SOC-i: A CubeSat Demonstration of Optimization-Based Real-Time Constrained Attitude Control

Conference Paper · March 2021

DOI: 10.1109/AERO50100.2021.9438540

24 authors, including:

Taylor Reynolds
University of Washington Seattle
25 PUBLICATIONS **167** CITATIONS
SEE PROFILE

Gorkem Caylak
University of Washington Seattle
1 PUBLICATION **1** CITATION
SEE PROFILE

James Rosenthal
University of Washington Seattle
18 PUBLICATIONS **73** CITATIONS
SEE PROFILE

Ellory Freneau
University of Washington Seattle
3 PUBLICATIONS **4** CITATIONS
SEE PROFILE

SOC-i: A CubeSat Demonstration of Optimization-Based Real-Time Constrained Attitude Control

Taylor P. Reynolds, Charles L. Kelly, Cole Morgan, Arnela Grebovic, Jerrold Erickson, Henry Brown, William C. Pope, Jonathan Casamayor, Kyle Kearsley, Gorkem Caylak, Kyle E. Fisher, Cameron Wutzke, Kylle Ashton, James Rosenthal, Devan Tormey, Ellory Freneau, Garrett Giddings, Hasan Emin Horata, Anika Dighde, Saharsh Parakh, Jiaping Zhen, John C. Purpura, Daniel B. Pratt, and Anders Hunt

University of Washington Aeronautics & Astronautics CubeSat Team
Seattle, WA 98105
aact@uw.edu

Abstract— **The Satellite for Optimal Control and Imaging (SOC-i) is a 2U CubeSat that will fly the first in-space demonstration of real-time optimization-based constrained attitude control. SOC-i is developed by AACT, a student team at the University of Washington (UW), and consists of two payloads and several supporting subsystems. The primary payload is a convex optimization-based attitude guidance algorithm called SOC-i's Optimal Attitude Reorientation (SOAR). SOAR will be the first software of its kind tested on an orbiting spacecraft. The secondary payload is a CMOS camera that serves as an instrument to demonstrate SOC-i 's pointing abilities. Though SOC-i is a CubeSat flight demonstration of the SOAR technology, it is relevant for spacecraft missions of all types given the prevalence of sun sensor-based attitude estimation, the need for sun-sensitive instrumentation in space, the finite control authority of all actuators, and the desire to execute rotations while minimizing some performance metric (e.g., electrical power). A small satellite mission is an ideal proving ground for SOAR. To reduce mission risk, SOC-i leverages COTS components from standard CubeSat suppliers; the EPS management system, battery, solar arrays, antenna, and actuators all have flight heritage, while the chassis and onboard computer are custom built by UW students. The design and development of these subsystems is presented in detail.**

Figure 1: The Satellite for Optimal Control and Imaging (SOC-i).

TABLE OF CONTENTS

1. INTRODUCTION

In-space autonomy is an important component in the design of satellite constellations, deep-space robotic exploration, and human spaceflight. Part of what allows autonomous systems to successfully achieve these objectives is the use of guidance and control techniques, and classical feedback control typically forms the nucleus of a spacecraft's automatic control system. A challenge increasingly faced by modern spacecraft is *constrained* attitude control, where a control system must achieve both traditional objectives (e.g., stability, disturbance rejection) and ensure the simultaneous satisfaction of constraints (e.g., multiple pointing constraints and actuator limitations). Typically, these attitude control constraints are mission-specific. For example, when a spacecraft has light-sensitive instruments – often in the form of imaging apparatus – reorientations must be carefully planned so that the instrument does not point directly at the sun, a type of pointing exclusion constraint. Pointing inclusion and exclusion zones are quite difficult to enforce using classical control design methods such as loopshaping or standard linear-quadratic-regulator (LQR). When the nonlinear equations of motion and torque and slew rate limits are also enforced as hard constraints, these difficulties compound. Historically, engineers have relied on a combination of heuristic algorithms or humans-in-the-loop to satisfy such constraints simultaneously, resulting in conservative designs or limited autonomy. In one example, the highly sensitive LORRI instrument on New Horizons was nearly destroyed when an attitude maneuver inadvertently pointed it at the sun during its voyage to Pluto [1]. The control system was designed to satisfy initial and final attitude constraints, but could not ensure the sun avoidance requirement was met during the intermediate attitudes.

In order to support autonomous operations, constrained attitude guidance maneuvers must be computed in real-time aboard the spacecraft. Optimization-based algorithms are quite good at finding feasible solutions to highly-constrained problems, with the fortuitous byproduct of optimality with respect to a desired metric. While the theory is well-developed, no spacecraft has flown and executed a real-time optimization-based attitude guidance algorithm to date. The Satellite for Optimal Control and Imaging (SOC-i) is a 2U CubeSat whose mission is the first in-space technology

978-1-7281-7436-5/21/$31.00

demonstration of such an algorithm. The payload, SOAR, uses optimization-based attitude guidance methods developed at the University of Washington to compute trajectories in real-time that meet a set of five constraints throughout the entire maneuvers. The constraints include maintaining line of sight between a sun sensor and the sun, ensuring the sun does not point directly at an onboard camera, bounded attitude rates, and bounded control inputs, among others. SOAR is integrated with SOC-i's guidance, navigation, and control (GNC) software and can autonomously compute attitude, attitude rate, and torque trajectories that together satisfy the nonlinear rotational dynamics of the spacecraft. Though SOC-i is a CubeSat flight demonstration of this technology, SOAR is relevant for spacecraft missions of all types given the prevalence of sun sensor-based attitude estimation, the need for sun-sensitive instrumentation in space, the finite control authority of all actuators, and the desire to execute rotations using minimum onboard power.

SOC-i is being developed by the all-student Aeronautics & Astronautics CubeSat Team (AACT) at the University of Washington. In this paper, we present the payload details and system-level design of SOC-i. A notional rendering of the spacecraft is shown in Figure 1. The mission is currently in its pre-CDR stage and we present our progress toward hardware development and a benchtop spacecraft prototype. The mission was selected by the 11$^{\text{th}}$ round of NASA's CubeSat Launch Initiative for a tentative launch date in 2022–2023.

2. PAYLOAD: SOC-I OPTIMAL ATTITUDE REORIENTATION

SOC-i's Optimal Attitude Reorientation (SOAR), our primary payload, is an optimization-based algorithm that computes attitude guidance commands in order to reorient the satellite. By attitude guidance, we mean a state (attitude and angular rates) and control (reaction wheel torque) that are implemented in a feedforward control architecture. The connection between SOAR and more classical feedback control architectures is highlighted in Figure 2.

SOAR is designed to solve a free-final time, nonconvex optimal control problem in real-time while meeting three key objectives: the state and control trajectories respect the rigid body equations of motion (dynamic feasibility); multiple state and control constraints are respected (constraint feasibility); and the control trajectories are computed so as to minimize the power drawn by our set of four reaction wheels.

Equations of Motion

To express the equations of motion that are used, we denote a body-fixed (rotating) coordinate frame $\mathcal{F}_\mathcal{B}$ with origin at the center of mass and coordinate axes as shown in Figure 8. We express the attitude of the satellite using a unit quaternion, $\boldsymbol{q} \in R^4$, that represents the orientation of $\mathcal{F}_\mathcal{B}$ with respect to the Earth Centered Inertial frame. The momentum of the satellite bus and a reaction wheel array, in the body frame, are denoted as $\boldsymbol{h}_\mathcal{B} \in R^3$ and $\boldsymbol{h}_w \in R^3$ respectively. Our control input is assumed to the reaction wheel torque, $\boldsymbol{\tau}_\mathcal{B} \in R^3$. The state and control vectors used by SOAR are then

$$\boldsymbol{x}(t) = \begin{bmatrix} \boldsymbol{q}(t) \\ \boldsymbol{h}_\mathcal{B}(t) \\ \boldsymbol{h}_w(t) \end{bmatrix} \quad \text{and} \quad \boldsymbol{u}(t) = \boldsymbol{\tau}_\mathcal{B}(t) \tag{1}$$

The quaternion kinematics and rigid body dynamics can be expressed as (suppressing the argument of time, t)

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}) = \begin{bmatrix} \frac{1}{2}\boldsymbol{q} \otimes \left(J^{-1}\boldsymbol{h}_\mathcal{B}\right) \\ -\boldsymbol{\tau}_\mathcal{B} + \left(\boldsymbol{h}_\mathcal{B} + \boldsymbol{h}_w\right) \times \left(J^{-1}\boldsymbol{h}_\mathcal{B}\right) \\ \boldsymbol{\tau}_\mathcal{B} \end{bmatrix} \tag{2}$$

where J is the inertia matrix of the satellite and $\otimes$ represents quaternion multiplication. The state and control solutions computed by SOAR satisfy the differential equation (2) to a desired numerical tolerance from a given initial condition.

Problem Constraints

The state, control and total maneuver time computed by SOAR satisfy several constraints. The final time, $t_f \in R$, is a free parameter, and we restrict it to a finite interval of the positive real line by imposing the constraint

$$0 < t_{f,\min} \leq t_f \leq t_{f,\max} \tag{3}$$

This helps to ensure a bounded optimization problem.

The control signal is bounded by the capabilities of the onboard reaction wheel array. Because SOC-i has an array consisting of four wheels, the set of feasible torques in the body frame is a polytope with 12 facets [2]. To maximize the volume of permissible torques would therefore involve enforcing 12 half-space constraints. However, SOAR does not use the full volume of this polytope for three reasons. First, SOAR is intentionally designed to minimize the power consumed by the wheel assembly during maneuvers, a goal that is consistent with not using large torques. Second, we need to ensure that there is some margin available for downstream feedback control that is used to reject external disturbances and correct for any model uncertainty relative to (2). Third, by implementing the bounded-torque constraint with an axis-aligned cuboid that strictly lies inside the feasible polytope, we can reduce the number of required constraints from 12 to 6, an important savings for a real-time implementation. After selecting an appropriate set of dimensions that define a cuboid strictly contained in the feasible torque polytope, $\boldsymbol{u}_{\max} \in R^3$, the following control constraint is enforced

$$-\boldsymbol{u}_{\max} \leq \boldsymbol{\tau}_\mathcal{B} \leq \boldsymbol{u}_{\max} \tag{4}$$

Note that this strategy for determining the control constraint is generalizable to any reaction wheel assembly with three or more wheels.

There are three different constraints imposed on the state vector. First, we limit the angular rates by enforcing

$$\|J^{-1}\boldsymbol{h}_\mathcal{B}\|_\infty \leq \omega_{\max} \tag{5}$$

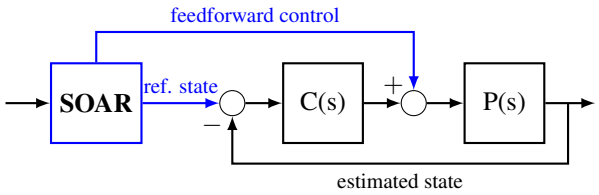


Figure 2: Conceptual integration of SOAR within a classical feedback control architecture with controller $C(s)$ and plant $P(s)$.

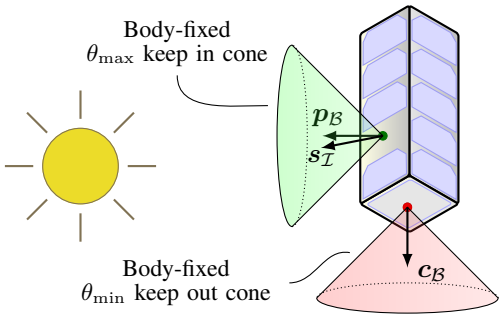


Figure 3: The geometry of the pointing inclusion (green) and exclusion (red) constraints enforced by the SOAR payload.

Next, we include one attitude inclusion constraint and one attitude exclusion constraint [3], [4]. The inertial spacecraft-to-sun vector is constrained to remain within a $\theta_{\max}$ degree cone (half-angle) centered around the sun sensor boresight direction in the body frame. At the same time, the inertial spacecraft-to-sun vector is constrained to remain outside of a $\theta_{\min}$ degree cone (half-angle) centered around the camera boresight direction in the body frame. Figure 3 provides a depiction of the geometry of each of these constraints relative to the body of the satellite. Both constraints can be expressed as quadratic inequalities on the attitude quaternion of the form

$$\boldsymbol{q}^\top M_i \boldsymbol{q} \leq 2 \quad \text{and} \quad \boldsymbol{q}^\top M_e \boldsymbol{q} \leq 2 \tag{6}$$

respectively. We refer the reader to [3], [4] for the construction of the matrices M_i and M_e from the (unit) vectors that define the constraints. Both matrices are positive semidefinite, and therefore (6) are each second-order cone constraints of the state variable, a type of convex constraint.

