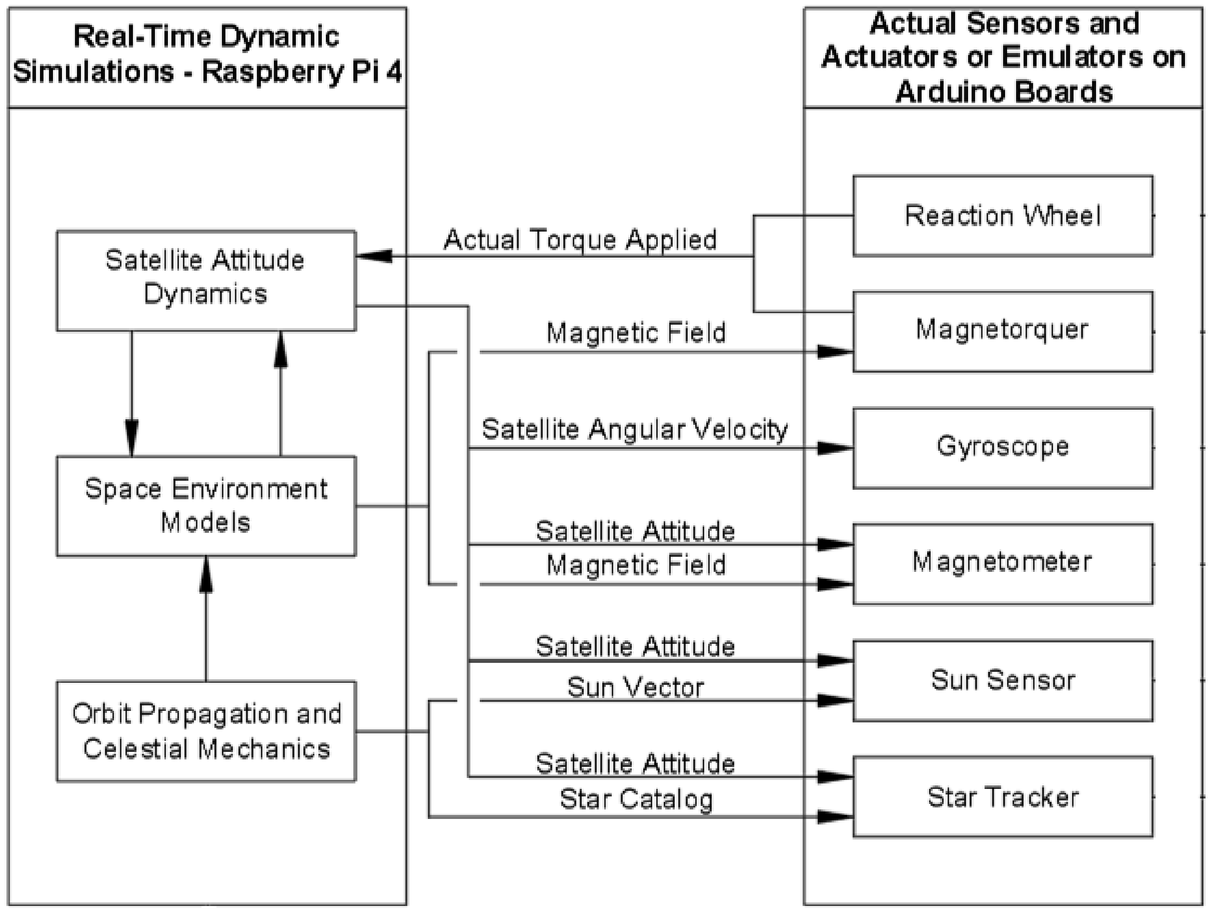
1. Provide accurate real-time satellite attitude and orbital mechanics simulations to enable hardware-in-the-loop testing
2. Be easily configurable by the user for different mission operations and satellite designs
3. Rely on widely used hardware, well documented processors, and high level programming languages for compatibility and interfacing
4. Mitigate numerical inaccuracies and asynchronous execution delays
5. Allow the implementation of real-time graphics and visualization

