The Development of an Autonomous Subsystem Reconfiguration Algorithm for the Guidance, Navigation, and Control of Aggregated Multi-Satellite Systems

by

Ryan Hua Duong

A Thesis Presented to the
Faculty of the USC Graduate School
University of Southern California
In Partial Fulfillment of the
Requirements for the Degree
Master of Science
(Astronautical Engineering)

December 2019

Abstract

As the current growth of space exploration and technology continues to skyrocket, two fundamental facts have relatively persisted: the desire to maximize space system performance while minimizing costs, and the traditional monolithic morphology of spacecraft. An inherent inverse relationship exists between the spacecraft performance and cost relative to the size of space systems, and increasing complexities in mission requirements tend to call upon solutions consisting of large-scale platforms. Simply put, doing more costs more money. However, this limiting barrier can be shattered by considering the concept of on-orbit satellite "aggregation" and "cellularization," or the potential to physically combine many smaller, individually functioning spacecraft to assemble a large space system. Ultimately, such satellite aggregation concepts must adhere to the fundamental spacecraft design considerations for any number of components. In particular, the ability to maintain an aggregate system's guidance, navigation, and control (GNC) subsystem presents an interesting challenge as additional aggregation changes system dynamics. Therefore, an autonomous methodology must be implemented to maintain complete control and successfully "reconfigure" an aggregate GNC subsystem, which we attempt to demonstrate through cellular spacecraft prototypes, dubbed "Satbots." The Satbots, bounded by 3DOF on a ground testbed, possess a simple thruster attitude control system which propel the Satbot to a desired state. The first tests of the reconfiguration algorithm focused on appropriately reallocating this thruster subsystem for a two-Satbot aggregate system. Upon integrating the algorithm onto the aggregate system, a successful reconfiguration of the GNC subsystem was observed, as new control outputs were distributed across both GNC subsystems.

ACKNOWLEDGMENTS

First, I would like to express my extreme gratitude for the opportunity to research such an incredibly visionary concept with Professor David Barnhart of the Department of Astronautical Engineering at the University of Southern California (USC) and its close collaboration with the Information Sciences Institute (ISI). Professor Barnhart's guidance, collaboration, and support throughout the year have taught me the fundamentals of research and granted me the ability to reach new milestones in my academic career.

This project would not have been realized without the generous sponsorship from the Michael Keston Grant from ISI. With the financial support, our team was able to develop the testbed and demonstrate our work.

I would also like to express my appreciation for the contributions to the project made by the members of the Space Engineering Research Center (SERC). Without them, this research would not have been able to achieve as many milestones in this small time frame; for that, I am eternally grateful for the help from Lizvette Villafaña, Shreyash Annapureddy, Jaimin Patel, Kirby Overman, Monica Campos, Michael Smat, and Justin Du Plesis.

Thank you to my family and friends for their encouragement and support. Ultimately, I would like to express my profound gratitude and love to my longtime girlfriend, Michelle Lok, for her everlasting support and continuous encouragement. Knowing that she will always be there for me lessens the fear of uncertainty and failure despite facing the toughest odds.

Thank you to everyone for always supporting me when the days seemed gloomy and the nights seemed darkest.

Contents

List of Figures

List of Tables

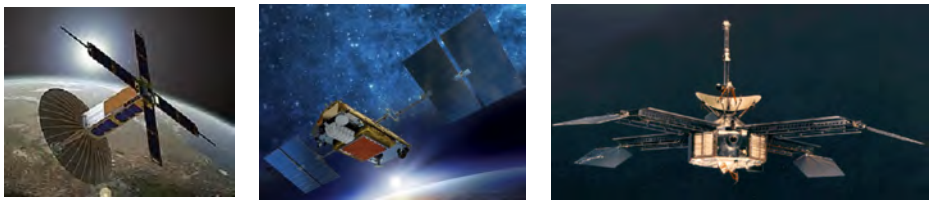
Chapter 1

Introduction

As time progresses, the space industry continues to expand in exciting ways. Currently, we can view the perspectives of interplanetary orbiters as they send back awe-inspiring pictures of the turbulent atmosphere of Jupiter (the Juno spacecraft by JPL) or the cold, rocky surfaces of asteroids (the Hayabusa2 sample return mission by JAXA) [1] [2]. Government space agencies, like NASA, the Israel Space Agency (ISA), China National Space Administration (CNSA), and Indian Space Reseach Organization (ISRO), are rushing back to the Moon at heightened pace to continue the lunar exploration started in 1969 [3] [4] [5]. Most importantly, the realm of space is no longer exclusive to the glory of nations. The rise of private commercial organizations, like SpaceX, Blue Origin, and Virgin Galactic, has opened the door for newfound competition and ambitions to achieve new innovations. Universities and start-up companies have been granted the opportunity to contribute through the growing popularity of CubeSats. The realm of space has opened up to millions.

However, despite the drastically dynamic nature of the space industry, there is one significant consideration that has remained the same: the spacecraft morphology. Simply put, current satellites and space systems still resemble those designed and created 50 years ago. This can be seen in Figure 1.1. Major spacecraft subsystems are combined in the same way. Despite the differences in component sizes, resources, and mission objectives, space systems like space telescopes, interplanetary voyagers, large geostationary communications satellite, constellations of hundreds of imaging spacecraft, and Cubesats all adhere to the same, traditional monolithic form.

Additionally, mission costs have become synonymous with size, mass, and



(A) (B) (C)

Figure 1.1: An example of the similarity in spacecraft morphology. (A) presents the Aeneas CubeSat that operated in LEO in 2010. (B) presents the communications satellite, Iridium-Next. (C) presents the Mariner 4 spacecraft, which performed the first successful flyby of Mars in 1964.

power, as every additional kilogram represents thousands of dollars. [6] This is visualized in Figure 1.2 [7]. Therefore, recent space objectives emphasize the search for low-cost solutions, which usually compromises a space system's performance capabilities.

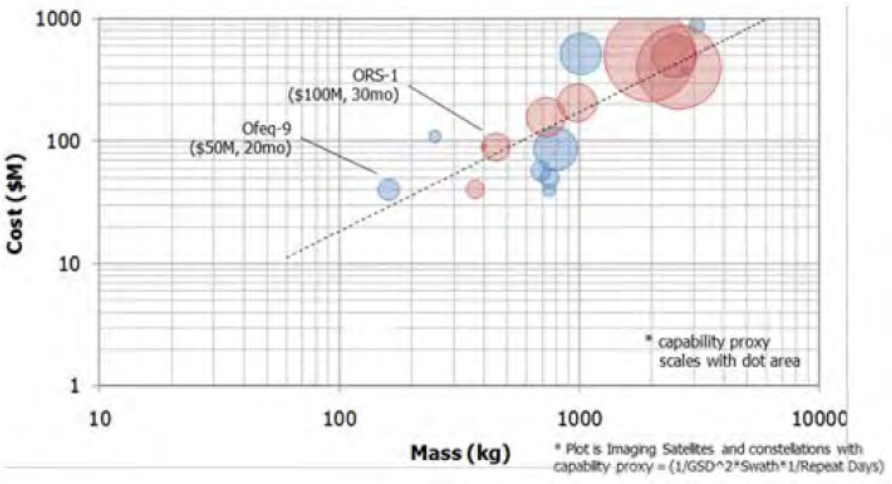


Figure 1.2: The linear trend between increasing mass and cost for a typical imaging mission.

With current technology and development in space systems, what if there were a way to mitigate the associated high costs of large space systems through the implementation of aggregation? What if a bountiful number of low-cost platforms can be launched into space and perform on-orbit aggregation to build large space systems? In doing so, how does one ensure that a large-scale space

system maintains seamless operation while adhering to fundamental spacecraft design considerations? These questions shed light on a new concept of space system morphology: "cellularization."

1.1 Applying Cellularization to Space System Morphology

This biologically inspired idea follows the principles of cellular amalgamation, in which many simple cells combine to form different functional groups (like tissues and organs) to achieve greater functions [7]. When applied to a space system, cellularization aims to transition away from the traditional monolithic entity to a decentralized network of smaller, low-cost platforms that combine resources to achieve mission objectives. Should the ability to aggregate and dock many electro-mechanical platforms on-orbit be mastered, then the current capabilities in space exploration and spacecraft performance will be able to expand in unprecedented ways. Applications may consist of (but are not limited to) seamless resource management and replacement through on-orbit servicing or the autonomous control of constructing complex multi-satellite systems like space telescopes.

This on-orbit cellularization concept proposes the means for a satellite or space platform to aggregate and deaggregate upon command. An on-orbit assembly may grow to any size or volume to adapt to mission requirements or reconfigure itself to address certain component level malfunction. However, several major challenges must be addressed before realizing true multi-satellite aggregation through a cellular morphology. These challenges consist of the following questions:

- How does one enable the ability for these platforms or "cells" to share their resources and capabilities seamlessly?
- How does one ensure that the aggregate system maintains functionality despite any number of "cells"?

Both of these challenges may be addressed by the implementation of a proposed aggregation architecture, which blends the potential of software with spacecraft hardware considerations to dictate component integration, resource management, and data flow.

1.2 Defining a New Computational Architecture for Aggregation

First, the proposed framework of a computational architecture to support applications of on-orbit aggregation is characterized and described. This top-level concept must encompass both hardware and software resources for every element of the aggregation. Additionally, various layers must include the ability to transport data and support the operation of the aggregate system.

This notional aggregation architecture is described briefly below and is visualized in Figure 1.3 [8].

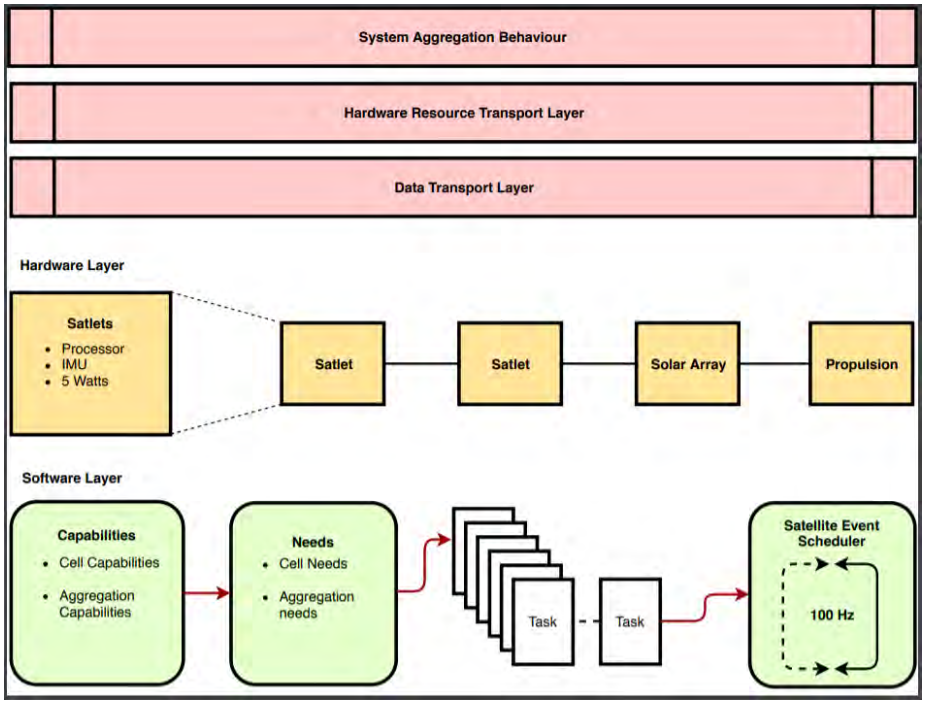


Figure 1.3: The notional computational architecture All five layers come together to support true aggregation between components, software, hardware,and satellite systems.

The aggregation architecture starts at the lowest level: the software layer. This section considers the characterization of internal capabilities (i.e., cell, component, element) that exist and the requirements that allow the aforementioned internal capabilities to function (i.e., power). Typical space systems possess an "event scheduler" that runs physical hardware systems at a certain processing

cycle from the main processor. Then, based on the current requirements of the component/cell, tasks are distributed to request certain needs or received to accept needs from other components.

The next level introduces the hardware layer, which represents the physical aggregation of various elements, like solar arrays or propulsion units. Each of these physical elements possess the embedded software layer that corresponds to the particular cell/component.

The third level, the data transport layer, enables aggregation across multiple physical systems by connecting multiple processing elements. This supports the data flow of pertinent information between cells/components.

A fourth level corresponds to the hardware resource transport layer, where the needs and capabilities of each hardware element are shared among other aggregate elements (e.g., maintaining functionality between two distinct on-board processors or IMUs).

The last level dictates the system aggregation behavior, where an aggregate system comprised of many smaller platforms can still function as the traditional monolithic spacecraft. The use of the hardware resource transport layer and the data transport layer support the seamless interaction between multiple systems to adhere to spacecraft design concepts. For example, an aggregate system identifies its required thermal response, then calls upon other elements of active or passive thermal control to ensure that the aggregate system maintains isothermal conditions.

1.3 The Motivation to Demonstrate System Aggregation Behavior for GNC

Continuing this concept of the aggregation architecture leads us to a particularly interesting instance of aggregation behavior: maintaining GNC performance for an aggregate system. Without a doubt, GNC serves one of the most vital supporting roles in spacecraft subsystem design for a variety of reasons. Simply put, the orientation of the spacecraft can directly influence the power received from solar panels, the communications link to ground systems, spacecraft dynamics in orbit, thermal control, and much more [9]. However, as a proposed space system continues to aggregation with N-number of satellite components, the system dynamics and mass properties (center of mass/gravity, moments of inertia) change. The continually shifting system dynamics must be accounted

for autonomously to support seamless functionality. Should there be an inconsistency in the GNC model, there is a real danger for potential mission failure. Therefore, the ability to *reconfigure the GNC operation autonomously* serves great interest in developing the stepping stones for realizing the cellularization morphology.

With this in mind, we focus on developing an algorithm to reconfigure the GNC subsystem and demonstrating the proof of concept through the physical aggregation of hardware.

1.4 Thesis Outline

The contents of this thesis encompasses the conceptual background and development surrounding the first instance of the subsystem reconfiguration algorithm and its implementation on the GNC subsystem of two distinct cellular spacecraft prototypes.