Optimal Control Problem

The free-final time nonconvex optimal control problem that is solved by SOAR can be summarized as

$$\min_{t_f, \boldsymbol{\tau}_{\mathcal{B}}} \quad \int_0^{t_f} \|\boldsymbol{\tau}_{\mathcal{B}}(t)\|_2^2 \, \mathrm{dt} \quad \text{s.t.} \tag{7a}$$

$$(2),\ (3),\ (4),\ (5),\ (6) \tag{7b}$$

$$\boldsymbol{q}(0) = \boldsymbol{q}_0, \quad \boldsymbol{h}_{\mathcal{B}}(0) = \boldsymbol{h}_{\mathcal{B},0}, \quad \boldsymbol{h}_w(0) = \boldsymbol{h}_{w,0} \tag{7c}$$

$$\boldsymbol{q}(t_f) = \boldsymbol{q}_f, \quad \boldsymbol{h}_{\mathcal{B}}(t_f) = 0 \tag{7d}$$

The cost function (7a) serves as a proxy for the power consumption of the reaction wheel assembly. The boundary conditions (7c) and (7d) are defined so that the satellite arrives at the target attitude $\boldsymbol{q}_f$ with zero body rates. The data $\{\boldsymbol{q}_0, \boldsymbol{h}_{\mathcal{B},0}, \boldsymbol{h}_{w,0}\}$ that defines the initial condition are provided by an onboard state estimation routine that is external to SOAR.

Solution Process

The SOAR payload uses successive convexification to solve the nonconvex optimal control problem given in (7). The nonconvexity of the optimal control problem (7) is a consequence of the nonlinear equations of motion. The algorithmic details are largely derived from the methods found in [5], [6], [7], which are also based on [8]. One key difference is that a trust region constraint is not required to solve the problem reliably [9].

SOAR solves a sequence of second-order cone programs, a sub-class of convex optimization, using the open-source ECOS solver [10]. ECOS solves problems of the form

$$\min_{\boldsymbol{z}} \quad \boldsymbol{c}^T \boldsymbol{z} \quad \text{s.t.} \tag{8a}$$

$$A\boldsymbol{z} = \boldsymbol{b} \tag{8b}$$

$$G\boldsymbol{z} \preceq_{\mathcal{K}} \boldsymbol{h} \tag{8c}$$

where the problem data $\{A, \boldsymbol{b}, \boldsymbol{c}, G, \boldsymbol{h}, \mathcal{K}\}$ are formed using the problem data from (7); a customized parsing process that is specifically tailored to the problem. The ECOS solution variable $\boldsymbol{z}$ contains the final time, the state and control vectors at a discrete set of temporal points, auxiliary variables required by successive convexification, and slack variables required to write the problem in the standard form shown in (8). Reference [6] contains a discussion of techniques to convert between optimal control problems of the form (7) and convex parameter optimization problems of the form (8) when using successive convexification. The notation $\preceq_{\mathcal{K}}$ denotes a generalized inequality with respect to a cone $\mathcal{K}$ [11]. For SOAR, two cones are used: the standard nonnegative orthant and a second-order cone.

At each iteration, in order to construct a problem of the form (8), the nonlinear dynamics are linearized about the previous solution. This leads to a linear time-varying (LTV) system of equations. These LTV dynamics are then discretized by solving the differential equation at a set of pre-selected temporal nodes by a assuming a profile for the control input between nodes. For SOAR, an affine interpolation and 10 equally space temporal nodes are used. The final result of this process is a set of discrete LTV equations that, while not converged, approximate the original nonlinear dynamics. Readers are referred to [5], [7] for details. Importantly, this discretization procedure ensures that upon convergence, the state and control pair computed by SOAR satisfies the original nonlinear dynamics (2) to a numerical tolerance selected by the designer. Stated another way, if the control solution obtained by SOAR is integrated through the nonlinear equations of motion (2), starting from the same initial condition defined by $\{\boldsymbol{q}_0, \boldsymbol{h}_{\mathcal{B},0}, \boldsymbol{h}_{w,0}\}$, then at each temporal node the resulting state vector will match the discrete state vector computed by SOAR.

Figure 4 provides a block diagram representation of how SOAR solves the optimal control problem (7). The initial guess is generated for each instance by using the SLERP algorithm (see [12]) to interpolate between the initial attitude $\boldsymbol{q}_0$ and the desired final attitude $\boldsymbol{q}_f$, along with zero satellite momentum, constant wheel momentum, zero wheel torque, and the midpoint of the final time interval. This initial guess is not necessarily dynamically feasible nor does it necessarily satisfy the attitude pointing constraints. This is one of the strengths of the successive convexification methods developed in [5], [7], [9], [8]; a simple infeasible initial guess is often sufficient. The stopping criteria is defined as recommended in [6].

Implementation Details

Because the core GNC flight software for SOC-i is being designed in Matlab/Simulink, the SOAR payload was designed in Simulink as an atomic library. The ECOS solver is available a set of C files, and Matlab's Legacy Code Tool was used used to create an S-function from these files that is callable from within the SOAR Simulink library. SOAR is integrated with the core GNC flight software and they are autocoded as a single unit. A similar procedure that inspired our approach was used in [13].

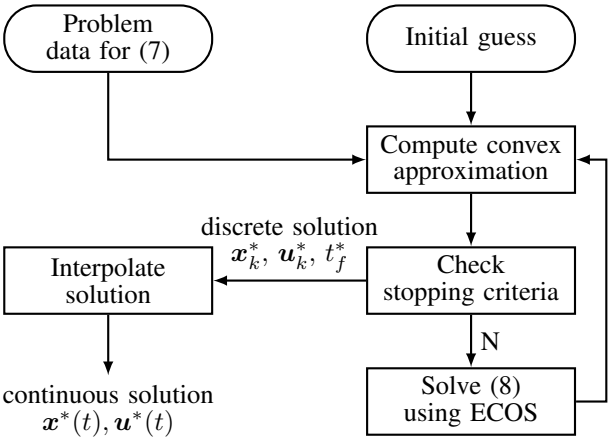


Figure 4: A block diagram of the SOAR algorithm.

By following the implementation recommendations in [6], all variable sizes can be predefined and all memory can be statically allocated. This is partly because the number of temporal nodes used to discretize the continuous time problem is a fixed constant. (The number of temporal nodes is also a significant driver of computation time.) In order to bound the runtime of the algorithm, we limit SOAR to a maximum of 15 iterations around the loop in Figure 4. Our initial testing has thus far not shown any trials that require this many iterations. Once a solution has been obtained, the main loop terminates and the discrete solution is fed into an interpolation function that acts as a discrete-to-continuous transformation. The discrete optimal state computed by the successive convexification procedure is interpolated via fourth order Runge-Kutta numerical integration with a step size equal to the sample time of the core GNC flight software, whereas the corresponding optimal control is interpolated by using the same linear interpolation that was assumed during the discretization of the equations of motion.

The outputs of the SOAR library are the optimal state and control, $\boldsymbol{x}^*(t)$ and $\boldsymbol{u}^*(t)$, and a custom telemetry package. After the optimal final time, t_f^*, is reached, SOAR will continue to output the final state $\boldsymbol{x}^*(t_f)$ and a zero torque command. The SOAR telemetry package includes the entire discrete state, control, and final time computed, as well as the time taken to solve each iteration, a set of custom-defined exit codes to report on the internal status of the ECOS solver and SOAR algorithm, as well as a unique integer to identify the maneuver.

Initial Validation

The preliminary flight implementation of the SOAR payload has been tested across a range of maneuvers that are representative of what we expect during the SOC-i mission. We present here the results of a 1000-trial Monte Carlo test case that was used to assess initial performance.

For every test case, the commanded attitude and angular rates were set to $\boldsymbol{q}_{cmd} = (1, 0, 0, 0)$ and $\boldsymbol{h}_{\mathcal{B},cmd} = (0, 0, 0)$ respectively. The initial attitude was selected by first choosing a random unit direction vector, and then uniformly sampling a total angle of rotation from the interval $[-90°, 90°]$. The initial quaternion was then constructed from this axis-angle pair, and the distribution of initial angles is shown in Figure 5a (top). The initial angular velocity was computed using a normal distribution with zero mean and covariance matrix $\mathrm{diag}\big(10^{-3}, 10^{-3}, 10^{-2}\big)$. The angular velocity was then mapped to an initial angular momentum using the spacecraft inertia matrix. Lastly, the initial reaction wheel speeds (in units of RPM) were sampled from a normal distribution with mean $1000 \cdot (1, -1, 1, -1)$ and covariance matrix $50 \cdot I_4$. The wheels speeds were then mapped to wheel momentum in the body frame using the wheel inertia and geometry.

Out of 1000 trials, all 1000 were successfully solved. The maximum number of iterations taken to solve a problem was 13, and the median was 4. To assess the accuracy of the solutions with respect to the dynamics, the final state output by SOAR was compared to the desired value of $\boldsymbol{q}_{cmd}$. The final state output by SOAR corresponds to an open-loop propagation of the SOAR control solution through the nonlinear equations of motion. The distribution of the net angle error at the final time is shown in Figure 5a. The maximum error was $2.2°$, and the median was $0.47°$. Figure 5b shows the relationship between the initial and final angle error – and reveals an intuitive trend that the final error grows as the maneuver angle increases. We stress that the errors shown here are solely *open-loop* errors, and do not account for the additional tracking performance available from feedback control.

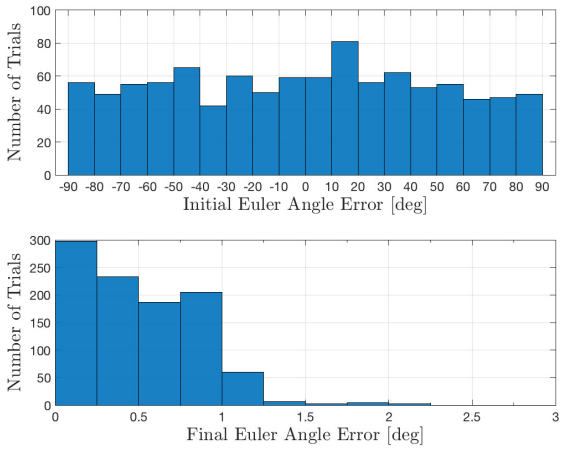
The number of temporal nodes selected for the discretization and the convergence tolerance are key drivers for the final error; and indeed it is true that the "V" shape becomes shallower with increased nodal density or decreased convergence tolerance. The trade-off is computational time; more temporal nodes leads to a larger optimization problem, and decreased tolerances require more iterations to converge, either case resulting in higher runtimes. For these trials, the maximum (cumulative) time required by the solver was 65 ms and the median solver time was 19 ms on a 2014 MacBook Pro with a 2.2 GHz Intel Core i7 processor with 16 GB of RAM. While these initial numbers are quite promising, final design decisions can only be made once the payload has been tested on the target flight processor.

3. SYSTEM DESIGN

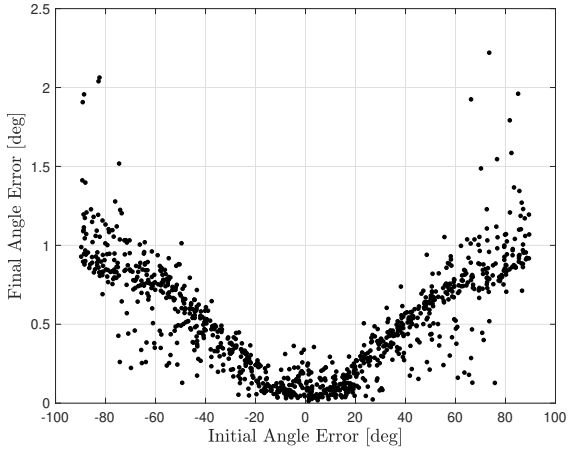
Concept of Operations

There are four main phases of the SOC-i mission. After launch, the mission begins autonomously in the *commissioning* phase. The commissioning phase begins with an automatic power-up sequence that boots only the EPS and onboard computer (OBC). The OBC performs a limited set of preliminary software functions. Once the battery is sufficiently charged, the power domains associated with the radio and attitude sensors are turned on. These allow a broader set of hardware checkouts to take place, and the radio is able to begin sending a basic "alive" beacon with a set of core system data. The commissioning phase ends once a link is established between the satellite and a ground station. Once this link has been established and the initial data has been processed, the satellite will be manually commanded into the *mission* phase from the ground station.