CLASS Design and Architecture



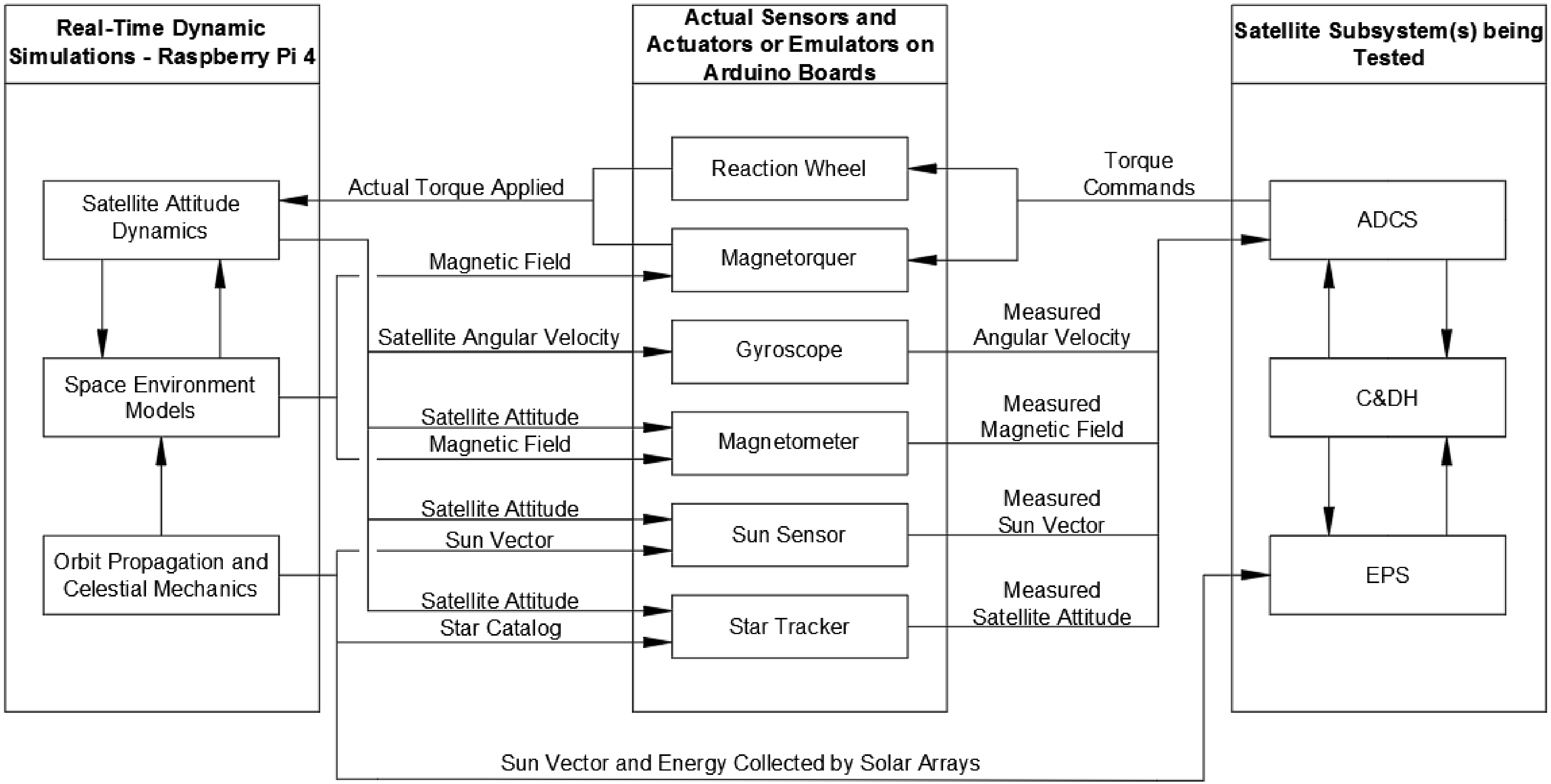
- Real time dynamics and space environment simulations
- Processor: Raspberry Pi 4
- PREEMPT_RT: Linux patch for emulating an RTOS
- Satellite attitude dynamics, orbit propagator, magnetic field, aerodynamics etc. all programmed in C++ for timely execution

CLASS Design and Architecture



- Dynamic simulation sends data about satellite attitude and orbital position to actual ADCS sensors or emulators
- Sensor emulators receive what the actual state of the satellite is and transform it to raw sensor data
- To implement actual sensors, tools such as Helmholtz cage, air bearing tables etc. are required. CLASS does interface with these tools

CLASS Design and Architecture



Section 3:
Simulation Algorithms

Attitude Simulation Algorithms

- Rigid body dynamics equation and quaternion kinematics:
  - Dynamics:

$$\dot{\boldsymbol{\omega}}^{\mathbf{b}} = {\mathbf{J}^{\mathbf{b}}}^{-1}\left[\mathbf{T}^{\mathbf{b}}_{\mathbf{ext}} - \boldsymbol{\omega}^{\mathbf{b}} \times \left(\mathbf{J}^{\mathbf{b}}\boldsymbol{\omega}^{\mathbf{b}}\right)\right]$$

  - Kinematics:

$$\dot{\mathbf{q}}^{\mathbf{I}}_{\mathbf{b}} = \frac{1}{2}\mathbf{E}^{\mathbf{b}}\left(\mathbf{q}^{\mathbf{I}}_{\mathbf{b}}\right)\boldsymbol{\omega}^{\mathbf{b}}$$

- Integrator is Runge-Kutta 4

Attitude Simulation Results

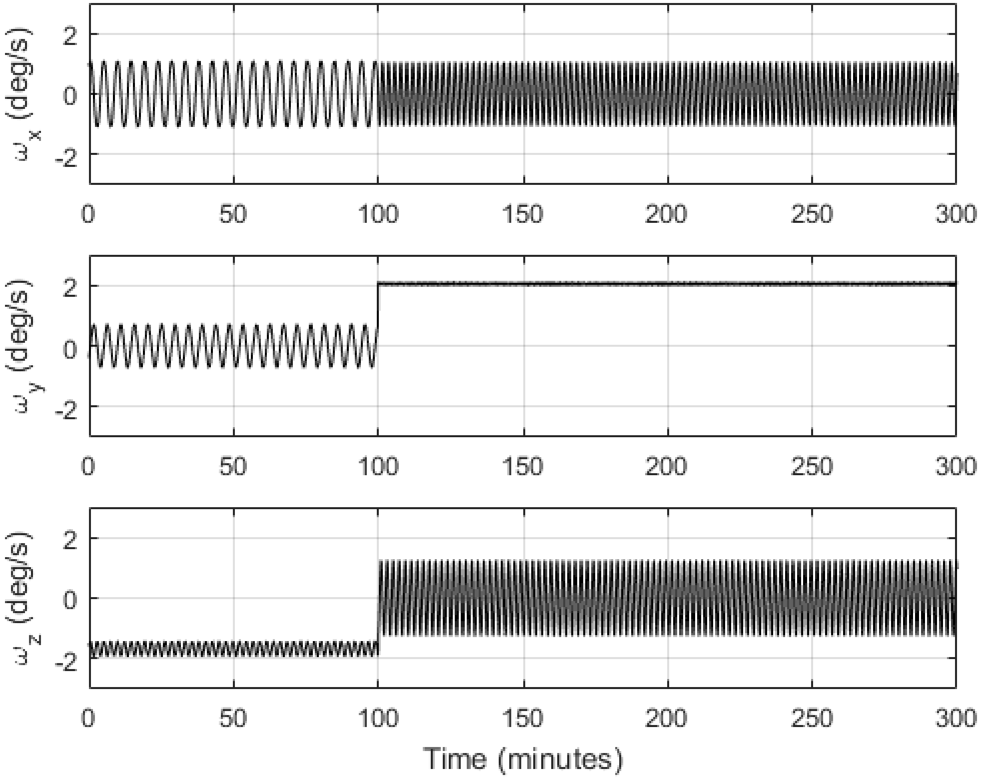


Figure 4: Angular velocity of the satellite expressed in the body frame (deg/s) vs time (minutes)