Firstly, the hardware architecture of the enabling technology is described to demonstrate the physical and tangible aspect of reconfiguration. Through this, we discuss the introduction of how we realize the concept of cellular morphology with fabricated pseudo-satellite prototypes called "Satbots." These Satbots represent a simple spacecraft with a processor and GNC thruster setup.

Next, we discuss the development of an autonomous GNC operation for a Satbot system in great detail, including the navigational sensors and validation, guidance techniques, and controller scheme. After the three sectors of the GNC operations are understood, we introduce the concept of the subsystem reconfiguration algorithm along with its ability to address N-number of aggregations with the computational architecture.

Finally, we address the first implementation of the subsystem reconfiguration algorithm on a two Satbot system, the corresponding preliminary results, and the next steps for further development.

Though we introduce the concept of an underlying computational architecture, we maintain strict focus on the development of the GNC subsystem and the reconfiguration algorithm. Usage of the proposed computational architecture will be hinted throughout certain sections when we discuss aggregate systems with N-number of Satbots; however, the current algorithmic testing does not actually implement the architecture's ability to transfer data.

Chapter 2

Hardware Architecture for Supporting Rendezvous and Proximity Operations

The Space Engineering Research Center has dedicated their efforts in advancing the current capabilities of rendezvous and proximity operations (RPO) ranging from the development of innovative docking mechanisms to the vision of establishing a new computational framework architecture. Ultimately, when exploring the necessity of resource management through multi-satellite aggregation, significant time must be spent to validate the software through the responsiveness from the provided hardware. Fusing hardware and software presents the opportunity to demonstrate the efficacy and the proof of concept for subsystem reconfiguration, where the physical hardware can be appropriately reallocated based on the given aggregate configuration. In support of developing the subsystem reconfiguration algorithm, we fabricate test hardware to represent a spacecraft with a GNC subsystem. In the following sections, the supporting hardware that realizes the testing for the reconfiguration algorithm is described in greater detail.

2.1 The Satbot Prototype

Researchers at the SERC developed several pseudo-satellite prototypes, dubbed "Satbots," to help realize the biologically inspired concept of cellular space platform morphology.[7] These Satbots represent the potential capability to construct more complex systems from many small, low-cost platforms, much like how stem cells amalgamate to form sophisticated, functioning organisms. A variety of spacecraft components may join together to serve multiple purposes, whether it be replacing worn-down components or forming creative configurations to achieve mission objectives. The first design of the aggregated Satbot system is presented in Figure 2.1.

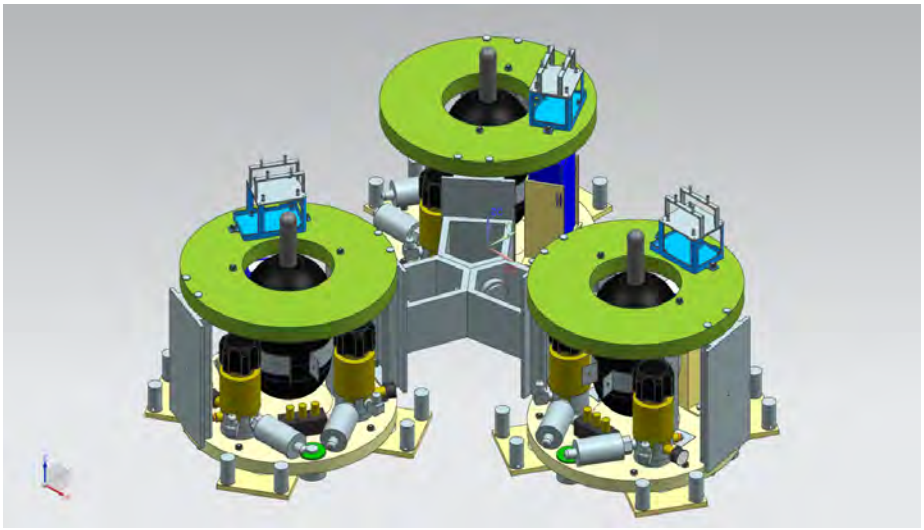


Figure 2.1: A computer-aided design model of the current Satbot design. This depiction shows the proposed aggregation between three Satbot protoypes.

The Satbot mimics the structural framework of a generic cell while maintaining the compact, symmetric design typically desired from spacecraft. Currently, each Satbot possesses two docking ports, from which a proposed docking mechanism may extend outward and latch onto similar systems. The Satbot is built to emulate orbit operations on a ground testbed, as seen in Figure 2.2.

An on-board compressed air tank feeds through three flat-round air bearings at 60 psi to lift the Satbot platform 5 microns on top of a float glass platen, establishing a near-frictionless environment resulting in 3DOF. Compressed air is also redirected through eight unidirectional output solenoid valves at 100

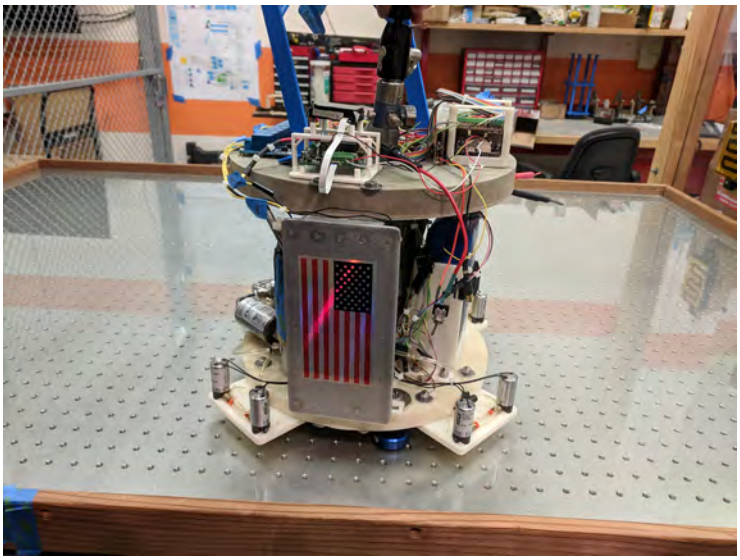


Figure 2.2: The first instantiation of the Satbot prototype.

psi to support its GNC subsystem. These solenoid valves output ports face tangentially to the circular base and generate thrust through the release of air through its opening port, as visualized in Figure 2.3. By placing these solenoid valves at the edge of its circular platform, the Satbot maximizes the torque generation about its center of mass, which is designed to co-locate with its own geometric center. For future reference in this paper, the terms "actuator" or "thruster" represent these solenoid valves.

A single six-cell battery powers the entirety of the Satbot's electronics, which is comprised of a Wi-Fi enabled Odroid-C2 single-board computer with a Linux Operating System, an eight channel LOW level relay module that feed into the eight solenoid valves, and the on-board navigational sensor.

2.2 Thruster Specifications and Thrust Validation

Significant consideration must be placed on designing and quantifying the Satbot's GNC operation to avoid potential mission failures; for example, the loss of the Lewis spacecraft was attributed to the Attitude Control System. [10] Therefore, in this next section, we focus on quantifying the solenoid valve's

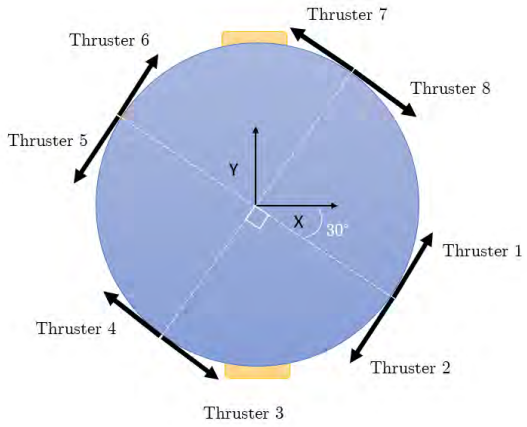


Figure 2.3: A visualization of the Satbot's top view. This does not represent the free body diagram of the Satlet since each arrow simply represents the direction of the expelled compressed air. Therefore, the resultant force for the Satbot from a particular thruster is the equal and opposite to the displayed unit vector. The yellow blocks represent the docking face ports. The body axis is defined by the orientation of the onboard sensor.

parameters to support GNC operation.

First, to understand each thruster's torque contribution, we measure the current X and Y positions of the thrusters relative to the Satbot's body frame and center of mass. The thrusters are mounted on an base that extends outward from the circular base, which are offset by -30 degrees from the x-axis in the body frame. Since the testbed enables 3DOF, the z-components are not considered for each thruster's position. When calculating torque contributions, we will only be interested in the moments about the outward z-component, so we assign the z-component positions to zero. The positions of all eight thruster are recorded below in Table 2.1.

Now, the force from the thrusters must be quantified, which we demonstrate analytically and experimentally. Given the specifications of the solenoid valve, some simplifying assumptions can be made to determine the flow properties of the compressed air [11]. From this, we can apply the isentropic flow equations to determine the output velocity and whether sonic conditions are met. Then, the fundamental rocket equation can be applied to determine the theoretical

Table 2.1: Thruster positions relative to the Satbot's body frame and center of mass

Thruster	X [m]	Y [m]
1	0.1296	-0.1244
2	0.0429	-0.1744
3	-0.1244	-0.1296
4	-0.1744	-0.0429
5	-0.1296	0.1244
6	-0.0429	0.1744
7	0.1244	0.1296
8	0.1744	0.0429

thrust output from a single solenoid valve.

The solenoid valves are regulated to 100 psi initially so that the maximum thrusts can be achieved. Typical laboratory settings are assumed for the ambient conditions. These conditions are collected and displayed in Table 2.2. A_e represents the outlet area, γ represents the specific heat ratio, R represents the gas constant for compressed air, T_0 and P_0 represent the stagnation temperature and pressure respectively.

Table 2.2: The initial conditions for solenoid valve flow calculations

Parameter	Value	Units
A_e	7.36e-4	[m^2]
T_0	298	[K]
P_0	100 (689476)	[psi] (Pa)
P_a	14.7 (101325)	[psi] [Pa]
R	286.7	[J / kg K]
γ	1.4	

We can model the solenoid valve as a simple chamber that is filled with compressed air at the initial time; this is possible because the central tank flows through every solenoid valve during GNC operation. Thrusts are not generated until the opening outlet port of the valves are opened. Starting with the isentropic flow relationship between stagnation pressure and the exit pressure, the conditions for sonic flow can be observed (i.e., the mach number $M = 1$) through Equation 2.1. The sonic flow conditions will be met if it can be shown that P_e is greater than the ambient pressure, P_a.

$$\frac{P_e}{P_0} = [1 + \frac{\gamma - 1}{2} M^2]^{\frac{\gamma}{\gamma - 1}} = [\frac{2}{\gamma + 1}]^{\frac{\gamma}{\gamma - 1}} = 0.528 \tag{2.1}$$

Confirming that sonic flow has been achieved, the exit temperature, T_e, can be determined with the relationship in Equation 2.2. Then, with the ideal gas law, the exit density, ρ_e, can be derived for the compressed air, as seen in Equation 2.3. Subsequently, the exit velocity, u_e, can be represented as the speed of sound, represented by Equation 2.4.

$$\frac{T_e}{T_0} = [1 + \frac{\gamma - 1}{2} M^2]^{-1} = [\frac{2}{\gamma + 1}] \tag{2.2}$$

$$\rho_e = \frac{Pe}{RT_e} \tag{2.3}$$

$$u_e = \sqrt{\gamma R T_e} \tag{2.4}$$

The mass flow rate, $\dot{m}$, can now be determined through Equation 2.5, and we can apply the thrust equation for a rocket, as seen in Equation 2.6.

$$\dot{m} = \rho_e u_e A_e \tag{2.5}$$

$$F_T = \dot{m} u_e + A_e (P_e - P_a) \tag{2.6}$$

The resulting exit conditions of the flow is organized and shown below in Table 2.3.

Table 2.3: Theoretical results for solenoid valve's flow properties

Parameter	Value	Units
T_e	248.3333	[K]
P_e	364240	[Pa]
ρ_e	5.1158	[kg/m^3]
u_e	315.72	[m/s]
$\dot{m}$	7.3271e-04	[kg/s]
T	350	[mN]

We find from the initial flow analysis that sonic flow is achieved from the solenoid valves output. Ultimately, we find that the thrust output is approximately 350 mN. Transitioning towards an experimental validation presents the

opportunity to understand more about the actual system response from the thrusters.

The experimental setup is depicted in Figure 2.4. To measure the force output from a given thruster, a single solenoid valve is fitted into a 3D printed mount. A hanging mass is used to calibrate a load cell, which is fixed to a location directly above the solenoid valve. Compressed air is fed through the solenoid valve at 100 psi, and the solenoid valve is actuated to open for the entire duration of the test.

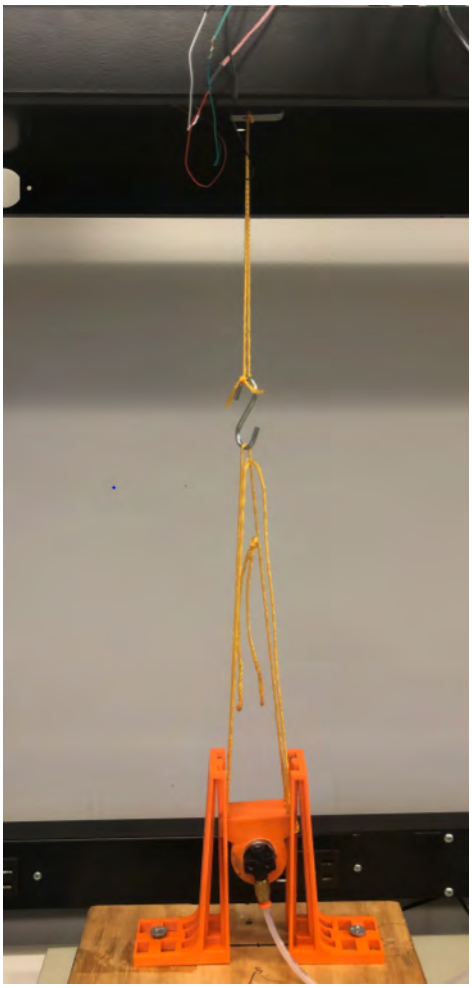


Figure 2.4: Test setup for the thruster validation test.