The mission phase, depicted in Figure 6a, is where the satellite will spend the vast majority of its mission life. During this phase, all power domains are nominally powered on when the batteries are sufficiently charged. First, the GNC system will de-tumble SOC-i, nulling out any angular rates that were obtained at deployment or have accumulated during the time since deployment. Next, the GNC system reorients the satellite so that the sun is in the field of view of the fine sun



(a) Initial and final open-loop error frequencies.



(b) Final versus initial open-loop error.

Figure 5: Preliminary Monte Carlo testing of the SOAR payload for a representative range of maneuvers.

sensor (sun acquisition) at an offset angle that maximizes the total solar cell area facing the sun. Nominal pointing proceeds by aligning the (offset) sun sensor boresight direction with the inertial sun vector, and the antenna boresight direction with the inertial nadir vector (with priority in that order). This nominal pointing objective is maintained at all times during the mission phase. Data collected during the mission phase is essential to assessing the performance of all non-payload functionality of SOC-i.

The remaining two phases, called *imaging* and *experimental* complete the primary phases of the SOC-i mission. These are similar in execution, so it is sufficient to describe only the experimental phase in which the SOAR payload will be demonstrated (Figure 6b). During a ground pass, we will send a command to the satellite enabling the GNC system to execute the SOAR payload at a specified time. The maneuver epoch and target orientation will be sent as part of this communication. Note that this target orientation can be anything; a small angle maneuver to perform an initial checkout of the payload, or a larger angle maneuver designed to test its capabilities. Once the maneuver epoch is reached, the SOAR payload is called, and the subsequent maneuver is executed. Once this is complete, the satellite resumes the same operations as in the mission phase until such time that the maneuver data is downlinked - at which point SOC-i returns to the mission phase. The imaging phase is executed identically, with the exception that SOAR is not used and an image is taken at the target attitude instead.

System-level Design Considerations

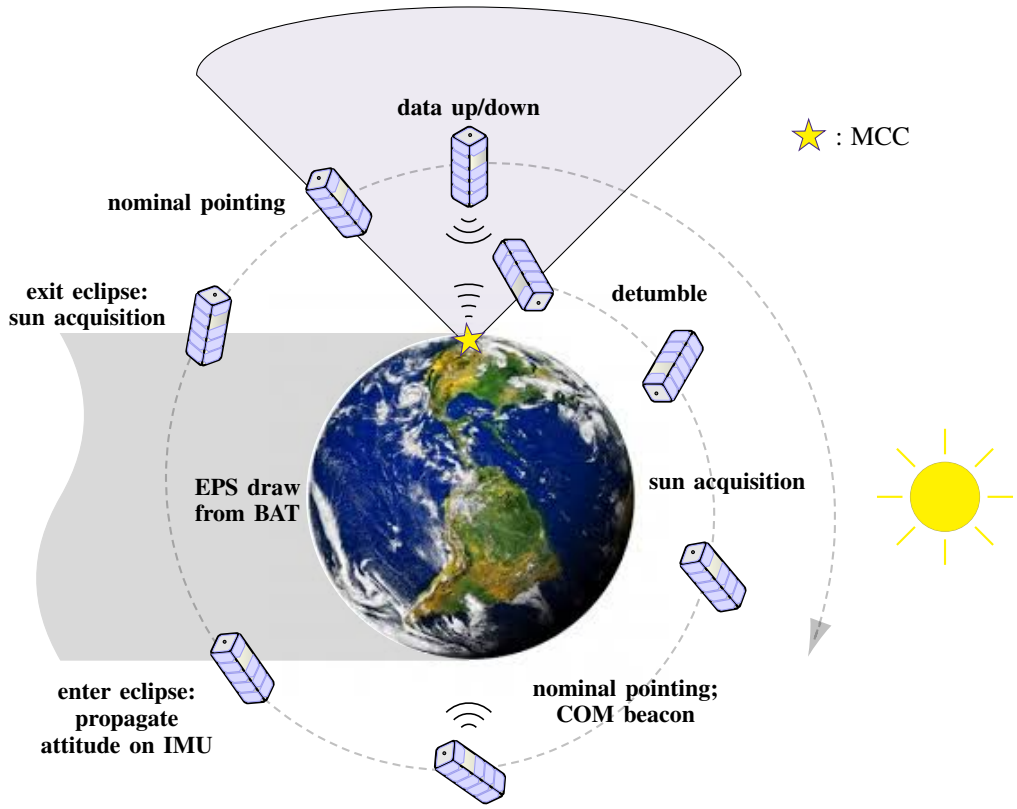
The preliminary design process for the satellite involved constant trade-offs between different subsystems being developed in parallel. We made a few design decisions upfront to initially constrain the system design. We decided to procure several components from off-the-shelf CubeSat suppliers; specifically, the EPS, battery, and solar arrays would be bought as a set, the antenna would be bought off the shelf to reduce risks associated with deployment, the communications system would be assembled from COTS components, the GNC actuators would be bought COTS, and the imager would be a cheap COTS component. Most of these decisions were made to substantially reduce the risk posture of SOC-i. Because it is our first CubeSat design, we chose to focus our custom development efforts on a few foundational elements such that the next mission(s) would provide a platform for fully custom hardware. The custom elements are the chassis and the flight computer, with additional customization in the integration of the GNC hardware consisting of many individual COTS components. With many components being off-the-shelf, our system design initially involved trade studies of available hardware on the market.

The primary design driver for SOC-i was the highly-coupled development of the communications system (COM) and the electrical power system (EPS). Due to the large volume of data that the SOAR payload and imaging system must downlink to the ground station, the initial link budget analysis suggested the COM hardware would require an orbit-average power of nearly 4 W. At the same time, the design objective of EPS was to meet the mission's power requirements without the use of deployed solar panels (a decision based on our risk posture and budget limitations). We found that it was not possible to generate enough power with body-mount panels on a 2U CubeSat to satisfy COM's power usage on top of all other mission-critical subsystem operations. This led to a crucial design choice: we could either a) accept the costs and risks associated with using deployable solar arrays; or b) reconsider the selection of the high-power COM hardware and the requirements and assumptions that led to their selection.

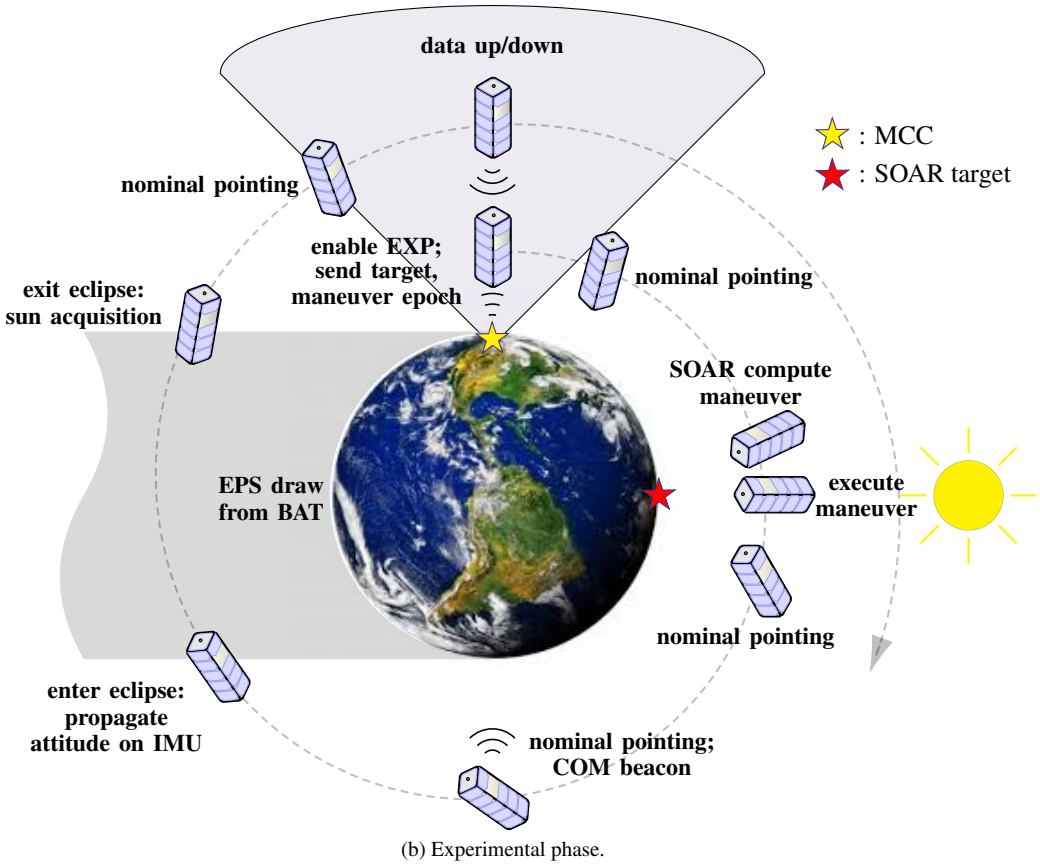
Ultimately, we took the latter approach. We reconsidered the volume of data required to be transmitted during both ground station downlinks and beacons, and we also refined our link budget calculations. This enabled SOC-i to transmit with lower power - reduced from a 2W transmission mode to a 0.5W mode - as well as for shorter durations, while still satisfying the mission requirements of COM. With this, EPS's power budget was closed without the need for deployable solar arrays.

Electrical Power System

The primary objective of the EPS design is to ensure that SOC-i operates with a positive "power budget" – the average electrical power generated during an orbit minus the average electrical power consumed by all loads – with sufficient margin during all phases of the mission. Therefore, we informed our design by developing a model of the spacecraft's power-generating and power-consuming elements. It is essential that



(a) Mission phase.



(b) Experimental phase.

Figure 6: A visualization of SOC-i's Concept of Operations for the mission and experimental phases of the mission.

both the power generation and the power consumption are modeled accurately enough in order to use the results to drive design.

For power generation, the model must include properties of the solar arrays and onboard batteries as well as information about the environment of the spacecraft. This includes, but is not limited to:

- The maximum power point of the solar cells and how this changes with temperature, degradation, and spacecraft attitude;
- Orbital parameters - particularly the amount of time spent in sunlight and eclipse;
- The attitude of the spacecraft throughout an orbit, and the number of solar cells exposed to sunlight as a result of this;
- The efficiency of the power electronics circuits used to control the solar arrays.

Accurately modeling these characteristics requires solar cell specifications and propagating notional orbits. Some assumptions were made to simplify the analysis, including:

- On the timescale of one orbit, losses due to the maximum power point tracking (MPPT) controllers of the EPS system were negligible;
- Power generated from earth's albedo is not to be relied on to close the power budget, and is thus neglected;
- Unless specifically considering other cases (such as the case of SOC-i tumbling), it is assumed that GNC is successful in controlling the attitude of the spacecraft.

Modeling the electrical loads of the spacecraft also involves several considerations. The main factors we used were the following:

- The mission's ConOps, which determines when different electrical loads are powered on and for how long;
- The orbital average power required for each system, sometimes modeled as a product of peak power and duty ratio;
- Losses in the system due to the efficiencies of the power conditioning modules (PCMs) that deliver power to the payloads and factors such as the round-trip efficiency of the battery.

Wherever possible, the power or efficiency of each system is derived from direct measurement. Otherwise, these parameters were found in datasheets, or in some cases, reasonable assumptions were made.

With both power generation and loads modeled, the orbital average power and average load power were directly compared to compute the mission's power budget in various operating states. This power budget has constantly changed throughout the design process as design decisions were made and hardware was selected. For SOC-i, the design goal of EPS is to have a positive power budget for all operations in all modes. The current power model for SOC-i, using the custom-built EPS hardware from DHV Technology, accomplishes this. A brief summary of the results of the current power model is shown in Table 1.

Table 1: A portion of the power budget for the mission phase, including a ground station downlink.