- Initial Conditions:
  - $\mathbf{q}_{\mathbf{I}}^{\mathbf{b}} = [0.836 \quad -0.029 \quad -0.52 \quad -0.173]^{\mathrm{T}}$
  - $\boldsymbol{\omega}^{\mathbf{b}} = [0.9 \quad -0.4 \quad -1.6]^{\mathrm{T}}$ deg/s
  - $\mathbf{T}^{\mathbf{b}} = [0 \quad 0 \quad 0]^{\mathrm{T}}$ N.m
- Impulse external torque applied at time $t = 100$ min
  - $\mathbf{T}^{\mathbf{I}} = [0.02 \quad -0.001 \quad 0]^{\mathrm{T}}$ N.m
- Expected behavior expressed in the inertial frame:
  - Angular momentum along x-axis: increase
  - Angular momentum along y-axis: decrease
  - Angular momentum along z-axis: no change

Attitude Simulation Results

- Expected behavior can be identified in the angular momentum vs time

Angular Momentum Vector Component	**Value before perturbation ($kg.m^2/s$)**	**Value after perturbation ($kg.m^2/s$)**
H_x	0.0004291	0.001429
H_y	-0.00009657	-0.00001459
H_z	0.00034	0.00034

Table 2: Angular momentum of the satellite expressed in the inertial reference frame before and after the impulse torque was applied

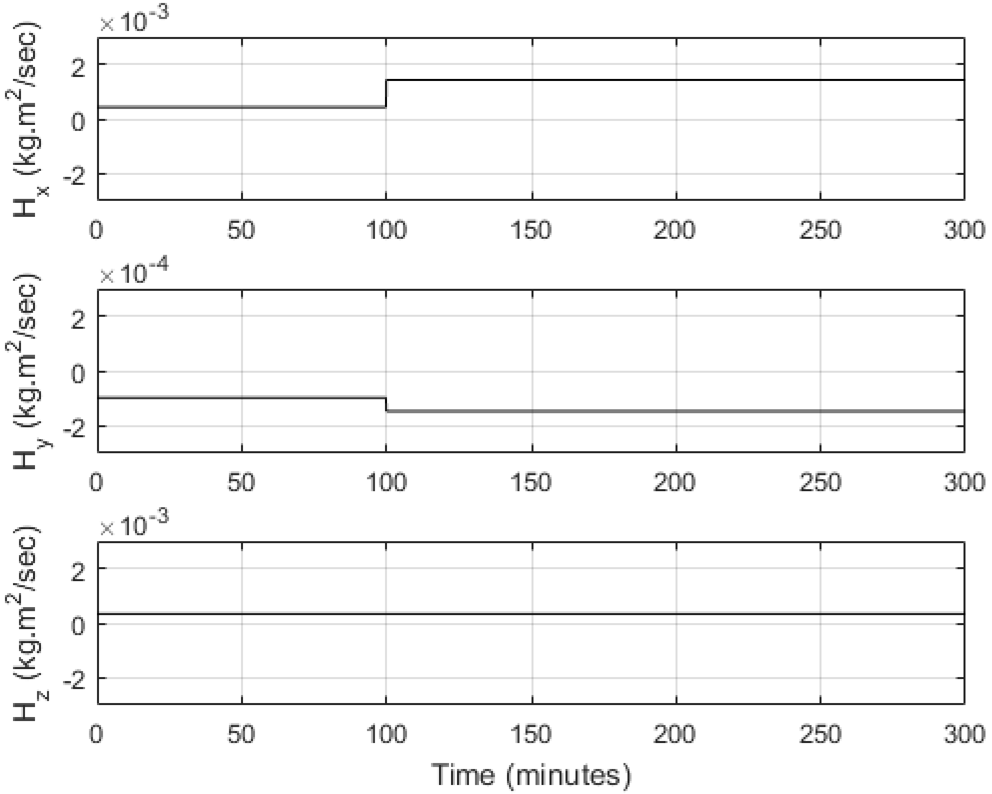


Figure 5: Angular momentum of the satellite expressed in the inertial frame ($kg.m^2/sec$) vs time (minutes)

Orbit Propagator

- SGP4: Simplified General Perturbation analytical model
- Widely used for LEO satellites
- Simplified model that runs efficiently and accurately