After testing for varying firing times, we find that the solenoid valve provides

the most consistent force outputs at approximately 559 mN, which is substantially larger than the calculated theoretical value. In hindsight, it has been determined that all the solenoid valve's orifices have been physically enlarged to allow for more thrust. This might result in unequal and unsteady force contributions from each solenoid valve. This will negatively affect in how precisely we can model the Satbot's current control system, but ultimately does not limit our ability in demonstrating the proof of concept of the reconfiguration algorithm With an understanding of the physical response of a solenoid valve, we assume that this determined force represents the maximum output value, denoted as F_{max}, which we discuss later in the section *Pulse-Width Modulation*.

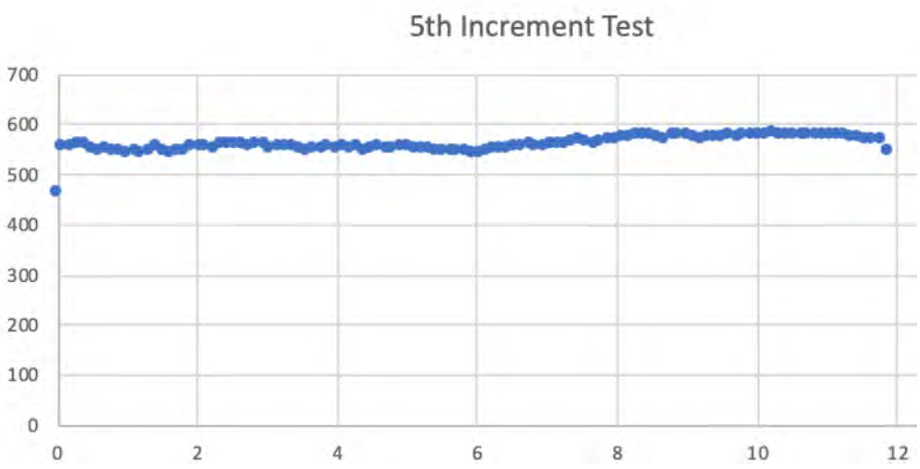


Figure 2.5: The thruster validation.

Chapter 3

Concept of GNC Operations

The proposed concept of operations of the Satbots is meant to demonstrate the potential of autonomous GNC and multi-spacecraft aggregation. Each has continuous communication between the other Satbots, thus each will determine the necessary states which drives the GNC subsystems.

Still in its early development phase, the Satbot is manually provided the desired states to perform GNC. On-board sensors measure the navigational state, which is stored for guidance and control calculations. The state data passes through a Kalman filter to determine the state estimates and corresponding deviations. The controller accepts these deviations to design the necessary forces and torques to derive the corresponding firing times from each thruster. This GNC operation continues until the Satbot achieves the desired state. This continuous process is visualized in Figure 3.1.

To support aggregation, the relative knowledge between all Satbots must be known at some level. Therefore, each Satbot possess an uploaded "identification file," which is a set of hardware parameters relative to the Satbot's body frame. This concept will be discussed later in the *Subsystem Reconfiguration Algorithm* section. If the Satbot identifies that a docking face is active, such that an aggregate system exists, then each Satbot transfers its parameters into the data transport layer of the computational architecture to create an aggregation identification file to support the system's GNC. To simulate a docking event (i.e., aggregation event) we created a new reconfiguration algorithm,

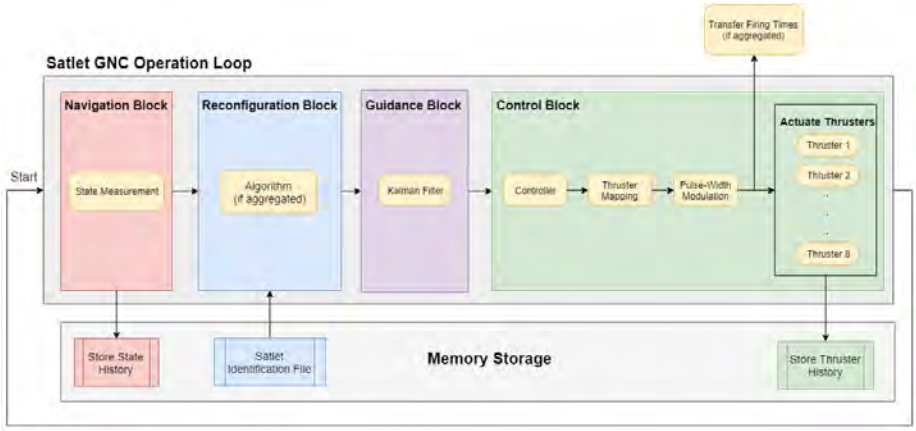


Figure 3.1: The GNC operation loop for the Satbot.

which imports the appropriate Satbot parameters from the architecture's data transport layer to calculate the new aggregate properties. The algorithm feeds this information back into *one* of the Satbots, and GNC operation continues for the aggregate system. As the aggregate system operates with the proposed decentralized computational architecture, each Satbot possesses the capability to carry on aggregated GNC for the system to operate as a single monolithic entity.

The following sections discuss the navigation, guidance, control, and reconfiguration algorithm in greater detail.

Chapter 4

Navigation

This section discusses the Satbot's onboard state measurement process. In performing the GNC operations, the Satbot must propel itself towards the desired state based on the current measurement. We discuss the selected sensor, its performance, and some supporting concepts for developing the GNC algorithm.

4.1 The Pozyx Tag

In the SERC labs, each Satbot uses a Pozyx tag, a real-time location system electronics board that measures the navigation state. Determining the positional state relies on multilateration, the methodology of GPS radio-navigation. Like GPS, four unique "anchors," or Pozyx modules that serve as fixed reference points, transmit an ultra-wide band signal to the Pozyx tag to locate its position with an accuracy down to 10 cm on the testbed. We measure the anchor locations relative to a particular corner of the testbed, hereby setting the origin at this point. The anchors are also mounted at varying heights so that only a single intersection point is identified. The Pozyx setup is depicted below in Figure 4.1. The Pozyx tags and the anchors are shown in Figure 4.2.

On-board the Pozyx tag lies a 3-axis accelerometer and a 3-axis gyroscope, which provide measurements for the accelerations and the orientation. The Pozyx also possesses its own coordinate frame, which we use to define the Satbot's body frame. Ultimately, each Satbot measures the presented measured state vector in Equation 4.1 for 3DOF. The orientation is determined via quaternions, where $q_{1,2,3}$ represents the quaternion vector components and q_4

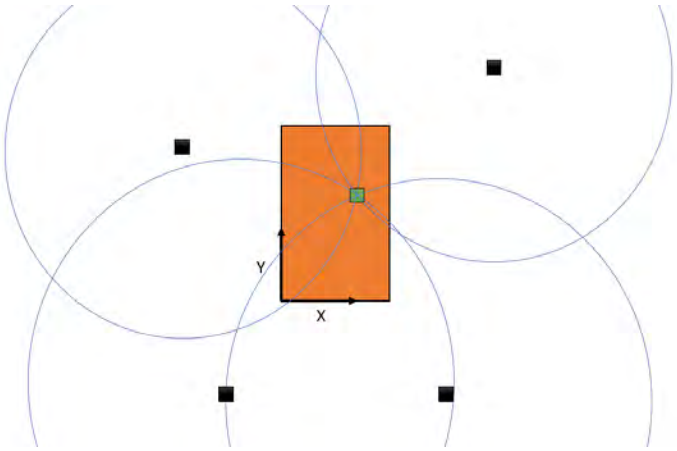


Figure 4.1: The Pozyx multilateration concept. With knowledge of the fixed anchor locations (black squares), the Pozyx tag (green square) will constantly transmit and receive signals to determine its positional state on the testbed (orange).

Figure 4.2: The Pozyx hardware suite.

represents the quaternion weight. The positions are represented by x and y, the accelerations are represented by a_x and a_y, and angular velocity is represented by ω_z

$$\mathbf{X}_{measured} = [x, y, q_1, q_2, q_3, q_4, a_x, a_y, \omega_z] \tag{4.1}$$

4.2 Orientation Through Quaternions

Representing the orientation through the use of quaternions has become the preferred methodology for space applications due to its robustness and computational superiority in comparison to the use of the more intuitively tangible Euler angles. [12] For instance, it is possible for Euler angles calculations to result in a singularity (gimbal lock); however, quaternions avoid this scenario through the redundancies of its parameters. Additionally, the Euler angles can be derived from the quaternions through Equation 4.2. However, this step will not suffice as a quadrant check is required to isolate the single solution for the Euler angle about the z-axis (3DOF). This can then be computationally solved through the use of *atan2* as seen in Equation 4.3.

$$\theta = 2cos^{-1}q_4 \tag{4.2}$$

$$\theta = atan2(2(q4 * q3 + q1 * q2), 1 - 2(q_2^2 + q_3^2)) \tag{4.3}$$

Another significant capability of the quaternions is the ease in developing the direction cosine matrix (DCM), in which transformations between coordinate frames can be quickly derived [12]. Equations 4.4 to 4.12 demonstrate how to compute each component of the DCM. The DCM is denoted as C, with each subscript corresponding to a particular index of the matrix, and is compiled into Equation 4.13. The superscripts for the DCM considers the two coordinate frames of interest. In Equation 4.13, the DCM represents the transformation from frame A to frame B.

$$C_{11} = 1 - 2q_2^2 - 2q_3^2 \tag{4.4}$$

$$C_{12} = 2(q_1 q_2 - q_3 q_4) \tag{4.5}$$

$$C_{13} = 2(q_3 q_1 + q_2 q_4) \tag{4.6}$$

$$C_{21} = 2(q_1 q_2 + q_3 q_4) \tag{4.7}$$

$$C_{22} = 1 - 2q_1^2 - 2q_3^2 \tag{4.8}$$

$$C_{23} = 2(q_2 q_3 - q_1 q_4) \tag{4.9}$$

$$C_{31} = 2(q_3 q_1 - q_2 q_4) \tag{4.10}$$

$$C_{32} = 2(q_2 q_3 + q_1 q_4) \tag{4.11}$$

$$C_{33} = 1 - 2q_1^2 - 2q_2^2 \tag{4.12}$$

$$^{A}C^{B} = \begin{bmatrix} C11 & C12 & C13 \\ C21 & C22 & C23 \\ C31 & C32 & C33 \end{bmatrix} \tag{4.13}$$

Two useful facts come with the DCM. Firstly, the DCM can be inverted (which is also the matrix transpose in this case) to reverse the coordinate transformation (e.g., the inversion of $^{A}C^{B}$ becomes $^{B}C^{A}$ and represents the transformation from frame B to frame A). Secondly, the DCM can incorporate any number of successful rotations to resemble the transformation between two select frames. For example, suppose there are three distinct coordinate frames, A, B, and D, of which we have knowledge of the following two DCMs: $^{A}C^{B}$ and $^{B}C^{D}$. Then, we can perform matrix multiplication to derive the transformation from frame A to frame D.

Another fact is considered: the Pozyx tag measures its quaternions with respect to the inertial frame that has been defined by the anchor. Therefore, when multiple Satbots are introduced for aggregation testing, the GNC algorithm will be able to derive the DCM to convert body-relative components to any other Satbot.

4.3 Assessing the Effectiveness of the Pozyx Tag

The Pozyx is currently rated to an accuracy of 10 cm, though numerous sources may contribute to additional errors in the measurement data. Ulti-

mately, it is important to identify the statistical outlook (like covariance matrices) before feeding the data through a filter to improve the Satbot's guidance and navigation capabilities. We activate a stationary Pozyx tag and measure state data for one hour to quantify the quality of the measurement and the initial covariance matrix to build the Kalman filter. The following results are shown from Figure 4.3 to Figure 4.8.

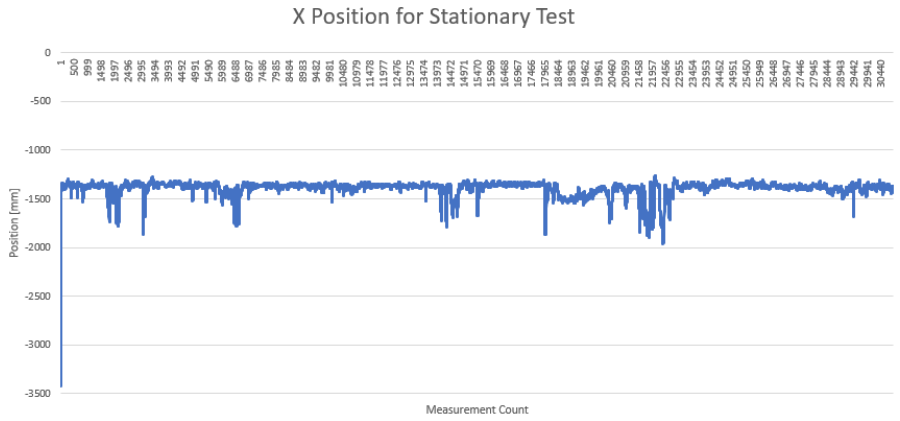


Figure 4.3: Measurement data for the X-position for a stationary Pozyx for approximately one hour of observations.

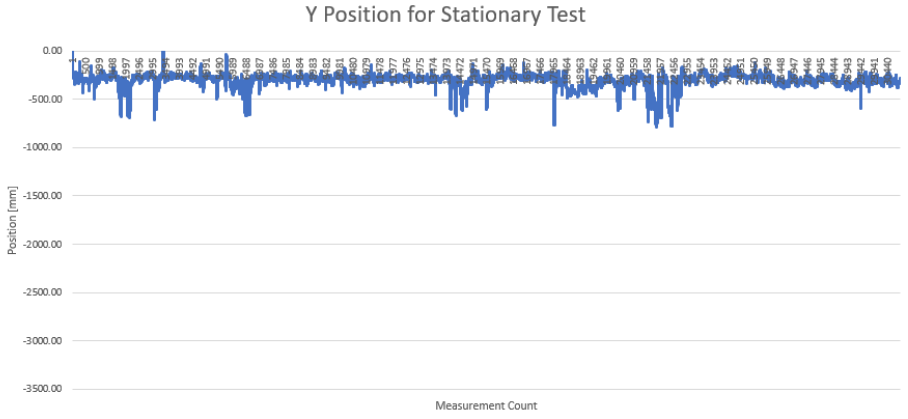


Figure 4.4: Measurement data for the Y-position for a stationary Pozyx for approximately one hour of observations.