Average Power	**Value**
Communications	652 mW
Reaction Wheel Assenbly	600 mW
Onboard Computer	398 mW
Battery Heater & Miscellaneous	425 mW
Sensors	104 mW
Efficiency Losses	308 mW

Average Sum Totals	
Orbital Load	2487 mW
Orbital Power	3182 mW
Power Budget	695 mW
Power Margin	28 %

Communication

The communication system (COM) acts as the two-way wireless link between the spacecraft and the Mission Control Center (MCC). This system is crucial for providing the data to validate the performance of the two payloads and ultimately determine mission success. The COM system provides data downlink capabilities from the spacecraft to the MCC while also providing a beacon of important health and status (H&S) data pertaining to the operation of the spacecraft that can be received by any ground station (Figure 6a). During all stages of the mission, COM additionally provides command uplink capabilities from the MCC to the spacecraft. This allows MCC to asynchronously set the spacecraft's current phase of ConOps as well as operational parameters of the spacecraft, such as a command to execute SOAR and/or imaging target orientation with a maneuver epoch.

The COM system is designed to operate in the ultra high frequency (UHF) 435-438 MHz (70 cm) amateur radio band. The UHF band supports easily deployable CubeSat-sized antennas with circular polarization and low directivity, such as a turnstile antenna, reducing pointing constraints on the GNC subsystem and mechanical complexity [14]. Further the UHF band is regularly used for small satellites, so there are abundant commercial off-the-shelf (COTS) parts available for developing the satellite and ground station transceivers. To determine a COTS radio system for the SOC-i mission, we considered components with demonstrated flight heritage on previous small satellite missions. This led us to choose the XDL Micro radio from Pacific Crest, a radio that is currently being used on Brown University's EQUiSat mission. The XDL Micro radio provides two-way half-duplex communications in the desired frequency range. It can provide configurable RF power at either 0.5 W or 2.0 W, receiver sensitivity down to -110 dBm for 10^{-5} bit error rates, and data rates between 4.8 kbps and 19.2 kbps with either Gaussian mean shift keying (GMSK) or 4-level frequency shift keying (4-FSK) modulation. During transmission at 0.5 W RF output power, the radio consumes 2.9 W of electrical power, during reception the radio consumes 0.45 W of electrical power. To close the power budget, we have designed the COM system to use the lower transmit power of 0.5 W during all phases of the mission. The XDL Micro radio will interface to an Endurosat UHF antenna featuring $>$0 dBi of gain with circular polarization. The antenna uses a turnstile configuration that can be deployed by the OBC via electric burn wires.

A link budget was developed for COM based on the intended GNC pointing accuracy, the available electrical power, the

Table 2: Link budget parameters for SOC-i

Parameter	Value
Transmit Power	27 dBm
Transmit Antenna Gain	0.9 dBic
Receive Antenna Gain	5.5 dB
Total Pointing Loss	3.1 dB
Free-Space Path Loss (400-740 km)	-142.6 to -137.2 dB
Receiver Sensitivity (FSK, 9600 baud, 1E-5 bit error rate)	-110 dBm
Link Margin	+1.7 to +7.1 dB

predicted frequency and duration of spacecraft passes over the MCC, and the downlink bandwidth that could be supported using COTS radios in the UHF band. A statistical analysis of potential orbits predicted that SOC-i would pass near the MCC with an elevation angle $>$30 degrees roughly once per day for an average of 5 minutes. Operating with a maximum data rate $<$19.2 kbps with the XDL Micro radio, we have limited the *experimental* mission phase to generating no more than 170 kBytes of science and H&S data per day (including 33% margin). With these constraints, the complete data payload can be transmitted to the MCC during a single pass if the spacecraft has a downlink rate $>$6700 bps, which falls within the XDL Micro radio's specifications.

Link budget parameters are shown in Table 2. By assuming an orbit similar to that of the International Space Station, a link margin between 1.7 and 7.1 dB was obtained, with the precise value in this range depending on the distance between the spacecraft and the MCC. While the link budget does not account for system and atmospheric losses due to moisture, it shows that the spacecraft downlink closes with margin using the selected COTS hardware. We are investigating additional measures to improve the link budget margin, e.g. by adding a higher gain Yagi-Uda antenna at the MCC with a rotator for pointing.

Guidance, Navigation, and Control

The GNC subsystem is responsible for the management of the spacecraft's physical state vector (position, velocity, attitude and angular rates). At a high level, this system encompasses onboard GNC flight software, operation of all spacecraft attitude actuators and sensors, and dynamic and environmental models used to validate said flight software.

We use an active control scheme that can achieve three-axis stabilization. The only states that are controlled in closed-loop are the attitude and angular rates; the position and velocity are only estimated. The mission requires that the spacecraft can arbitrarily point to generate maximum power using body-mounted solar panels, communicate with the MCC ground station in Seattle, and fulfill its Earth-imaging and technology demonstration objectives. GNC performance is paramount to the success of these mission goals; design of all GNC hardware and software components was therefore aimed at maximizing our ability to reliably satisfy these goals. A suite of sensors will be used to determine the attitude and angular rate of the spacecraft, and the flight software will compute any necessary attitude or angular rate corrections that should be executed to achieve the desired pointing objective.

A key mission requirement driven by the Earth-imaging objective is the need for the GNC subsystem to achieve a pointing accuracy of less than 7 deg between a body-fixed vector and an inertial target (attitude accuracy). Given the field of view of our camera, and the expected orbital altitude, we estimate that we will be able to accurately capture images of a desired ground target with a maximum error of this magnitude. Designing to this 7 deg pointing requirement was split between our ability to estimate our attitude to within 3.5 deg, and our ability to control the attitude to within 3.5 deg of a target. The attitude must also be kept sufficiently stable in order to take good-quality images (attitude stability). The SOAR payload requires tighter error bounds to adequately verify its performance, and this drove component selection. To control the attitude of the spacecraft, we use a set of four reaction wheels arranged in a tetrahedral configuration from NanoAvionics (see Figure 11) and 5 air-core magnetorquers built into the solar panels on the body of the spacecraft. To estimate the attitude of the spacecraft, we use a standard configuration of one digital sun sensor, four analog sun sensors, three 3-axis gyroscopes, and three 3-axis magnetometers.

The sun sensor provides the spacecraft-to-sun vector in the body frame, and the magnetometer provides the local Earth magnetic field vector in the body frame. The flight software must therefore maintain knowledge of these vectors in the Earth-Centered Inertial frame (ECI) to infer the orientation of the spacecraft. The position of the spacecraft in ECI will be continuously estimated using the SGP4 algorithm [15]. In the ECI frame, with a knowledge of the current time and the position generated by SGP4, the inertial spacecraft-to-sun vector can be computed (see [16]) and an estimate of the local magnetic field can be generated using standard magnetic models [17]. These two inertial vectors are combined with their measured body frame counterparts and the gyroscope measurements in an extended Kalman filter to estimate a unit quaternion and gyroscope bias vector [18]. The overall architecture of this state estimation method is shown in Figure 7. In Figure 7, the inputs TLE, RTC, SS, MAG, and GYR represent a Two-Line Element, a Real Time Clock, the digital Sun sensor, the magnetometer(s), and the gyroscope(s), while the outputs r, v, q, and ω represent the position, velocity, attitude quaternion, and body rates of the spacecraft to be estimated.

The reaction wheel assembly (RWA) is a momentum exchange device that (in our case) allows for relatively large torque commands relative to the inertia of the spacecraft. In general, RWAs allow for high-precision attitude control about all three body axes. However, the RWA can only create an internal torque that exchanges momentum with the spacecraft bus in order to rotate the spacecraft. A consequence of this is that the wheels can become saturated (i.e. they reach their maximum RPM) as external torques in the space environment add momentum to the closed spacecraft/wheel system. In order to minimize the power consumption of the RWA and avoid saturation (which can lead to a loss of control), it is necessary to reduce the angular momentum stored in the RWA periodically. Decreasing the speed of the wheels in the RWA will have the opposite rotational effect on the spacecraft bus, and it is therefore necessary to have another means of torquing the spacecraft. We use the magnetorquers placed on all three body axes to generate a torque on the spacecraft that is simultaneously nulled by the spinning-down of the reaction wheels [19].

Having four reaction wheels provides redundancy and the ability to satisfy a three-dimensional torque request even in the event of an individual wheel failure. Strictly speaking, it is not mandatory to have magnetorquers on all three axes of

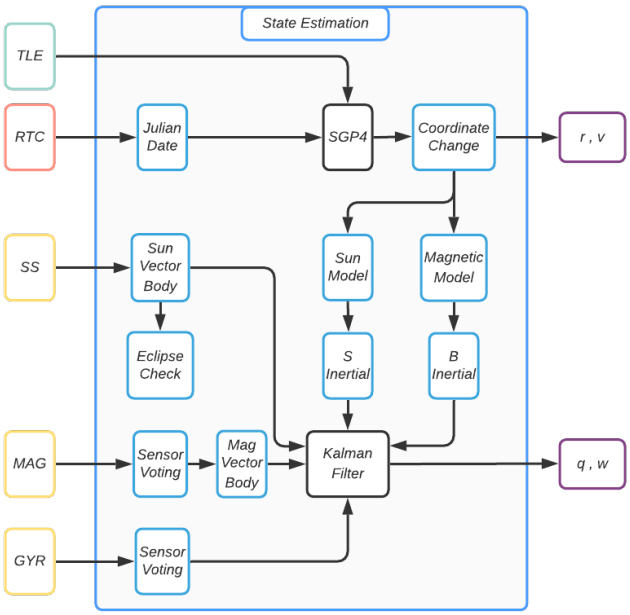


Figure 7: Representative Attitude Determination/Estimation scheme.

the spacecraft; two axes of magnetorquers can provide three axes of external torque on average, and would therefore be sufficient to unload momentum from a RWA. In our case, the availability of magnetorquers built into COTS solar panels permits some free redundancy with no additional risk. The magnetic actuation system is therefore two-fault tolerant, whereas we consider the loss of two reaction wheels to be a condition for mission failure.

A detailed simulation of the space environment, spacecraft dynamics, sensors, actuators, and flight software has been developed based on [20]. Everything outside of the flight software is considered to be the "truth" model of the simulation, and we are currently validating the hardware models against the actual performance of the selected components. On orbit, the dynamic and environmental models will be replaced by the spacecraft and space environment themselves, and the sensor/actuator models will be replaced by the actual hardware. The simulation allows us to validate the performance of the GNC flight software against representative inputs. We developed this simulation in Matlab/Simulink leveraging existing work from the University of Washington [20], and we use the autocoding feature to generate C code that can be run on our flight computer.

Command and Data Handling

A high-performance flight computer is required to meet the needs of SOC-i's mission. The computer must execute flight software, manage a number of subsystems, and interface with a wide variety of sensors simultaneously. The fundamental design decisions for the flight computer are the hardware and firmware selection. The flight computer's hardware consists of the microcontroller, memory and data storage devices, peripherals, and other electrical circuit components. The firmware must run flight software, communicate with other subsystems, and schedule tasks for all operations. This functionality can in theory be built with a bare-metal embedded design, a real-time operating system (RTOS) (e.g., FreeRTOS), or a general purpose operating system (e.g., a Linux distribution).

Hardware— The hardware requirements for the OnBoard Computer (OBC) were informed primarily by the processing needs of the GNC subsystem and the SOAR payload. The flight computer's microprocessor selection was driven by a trade study of COTS embedded development boards. NXP offers high-performance development boards with detailed documentation and support. Basing the OBC design on a development board – a commercially proven design – minimizes the risk of component integration issues. The schematics and bill of materials of NXP development boards are freely available.

Initially, the OBC design was based on the NXP i.MX RT1050 EVK development board. However, lessons were learned during PCB layout because the microprocessor's package is only available as ball grid array (BGA). This small form factor created PCB routing challenges and this package type would have substantially increased board fabrication costs. The microprocessor component selection had to be revisited, along with which development board to use as a reference design. Fortunately, the microprocessor in the NXP i.MX RT1020 EVK development board is available in a LQFP chip package. This form factor allows for cleaner PCB routing and reduced board fabrication costs. Select integrated circuit chips from the i.MX RT1020 EVK development board were incorporated into the OBC design to meet mission requirements. The OBC's microcontroller (MIMXRT1021DAG5A) is based on the ARM Cortex-M7 Platform and has a maximum clock frequency of 500 MHz. The OBC has an SDRAM (256 Mbit) device for memory and a QSPI Flash (64 Mbit) device for storage. An SD card is also included for additional data storage. A 20-pin JTAG connector is used to program and debug the OBC's microcontroller. The PC104 standard stack-through header provides power to the OBC and a data link to other subsystem's boards. A DC power connector will be used for post-assembly trickle charging of the battery. The 20-pin JTAG and DC power connectors will be accessible via the side panel cutout. The CDH subsystem is responsible for managing the spacecraft's knowledge of time, and this is done by using a real time clock (RTC) that is external to the OBC's circuitry.