- Limitation exits for long propagation times: loss in accuracy

Orbit Simulation Results

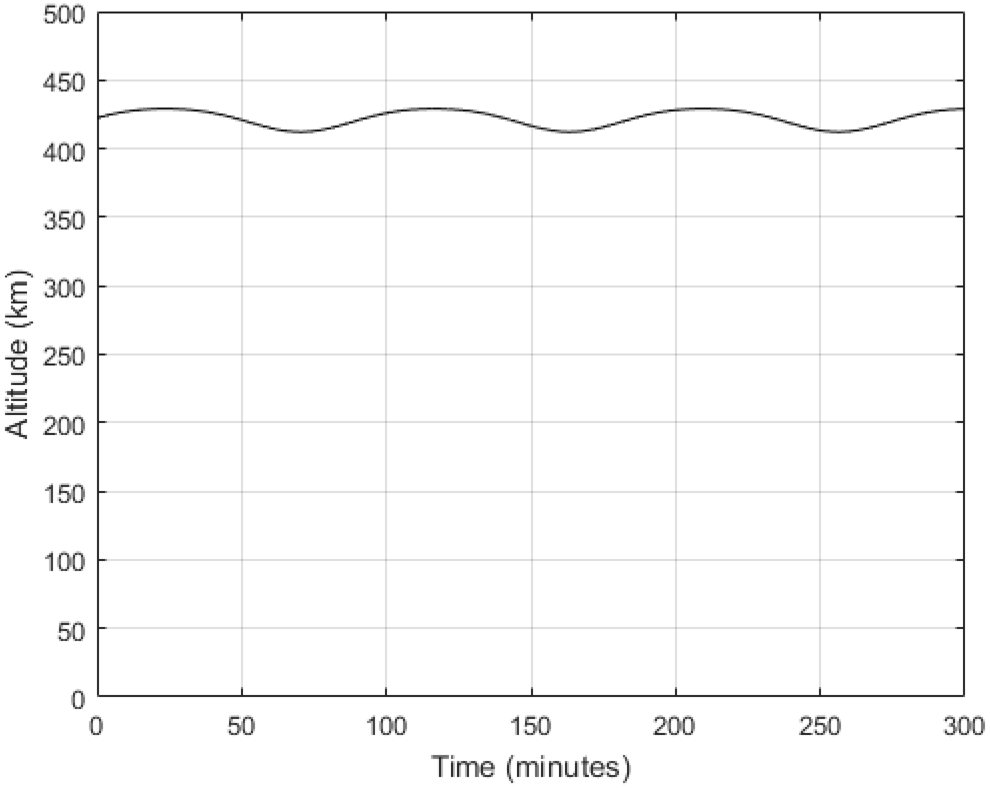


Figure 6: ISS altitude (km) vs time (minutes) simulated in CLASS

- To test the performance of the orbit simulation, a TLE file for the ISS was uploaded to CLASS on April 2020
- The ISS was orbiting at an apogee of 423 km and a perigee of 418 km according to NASA's online ISS tracker
- CLASS results indicate an apogee of 429 km and perigee of 412 km
- The numerical errors of 1.42% for apogee and 1.44% for perigee are acceptable

Orbit Simulation Results

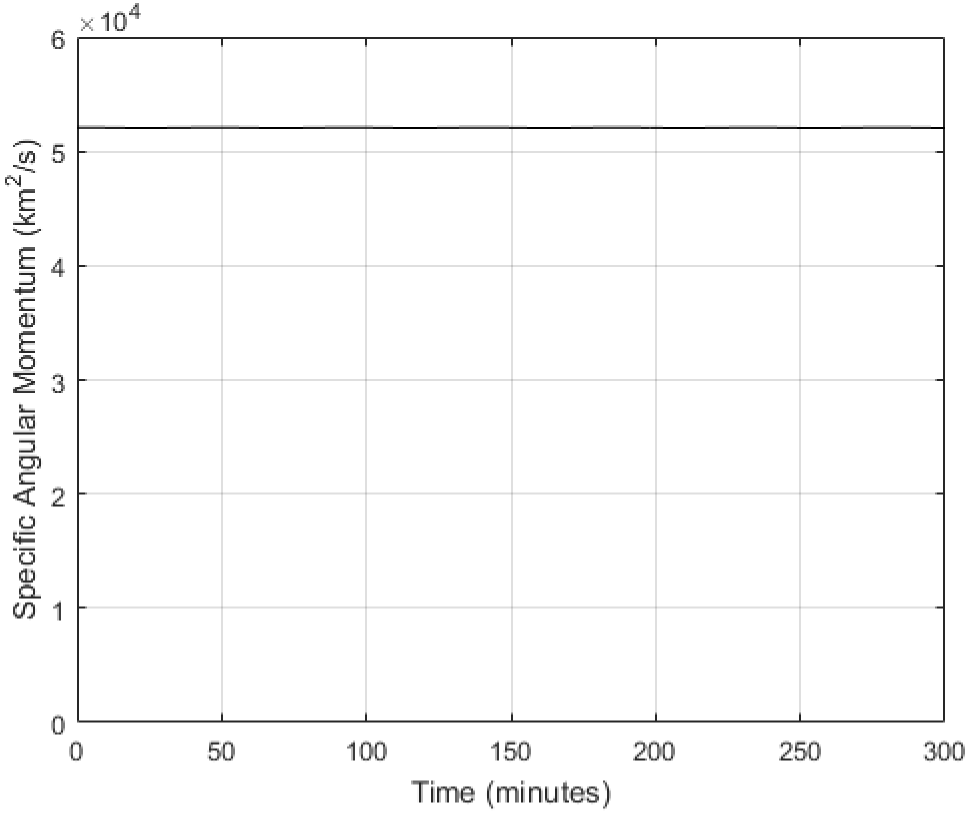


Figure 7: ISS simulated specific angular momentum (km^2/s) vs time (minutes)

- The simulated orbit period of 92.83 minutes is in accordance with the expected orbital period
- Further validation was performed by relying on the fact that the specific angular momentum of the satellite for only 3 orbits will negligibly change and can be considered conserved:

	Expected Value	**Simulated Value** t<90 min	**Simulated Value** 90 min < t < 180 min	**Simulated Value** 180 min < t < 270 min
Specific Angular Momentum (km^2/sec)	52033	52070	52070	52060

Table 3: Specific angular momentum of the satellite orbit expected value and the simulated values per orbit period

Magnetic Field Model

- International Geomagnetic Reference Field: IGRF 13

$$\mathbf{B} = -\nabla V$$

$$V(r, \theta, \phi, t) = a \sum_{n=1}^{N} \sum_{m=0}^{n} \left(\frac{a}{r}\right)^{n+1} [g_n^m(t) \cos(m\phi) + h_m^n(t) \sin(m\phi)] P_m^n \cos \lambda'$$

- Algorithm was coded into C++
- Magnetic field vector in the satellite body frame is calculated every time the attitude of the satellite changes

Environmental Disturbances: Aerodynamics

- Aerodynamic torque:

$$\mathbf{F}^{\mathrm{b}}_{\mathbf{areo\ i}} = -\frac{1}{2}\rho C_D v^{\mathrm{b}}_{\mathrm{rel}} \mathbf{v}^{\mathbf{b}}_{\mathbf{rel}} S_i \max(\cos\theta_i\,, 0)$$

$$\mathbf{T}^{\mathrm{b}}_{\mathbf{aero}} = \sum_{i=1}^{N} \mathbf{r}^{\mathbf{b}}_{\mathbf{cm_{cp}i}} \times \mathbf{F}^{\mathbf{b}}_{\mathbf{aero\ i}}$$

- User can select the drag coefficient, change geometrical properties of the CubeSat, and select an atmospheric air density model (future addition)