It is visible that the Pozyx, like every sensor, is accompanied by some in-

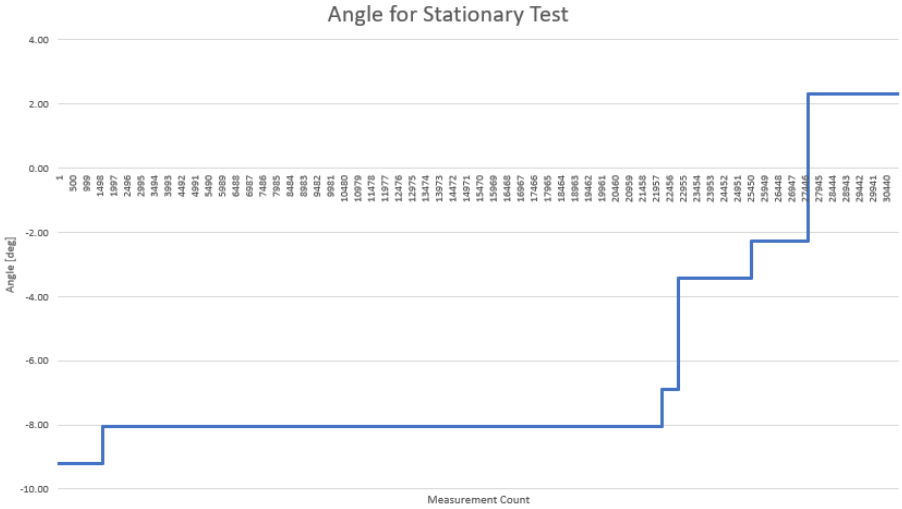


Figure 4.5: Measurement data for the angle for a stationary Pozyx for approximately one hour of observations.

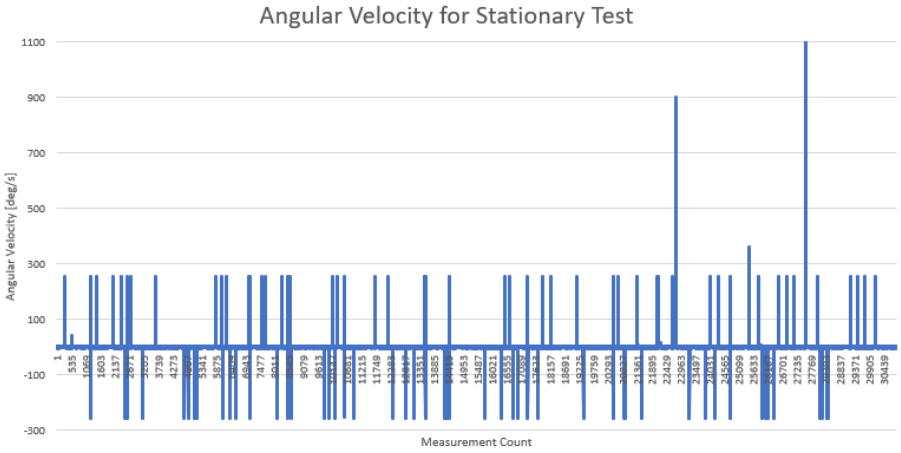


Figure 4.6: Measurement data for the angular velocity for a stationary Pozyx for approximately one hour of observations.

herent noise in its data collection. When resting completely still, the Pozyx measurements suggest that there is continual shifts in state. An impulse seems to have occurred closer to the end of data collection, as a large spike in measurements occur across the acceleration and orientation terms.

All things considered, a few conclusions can be drawn regarding the Pozyx

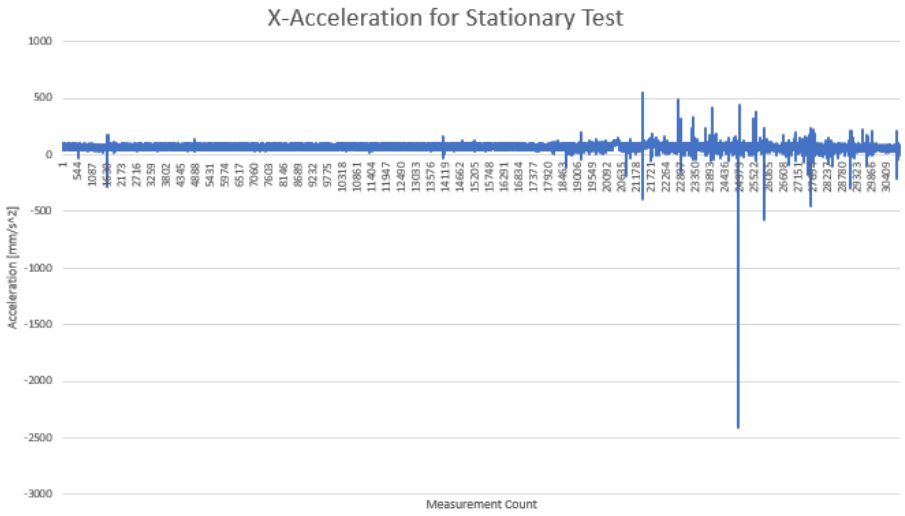


Figure 4.7: Measurement data for the x-acceleration for a stationary Pozyx for approximately one hour of observations.

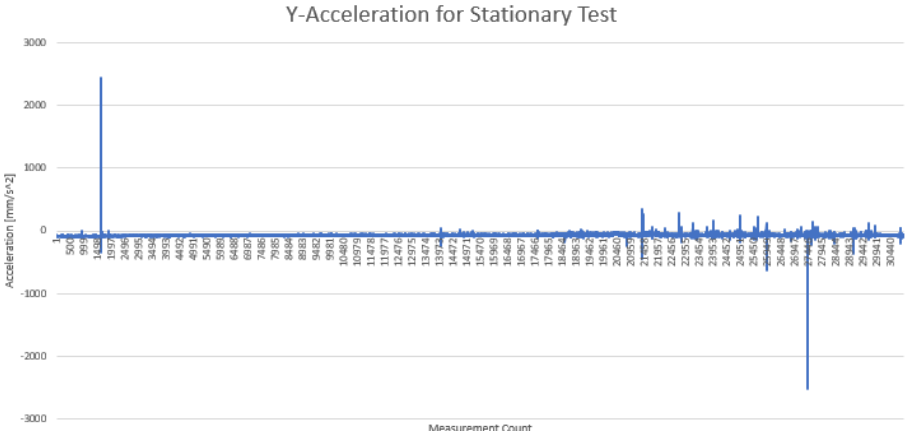


Figure 4.8: Measurement data for the y-acceleration for a stationary Pozyx for approximately one hour of observations.

data:

- There is significant noise / error in the positioning measurement. This can be attributed to inherent sensor noise, errors in calibration of the anchor locations, and/or improper signal return from metallic reflectivity.
- The orientation (quaternions) is very stable, with the exception of the

unexpected impulses throughout the test.

- The angular velocity has periods of stability, but is plagued by consistent and unreliable spikes measure ± 255 degrees per second.

We attempt to remedy this situation by implementing an *additional* filter to smooth the position and angular velocity data before processing it through the Kalman filter. We do not filter the angle or acceleration terms, as its response demonstrated stable results until perturbed by some anomalous impulse. The data filter we add simply detects an outlier, then forces the outlier's measurement index to be represented by the previous valid measurement. An outlier in this case would be defined as a measurement that exceeds the specified allowable deviation; for position and angular velocity, we set this value at 200 mm and 5 deg/s respectively. Upon encountering an outlier value, a counter is incremented. Should the counter exceed a value of 10 instances (where a measurement is taken every 0.04 seconds), then the measurement that appeared to be an outlier is deemed a new measurement. There is no concern for any conflicts between the incoming measurement data because thruster actuation only uses the latest values after all control sequences are finished.

This does not drastically affect the control output as data is measured every 0.04 seconds, and the control firing times actuate for approximately 0.8 seconds at a time.

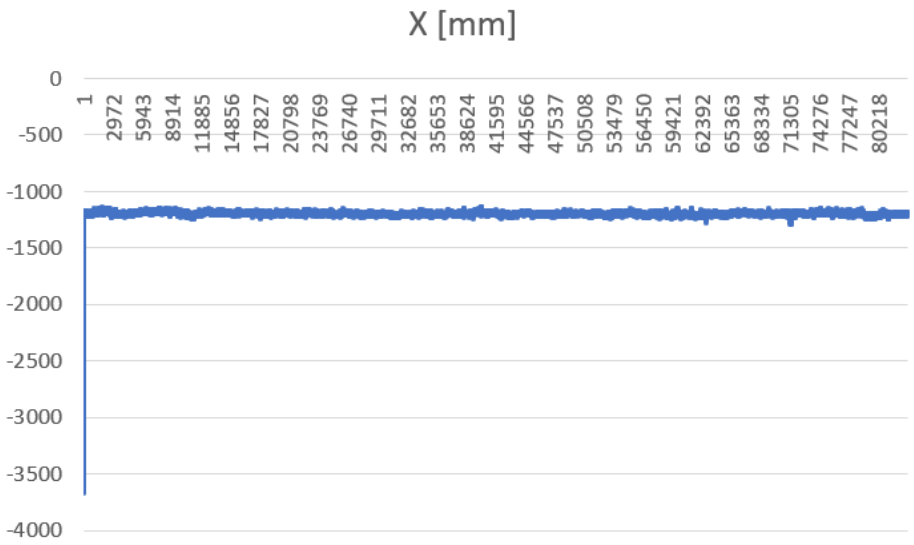


Figure 4.9: The filtered data for the X-position of a stationary Pozyx over an hour.

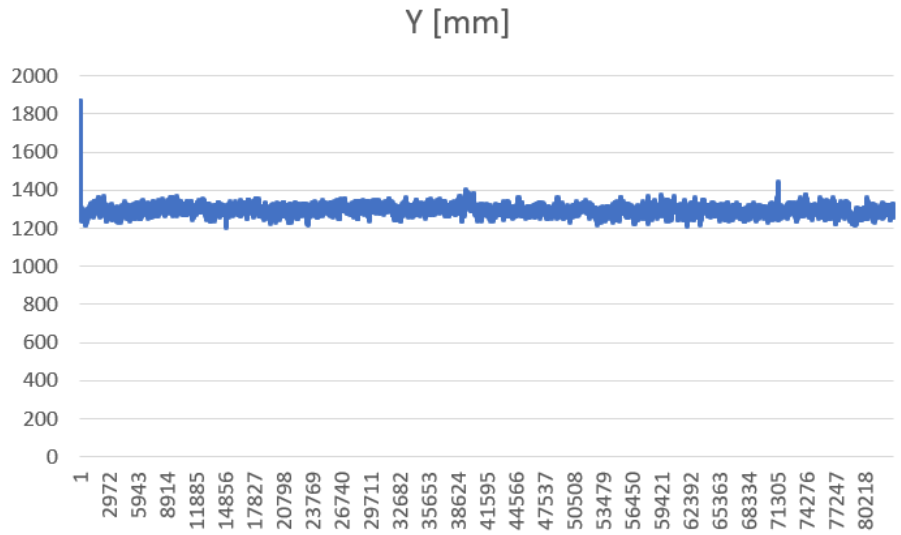


Figure 4.10: The filtered data for the Y-position of astationary Pozyx over an hour.

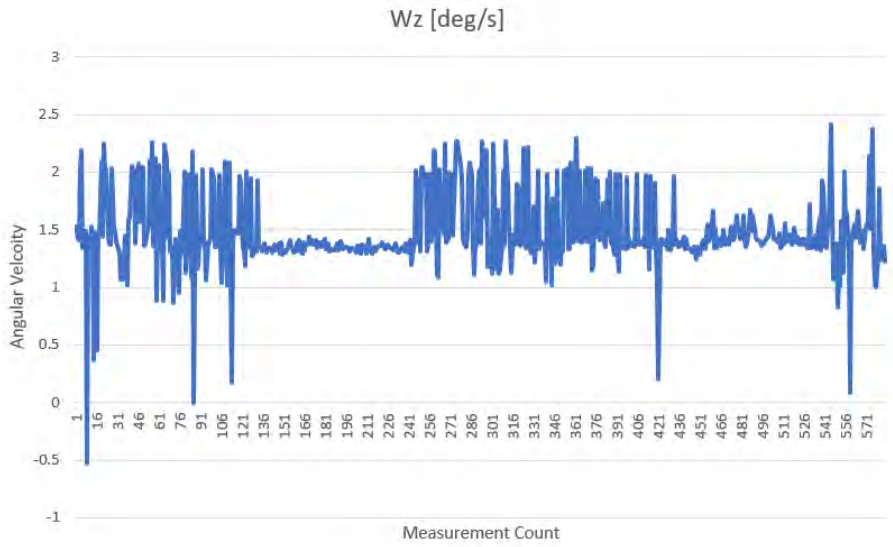


Figure 4.11: The filtered data for the angular velocity of a stationary Pozyx over some period of time.

Then, after these trials with the Pozyx, we can determine the state covariance matrix, as seen in Table 4.1. The blank indices correspond to the terms reflected across the matrix's diagonal (i.e., $P_{xy} = P_{yx}$). We note that the velocity terms are calculated through simple linear dynamics from the time between measurements and the difference in accelerations.

Table 4.1: The initial covariance matrix of the filtered pozyx data

	X [m]	Y [m]	Vx [m/s]	Vy [m/s]	θ [deg]	ω [deg/s]
X [m]	0.0216					
Y [m]	0.0018	0.1057				
Vx [m/s]	1.986e-5	1.395e-5	1.844e-6			
Vy [m/s]	1.144e-5	-3.147e-5	6.729e-7	3.151e-6		
θ [deg]	-5.521e-31	9.364e-32	1.846e-35	-2.638e-35	8.292e-32	
ω [deg/s]	0.0019	0.1034	1.352e-5	-8.388e-6	1.091e-31	0.1065

With more precise state data, we transition towards the guidance portion of the GNC operation.

Chapter 5

Guidance

Though the Satbot is capable of understanding its current state through the navigational sensor, there are several components that cannot be directly measured (e.g., velocity). Additionally, there is the potential for inconsistency in data, as demonstrated through the inherent sensor noise and disturbances. Therefore, using measured sensor data as the reference for determining state deviations may lead to improper control outputs. However, we may ameliorate these concerns by implementing a Kalman filter [13].