A secondary board was designed to integrate the magnetometer and gyroscope sensors, the RTC circuit, and several cabling connectors – a board that we have named OB1. The external RTC is powered by an independent coin cell battery to maintain time-keeping even while the main bus is not powered during integration, launch, and deployment, and this circuitry resides on the OB1 board. The OB1 board interfaces with the OBC via the PC104 headers. Additionally, the OB1 includes four two-pin DF13 connectors for the photodiodes and its analog-to-digital conversion circuitry. A 10-pin Hirose FPC connector is used to connect to the digital sun sensor and receive its data. Four seven-pin Molex Pico-Lock connectors are used to drive the four reaction wheels. A final connector provides power and data to the CIA board on the bottom of the spacecraft, and uses a 10-pin Molex Pico-Lock connector.

Software—Energy consumption is an important concern when choosing a software system architecture. The main goal, therefore, is to choose the most power efficient option that meets our requirements: A bare-metal software implementation that runs a single thread of operations, only utilizing interrupt routines on hardware/software events; RTOS enables multi-threaded operation, imitating parallel operation by the utilization of context switching; Linux has the ability

to spawn multiple processes as well as threads, and switch between them. The Linux option provides flexibility to run multiple programs with different processes. It also enables utilization of a diverse set of tools to use concurrently for the overall functionality. However, this option lacks real-time operation capabilities. This is an important property in the case of a satellite system where timely responses to events is a top priority. On the other end of the spectrum, the bare-metal option lacked the inherent ability to use multiple tasks, which is useful for a system with many subsystems, all of which run multiple operations. Disregarding different functionalities and abilities they offer, a bare-metal embedded implementation has the smallest overhead, while Linux has the largest.

RTOS provides the ability to assign tasks to every operation with different priorities, tools to use for inter-thread communications, and mechanisms to control shared resource usage. Taking these into account, RTOS proved to be the optimum option for SOC-i's flight computer. The NXP microcontroller chip was also chosen because it officially supports FreeRTOS. Having extensive documentation, a large community, and a proven record of successful use in industry, FreeRTOS seems to be a suitable option for our purposes.

A key set of decisions was choosing the schemes to communicate with various sensors, actuators, and other modules. Most embedded devices, including the sensors we chose to incorporate, and other microcontrollers running different subsystem boards, use and support some common communication protocols. The protocols supported by most development boards are I2C, SPI, and UART. CAN is another common protocol, but it is designed to be distance-tolerant, an important feature for the automotive industry where it is typically implemented. It also introduces some advantages when tens of modules are sharing a line. Because our system has only a few modules, CAN does not provide any significant advantages, notwithstanding the lack of support for almost all of our sensors. I2C, SPI, and UART, all being serial communication protocols, differ on whether they are synchronous (using a clock) or asynchronous, supporting multiple masters/slaves on a communication line, performance in terms of communication rate and payload overhead, and error tolerance. Our software is designed to support each of these three protocols; subsystem-specific decisions among these options were made considering the associated trade-offs and were mostly influenced by the ability of specific components (e.g., our selected magnetometers require an I2C interface).

Structures, Thermal, and Configuration

Chassis Design— SOC-i 's chassis is custom designed to house all hardware and circuit boards, allow easy access to internal components, and survive launch loads. Five parts assemble to form the chassis: the camera end plate (-Z face), the antenna end plate (+Z face), the rib, and two wall pieces comprising the rails. Each chassis part is made from 7075 alloy aluminum, which is chosen for its similar thermal expansion coefficient to that of the CubeSat deployer. Additionally, each part will be hard anodized to prevent cold-welding where the rails contact the deployer. The chassis is depicted in Figure 8.

The two end plates comprise the $\pm$Z faces of the satellite. Holes in the end plates are used to mount the rail walls and the solar panels. On the interior, each end plate has four holes patterned to the PC/104 standard, which are used to attach the board stack and camera assembly via hexagonal

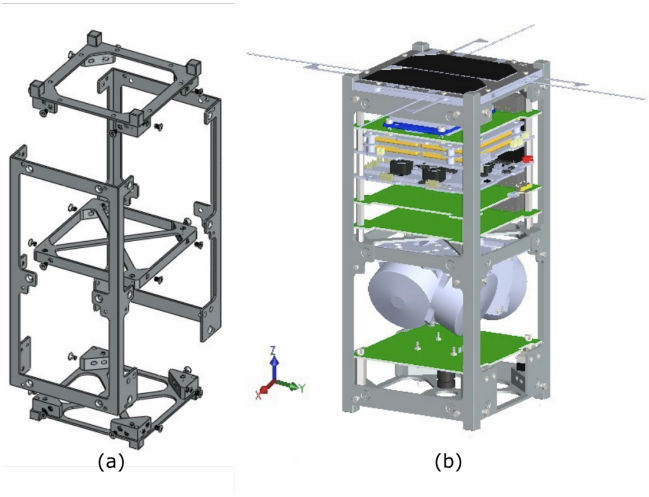


Figure 8: (a) SOC-i's chassis is comprised of five individually machined parts. (b) The internal configuration of components.

standoffs. All mounting holes in the end plates and rib are tapped and fitted with helicoils to prevent damage to threads in the chassis itself. The two rail walls join to form the $\pm$X and $\pm$Y faces of the satellite, each of which mounts a 2U-sized (211.5x82.6 mm) solar panel. Additionally, three plunger-type deployment switches mount to tabs on the rails near the camera end plate. These switches are depressed by contact with adjacent satellites within the deployer (or the deployer itself), which prohibits SOC-i's onboard battery from providing power while it is contained in the deployer. The rib is mounted inside the rail walls near the midplane of the chassis parallel to the end plates in order to stiffen the chassis. In addition, the rib provides the anchor point for the main board stack above and the RWA below.

The biggest consideration in designing the chassis was for it to readily allow access to the internal hardware during assembly, testing, and leading up to the final integration into the launch deployer. This was important because after reviewing existing CubeSat chassis designs, it was clear that many problems during assembly and integration could be solved by having a skeletonized chassis. It also provides ease of disassembly in case this is required during testing and fit checks. The chassis provides large access openings to SOC-i's internals even when fully assembled. Also, the board stack can be removed, worked on outside the satellite, and then assembled back into the chassis by simply removing a single end plate. These features serve to facilitate proper integration of SOC-i 's boards, hardware, and harnesses and reduce risk of damage to sensitive hardware during testing and assembly.

Coordinate System—The camera end plate defines the -Z face of the CubeSat and the solar panel housing the remove before flight (RBF) pin defines the +Y face. This alignment meets NanoRacks' requirement that any plunger-type deployment switches be near the CubeSat's -Z rail ends. Our coordinate system also aligns the CubeSat's +Y axis with the +Y axis of the NanoRacks CubeSat Deployer (NRCSD), which allows for access to both the RBF pin and a cutout for OBC programming and battery trickle charging through the access panels on the deployer.

Board Configuration—The configuration of SOC-i's boards (Figure 8(b)) was driven by each component's required proximity to other hardware in order to reduce the lengths and masses of harnesses. Boards in the main stack are each separated by four hexagonal standoffs and connected to adjacent boards via two 52 pin PC/104 headers. The COM board, to which the radio mounts, is placed closest to the antenna and is the topmost (+Z) board in the stack. The battery and EPS are placed together, as are the OBC and OB1 boards. This organization facilitates interfacing between paired boards. Cabling and trickle-charge connectors on the OBC center on a cutout in the +Y solar panel, providing access while SOC-i is fully integrated. The OB1 is located below the OBC at the bottom of the stack so that it can route easily to the reaction wheels, which are oriented with connectors facing in the +Z direction for this purpose. The CIA board, to which the camera mounts, is the only board separate from the main board stack; it attaches to the camera end plate and is the bottom-most (-Z) board. This provides the camera with an unobstructed view through the viewport in the camera end plate.

Mass Properties—The full chassis by itself has a mass of 299.7 g while SOC-i 's total mass is 2750.7 g; these include an average mass growth factor of 1.09. This leaves a margin of 849.3 g under NanoRacks' 3.60 kg mass limit and also puts it within typical mass allowances for P-POD deployment.

Structural Analysis—Structural analysis for SOC-i was used to drive the chassis design and to determine the spacecraft's fundamental frequency and significant resonant modes. The analysis was conducted in ANSYS using a defeatured model of the CubeSat. Larger components including the battery, radio, and reaction wheels were represented by point masses attached at the center of mass of their respective boards, while the remaining boards were represented simply as plates with a mass equivalent to that of the board and its components. Cases in the analysis used an acceleration of 16 g applied singly to each axis and also simultaneously to multiple axes (XZ as well as XYZ).

This analysis allowed the team to make a significant design decision on the necessity of the rib plate. To prevent amplification of loads from the launch environment, we aimed for the satellite to have a first mode above 100 Hz. Our analysis indicated that the first mode of the satellite occurs at 159 Hz in the case with the rib and at 103 Hz in the case without the rib. Without the rib, this gives the satellite a factor of safety of 1.03 above the necessary frequency. Additionally, analysis showed that maximum board deformation was reduced by 66% when including the rib. However, the rib configuration had a smaller factor of safety against yielding under maximum principal stress: 2.64 for the rib case as opposed to 8.41 without the rib. The maximum stress in both cases occurred near the holes in the boards, stemming from a difference between the attachment constraints in the two cases. Mounting the reaction wheel plate directly to the rib limits its degrees of freedom, leading to stress concentrations near the holes. On the other hand, in the case without the rib, bending of the board distributes force across a larger area.

Although the rib design has a higher maximum stress, it still provides sufficient safety factor against yielding (2.64). Moreover, this design was selected because it significantly reduces the maximum board deformation and has a more favorable fundamental frequency. Structural analysis results for SOC-i will be validated against data collected from vibration testing of an engineering prototype, discussed in Section 4.

Thermal Analysis—Unlike other similarly sized CubeSats that have issues with overheating, our thermal analysis for SOC-i indicates that the minimum allowable flight temperatures are more of a concern. This issue primarily stems from SOC-i having body mounted solar panels on five of its six faces (the camera endplate being the exception), which means that power margins are tighter, its internal components dissipate less heat, and there is a large high-emissivity surface area that radiates heat away. By contrast, CubeSats that have deployable solar panels can provide more power for heat dissipating components and allow for a tailored optical coating of the surfaces not nominally exposed to the sun to maintain an optimal orbit average temperature.

The most restrictive temperature range is that of the battery, and this is the main driver of SOC-i 's thermal control design. The extreme cold case potentially occurs during low power mode if the -Z face is positioned with a range of small incidence angles to the sun for an extended amount of time (i.e., the camera points towards the sun and the spacecraft is inertially fixed). GNC software should prohibit this during any nominal operation *after* the commission phase, but redundancy via passive thermal control is desirable. Approaches to resolve this issue are the focus of this section.

For CubeSats with body-mounted solar panels on most major faces, there is a limited amount of surface area to customize the optical coating for passive thermal control. Both the hard-anodized aluminum rails and solar panels have high emissivities that dissipate heat and cool the CubeSat. Thus, the only area available to apply optical coatings is the exposed printed circuit board mask surrounding the solar panels. Copper tape was chosen to cover this area due to its low emissivity (to limit radiative heat loss) and relatively high absorptivity ($\alpha = 0.32$; $\varepsilon = 0.02$). It is projected to increase the overall CubeSat temperature by 12 °C during the cold case at the expense of slightly reducing the already large margins on the hot case. This was determined by creating a single node MATLAB model that calculated the environmental heat fluxes (solar, albedo, and Earth IR) and heat radiated in conjunction with the internal heat dissipation to determine an approximation for the average CubeSat temperature over an orbit. By varying the optical properties of the surface areas of the CubeSat, different coatings can significantly affect the average temperature, as seen in Figure 9.