Environmental Disturbances: Gravity Gradient

- Gravity gradient torque:

$$\mathbf{T}_{\mathbf{gg}}^{\mathbf{b}} = \frac{3\mu_{\mathrm{Earth}}}{r^3}\left[\hat{\mathbf{n}}^{\mathbf{b}} \times \left(J^b \hat{\mathbf{n}}^{\mathbf{b}}\right)\right]$$

- Moment of inertia and other geometrical properties are initially set by the user

Environmental Disturbances: Solar Radiation Pressure

- Solar radiation pressure disturbance torque:

$$\mathbf{F}^{\mathbf{b}}_{\mathbf{SRP\ i}} = \mathrm{P_{SRP}} \mathrm{S_i} \left[2 \left(\frac{\mathrm{R_{diff\ i}}}{3} + \mathrm{R_{spec\ i}} \cos \theta_{\mathrm{SRP\ i}} \right) \mathbf{n}^{\mathbf{b}}_{\mathbf{i}} + \left(1 - \mathrm{R_{spec\ i}} \right) \mathbf{s}^{\mathbf{b}} \right] \max(\cos \theta_{\mathrm{SRP\ i}}, 0)$$

$$\mathrm{P_{SRP}} = \frac{\mathcal{F}}{\mathrm{c}\,(\mathrm{r_{sat\ to\ sun}})^{2}}$$

$$\mathbf{T}^{\mathbf{b}}_{\mathbf{SRP}} = \sum_{\mathbf{i=1}}^{\mathbf{N}} \mathbf{r}^{\mathbf{b}}_{\mathbf{cm_{csrp}i}} \times \mathbf{F}^{\mathbf{b}}_{\mathbf{SRP\ i}}$$

- Programmed into CLASS but not yet fully tested

Section 4:
Closed-Loop Test on CAPSat

Closed-Loop Tests: CAPSat

- CLASS was used to test the ADCS of CAPSat, a LASSI satellite
- CAPSat is a 3U satellite with three payloads:
  - Radiator to test a novel active cooling technique
  - Quantum annealing experiment
  - Fine pointing attitude control technique by creating controlled solar panel deformations
- The control requirements are: de-tumble and then maintain Earth pointing

Test Setup

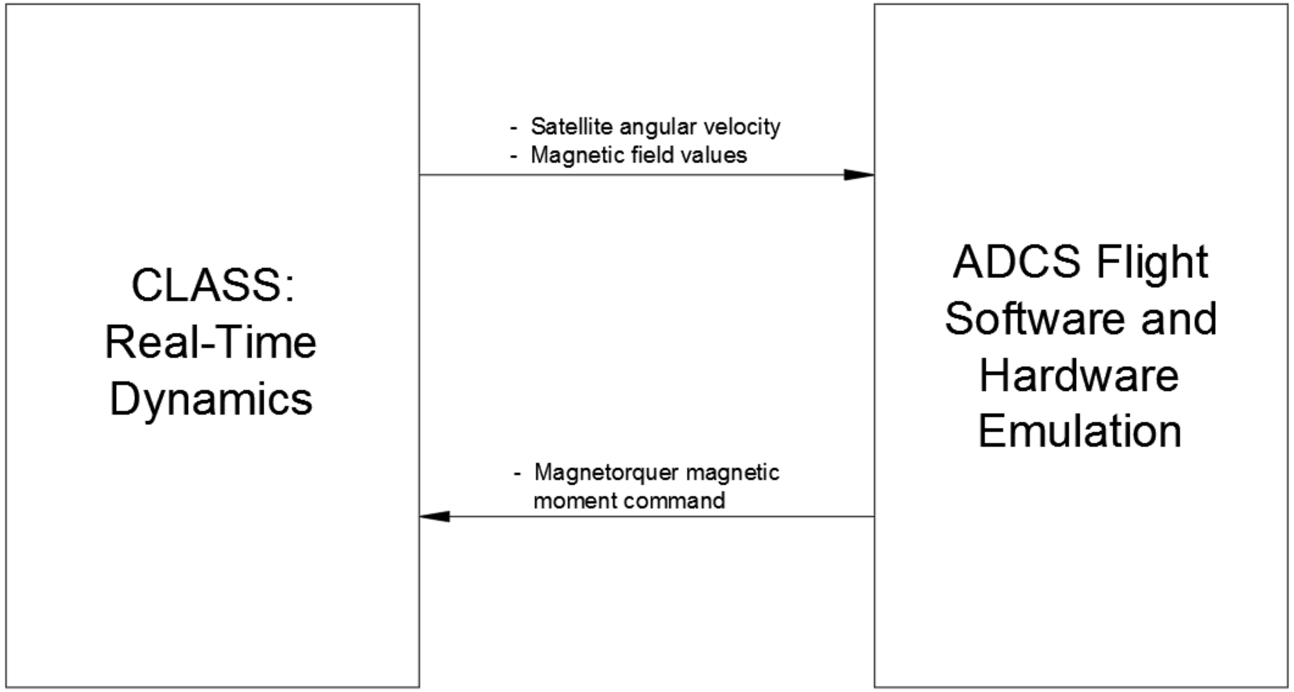


Figure 8: Data interface diagram between the ADCS flight computer emulation and CLASS dynamic simulation.