5.1 Kalman Filtering

The Kalman filter is a technique for computing the best estimate of a state through a predictor-corrector methodology. The Kalman filter can accept imperfect measurement data and uncertainty values to model a more true state. There exists several types of Kalman filters, but we choose to employ an Extended Kalman filter (EKF) because it addresses the non-linear dynamics of orbit operations. We acknowledge that the current Satbot setup is not necessarily non-linear, but implementing the EKF is still valid for our application. The ultimate outcome from the Kalman filter should be a suitable state estimate vector that can be used to more accurately determine the control output. More specifically, we aim to determine the following in Equation 5.1, where $\hat{x}$ represents the estimate of the state.

$$\hat{x} = [x, y, v_x, v_y, \theta, \omega_z]^T \tag{5.1}$$

Since the kinematics of the Satbot motion on the 3DOF testbed is linear, the equations of motions can be described as simple translational and rotational motion as seen Equation 5.2 to Equation 5.4 and Equation 5.5 to Equation 5.7.

$$a(t) = a_i \tag{5.2}$$

$$v(t) = v_i + a_i \Delta t \tag{5.3}$$

$$r(t) = r_i + v_i \Delta t + \frac{1}{2} a_i \Delta t^2 \tag{5.4}$$

$$\dot{\omega}(t) = \ddot{\theta}_i \tag{5.5}$$

$$\omega(t) = \omega_i + \dot{\omega}_i \Delta t \tag{5.6}$$

$$\theta(t) = \theta_i + \omega_i \Delta t + \frac{1}{2} \dot{\omega}_i \Delta t^2 \tag{5.7}$$

With the equations of motion, the state transition matrix (STM) can be derived. The state transition matrix essentially maps a initial state vector to another state at any given time, a very powerful concept in linearizing the dynamics of orbit trajectories (or the trajectory of the Satbot). This idea is mathematically expressed in Equation 5.8, where the Φ represents the state transition matrix, δx represents a state deviation at any time, and δx_0 represents the initial state deviation.

$$\delta x = \Phi(t, t_0) \delta x_0 \tag{5.8}$$

Similarly to Equation 5.8, the time derivative of the STM can be determined through the use of the Jacobian matrix, which is the matrix of partial derivatives of the state rates with respect to the state. We show the Jacobian matrix in Equation 5.9, which becomes a 6x6 matrix for our defined state, and the differential equation for the STM in Equation 5.10.

$$F(t) = \frac{\partial \dot{\mathbf{X}}}{\partial \mathbf{X}} \tag{5.9}$$

$$\dot{\Phi} = F(t) \Phi \tag{5.10}$$

The STM possesses several unique properties; most fundamentally, the initial

STM ($\Phi(t_0, t_0)$) is the identity matrix. With this knowledge, directly integrating the differential equation for the STM allows us to derive the following in Equation 5.11, assuming that the STM is constant.

$$\Phi = \begin{bmatrix} 1 & 0 & \Delta t & 0 & 0 & 0 \\ 0 & 1 & 0 & \Delta t & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.11}$$

Then, the EKF algorithm uses the previous (or initial) estimate to predict the current state and covariance matrix, update the Kalman gain, and determine the newest state estimate. The following sets of equations are performed in sequence. We note that variables with a bar represent a prediction. Equations 5.12 and 5.14 calculate the predicted state error and predicted error covariance, where P represents the covariance matrix, and Q represents the process noise matrix. The observation (what is actually measured) is represented by ρ. The Kalman gain is represented by K, and R represents the measurement-noise matrix.

PREDICTION STEP

$$\bar{\mathbf{X}}_{\mathbf{k}} = \Phi \hat{X}_k \tag{5.12}$$

$$\delta \bar{x}_{k+1} = 0 \tag{5.13}$$

$$\bar{\mathbf{P}}_{k+1} = \Phi \hat{\mathbf{P}}_{\mathbf{k}} \Phi^T + \mathbf{Q} \tag{5.14}$$

UPDATE STEP

$$\widetilde{b}_{k+1} = z - H_{k+1}\bar{X}_{k+1} \tag{5.15}$$

$$H = \frac{\partial \rho}{\partial \mathbf{X}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.16}$$

$$K_{k+1} = \bar{P}_{k+1}H_{k+1}^T[H_{k+1}P_{k+1}H_{k+1}^T + R]^{-1} \tag{5.17}$$

$$\delta\hat{x}_{k+1} = K_{k+1}\widetilde{b}_{k+1} \tag{5.18}$$

$$\hat{P}_{k+1} = \bar{P}_{k+1} - K_{k+1}H_{k+1}\bar{P}_{k+1} = [I - K_{k+1}H_{k+1}]\bar{P}_{k+1} \tag{5.19}$$

$$\hat{X}_{k+1} = \bar{X}_{k+1} + \delta\hat{x}_{k+1} \tag{5.20}$$

Through careful initialization of the statistics surrounding the estimate, like the sensor noise matrix and covariance matrix, we can implement the EKF into the GNC operations. However, if not modeled properly, the EKF may diverge and result in a scenario called "smugness," where the model believes its correct and rejects new sensor measurements. With the state estimate, the Satbot can determine the necessary control output to realize thruster actuation.

Chapter 6

Control System

The control system plays the final role in developing the implementation of autonomous GNC operation. A controller is implemented to determine the required control inputs necessary from the previously determined deviations from the ideal trajectory or state (provided from guidance and navigation). With the control output vector, which consists of the required force and torques, the Satbot determines the required firing times for each thruster and actuates accordingly with a pulse-width modulation (PWM) scheme. A variety of possible controllers exist and present their own advantages. However, when considering the primary objective of achieving autonomy, only a few controllers are deemed suitable to support the proposed GNC operations, where stable performance is maintained for potentially varying system dynamics from physical aggregation. Therefore, the designed controller must maintain robustness despite the varying configurations and scale of the aggregate system.

In the next section, we survey a number of common controllers and remove some accordingly based on our conceptual requirements. With the selected controller design, we discuss in greater detail the conceptual background that structures the framework for autonomous control.

6.1 Survey of Common Controllers

We list a variety of controller schemes below:

1. **Proportional-Integral-Derivative (PID)**: This is the most fundamental controller design based implementing three controller gains to modify

the system response. [14] The PID controller is effective, robust, and simple, making it a popular choice for many applications. However, these three gains must be predetermined through either trial and error or the Ziegler Nicohls tuning methodology. Achieving autonomous operations will prove difficult when considering the possibility of widely varying configurations; the constantly changing mass properties will eventually drive the initially designed controller into instability, unless new controller gains are uploaded. This solution counters the concept of autonomy, dissuading the selection of the PID controller. However, the PID represents a valid starting point in developing the framework of the GNC operation and will suffice in demonstrating the proof of concept of the subsystem reconfiguration algorithm.

2. **Linear-Quadratic Regulator (LQR)**: This controller introduces the optimal control theory (calculus of variations) to minimize the operating cost function by optimizing the control feedback gains [14]. Should the aggregate system's configuration be known prior to assembly, then the proper gains can be uploaded to ensure the optimal response. However, for the reasons listed above for the PID controller, this is disqualified as a candidate for not striving for true autonomy for a varying dynamical system.

3. **Gain Scheduling**: This controller scheme is a popular technique for adaptive control [15]. Gain scheduling allows for a continual modification of system performance based on the certain scenarios. The appropriate gains are determined beforehand and stored on a look-up table on the system. However, this concept does not completely embody adaptive control and still requires knowledge of the configuration of the assembly, which can be known to an extent, but not for N-number cases. With this in mind, we turn our attention to controllers that gives the aggregate system adaptability regardless of any configuration knowledge.

4. **Self-Tuning**: This is a controller that adjusts, or tunes, the gains throughout operation based on performance requirements [15]. A well designed self-tuning controller presents a valid option for enabling autonomous control for an aggregate system.

5. **Model Reference Adaptive Control (MRAC)**: This adaptive controller takes account of two system plants: one reference model for the

ideal system response and another for the true system response [16]. The difference between these two responses are fed into an adaptation law that changes the input gains to achieve the ideal system response. This presents another valid controller option as this reference model remains independent of the true system response (which can correspond to N-number of aggregation).

Ultimately, a control system must establish adaptation to achieve true autonomous control for any number of aggregations. Without adaptation, the control system designed for one individual cell will prove ineffective, as it was designed for a system with different mass properties. However, before attempting to design high fidelity adaptive control, we seek to develop the baseline controller model to demonstrate the reconfiguration algorithm.

6.2 P-D Controller Design

With that being said, we designed a simple P-D controller for the initial stage of testing; the P-D controller resembles a PID controller without the integral term. Therefore, the control output term can be derived from the second order canonical form to that of Equation 6.1.

$$u(t) = K_p e(t) + K_d \frac{de(t)}{dt} \tag{6.1}$$

The error terms, denoted as $e(t)$, can be replaced by the difference between the final desired state and the current state estimate provided from the Kalman filter. Expanding the control output from Equation 6.1, we derive the following notation from Equations 6.2 to 6.4. [17] Now, the state errors are accurately denoted in terms of δ, where the subscript corresponds to the particular element error. Each component of the control output vector, u, is also expanded to visually represent the corresponding forces and torque.

$$F_x = K_{p,position}\delta_x + K_{d,position}\delta_{\dot{x}} \tag{6.2}$$

$$F_y = K_{p,position}\delta_y + K_{d,position}\delta_{\dot{y}} \tag{6.3}$$

$$T_z = K_{p,attitude}\delta_\theta + K_{d,attitude}\delta_{\dot{\theta}} \tag{6.4}$$

The K terms represent the controller gains, where the subscripts p and d correspond to the proportional and derivative terms respectively. These can be

determined from the second order canonical dynamical state, which take into account of the mass properties, natural frequency, ω_n, and damping ratio, ζ. These components are typically predetermined and used to calculate the gain equations from Equations 6.5 to 6.6. We acknowledge that there are additional terms in calculating these gains; however, these terms correspond to the implementation of the integral term, hence their omission.

$$K_p = m(\omega_n^2) \tag{6.5}$$

$$K_d = m(2\zeta\omega_n) \tag{6.6}$$

We note that the mass term is replaced by the largest moment of inertia component for the previous equation when determining the necessary torque output.

6.3 The Thruster Mapping Matrix

After determining the necessary control input for the Satbot to achieve the desired state, the GNC algorithm computes the required firing times for each thruster. However, controlling multiple thrusters on different Satbots simultaneously requires the concept of a scalable thruster mapping matrix, which translates the physical contribution from each thruster into a mathematical model. [18] [19] The diagram in Figure 6.1 visualizes our approach to orient forces and torques equal about each axis for simplicity.

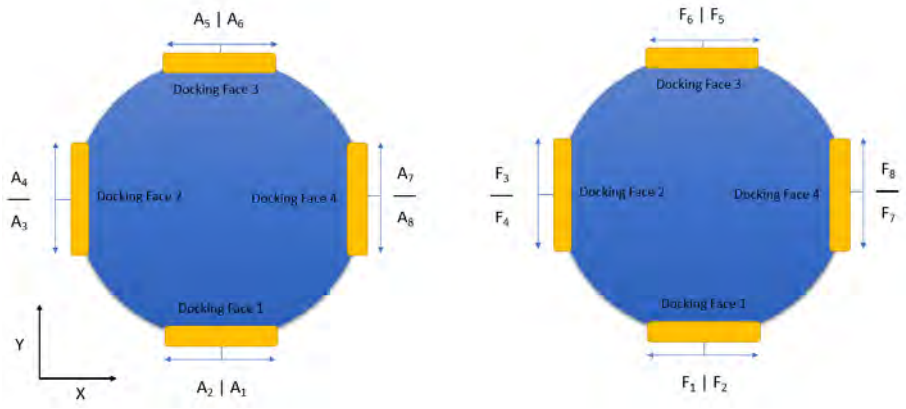


Figure 6.1: An illustration of the top view of a single Satbot model is presented here. In both diagrams, the yellow rectangles represent the docking face mechanisms. The left image depicts the actuators' thrust output as the arrows extending tangentially from the Satbot's circumference. The right image depicts the corresponding resultant force as specified by the thruster label.

To simplify several different thrusters and orientations, each force contribution is collected into a simple table, as seen in Table 6.1.

Furthermore, firing a single thruster generates a body-axis torque. Since the thrusters are mounted on the edge of the Satbot's circular base, determining each thruster's torque calculation is straight forward. We can map the Satbot's body-axis torque similarly to Table 6.1. When combining the body-axis force (excluding the z-component) and body-axis torque contributions (about the z-axis) from each thruster, we create its thruster mapping matrix, as seen in Equation 6.7. Here, the first two rows represent the forces in x and y components respectively, and the third row represents the torque components. Each column corresponds to the particular thruster of that current value; e.g., column one

Table 6.1: Directional force contributions from each thruster for the Satbot

Thruster	Resultant Body-Axis Force			
	$+\hat{x}$	$-\hat{x}$	$+\hat{y}$	$-\hat{y}$
1		**x**		
2	**x**			
3			**x**	
4				**x**
5	**x**			
6		**x**		
7				**x**
8			**x**	

corresponds to thruster one. The benefit to this mapping matrix lies in its scalability, since incorporating more or less thrusters will simply change the number of columns. Additionally, the thruster mapping matrix can be modified accordingly to the location and orientation of each thrusters.

$$\mathbf{M} = \begin{bmatrix} -1 & +1 & 0 & 0 & +1 & -1 & 0 & 0 \\ 0 & 0 & +1 & -1 & 0 & 0 & -1 & +1 \\ -1 & +1 & -1 & +1 & -1 & +1 & -1 & +1 \end{bmatrix} \tag{6.7}$$

Another benefit to the thruster mapping matrix arises in its conceptual simplicity in devising autonomous control as the Satbot will understand which thrusters to actuate. Continuing with Figure 6.1, if the Satbot identifies that it must move *only* in the positive x-direction (relative to its body frame), then the thruster mapping matrix dictates that the Satbot must use both thrusters two and five. Firing individual thrusters contribute to a resultant rotation, which may not be desirable in achieving the final state. Therefore in this example, the system must identify two (or more) thrusters that generate positive force contributions while offsetting each other's torque contribution.