The current EPS hardware, supplied by DHV Technology, has battery heaters to assist in thermal management, but our narrow power margin requires limiting their use. Hand calculations showed that thermal isolation of the battery could be achieved by replacing the aluminum standoffs contacting the battery board with a low conductivity titanium rod and G-10 outer sheath. This was combined with minimizing radiative losses by covering the battery in a layer of Kapton tape for electrical insulation and then a layer of aluminum tape ($\varepsilon = 0.03$). This insulation scheme was chosen over traditional multi-layer insulation (MLI) due to the small size of the battery; MLI suffers from significant degradation near seams due to conduction between the layers [21, Figure 5.7]

Temperature margins were calculated in an ANSYS transient thermal model for a variety of cases based on a similar defeatured CAD model as mentioned for the structural analysis. Figure 10 shows the margins calculated for both hot and cold temperature limits. Healthy margins on the hot case indicate no components are at risk of overheating while a very conservative analysis on the cold case indicates the previously described thermal control measures should be sufficient with

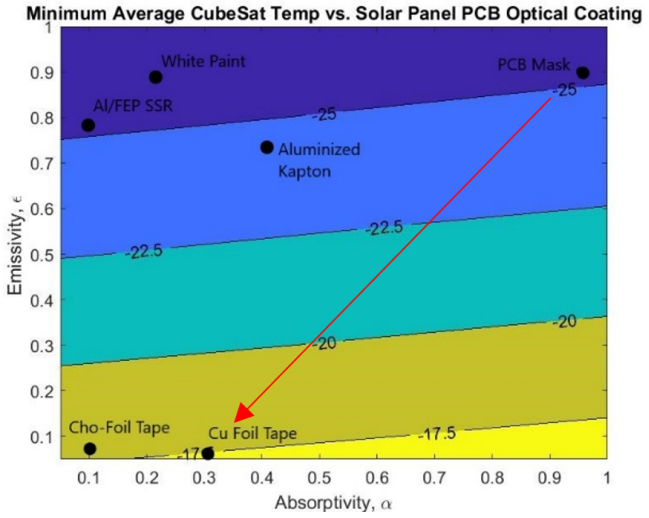


Figure 9: Changing the optical properties of the ±X and ±Y solar panels improves the worst-case cold temperatures from -25 to -17.5 °C. A few typical coatings are shown with copper or aluminum tape being ideal for preventing the CubeSat from getting too cold.

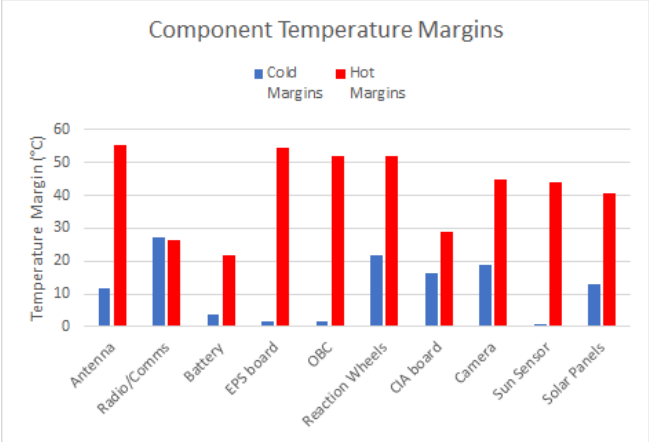


Figure 10: Orbit-average temperature margins for the worst-case hot scenario of SOC-i indicate little risk of overheating. The worst-case cold scenario reveals some components with small margin, however, based on the conservative nature of the model and our risk posture, we deemed this risk to be acceptable.

a non-zero margin. Based on the conservative nature of the cold case model and our overall mission risk posture, these small margins were deemed acceptable.

Imaging System

The imaging system used to perform Earth-imaging serves as a secondary payload for the SOC-i mission. With SOAR already the main technology demonstration payload, we designed the imaging payload to utilize easily available parts so that it has high educational and outreach value. An inexpensive imager is also an ideal first use case of SOAR, as it is a low-risk representation of a sensitive optical sensor to be protected by constrained attitude control. The uCam-III from 4D Systems was chosen as the onboard camera due to its low cost, low weight, and flight heritage. A 2013 study compared commercial off the shelf cameras and rated the uCam-II highly against competition [22]. Following the study's findings, the newest version of uCam was chosen.

The camera resides on the Capstone, Imaging, and Astrionics (CIA) board on the -Z face of the satellite (so named because of the components on the board and because a senior capstone project contributed significantly to the board design). An Arduino Pro Mini mounted to the CIA board controls the uCam-III and stores images on a microSD card dedicated to this purpose. The Arduino was chosen to simplify communications between the flight computer and camera, allowing the imaging system to operate as a standalone system. For example, the flight computer can send a single command to the Arduino to take a picture, and the Arduino handles the sequence of commands and responses required by the uCam-III to actually perform this operation. This simplification extends to image data retrieval and adjustments to camera settings. An auxiliary benefit of this design is that the entire imaging system was able to be prototyped, built, and tested independently from the OBC.

4. Development Progress

Guidance, Navigation, and Control

The GNC system employs an array of hardware to achieve its mission objectives as discussed in 3. In order to provide accurate sensor data and actuator responses to the flight software, each peripheral device needs to be characterized and well understood. For successful system integration, there are two main areas of focus: operational behaviors and software interfaces. The operational behavior covers the ways in which a device carries out its primary task, such as a gyroscopes's bias stability or an actuator's transient response. These can be characterized through bench tests that simulate the expected on-orbit range of operation, or by delving into the device's design and basic input/output response. The software interface is the method by which the device communicates with the onboard computer. All GNC peripherals use some form of serial communication, namely SPI, I^2C, and UART. Each device requires its own hand-tailored driver code on the OBC to query data, send commands, and detect errors. Understanding each software interface is central to writing those drivers, and doing so in a way that optimizes device performance and minimizes communication errors.

To develop these focus areas for each sensor and actuator, a common cycle was followed. Each of the five peripherals was tackled by 1-2 people working together, and work started by developing a basic interface with the target device by using an embedded device as a stand in for the OBC (e.g., an arduino connected to a sensor). Using breakout boards and pre-built libraries, teams had an easy approach to study the behavior of the device while slowly stripping down to the base level of the software interface. Over the course of months, this approach progressed each device from simple bench top interface toward flight representative operation.

RWA Testing—The first peripheral subjected to this development cycle was the RWA, procured from Lithuania-based NanoAvionics. We needed to perform acceptance testing as soon as the device was delivered, so development began many months ahead of time. The objective of this testing was to evaluate all facets of the RWA, covering software, electrical, hardware, and motor performance. The testing driver code contained all elements necessary for software interface, but also included keyboard interaction, automated

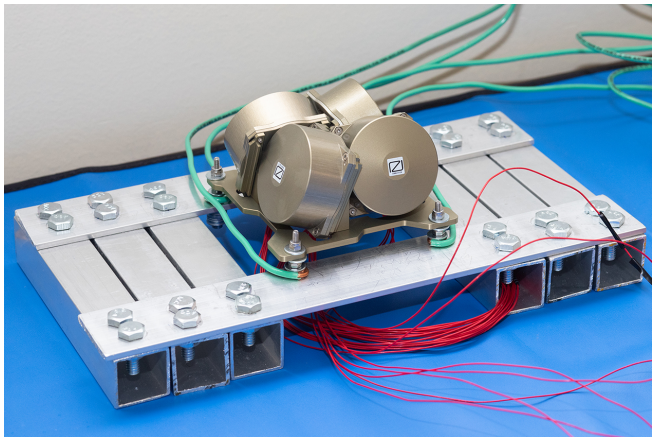


Figure 11: The reaction wheel assembly mounted on its test stand, with wiring leading off to the master board and an ESD protection circuit.

test routines, and test data logging, written in Python and run on a Raspberry Pi (RPi) device. Without the RWA in hand, the best way to prepare this driver code was to create a dummy version of the RWA using an Arduino Uno board. The Arduino was programmed to respond to commands in identical fashion to the RWA, allowing the RPi to send and receive SPI transmissions as if it were connected to the real device. Once the RWA was delivered in September 2020, the pre-written tests were successfully executed in a matter of days.

The RWA was the first piece of high-value flight hardware tested, which presented another challenge. We had little practical knowledge of laboratory setup or electrostatic discharge (ESD) protection for sensitive devices, so it was a rapid learning process in the weeks leading up to delivery. Without access to machine shop or lab spaces due to a global pandemic, plans for a machined test base were abandoned in favor of a lightweight aluminum stand built from hardware store materials. Additionally, a variety of procedures were written to ensure ESD protective practices were followed at all times by those handling the device. Mounted on its stand and connected to the RPi and a bench power supply (see Figure 11), the RWA was fully operational and protected from ESD damage. Guidelines written for this testing will serve as the basis for a general flight hardware safety doctrine used across all similar SOC-i projects.

Sensor Integration—Concerted work began on the gyroscope and magnetometer characterization in mid-2020. Due to the low cost of these sensors, they were readily available from electronics suppliers. Teams took advantage of development-friendly Adafruit breakout boards and libraries to begin developing calibration procedures for both sensors. Thoroughly determined calibration coefficients are needed for the flight controller to correctly process sensor inputs during flight. The calibration procedures were used as a means to develop basic interface code for each sensor, after which more efficient driver code could be developed that will run on the OBC during the mission.

The magnetometer selected is the STMicroelectronics LSM303DLHC sensor chip, which comes packaged on an Adafruit breakout board for breadboard development. With a ready-made sensor library, the team was able to quickly gather measurements from the sensor using an Arduino. Using the sensor's dataset, we wrote a program in C++ to perform a simple calibration to account for hard and soft iron distortions. With the simple calibration task complete, we proceeded onto a more sophisticated calibration method that corrected for hard and soft iron magnetic interference, as well as instrumentation errors unique to individual sensors. By making slight alterations to an already-written Matlab sphere-fitting algorithm and inputting the magnetometer's uncorrected data into said program, an automated calibration procedure was refined. At the time of writing, the magnetometer team has begun to write the flight driver code for the magnetometer to communicate via I^2C with the OBC.

The gyroscope selected for SOC-i is the NXP FXAS21002 sensor chip, which comes built onto another Adafruit breakout board. Testing and calibration began alongside the magnetometer. Testing evolved from a simple Arduino connection to a comprehensive, long form static sensor routine. The preliminary code was developed in C++, where we could easily configure the gyroscope's time, full scale range, output data rate, and bandwidth variables. We tested different combinations of these variables, looking for ideal values that would reduce static noise and yield a set of desirable coefficients from an Allan variance test, while keeping in mind the intended sample time of the GNC control system. The Allan variance tests were implemented in a separate Matlab script using data generated from the (static) sensor. From this test, we were able to compute coefficients for rate random walk and angle random walk, which can be visualized to compare different sensor configurations. At the time of writing, the gyroscope team is refining the Allan variance tests and beginning to write the flight driver code for the gyroscope to communicate via I^2C with the OBC.

The final sensors are the digital sun sensor (SolarMEMS NanoSSOC-D60) and surface photodiodes (SLCD-61N8). Both components are integrated into COTS solar panels, so peripheral development again began without hardware in hand. At the time of writing, we have begun developing simulated sensors using an embedded device to replicate the true sensor's input/output behavior so that we can begin to prototype basic interface code. The team has recently acquired a solar simulator, and we are designing test plans using this simulator to both validate the interface code and characterize the sensors once they have been procured.

FlatSat System Design—The common goal for all peripheral work is full integration into a hardware-in-the-loop (HIL) testbed, commonly referred to as FlatSat. FlatSat will take the form of several modules laid out on an optical bench, each representing a different subsystem or element of the satellite. Some will be flight hardware, such as the RWA or the imaging board, while others will be abstractions of the mission function, such as a power supply and circuitry emulating the EPS. FlatSat will be the first time sensors and actuators are connected to the OBC, creating the first marriage of GNC's flight software and its hardware components. To properly run a HIL test, the controller needs to tricked into believing that it is in the intended orbital environment by receiving appropriately-simulated sensor inputs. Based on these, the software will compute commands for its actuators, which can be routed to the actual hardware. Because the actual sensors will not return appropriate orbital values in a lab setting, we use an embedded device connected to the GNC environmental simulation to serve as an intermediary between the OBC. This intermediary device allows the OBC to interact with a "sensor" using the same input/output signals and serial communication.

State Estimation—The attitude knowledge error of the GNC system is to remain below 3.5°. To properly control the orientation of the spacecraft, we also require an estimate of the chassis's angular velocity. Therefore, it is necessary to employ an algorithm that filters raw sensor data, then fuses noisy sensor measurements into attitude and gyroscope bias estimates. The Multiplicative Extended Kalman Filter (MEKF) algorithm achieves this objective [18]. The MEKF provides accurate state estimates by calculating the best weighted average between sensor measurements and predictions of the state at the current time step. The predictions of the state come from a known dynamic model of how the state evolves over time (i.e., (2)). The combination of the measurements and the predictions yields estimates of the state that are more accurate than either sensor measurements or predictions alone. We use a variant of the MEKF whereby we maintain an estimate of the square root of the state covariance matrix. The square root is achieved via a Cholesky decomposition, and ensures that the covariance matrix remains positive definite for all times. This positive definiteness is a necessary condition for the numerical stability of the MEKF.