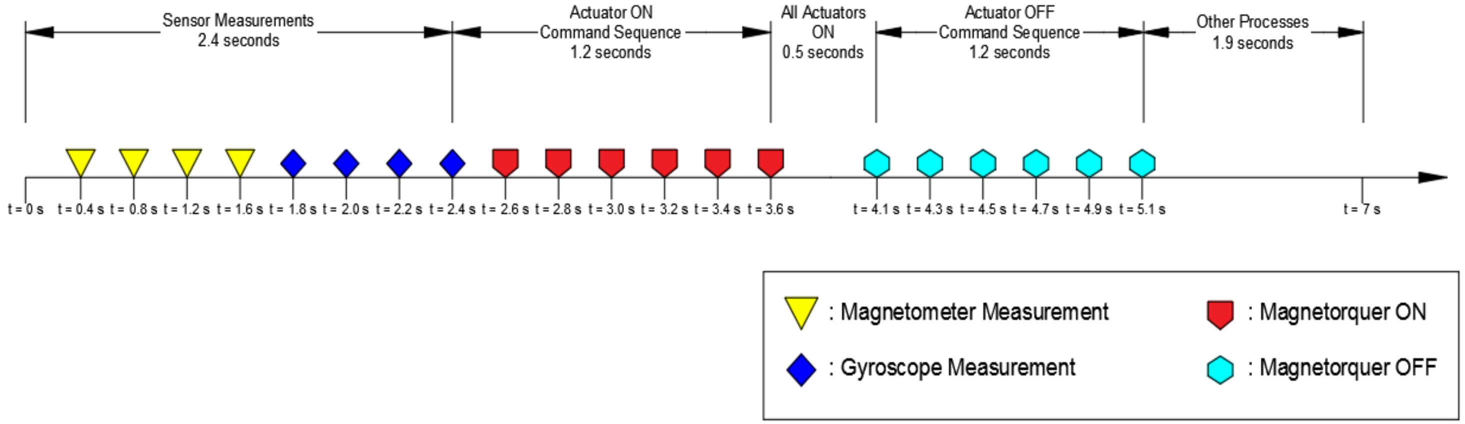
- A program that emulates CAPSat's ADCS flight computer was built
- The flight computer emulation communicates with the CLASS Real-Time dynamics to collect sensor data and send actuator commands
- Hardware delays and constraints were carefully modeled in the flight computer emulation

CAPSat ADCS Flight Computer Model

- Hardware delays play a critical role in the performance of the ADCS

	Sensors	Actuators activation and deactivation	Other tasks	**Total Control Cycle**
Execution Time (s)	2.4	2.4	1.9	**7.2**

	Execution Time (s)	**Quantity**
Magnetomter	0.4	4
Gyroscope	0.2	4
Magnetorquer ON	0.2	6
Magnetorquer OFF	0.2	6



Phase 1: De-tumbling

- B-dot control law:

$$\mathbf{u^b} = -\mathrm{K}\frac{\mathrm{d}\mathbf{B^b}}{\mathrm{dt}}$$

Where:

- $\mathbf{u^b}$ = control input expressed in the body coordinate frame
- K = proportional gain
- $\frac{\mathrm{d}\mathbf{B^b}}{\mathrm{dt}}$ = time derivative of the magnetic field vector expressed in the satellite body frame

- Initial conditions for B-dot test:
  - $\boldsymbol{\omega}^{\mathbf{b}} = [5.9 \quad 4.2 \quad -6.3]^{\mathrm{T}}$ deg/s
  - Angular velocity of 9.6 deg/s
  - Random attitude

Phase 1: De-tumbling Results

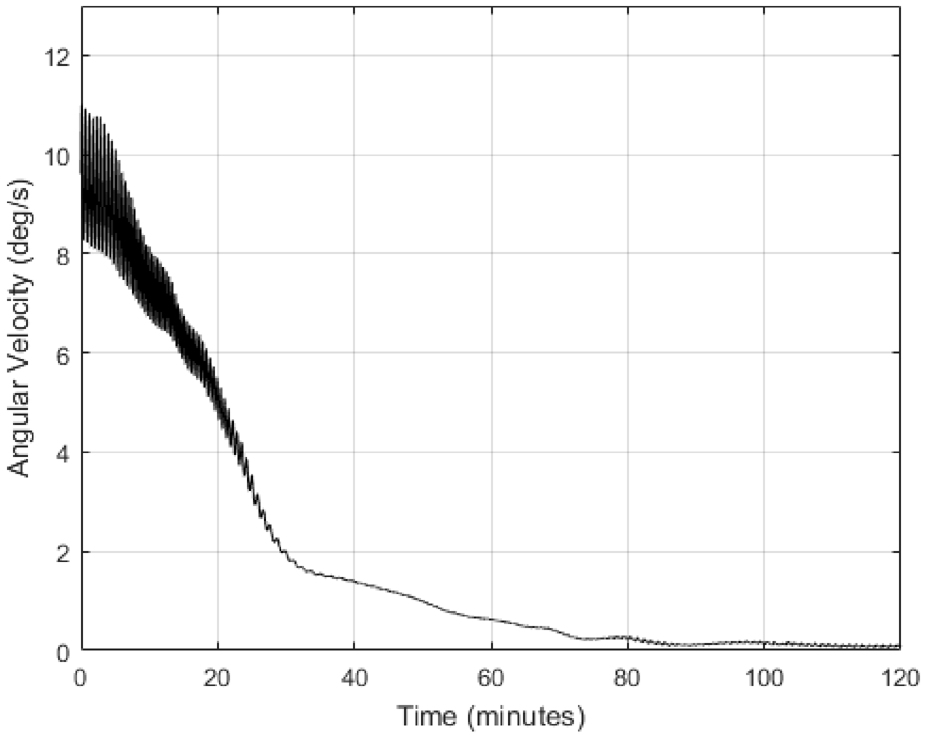


Figure 9: De-tumbling results for CAPSat. Satellite angular velocity (deg/s) vs time (minutes). Angular velocity reaches values near zero in less than one orbit.

- From 9.6 deg/s to 0.065 deg/s in 117.2 minutes
- Dense oscillations in the angular velocity can be seen
- The oscillations are primarily due to:
  - Slow response time of the actuators due to bus design
  - Under-actuated nature of magnetorquers: max magnetic moment of $0.18\ \mathrm{A.m^2}$