With this concept, we form the equation that demonstrates how the control output is achieved by the set of thrusters, as seen in Equation 6.8. The control output vector, thruster mapping matrix, and firing forces vector are represented by $\vec{u}$, $\mathbf{M}$, and $\vec{f}$ respectively.

$$\vec{u} = \mathbf{M}\vec{f} \tag{6.8}$$

The Satbot needs to solve for the necessary forces and torques to be supplied by each thruster to achieve the desired control output, which is previously determined by the selected controller scheme. Since $\mathbf{M}$ is a non-square matrix, the pseudoinverse, denoted as $\mathbf{M}^{+}$, is computed to find the appropriate firing times. The pseudoinverse can be achieved through multiple techniques, we apply a simple *right inverse* to the mapping matrix. [cite stuff] This is a computationally sound way in deriving the pseudoinverse because $\mathbf{M}$ will always be linearly independent in its rows ($\mathbf{M}$ is always a 3x8 matrix in our scenarios). The pseudoinverse can be calculated via Equation 6.9, where $\mathbf{A}$ represents a generic matrix that is linearly independent in its rows.

$$\mathbf{A}^{+} = \mathbf{A}^{T}(\mathbf{A}\mathbf{A}^{T})^{-1} \tag{6.9}$$

Calculating the required firing forces is shown in Equation 6.10. A scaling factor of two is applied to account for both the positive and negative contributions of the thrusters. [19] In essence, if a force is requested in the positive x-direction, thruster two and thruster five would be activated, where each thruster provided half the requested force. Additionally, scaling the firing forces vector allows the Satbot to ignore any negative force components from the calculation. Logically in terms of the solenoid valve's unidirectional output, a given thruster cannot contribute a negative force, but rather a positive force in its fixed directional output.

$$\vec{f} = 2\mathbf{M}^{+}\vec{u} \tag{6.10}$$

However, two major design considerations changes the thruster mapping matrix. The thruster positions are not exactly on the radius of the circular base; therefore, the positions of each thruster must be accurately accounted (see section *Thruster Specifications and Thrust Validation*). A more accurate depiction of the directional thruster output for the Satbot is provided below in Figure 6.2.

Now, the thruster mapping matrix for the current Satbot design can be portrayed as seen in Equation 6.11. Here, we use $\tau = 0.1725$ to represent the torque to shorten the length of the matrix.

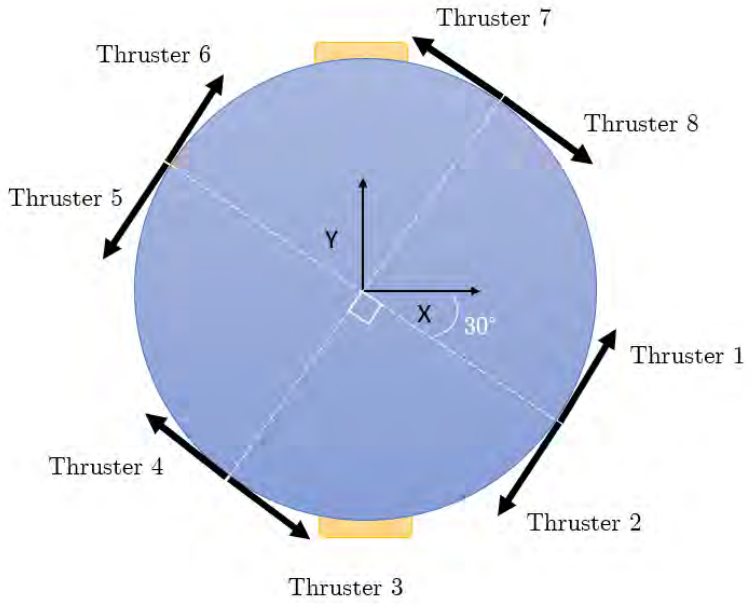


Figure 6.2: The top view thruster model of the actual Satbot design.

$$\mathbf{M} = \begin{bmatrix} -0.5 & +0.5 & -0.866 & +0.866 & +0.5 & -0.5 & +0.866 & -0.866 \\ -0.866 & +0.866 & +0.5 & -0.5 & +0.866 & -0.866 & -0.5 & +0.5 \\ -\tau & +\tau & -\tau & +\tau & -\tau & +\tau & -\tau & +\tau \end{bmatrix} \tag{6.11}$$

6.4 Pulse Width Modulation Scheme

Since the thrusters actuate in an ON/OFF manner, a modulation scheme must be implemented to perform the maneuvers. [20] Though a variety of modulators may achieve the proper control, like Schmitt Triggers and sigma-delta modulators, we opt for the common pulse-width modulator (PWM) to enable Satbot functionality for ease of initial testing. [21]

The PWM converts the requested, non-negative firing forces to solenoid valve

opening times. This results in impulses for each thruster for a defined actuation interval or duty cycle. This active time for a given impulse can be determined from Equation 6.12 and is represented by $T_{on,i}$.

$$T_{on,i} = \frac{f_i}{F_{max}} t_{PWM} \tag{6.12}$$

The maximum force output represents the physical capability of the thruster and is denoted as F_{max} and has been determined through experimentation. The PWM pulse duration is t_{PWM} and is determined by the duty cycle and total modulation period. Each component of the requested firing forces is denoted by f_i.

Additionally, if the determined firing time is less than the minimum opening time of the solenoid valve (approximately 10 ms), then that thruster's contribution is set to zero. If the requested force exceeds the maximum force capability that the thruster can supply, then the firing time is constrained to the maximum pulse duration.

Chapter 7

Spacecraft Subsystem Reconfiguration

The concept of multi-satellite aggregation introduces the importance of appropriate subsystem reconfiguration, in which the space system must manage N-number of subsystems while adhering to the fundamentals of spacecraft operations. When considering the GNC subsystem, new questions appear regarding the logistics of operation. Firstly, how is spacecraft control ensured for varying configurations and mass properties? Will all actuators remain necessary to function; if not, how can the aggregate system automatically determine which actuators to maintain and disable? In answering this question, we develop an algorithm that supports the N-number configuration of Satbots by determining the aggregate mass properties and remapping of the collective thrusters.

Recall the previously proposed GNC operations. After performing navigation, the Satbot will identify whether or not a docking event has occurred. Upon recognizing an active docking face, the reconfiguration portion will trigger. A single Satbot of the N-Satbot assembly will randomly be selected as the reference Satbot, from which all the other Satbots will measure their position. After performing the algorithm, the new aggregate properties will be sent to the initially selected reference Satbot, where the bulk of the remainder GNC operations is performed. Finally, the reference Satbot distributes the collective firing times for each Satbot so that the aggregate system responds as a whole.

Now, we discuss the enabling properties that lays the foundation for the reconfiguration algorithm.

7.1 Spacecraft Identification File

To calculate the aggregate properties for the system, i.e., the new center of mass, moments of inertia, and thruster mapping matrix, we need a system to account for the parameters of N-number of Satbots. Therefore, for each Satbot, we define "spacecraft identification file," or a collection of its GNC components relative to its body axis. This spacecraft identification file is intentionally generic as this collection can be extended to include other resources and parameters. With the support of the data transport layer from the proposed architecture, the allocation of the pertinent information can be seamlessly implemented at specific points in the algorithm to support the calculations as needed. For a given Satbot, the spacecraft identification file is comprised of the parameters seen in Table 7.1.

Table 7.1: The spacecraft identification file parameters

Number	Parameter	Description	Size
1	Spacecraft Identifier	The spacecraft identification tag	[1x1]
2	Docking Status	This provides the active status of each docking face	[2x8]
3	Mass	Current spacecraft mass	[1x1]
4	Moments of Inertia	Spacecraft moments of inertia	[3x3]
5	Docking Face Position	Position of the 4 docking faces	[3x4]
6	Sensor Position	Position of all sensors	[1x3]
7	Thruster Position	Position of all thrusters	[3x8]
8	Force Matrix	Directional forces output	[3x8]
9	Torque Matrix	Directional torque output	[3x8]
10	Thruster Mapping Matrix	Combined force and torque matrix	[3x8]
11	Direction Cosine Matrix	Orientation transformation from body frame to the inertial frame	[3x3]

7.2 Reconfiguration Algorithm

When aggregation occurs, each Satbot exports its identification file through the data transport layer to build a system level "aggregated" identification file, which encompasses all actively docked Satbots. Per the concept of operations

described in the earlier section, the aggregate identification file is distributed to the operating Satbot to continue guidance and control. However, to model the space system dynamics accurately for GNC, the mass properties must be accounted as any instance of thruster misalignment will result in improper control. Therefore, the challenge in GNC reconfiguration depends on the center of mass and moments of inertia, parameters which continually change upon additional aggregation. Ultimately, the devised algorithm must maintain robustness regardless of the number of components to achieve true autonomy.

First, we discuss the theory in calculating the aggregate mass properties, then we go through an example in demonstrating how the algorithm iterates through the established data structure for each calculation.

7.2.1 Key Assumptions

We collect the significant assumptions that enable the reconfiguration calculations so far:

1. The Satbot's center of mass lie at its geometric center.
2. The Satbots are directly docked to each other on the docking face with no additional length between connections.
3. The body frame is represented by the sensor's frame.
4. The proposed aggregation architecture will be responsible for data transfer in and out of the algorithm.
5. A proposed docking mechanism will be responsible for signaling whether an aggregation has occurred.
6. Two equal Satbots (equal mass upon aggregation) will not experience mass differentials during aggregate operations; therefore, the center of mass does not change.
7. All commanded actuations are synchronous.
8. All thrusters possess the same maximum force.

7.2.2 Aggregate Center of Mass

To determine the mass properties, the algorithm must determine the system's aggregate mass and center of gravity. The aggregate mass is simply the

sum of all current Satbot masses, as depicted in in Equation 7.1. Then, the center of gravity calculation is determined as seen in Equation 7.2.

$$M_{agg} = \sum_{i=1}^{N} m_i \tag{7.1}$$

$${}^{\mathrm{A}}\vec{r_{cg}} = \frac{1}{M_{agg}} \sum_{i=1}^{N} m_i^A \vec{r_{cg,i}} \tag{7.2}$$

Denoted in Equation 7.2 is the center of gravity relative to reference frame A. Careful consideration must be given to the center of gravity term as each Satbot distance must be measured from the initially selected reference Satbot. Therefore, as an example, the relative distance to Satbot B's center of gravity to the initial Satbot A's center of gravity can be represented by Equation 7.3.

$${}^{A}\vec{r}_{AB} = {}^{A}\,\vec{r}_B - {}^{A}\,\vec{r}_A \tag{7.3}$$

An important fact to acknowledge is that Satbot A represents the reference point of this assembly, i.e., its center of gravity relative to its center is the origin and is the zero vector. Another key point is that it is not guaranteed that the docked Satbots' body frames align with the body frame of the reference Satbot. In this case, when multiple docked Satbots have differently aligned body frames, transforming between body frames must be accomplished through the use of the direction cosine matrix ($\mathbf{C}$), which has been derived from the measured quaternions. We note that the Pozyx board measures the quaternions relative to the set inertial frame; simple matrix mathematics allow us to derive the transformation between two different Satbot frames.

Then, the position of the Satbot simply becomes the difference between the active docking faces location, which are known from the spacecraft identification file. We present the following example in Figure 7.1, where two Satbots are aligned. For this assembly, the Satbot reference frame is denoted by A. Docking face 4 is the active docking mechanism for both Satbots, which is standardized to be a positive radial distance away from each respective Satbot's body frame. The position of each docking face is known to the Satbot from the ID file.

Because frames A and B are offset by 180 degrees, the appropriate direction cosine matrix must be applied. In this scenario, the more generic position of Satbot B relative to Satbot A is represented by Equation 7.4. We note that A represents the origin (as we calculate subsequent distances from this reference).

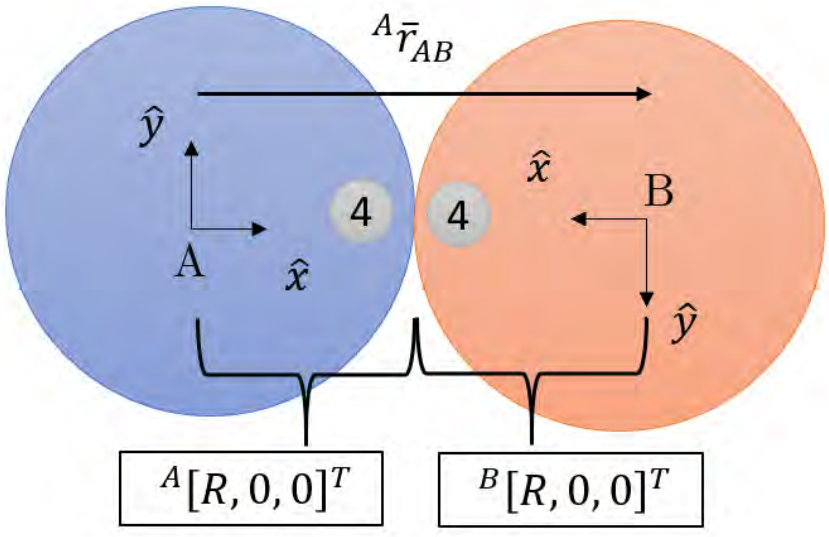


Figure 7.1: An example aggregation between two Satbots with differently oriented body frames.

$$^{A}\vec{r}_{AB} = {}^{A}\vec{r}_{B} = {}^{A}\vec{r}_{DF,4} - {}^{A}\mathbf{C}^{B}\,{}^{B}\vec{r}_{DF,4} \tag{7.4}$$

Performing this quick calculation (considering that the direction cosine matrix rotates frame B -180 degrees), one would find that the current distance of Satbot B is $2R$ away from Satbot A. When considering additional Satbots, the process remains the same, but the additional distances bridging the Satbots together must be accounted. More specifically, if we considered a third Satbot C which were connected to Satbot B, then the distance from Satbot C relative to Satbot A must include two relative distances: the distance from Satbot C to B and from Satbot B to A.