In the absence of adequate angular rate measurements, the MEKF cannot accurately predict the attitude or gyroscope bias vector. We have included three gyroscopes to mitigate the likelihood of this occurring, but nonetheless a secondary estimation scheme was designed to address this edge case. The chosen scheme for such redundancy is the Triad method [23]. The Triad method still facilitates attitude estimates derived from the (assumed) operational sun sensor and magnetometer when angular rate data is unavailable, albeit with a reduced accuracy. A secondary benefit to the Triad method is that it allows for an accurate initial guess of the spacecraft attitude, which is used to initialize the MEKF. We have observed that this initial guess significantly increases the ability of the MEKF to converge to an attitude estimate quickly.

An ancillary benefit of the MEKF is its inherent ability to continue providing short-term attitude estimates even in the presence of sun sensor or magnetometer failures. In the case where one of these sensors fails to provide valid measurements (e.g., during eclipse) the MEKF bypasses the usual incorporation of the sensor measurements, and instead provides state estimates based solely on the spacecraft attitude dynamic model. The gyroscope's data continue to provide angular rate information, and so we can continue to estimate the gyroscope's bias vector, thereby completing the full filter state estimate. We have mitigated the likelihood of this occurring during non-eclipse portions of the orbit by incorporating three magnetometers (each oriented differently with respect to the body frame).

Electrical Power System

We have chosen a COTS EPS in lieu of designing a system in-house due to the significant reduction in risk it provides to the mission. A system that includes a main EPS motherboard, a battery board, and solar panels is currently under development by DHV Technology and is being customized to our mission's needs. The SOC-i team is responsible for working with DHV during this development process to ensure that the delivered product meets all mission requirements, as well as integrating the EPS with the other subsystems.

While the DHV EPS will be used on the mission, the SOC-i EPS team is currently developing a mock version to be used for FlatSat testing. The purpose of this is to test the operation of other satellite systems without needing to integrate the solar panels and battery, and without risking any damage to the flight EPS hardware. The FlatSat prototype under development uses an external 8 V DC power supply in place of a battery and uses DC-DC converters to generate 3.3V and 5V voltage busses, mimicking the power conditioning modules on the actual EPS. Additionally, an MSP 430 microcontroller from Texas Instruments will be used to receive commands from the onboard computer and control power flow through multiple power switch circuits, which will operate in the same way as the power distribution modules on the actual EPS. These will be used to power on/off individual subsystems as needed to simulate the mission's ConOps. The mock EPS board will be configured following PC/104 standards in order to integrate with the other systems using the same interface as the flight hardware.

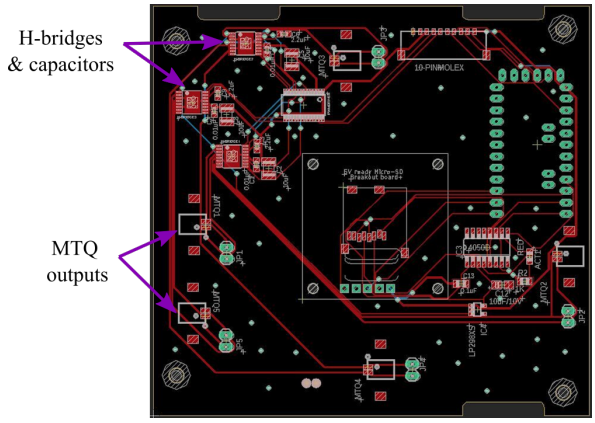


Figure 12: The board layout of CIA is designed for heat distribution, signal integrity, and ease of connection to the magnetorquers embedded in the solar panels.

Imaging and Astrionics

For the camera to image objects without obstruction from the spacecraft, it is protruding form the lowest (-Z) face of SOC-i. A circuit board was designed based on the PC/104 standard (but without the usual header pins) to house the imaging hardware for mounting at the viewport in the -Z face. The CIA subsystem receives commands from the OBC and power from the electrical power system through a cabled connection. The CIAB subsystem also contains driving circuitry for the magnetorquer control system because the location of the magnetorquer connectors in the solar panels is near the bottom of the chassis. Magnetorquer dipole commands are received from GNC flight software running on the OBC, and then translated to a pulse-width modulated (PWM) signal with adjustable direction, duty cycle, and current. The low-level software for this conversion has been developed and is currently undergoing unit testing before integration with the main flight software.

Figure 12 shows the current design of the CIA board. In the upper left side, there are H-bridges and the PWM driver arranged in a quadrilateral position to help create an even distribution of heat. The capacitors are also located next to the H-bridges to help keep the time constant as close to the specifications of the H-bridge as possible. The connectors to the magnetorquers are also located on the board near their respective faces, i.e. those interfaced with the $\pm$X panels are located on the left and right side of the board and those responsible for the $\pm$Y panels are on the top and bottom to minimize cable length.

Imaging Payload—The Arduino Pro Mini, micro-SD breakout board and uCam-III are all co-located on the CIA board. Data and power are routed to the imaging system by a cable that connects to the OB1 board in the main board stack. The Arduino communicates with the flight computer using UART communication, and there are ten possible commands that the flight computer will send. These commands check the health of a component, take a picture and save it to the micro-SD card, retrieve a picture/thumbnail from the micro-SD card, get the number of bytes in a picture/thumbnail, change the uCam-III's contrast, exposure or brightness, or set a sleep time. As an example command flow, if the flight computer sends a command to the Arduino to take a picture, then the Arduino will command the uCam-III to take the picture, and then save the image data on the micro-SD card and report back that the operation was successful. Later, if the flight computer sends a command to retrieve the image data, the Arduino accesses the corresponding image on the micro-SD card and begins to send the data to the OBC.

One of our biggest challenges has been to develop the system in way that it reliably interacts with the flight computer. To develop this interface in the absence of the OBC, we are testing with a second Arduino that acts as the flight computer to send commands to the imaging system. This allows us to probe the imaging system and test whether it responded the way that it was designed to. The second Arduino is designed to automatically test all possible commands that system can receive. It begins by testing the health of individual components to see if all of them are responding. Then, we take test pictures and retrieve them from the SD card to ensure that the Arduino is successfully interfacing with the camera and SD card. We take a picture at each possible brightness, exposure, and contrast setting and save them to the SD card. This way we are able to test all individual components and possible commands to ensure that the system is reliably communicating with the OBC.

Another challenge we faced was to find the best possible configuration for the camera to provide good images from space. We are currently developing a test case that runs through all possible image configurations, and images an Earth-like object with a black background. The aforementioned solar simulator will be leveraged to conduct image testing in a representative lighting environment. We then plan to manually choose the configuration that provides the best image quality.

Communications

We are developing a COM flight board (Fig. 13) that contains the XDL MicroUHF transceiver and interfaces with EnduroSat UHF antenna. The main purpose of the board is to receive and transmit data from ground control using the antenna and transmit the information to the flight computer. The information can then be relayed to the rest of the satellite. The XDL Micro interfaces to the antenna via an MCX-to-SMA coaxial cable, and it interfaces to the flight computer via a UART serial interface. In addition to mounting the radio, the COM board acts as a wiring interface, enabling two EnduroSat solar panel connections to connect to the EPS with reduced cabling. The solar panels are located on the +Z face and the connection to the solar panels also contains the controls for the magnetorquer that are received via a CIA board.

When choosing the orientation of the radio during development, an important factor was the bend radius of the coax cable between the radio and antenna board. The placement

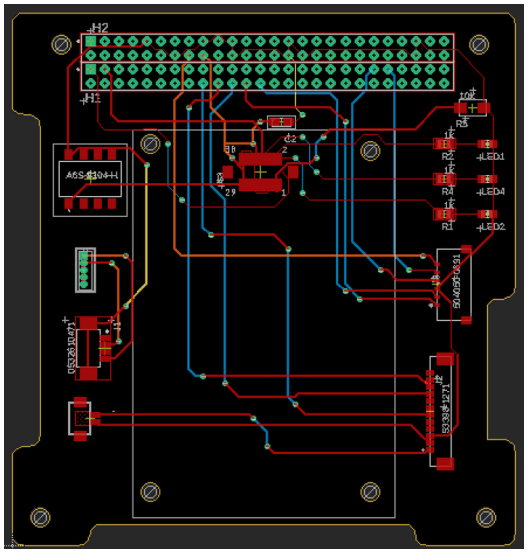


Figure 13: The latest version of the COM board contains the XDL Micro radio and several connector interfaces.

that has the optimal bend radius is with the radio coax port pointing towards the bottom of the PC104 header with the radio as close as possible to the +X edge. To simplify routing, both solar panel connections have been placed as close to the EPS connection as possible which is on the -Y face.

The antenna is mounted on the +Z face of the satellite to facilitate deployment and to achieve an optimal RF gain pattern. The antenna is deployed using a burn wire mechanism that is activated by the flight computer using one of three redundant mechanisms: a digital I2C protocol command and two independent voltage signals.

Flight Computer

The schematic design and PCB layout of the OBC and OB1 boards are being finalized, and will be fabricated by external companies. Autodesk's EAGLE software is being used for circuit schematic capture and PCB layout. An important lesson learned came from trying to implement all required flight computer computer onto a single board. This configuration produced an immensely dense board, which led to unnecessarily complex PCB routing. Continuing with that approach would have led to signal integrity and noise susceptibility issues. In an effort to reduce these effects and facilitate debugging and testing, the flight computer hardware was split into two boards (OBC, OB1).

RTOS development increases modularity of system functions. This requires assigning tasks for every unit function or a group of related functions. These tasks are instantiated with respective priorities, thereby enabling correct prioritization of functionalities. A proper configuration of tasks enables seamless transition between different tasks, technically known as context switching. While prioritization of GNC tasks over others is desirable, this might lead to starvation of lower priority tasks (e.g., downlinking an image to the ground station). This issue can be resolved by the use of carefully picked priorities and widely used techniques like aging of tasks.

Another requirement for an RTOS system is the use of shared resources. In the case of SOC-i, these resources might

reside in software (e.g. an image stored in memory) or as a hardware component (any sensor or actuator). FreeRTOS has the mutex, semaphore, and their variations to facilitate this. These tools lock a resource while it is being used, so that only one or a limited number of tasks can access that resource. In our early tests, we were able to initialize and use multiple UART ports for communication. With the help of their assigned buffers and the use of mutexes, concurrent communication with no information loss was achieved.

We have developed drivers for UART, and currently in the process of testing out SPI and I2C. The first step of development for these protocols is to understand the needs of the subsystem to be communicated with. While the initial tests used a Raspberry Pi or an Arduino on the other side of the communication for testing purposes, all sensors/actuators/modules we interface to will have baud rate, parity, frequency limitations among others, or will operate on different number of lines. Configuration of these parameters, along with selecting the relevant function for individual pins will enable communication with all different modules.

An important step necessary for communicating with other subsystems and their components is to structure the communication in software. This will involve creating the correct header for the payload to be sent, include the necessary signals to select and switch states of communication, etc. This, in turn, requires inspecting the datasheets for all modules. As an example to this and as a model for the future ones, we are developing the UART communication module for the radio. Building on the simple send, receive functionalities of basic UART, it is implementing header construction, error checking functions among others to maintain the communication in a higher level throughout the system.

Structural Design and Testing

SWaP Model—The objective of the Size Weight and Power (SWaP) project is to create an engineering prototype of SOC-i that replicates the size, mass, and thermal properties of the actual satellite so that it may be put through initial vibration and thermal testing without damaging the flight model. The information gathered from these tests will validate structural and thermal modeling and inform further development of SOC-i.

The model uses a replica of the aluminum chassis, individual circuit boards cut from copper clad PCB with added aluminum weights, as well as custom machined parts for more eccentric geometries. Actual components are used whenever it is financially feasible to do so, such as the CIA board and camera. Heat dissipations are accounted for by attaching patch-style resistive heaters to each board and running the wires out to a breadboard for power and voltage management. Thermocouples are placed at locations of interest to collect data on the heat transfer through the system as a whole.

The power and mass distributions for each component have been collected from each subsystem and documented. Power is managed by taking the given power dissipation of each board, the resistance of each patch heater, and calculating the voltage that needs to be delivered to the heaters. An aluminum weight is used to match the mass of each component if the representative PCB does not meet the mass of the original component.