Phase 1: De-tumbling Results

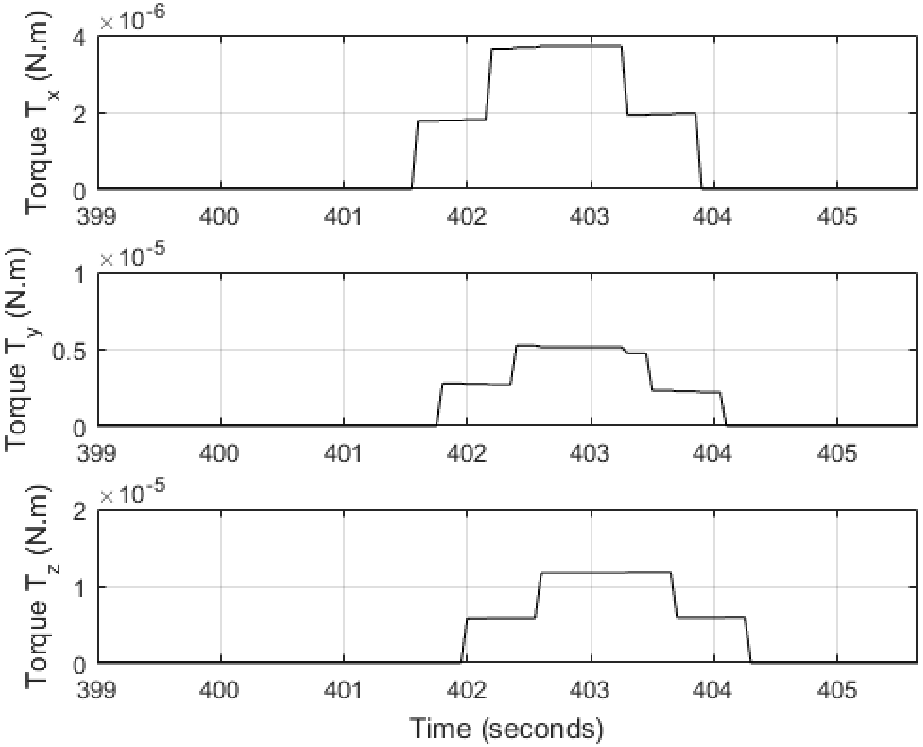


Figure 10: De-tumbling results for CAPSat. Applied torque (N.m) vs time (seconds) for one control cycle.

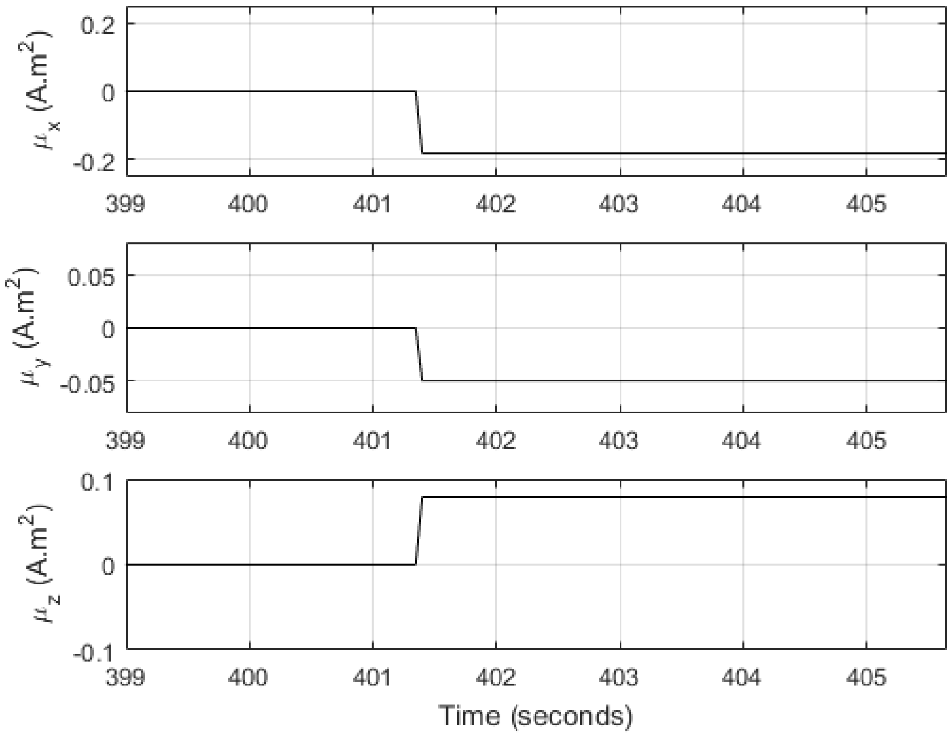


Figure 11: De-tumbling results for CAPSat. Commanded magnetic moment (A.m^2) vs time (seconds) for one control cycle.

Phase 2: Earth Pointing

- To achieve antenna Earth pointing, the actuators will attempt to align the satellite body frame and LVLH
- Control law used:

$$\mathbf{u^b} = -\mathrm{k_p}\mathbf{q_b^{LVLH}}(\mathbf{2:4}) - \mathrm{k_d}\boldsymbol{\omega}^{\mathbf{b}}_{\mathbf{b/r}}$$

Where:

- $\mathbf{u^b}$ = control input expressed in the body coordinate frame
- $\mathrm{k_p}$ = proportional gain
- $\mathbf{q_b^{LVLH}}(\mathbf{2:4})$ = vector containing the last three elements (the non-scalar) of the quaternion describing the rotation between the body frame and the LVLH frame
- $\mathrm{k_d}$ = derivative gain
- $\boldsymbol{\omega}^{\mathbf{b}}_{\mathbf{b/r}}$ = angular velocity of the body coordinate frame relative to the LVLH coordinate frame expressed in the body frame

Phase 2: Earth Pointing

- Arbitrary attitude was reached after de-tumbling:
  - $\mathbf{q_b^I} = [0.427 \quad 0.468 \quad 0.137 \quad 0.762]^\mathrm{T}$
  - Which corresponds to **113 degrees** between the antenna and the nadir vector
  - Angular rates near zero

- Attitude is determined using a seven state Extended Kalman Filter
- The LVLH frame attitude is calculated on-board during every control cycle
- The quaternion expressing the rotation between the body frame and the LVLH frame must go to $[\pm 1 \;\; 0 \;\; 0 \;\; 0]^\mathrm{T}$ for alignment