Ultimately, we can expand the center of mass calculation to resemble that of Equation 7.5. This generically represents the center of mass relative to the A-frame, though this idea is applicable for any selected reference.

$$^{A}\vec{r_{cg}} = \frac{1}{m_{agg}}[m_A^A\vec{r_{cg,A}} + m_B(^{\mathrm{A}}\vec{r}_{DF,A} - {}^{\mathrm{A}}\mathbf{C}^{\mathrm{BB}}\vec{r}_{DF,B}) + \\ \ldots + m_N\,{}^{\mathrm{A}}\vec{r_{cg,N}}] \tag{7.5}$$

7.2.3 Aggregate Moments of Inertia

Similarly, the moments of inertia change with each additional Satbot, which can be determined by using the general form for the parallel axis theorem in Equation 7.6.[17] Here, $\mathbf{I}_{\text{agg}}$ represents the aggregate moment of inertia tensor. $\mathbf{I}_{\text{cg, i}}$ represents the inertia tensor for a given Satbot. $\mathbf{E}_3$ is the identity matrix. From before, the Satbot reference frames may differ, hence the transformation of the inertia tensor must be applied as shown in Equation 7.7.

$$ {}^{\text{A}}\mathbf{I}_{\text{agg}} = \sum_{i=1}^{N} {}^{\text{A}}\mathbf{I}_{cg,i} + m_i[({}^{\text{A}}\vec{R}_{cg,i} \cdot {}^{\text{A}}\vec{R}_{cg,i})\mathbf{E}_3 - {}^{\text{A}}\vec{R}_{cg,i} \otimes {}^{\text{A}}\vec{R}_{cg,i}] \tag{7.6} $$

$$ {}^{\text{A}}\mathbf{I}_{cg} = ({}^{\text{A}}\mathbf{C}^{\text{B}})({}^{\text{B}}\mathbf{I}_{cg,B})({}^{\text{A}}\mathbf{C}^{\text{B}})^T \tag{7.7} $$

Combining Equations 7.6 and 7.7 expands the parallel axis theorem to become Equation 7.8 for N-number of Satbots.

$$ \begin{aligned} {}^{\text{A}}\mathbf{I}_{\text{Agg}} = {}^{\text{A}}\mathbf{I}_{cg,A} + m_A[({}^{\text{A}}\vec{R}_{cg,A} \cdot {}^{\text{A}}\vec{R}_{cg,A})\mathbf{E}_3 \\ - {}^{\text{A}}\vec{R}_{cg,A} \otimes {}^{\text{A}}\vec{R}_{cg,A}] + ... + ({}^{\text{A}}\mathbf{C}^{\text{i}})({}^{\text{i}}\mathbf{I}_{cg,i})({}^{\text{A}}\mathbf{C}^{\text{i}})^T \\ + m_i[({}^{\text{A}}\vec{R}_{cg,i} \cdot {}^{\text{A}}\vec{R}_{cg,i})\mathbf{E}_3 - {}^{\text{A}}\vec{R}_{cg,i} \otimes {}^{\text{A}}\vec{R}_{cg,i}] \end{aligned} \tag{7.8} $$

7.2.4 Aggregate Mapping Matrix

Lastly, the change in the center of mass and the addition of thrusters will modify the thruster mapping matrix. Since the Satbots are designed with eight similar thrusters, the size of $\mathbf{M}$ simply becomes [3x8N] for N-number of Satbots. More importantly, the force and torque contributions change according to the selected reference frame and the thruster distances from the new center of mass. Therefore, the appropriate direction cosine matrix must transform the relative thruster position and directional force output vectors. As a result, the new torque contributions from each thruster must be recalculated, then the aggregate mapping matrix can be formed.

Continuing with the GNC operations with the new aggregate mapping matrix, the Satbot must calculate the firing times for every thruster (8N number of thrusters). Additionally, each component of the mapping matrix is tagged with the Satbot's identifier so that the aggregate system always maps the correct

thrusters to the correct Satbot.

Another important consideration requires that the thrusters adjacent to the active docking faces deactivate to prevent potential damage or interference to any Satbot. Pairing the aggregated docking status with **M** allows the Satbots (and thus future space systems) to identify which thrusters to disable. This can be seen in an example provided in Table 7.2. Here, DF and DS correspond to the docking face number and the docking face status respectively. Each docking face is aligned with its corresponding closest thruster and is marked either 1/0 to depict an active docking face. The rest of the table correspond to the thruster mapping matrix.

Table 7.2: An example of an aggregate mapping matrix paired with the docking status

DF:	1	1	2	2	3	3	4	4	1	1	2	2	3	3	4	4
DS:	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1
Fx:	-1	1	0	0	1	-1	0	0	1	-1	0	0	-1	1	0	0
Fy:	0	0	1	-1	0	0	-1	1	0	0	-1	1	0	0	1	-1
Tz:	-1	1	-2	2	-1	1	0	0	-1	1	-2	2	-1	1	0	0

7.2.5 Algorithmic Process

The algorithm is driven by the influx of aggregate information, which is organized into a central data structure to support calculations; an example of this data structure is presented in Table 7.3, which corresponds to Figure 7.2. For brevity, the columns for the moments of inertia, thruster positions, and force matrix are omitted from Table 7.3. The data structure holds a number of rows equal to the number of total active docking faces. For example, if there are only two Satbots in an aggregate system, each Satbot has one active docking face (they are docked to each other); this results in two rows. Should three Satbots be aggregated together, then a total of four docking faces are active (one from the end points, and two from the central Satbot). As a reminder, the Pozyx measures the direction cosine matrix relative to the inertial frame, and the docking face locations are measured relative to each Satbot's body frame. The attachment column provides the Satbot insight as to what is currently attached to its docking face. The "accessed" column represents a counter that prevents the algorithm for accounting for a docked Satbot more than once.

Table 7.3: An example of the data structure established to support the algorithm's calculations.

ID	Mass	Docking Face	r_{DF}	DCM	Attachment	Accessed
A	10 kg	4	[+1, 0]	${}^{A}\mathbf{C}^{I}$	B	1
B	10 kg	3	[0, +1]	${}^{B}\mathbf{C}^{I}$	C	0
B	10 kg	4	[+1, 0]	${}^{B}\mathbf{C}^{I}$	A	0
C	10 kg	4	[+1, 0]	${}^{C}\mathbf{C}^{I}$	B	0

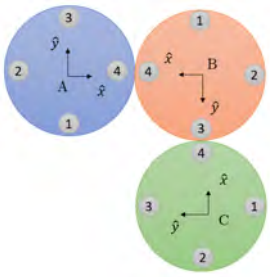


Figure 7.2: An example of an aggregate system consisting of three different Satbots.

With the constructed data structure, the algorithm understands which Satbots are involved in the aggregate system. The following describes the steps of the algorithm.

1. **Select a random Satbot as the reference frame and origin.** From this, the center of mass, moments of inertia, and aggregate mapping matrix are computed. To accommodate a decentralized network of Satbots with this computational architecture, any Satbot is capable of acting as the initial reference.

2. **Iterate through the rows of the ID column to find the first matching instance of the reference Satbot.** Here, we identify the reference Satbot's parameter and store its information to start calculations.

3. **Identify the current row's attached Satbot's ID.** Here, the algorithm identifies the attached Satbot so that its relative position from the reference can be determined.

4. **Mark the "Accessed" column to 1.** This is done to ensure that the algorithm will not account for a row multiple times.

5. **Iterate through the rows of the ID columns to find a match with the previously determined attachment ID.** If this current row's attachment column matches the current reference frame, then the data is stored for calculation, and the "accessed" column is marked to 1. If the attachment does not match the current reference frame, then the code moves on to the next row until this step is satisfied. If no other attachments are identified, then the algorithm skips to step 8.

6. **Calculate the relative position of the current Satbot to the current reference.**

7. **Repeat steps 2 - 6**. Looping through each instance of the reference Satbot ensures that all attached Satbots are accounted.

8. **Assign one of the attachments to the initial reference Satbot as the new reference**. Doing this allows the algorithm to consider additional Satbots that are not directly docked to the initial reference.

9. **Repeat steps 2 - 7 until all Satbots are accounted**.

By incorporating the aforementioned equations in the previous section, the relative positions of each Satbot, aggregate center of mass, moments of inertia, and thruster mapping matrix can be determined relative to the initially selected reference frame.

Consider the aggregate system again in Figure 7.2, where the initially selected reference in Satbot A. The initial reference frame is designated as "A." The code loops through the data structure in Table 7.3 to find the first row whose ID matches the initial frame. This is found to be the first row, where ID is "A." The pertinent information in this row is stored, and the accessed column is set to 1. Then, the algorithm assigns a variable to the identifier in the attachment; this is designated as "B."

Now, the code loops through the entire data structure again, but this time the algorithm attempts to find the row whose ID matches "B." The first instance is row 2; however, this row's attachment ("C") does not match "B." This row is neglected for now, then the algorithm moves on to the next row where the ID is "B." Row 3 satisfies this condition, and its attachment corresponds to "A," the current reference. The accessed counter for row 3 is changed to 1, and the data is stored for calculations. Since there are no more valid rows that match the attachment "B," the algorithm steps out of this inner loop and attempts to find the next row instance that corresponds to the initially selected reference "A" to repeat this process.

Since the all of Satbot A has been accounted for (row 1 and its attachment row 3), the algorithmic selects a new current reference frame, which is chosen to be the attachment of Satbot A; this is "B." In doing so, the algorithm accounts for additional Satbots that are not directly docked to the reference Satbot. We repeat the previous process by trying to match the row's ID with "B." Now, the first matching instance is row 2, whose attachment is Satbot C. Then, as we loop through to find the row that matches "C," we come across row 4. Row 4's attachment corresponds to the current reference "B." The condition is satisfied and calculations are performed.

Chapter 8

Test Setup

For demonstrating the proof of concept of the reconfiguration through the Satbot prototypes, an autonomous distributed resource management system is not required. At this development stage, the extent of the implemented computational architecture is simply the communications protocol that links the Satbots for the test. Each Satbot will possess an Odroid C-2 processor and Pozyx sensor. However, this demonstration will resemble the more traditional "master-slave" framework, i.e., only one Satbot will perform the GNC operations. Additionally, this selected Satbot will perform the subsystem reconfiguration algorithm and calculate the firing times for both Satbots. After this, the primary Satbot transfers this aggregate array of firing times to the other Satbot, where a continuously active receiver code imports these firing times. Then, the aggregate system actuates its collective thrusters concurrently to propel itself towards the predetermined desired state. A successful demonstration of the algorithm would be the observation of correct thruster actuation for the aggregate system. The aggregate system was directed to rotate to a final state of +15 degrees.

Two Satbots will be docked via Velcro prior to the start of the demonstration as specified below in Figure 8.1 and in Figure 8.2.

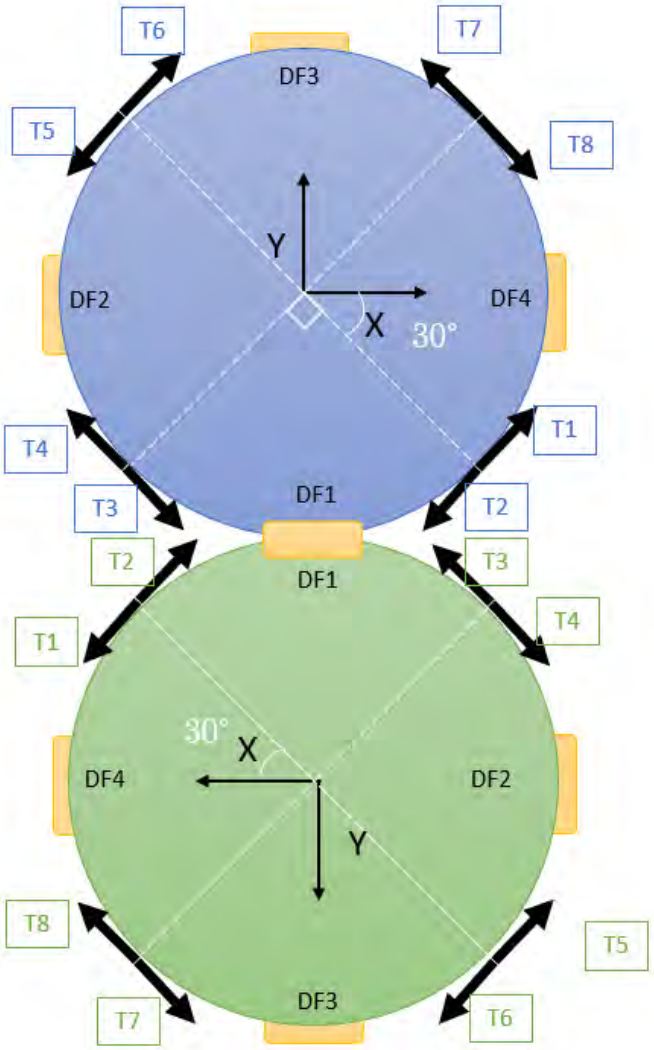


Figure 8.1: The orientation of the two Satbot aggregate system for the algorithmic demonstration.

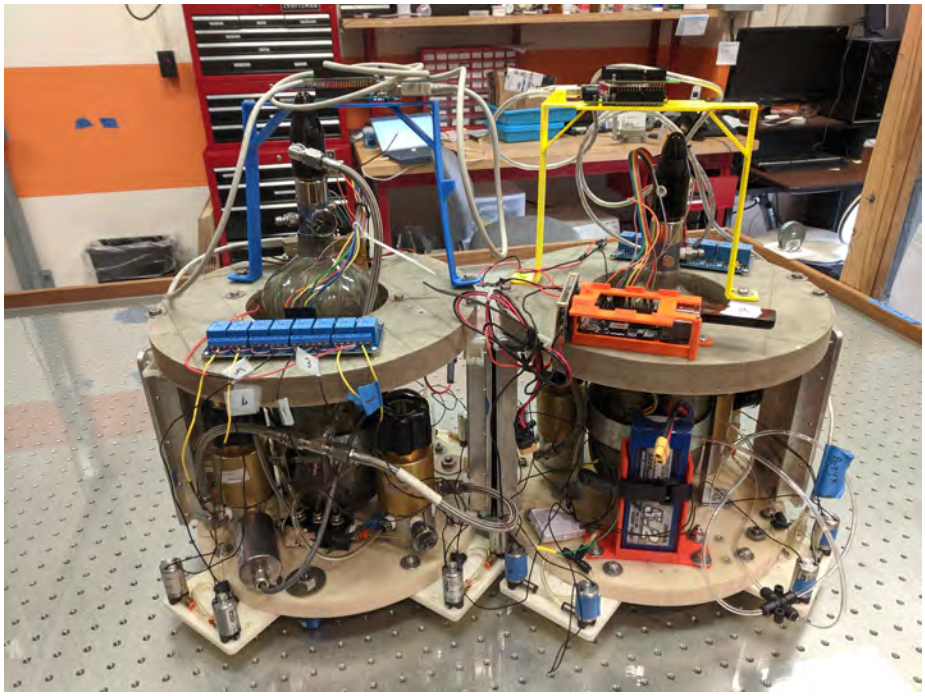


Figure 8.2: The physical hardware of the two Satbot aggregate system.