Students used CNC mills at the University of Washington Mechanical Engineering machine shop to machine the prototype chassis for the SWaP unit. This project came with some unique challenges, particularly with meeting tight tolerance specifications on thin-walled parts. Students addressed this challenge by creating custom aluminum fixtures for the chassis components that enabled parts to be oriented and machined from all sides while fixed rigidly to the mill's table. To reduce chatter induced by resonance between the cutter and the rail walls during machining, students developed a sacrificial 3D printed fixture to encapsulate the thin part and allow for accurate thin-wall machining.

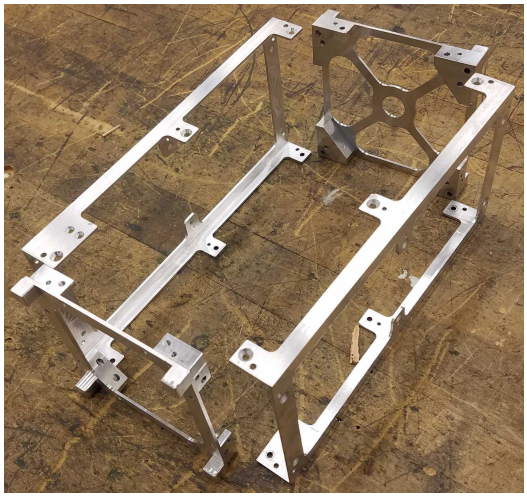


Figure 14: Prototype rail walls and endplates have been machined by students in the UW Mechanical Engineering machine shop.

At the time of writing, prototypes of both end plates and rail walls have been completed (see Fig. 14), and the structures team is working to update several recent design changes. These include adapting to the NanoRacks IDD, adding the OB1 sensor board, and adjusting to new solar panel configurations as the DHV design is finalized. The most significant change involves moving the rib from its central location in the chassis to being nearer to the camera end plate, in the -Z direction. We are planning to outsource final machining of the SWaP unit because of the global pandemic and associated closure of the university's machine shops.

5. CONCLUSION

Lessons Learned

With SOC-i being AACT's first CubeSat, our institutional knowledge is rapidly growing as we advance through the spacecraft design stages. A stated non-technical objective of SOC-i is to share knowledge and resources with the broader CubeSat community, so here we briefly discuss some of the major takeaways from our work so far.

Early in the development of SOC-i, the team was very small and we had few laboratory resources. During these several months, we focused heavily on developing a clear, well-structured mission scope and defining the mission requirements. This upfront effort has benefited SOC-i greatly by firmly anchoring our design choices to the mission objectives and requirements.

Our initial research into past missions indicated that a common pitfall for university CubeSat teams was to add greater

and greater complexity as (natural) student turnover introduced new and eager students to a partially designed satellite. This could, in the worst case, result in a broad mission descoping when incomplete hardware and payloads must be scrapped to meet a launch deadline, and the risk of mission failure in orbit is increased. While AACT is not immune to this kind of scope creep, we have found that our early efforts to define carefully considered mission requirements provides a common anchor point with which to evaluate any potential design decision.

We also intentionally try to avoid jarring design changes by maintaining continuity of subsystem leadership. The nature of any student engineering team is that students will come and go frequently. SOC-i's leadership is structured such that the subsystem leads hold their positions indefinitely until they leave the team or decide to step down. New leads are given a few months to "apprentice" for the outgoing lead so that knowledge is not lost in the transition. We maintain one non-technical project manager role and one technical systems-level role that are held by students committed to the project over a multi-year span. These objective of these roles is to provide consistent long-term direction and to ensure that subsystem leads can focus on developing their subsystems as much as possible.

There have also been lessons learned in hardware design and development. One takeaway is the importance of starting the satellite's detailed CAD and configuration design as early as possible. We selected the major elements of flight hardware from several trade studies. However, we neglected to determine the feasibility of configuring this hardware together until near PDR. The initial CAD model revealed that more than 1.5U of our proposed 3U bus was empty. This eventually prompted the change to a 2U size to make more efficient use of volume, but this change proved challenging because we had already made several design choices based on the availability of power from 3U-sized solar panels. The reduction in power budget required changes to the communications system operations and trade studies of new battery and deployable solar panel options. This was a tough lesson learned not to commit to arbitrary design choices (in this case, a 3U volume) early in the development cycle but to let the mission requirements drive them instead. A similar example is the choice of flight computer. We are designing our OBC based on the development board of the chosen microprocessor. However, in doing so we neglected to consider the challenges associated with integrating this particular microprocessor onto a custom board. The chip only came in a ball-grid array package, which is very difficult to design into a student-built PCB with budgetary limitations, not to mention challenges with signal routing and trace width. This led to late-stage changes to the microprocessor to one in a low-profile quad flat package, a choice that propagated through the PCB design and required a substantial redo of circuit design. Along similar lines, we found that far too many components were designated to go on the custom OBC board. Signal routing to ensure an easily testable design became an immense challenge. Fortunately, we had enough margin in our volume and mass budgets to accommodate a second board to house some of these components, and we ultimately split the OBC into what are now the OBC and OB1 boards. This was a setback in development time, but will ultimately provide hardware that is much simpler to test and debug, and the two boards can be tested without relying on proper operation of the other one, allowing development to proceed in parallel.

Summary

SOC-i is rapidly advancing toward a working benchtop prototype, flatsat, that will be used as a testbed for flight hardware and software. We anticipate this flatsat will be operational in early 2021, paving the way for flight unit integration and testing in late 2021 and eventually launch in 2022–2023. SOC-i will demonstrate an important technology for in-space autonomous control systems, while also serving as the first use of real-time convex optimization-based attitude control of an orbiting body. We believe that a successful demonstration of this technology will help to open the door to advanced control design for the space industry.

ACKNOWLEDGMENTS

We wish to thank the faculty advisors for AACT, Professors Mehran Mesbahi and Behçet Acikmeşe; members of the community who have provided valuable guidance, in particular Krish Kaycee, Hannah Varner, and Kelly Hering; and the UW College of Engineering and Aeronautics & Astronuatics department for their financial support.

All authors contributed to writing sections of this paper. Taylor P. Reynolds is the chief engineer of SOC-i, designer of SOAR, and co-founder of AACT. Charles L. Kelly is the project manager of SOC-i and co-founder of AACT. Cole Morgan was the GNC subsystem lead from 2019–2020. Arnela Grebovic was the STC subsystem lead from 2018–2020 and an original member of AACT. Jerrold Erickson was the EPS subsystem lead from 2018–2020 and an original member of AACT. Henry Brown is the current STC subsystem lead. William C. Pope is the current GNC subsystem lead. Jon Casamayor is responsible for the development of the flight computer as a member of the CDH subsystem since 2019. Kyle Kearsley was responsible for the thermal analysis of SOC-i as a member of the STC subsystem from 2019–2020. Gorkem Caylak is the current CDH subsystem lead. Kyle E. Fisher was responsible for the SWAP model development as a member of the STC subsystem from 2019–2020. Cameron Wutzke is the current IMG subsystem lead. Kylle Ashton was responsible for state estimation as a member of the GNC subsystem from 2019–2020. James Rosenthal is a technical advisor to and original member of AACT and a member of the COM subsystem. Devan Tormey was the GNC subsystem lead from 2018–2020 and an original member of AACT. Ellory Freneau was the COM subsystem lead from 2018–2020 and an original member of AACT. Garrett Giddings is the current EPS subsystem lead. Hasan Emin Horata is the current CIA subsystem lead. Anika Dighde is responsible for the flight radio PCB as a member of the COM subsystem. Saharsh Parakh is responsible for IMG system testing and was the IMG subsystem interim lead in 2020. Jiaping Zhen, John C. Purpura, Daniel B. Pratt, and Anders Hunt are responsible for sensor integration as members of the GNC subsystem since 2020.

REFERENCES

[1] A. Stern and D. Grinspoon, *Chasing New Horizons: Inside the Epic First Mission to Pluto*. New York: Picador, 2018.

[2] F. L. Markley, R. G. Reynolds, F. X. Liu, and K. L. Lebsock, "Maximum Torque and Momentum Envelopes for Reaction-Wheel Arrays," *Journal of Guidance, Control, and Dynamics*, vol. 33, no. 5, pp. 1606–1614, 2010.

[3] Y. Kim, M. Mesbahi, G. Singh, and F. Y. Hadaegh, "On the Convex Parameterization of Constrained Spacecraft Reorientation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 46, no. 3, pp. 1097–1109, 2010.

[4] U. Lee and M. Mesbahi, "Feedback Control for Spacecraft Reorientation Under Attitude Constraints via Convex Potentials," *Transactions on Aerospace and Electronic Systems*, vol. 50, no. 4, pp. 2578–2592, 2014.

[5] T. P. Reynolds, M. Szmuk, D. Malyuta, M. Mesbahi, B. Açıkmeşe, and J. M. Carson III, "Dual quaternion based 6-dof powered descent guidance with state-triggered constraints," *Journal of Guidance, Control, and Dynamics*, vol. 43, no. 9, pp. 1584–1599, 2020.

[6] T. P. Reynolds, D. Malyuta, M. Mesbahi, B. Acikmese, and J. M. Carson, "A Real-Time Algorithm for Non-Convex Powered Descent Guidance," in *AIAA SciTech Forum*. Orlando, FL: AIAA, 2020.

[7] M. Szmuk, T. P. Reynolds, and B. Açıkmeşe, "Successive convexification for real-time 6-dof powered descent guidance with state-triggered constraints," *Journal of Guidance, Control, and Dynamics*, vol. 43, no. 8, pp. 1399—-1413, 2020.

[8] Y. Mao, M. Szmuk, X. Xu, and B. Acikmese, "Successive Convexification: A Superlinearly Convergent Algorithm for Non-convex Optimal Control Problems," *arXiv e-prints*, 2018. [Online]. Available: http://arxiv.org/abs/1804.06539

[9] Y. Mao, D. Dueri, M. Szmuk, and B. Açıkmeşe, "Successive Convexification of Non-Convex Optimal Control Problems with State Constraints," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 4063–4069, 2017.

[10] A. Domahidi, E. Chu, and S. Boyd, "ECOS: An SOCP solver for embedded systems," in *European Control Conference*, 2013, pp. 3071–3076.

[11] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, UK: Cambridge University Press, 2004.

[12] K. Shoemake, "Animating Rotation with Quaternion Curves," in *Conference on Computer Graphics and Interactive Techniques*, 1985, pp. 245–254.

[13] D. Malyuta, "An Optimal Endurance Power Limiter for an Electric Race Car Developed for the AMZ Racing Team Semester Project," M.S., Swiss Federal Institute of Technology (ETH) Zurich, 2016.

[14] EnduroSat. UHF antenna III. [Online]. Available: https://www.endurosat.com/cubesat-store/all-cubesat-modules/uhf-antenna/?v=7516fd43adaa (accessed Oct. 6, 2020).

[15] T. Kelso, F. Hoots, and R. Roehrich, "Spacetrack Report no. 3 - Models for Propagation of NORAD Element Sets," NASA, Tech. Rep., 1988.

[16] D. Vallado, *Fundamentals of Astrodynamics & Applications*, 4th ed. Springer Science & Business Media, 2014.

[17] A. Chulliat, W. Brown, P. Alken, C. Beggan, M. Nair, G. Cox, A. Woods, S. Macmillan, B. Meyer, and M. Paniccia, "The us/uk world magnetic model for 2020-2025," National Centers for Environmental Information, NOAA, Tech. Rep., 2020.

[18] J. L. Crassidis and J. L. Junkins, *Optimal Estimations of Dynamic Systems*. Chapman and Hall/CRC Applied Mathematics and Nonlinear Science Series, 2012.

[19] M. J. Sidi, *Spacecraft Dynamics and Control*. Cambridge University Press, 1997.

[20] T. P. Reynolds, K. Kaycee, B. Barzgaran, M. H. D. Badyn, S. Rice, E. Hansen, and A. Adler, "Development of a Generic Guidance Navigation & Control System for Small Satellites: Application to HuskySat-1," in *AIAA Space Conference & Exposition*, Orlando, FL, 2018.

[21] D. G. Gilmore, Ed., *Spacecraft Thermal Control Handbook*. El Segundo, CA: The Aerospace Press, 2002.

[22] K. Khurshid, R. Mahmood, and Q. ul Islam, "A survey of camera modules for CubeSats - Design of imaging payload of ICUBE-1," in *International Conference on Recent Advances in Space Technologies (RAST)*, 2013, pp. 875–879.

[23] L. F. Markley, "Attitude determination using two vector measurements," NASA Goddard Space Flight Center, Tech. Rep. 19990052720, 1998.