Phase 2: Earth Pointing Results

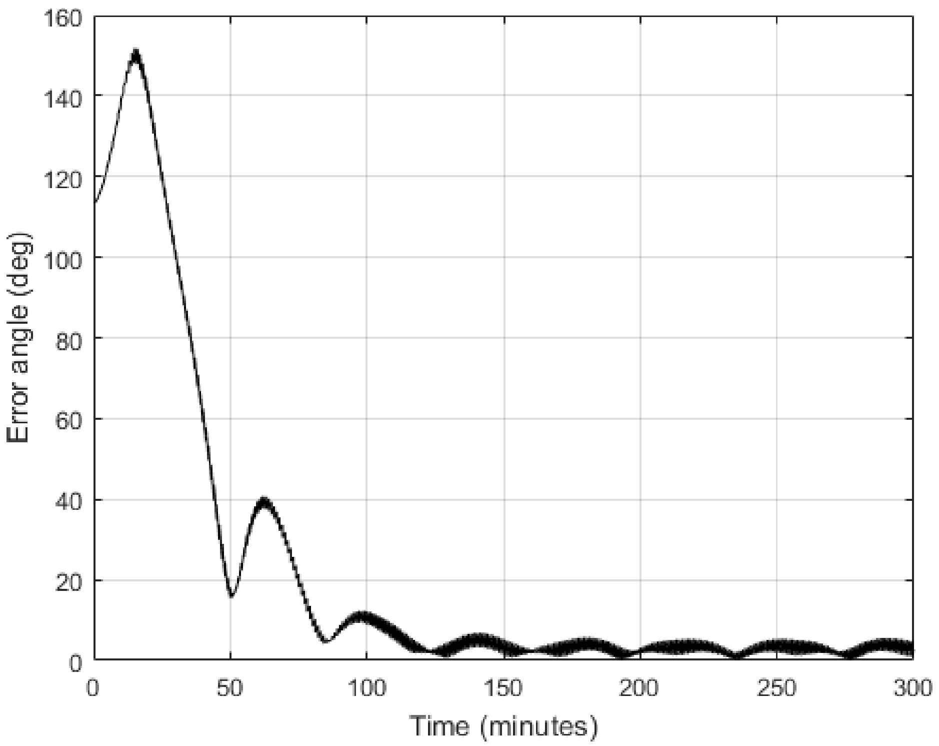


Figure 11: Earth-pointing results for CAPSat. Error angle (degrees) between the antenna and the nadir vector vs time (minutes).

- The error angle (angle between antenna and nadir vector) decreased to 0 ± 6 degrees in 142.2 minutes (less than 2 orbits)
- Oscillations are also present. They are also due to the hardware constraints and slow actuator response

Phase 2: Earth Pointing Results

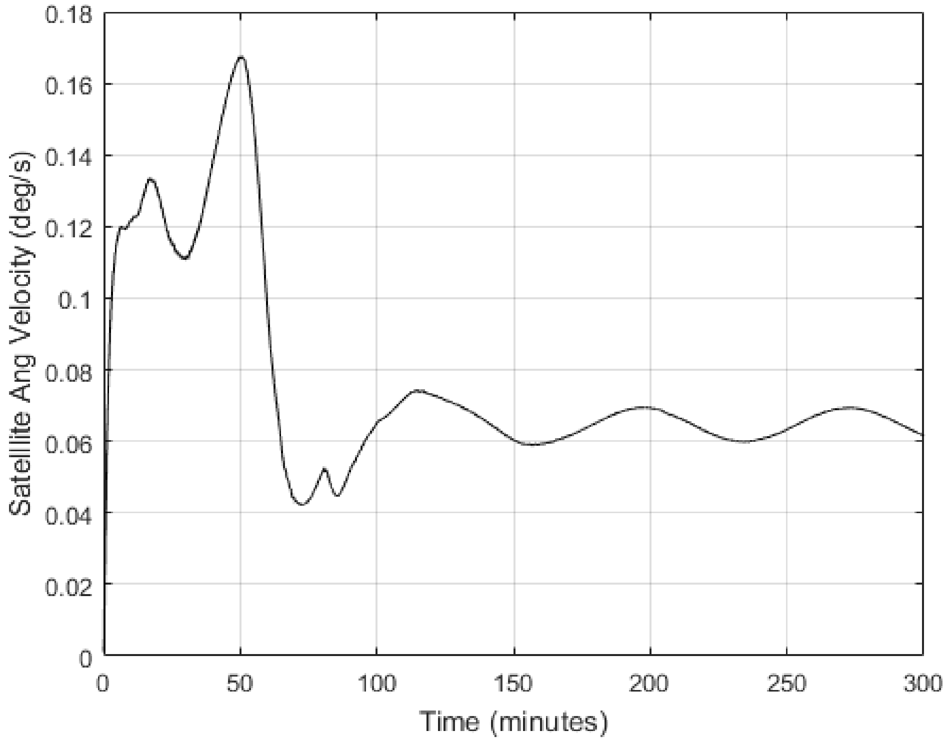


Figure 12: Earth-pointing results for CAPSat. Satellite angular velocity (deg/s) vs time (minutes).

- In LEO, satellites have an almost circular orbit. To remain nadir-pointing, the satellite should rotate at about the circular rate:

$$\omega_{\text{circ}} = \frac{360°}{T_{\text{CAPSat}}} = \frac{360°}{92.83} = 0.065 \text{ deg/s}$$

- The satellite angular velocity does settle down at $0.065 \text{ deg/s} \pm 7.7\%$.

Section 5:
Conclusion and Future Work

Conclusion

- CLASS was able to identify the following errors in the CAPSat bus design:
  - High sensor resolutions
  - Sign flip in control algorithm
  - Wrong magnetorquer installation

- All of the above errors are costly and mission critical
- CLASS offers an easily configurable and modular test system that helps reduce mission failures and improve verification and validation procedure in CubeSat development

Future Work

- The initial development phase of CLASS, mainly including the general infrastructure/hardware and the main simulation algorithms are complete

- Upcoming additions to CLASS:
  - Space environment models (atmosphere density calculations, SRP, radiation etc.)
  - Celestial mechanics for Sun sensor and star tracker testing
  - A real-time graphics visualization of the mission and satellite during testing
  - Implementation of other subsystems (such as EPS, C&DH etc.)

Thank You for Listening