Chapter 9

Results

Ultimately, the reconfiguration algorithm demonstrated the capability to determine the new center of mass relative to Satbot A (which lies at the docking face between the two Satbots) and calculate the corresponding firing times to rotate the aggregate system, as seen in the thruster history presented in Figure 9.1 and 9.2 for both Satbots. The initial angle reading showed -140 degrees, which corresponds to a control output required positive torque contribution to rotate towards +15 degrees. As a result, both Satbots fire the thruster combinations that contribute to the maximum positive torque (thrusters 6A/8A and thrusters 6B/8B).

However, there exists a large discrepancy in the angle history; the sensor reading suggests a continuously decreasing angle of the aggregate system's frame, which counters the idea of the positive torque contribution (the negative angle reading should be increasing towards 0) and the expected definition of the sensor's frame. This angle history is shown in Figure 9.3. This could be attributed to a couple of factors: the navigational sensor or the divergence of the Kalman filter through smugness. [22] Through several observations, we found that the Pozyx would provide inconsistent measurements while remaining stationary at different initialization times. Additionally, since the Kalman filter may be improperly modeled statistically, the updated covariance matrix over time is driven to a zero matrix, and the state estimate diverges. Data collection ended prior to the Satbot reaching the final destination due to an unexplained shutoff of the sensor during operation.

Also, it is observed that the initialization of the sensor provides large errors in positioning, as seen in Figure 9.4, which starts the control output with improper

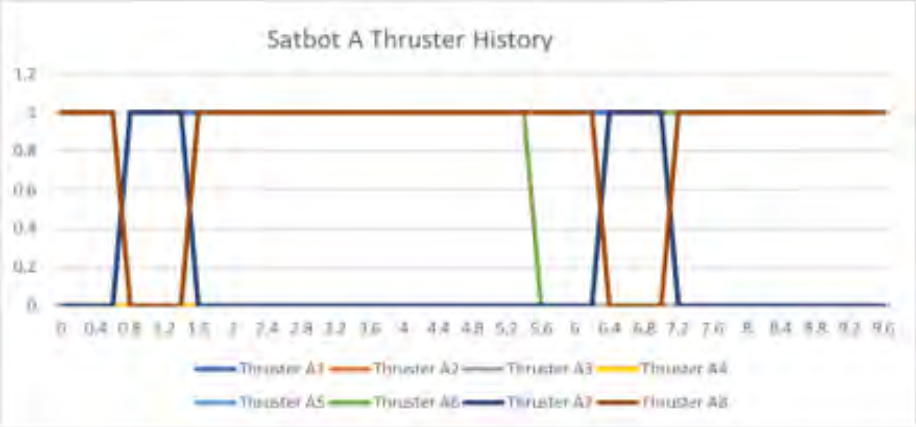


Figure 9.1: The thruster history of Satbot A. It is seen that thrusters 6A and 8A are the most active as the controller scheme dictates the aggregate system to rotate with a maximum positive torque.

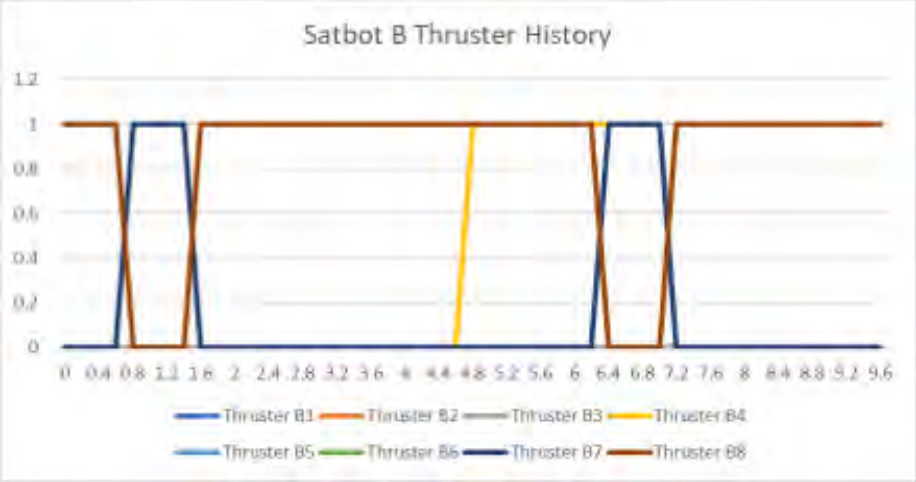


Figure 9.2: The thruster history of Satbot B. It is seen that thrusters 6B and 8B are the most active as the controller scheme dictates the aggregate system to rotate with a maximum positive torque.

control.

Despite these setbacks, the reconfiguration algorithm's demonstration in dictating new thruster commands for an aggregate system was successful, even though further improvements must be made to the general GNC operations.

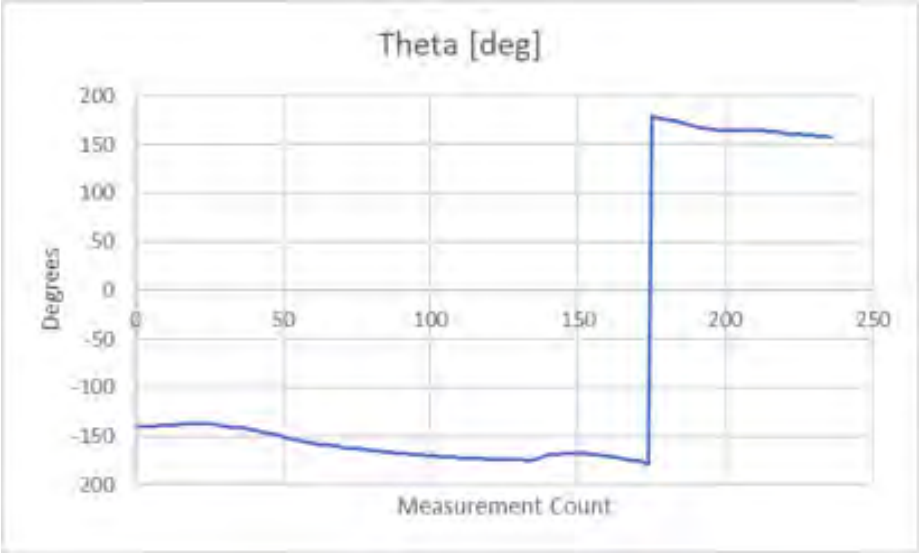


Figure 9.3: The angular history of the aggregate system. Contrary to the firing of the positive torque thrusters, the measurement suggests that the sensor continues to grow more negative.

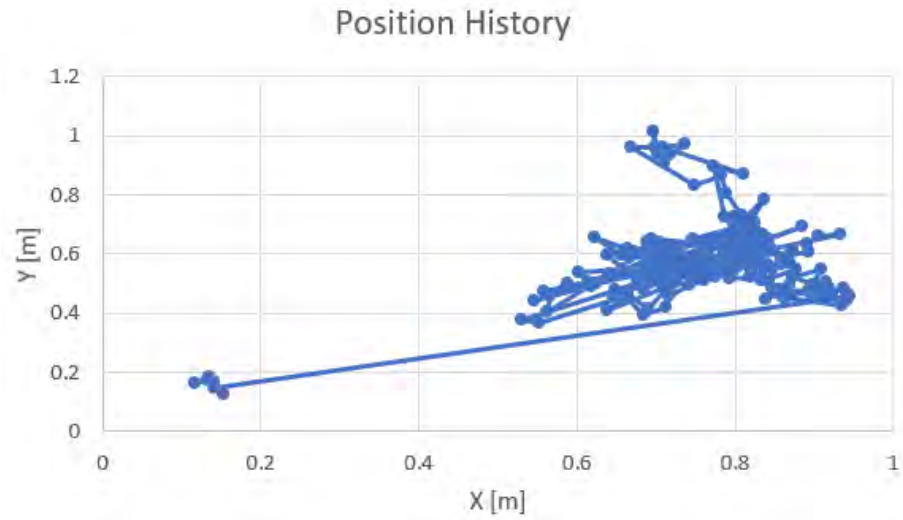


Figure 9.4: The X-Y state history of the aggregate system throughout the test. As seen, the initial points (close to the origin) demonstrate an initialization error in the state, which may lead to improper control.

Chapter 10

Conclusion

Despite the evolution of space technologies and exploration, the current concepts in space systems have seemingly plateaued to a monolithic morphology. However, by transitioning to a new concept of space system cellularization, innovative solutions may be derived from the mastering of on-orbit assembling large-scale spacecraft from a multitude of low-cost platforms. In establishing such an idea, we introduce the notional aggregation architecture to support seamless integration among N-number of spacecraft and autonomous GNC operation.

10.1 Thesis Summary

In this thesis, we explored the application and theory behind developing our RPO testbed to support multi-satellite aggregation. We introduced the physical hardware included to build the Satbot, a cellular spacecraft prototype with a simple GNC thruster subsystem. Each component of GNC operation was discussed in great detail, ranging from the validation and testing of navigational sensor data to the realization of thruster actuation control to the implementation and logistics of the concept of reconfiguration. A simple two Satbot aggregate system was constructed and demonstrated the viability of the reconfiguration algorithm.

10.2 Recommendations for Future Work

Through this thesis, the first instantiation of the aggregation behavior has been demonstrated with the reconfiguration of the GNC subsystem between two spacecraft prototypes. The framework of a newly proposed computational aggregation architecture has been suggested to accommodate the futuristic needs of realizing space system cellularization. Still, significant work must still be accomplished to improve the current state of the research being conducted.

The next steps are three-fold:

- Develop the enabling computational architecture that enables seamless flow of data and resource management across a suite of N-number of systems.
- Establish of true autonomous GNC and RPO with the testbed to support the endeavors and demonstrate the computational architecture.
- Fabricate and improve upon the technology used for cellularization, like an androgynous docking mechanism and the new generation of Satbot models that are more light-weight and possess additional capabilities.

The first point relays the concept that the future rests in the power and capability of the proposed architecture. The overarching system that manages a decentralized network of spacecraft will expand the realm of spacecraft design by enabling the possibility of innovative assemblies and configurations from many low-cost options.

The second point emphasizes the role true autonomy will play in the future of space. By developing the necessary software and interfacing it with the proposed architecture, RPO may be able to step away from the traditional "Master-Slave" rendezvous techniques and usher in a new era of decentralization. Implementing the concepts of adaptive control and spacecraft communication protocol will be the next major stepping stones in demonstrating these concepts on our testbed.

Lastly, by creating a lighter and more capable Satbot along with a suitable docking mechanism, we will be able to expand upon these aforementioned concepts on our testbed.

Bibliography

[1] S. J. Bolton, "The juno mission," *Proceedings of the International Astronomical Union*, vol. 6, no. S269, p. 92100, 2010.

[2] T. Yoshimitsu, J. Kawaguchi, T. Hashimoto, T. Kubota, M. Uo, H. Morita, and K. Shirakawa, "Hayabusa-final autonomous descent and landing based on target marker tracking," *Acta Astronautica*, vol. 65, no. 5, pp. 657–665, 2009.

[3] S. Loff, "Artemis program," Jun 2019.

[4] M. Wall, "Why it'll take israel's lunar lander 8 weeks to get to the moon," Feb 2019.

[5] A. Witze, "China aims for the moon.," *Nature*, vol. 503, pp. 445–6, Nov 2013.

[6] J. M. LONGUSKI, *Optimal control with aerospace applications*. SPRINGER-VERLAG NEW YORK, 2016.

[7] D. Barnhart, L. Hill, M. Turnbull, and P. Will, "Changing satellite morphology through cellularization," in *Proceedings of the AIAA Space Conference 2012*, 2012.

[8] D. Barnhart, R. Duong, and et. al, "The development of dynamic guidance and navigation algorithms for autonomous on-orbit multi-satellite aggregation," in *International Astronautical Congress*, 2019.

[9] J. R. Wertz, D. F. Everett, and J. J. Puschell, *Space Mission Engineering: The New SMAD*. Microcosm Press, 2011.

[10] C. Anderson, C. Vanek, R. Freeman, D. Furlong, A. Kirschbaum, and R. Roy, "Lewis spacecraft mission failure investigation board," tech. rep., 1998.

[11] P. G. Hill and C. R. Peterson, *Mechanics and thermodynamics of propulsion*. Addison-Wesley Longman, 2010.

[12] B. Wie, *Space Vehicle Dynamics and Control*. 1998.

[13] B. D. Tapley, B. E. Schutz, and G. H. Born, *Statistical orbit determination*. Elsevier, 2004.

[14] K. Ogata, *Modern control engineering*. Pearson, 2016.

[15] L. I. Dore, R. Lozano, and M. MSaad, *Adaptive Control Algorithms, Analysis and Applications*. Springer-Verlag London Limited, 2011.

[16] N. T. Nguyen, *Model-reference Adaptive Control: a primer*. Springer, 2018.

[17] C. M. Jewison, B. McCarthy, D. C. Sternberg, D. Strawser, and C. Fang, "Resource aggregated reconfigurable control and risk-allocative path planning for on-orbit servicing and assembly of satellites," in *AIAA Guidance, Navigation, and Control Conference*, p. 1289, 2014.

[18] M. O. Hilstad, "A multi-vehicle testbed and interface framework for the development and verification of separated spacecraft control algorithms," 2002.

[19] R. Zappulla, J. V. Llop, H. Park, C. Zagaris, and M. Romano, "Floating spacecraft simulator test bed for the experimental testing of autonomous guidance, navigation, and control of spacecraft proximity maneuvers and operations," *AIAA/AAS Astrodynamics Specialist Conference*, Sep 2016.

[20] A. Chen, *Propulsion System Characterization for the SPHERES Formation Flight and Docking Testbed*. PhD thesis, 2002.

[21] R. Zappulla, J. Virgili-Llop, and M. Romano, "Spacecraft thruster control via sigmadelta modulation," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 11, p. 29282933, 2017.

[22] D. A. Vallado, *Fundamentals of Astrodynamics and Applications*. 2013.