Optimization-based Methods for Satellite Control

Jacob Willis

CMU-RI-TR-23-84

December 05, 2023

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Zachary Manchester, *chair*
Brandon Lucia,
Guanya Shi,
Kevin Tracy

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.

To those in the future whose lives and work are impacted by mine. To those in the past whose lives and work impact mine. We are all connected.

Abstract

Since 2012, the number of satellites launched into orbit has grown from a maximum of 168 per year to over 2000 per year. Over that same timeframe, incredible advances have been made in control systems for terrestrial robotics and autonomy. Despite the increased quantity of satellites in orbit and the advances made in terrestrial control systems, satellite control systems have not followed the same growth trajectory. This thesis aims to close that gap. We take advantage of computational tools to develop modern approaches to several common satellite control problems. We develop a linear relative motion model for satellites that has higher accuracy than the state of the art. This linear relative motion model allows for convex optimization of low-thrust relative orbital maneuvers, greatly reducing their computational burden. We develop a differential drag satellite formation flying algorithm that computes maneuvers to separate satellites in both the along-track and cross-track directions. The algorithm is a convex optimization that can be solved to optimality within the computational constraints onboard a satellite. The final algorithm we develop is an improved method for detumbling a satellite using electromagnetic torque coils. This algorithm reduces the detumbling time to 50% of conventional methods. In addition to algorithm development, this thesis also describes hardware and flight software development for the PyCubed-mini and Py4 satellite projects. The first PyCubed-mini satellite was launched in November 2023 and the Py4 satellites will be launched in spring 2024.

Acknowledgments

The work I have done for this thesis has been supported by an incredible number of people.

Katie, my wife, deserves acknowledgement first. For all of the changes we have experienced together, for the hundreds of hours of conversations, and for the endless encouragement, thank you. You have truely been a partner, I hope I have been the same.

I wouldn't have made it to Carnegie Mellon without the mentorship of Professors Randy Beard, Cammy Peterson, Tim McClain, and Doran Wilde at BYU. In addition, I would never have gotten to work on satellites without the continued mentorship and friendship of Patrick Walton and the whole PICs team.

My research wouldn't be nearly as interesting or impactful without the continuous support of my advisor, Zac Manchester. Writing this thesis was an amazing reminder of how much I have learned from Zac over the past two years.

One of my greatest joys has been building relationships with the incredible members of my research lab. Thanks (in an attempt at chronological order) to Brian, JJ, Kevin (who taught me orbit dynamics), Benj, Fausto, Paulo, Alex, Chiyen, Brandon, Ben, Swami, Mitch, Giusy, Arun, John, Sofia, Sam, Khai, Aaron, Anoushka, Ibrahima, Will, Ashley. You have made my life better.

A portion of my time has been dedicated to working on a Pocketqube satellite with a team of undergraduate and Master's students. Losha, Karissa, Yashika, Guarav, Krrish, Thomas, and Neil: it was a pleasure to work with and learn from you.

This thesis marks a turning point for me. Earlier this year I made the challenging decision to leave behind the incredible work and people I've gotten to know at CMU and to join another remarkable organization, Albedo. To the people there who took a chance bringing me on as an intern and who have given me the chance to stay for longer: Ankur, Brian, Tom, Brandon, Warren, Topher, AyJay, and Winston. Let's build!

Funding

This work was supported by the United States Department of Defense National Defense Science and Engineering Graduate Fellowship (NDSEG).

Contents

When this thesis is viewed as a PDF, the page header is a link to this Table of Contents.

List of Figures

List of Tables

Chapter 1

Introduction

Over the past two decades the number of satellites launched into Earth orbit has increased from 100 per year to over 2000 per year [66]. While the growth is remarkable when measured in quantity, the growth in satellite autonomy has remained more modest; satellite capability is more limited than satellite quantity. There are a variety of factors at play here, but two of the dominant factors are limited computational resources and conservatism in control system design. As the quantity of satellites continues to grow, satellite hardware and control systems will need to become more capable.

This thesis presents three research problems that extend the state-of-the-art in satellite control. The first two problems focus on orbital dynamics and relative motion between multiple satellites, applying transformations and approximations to make challenging formation flying problems solvable as convex optimizations. The final problem was motivated by noticing the poor performance the classic B-cross detumbling control law experiences and using recent results in discrete-time Lyapunov analysis to improve its performance. The final technical chapter of the thesis is devoted to describing software development for two different satellite experiments: a tiny 5 cm Pocketqube satellite, and the larger 1.5U Py4 cube satellites.

More detail on each chapter follows:

Chapter 2: Background

In this chapter we introduce background concepts that are relevant for spacecraft control system development and that are used in the remaining chapters. Concepts covered include orbital mechanics, attitude dynamics, and rigid body dynamics. Each is approached from a first-principles perspective to help the reader gain understanding and intuition in these challenging topics.

Chapter 3: Convex Optimization of Relative Orbit Maneuvers Using the Kustaanheimo-Stiefel Transformation

As small-satellite constellations continue to grow in size and complexity, there is an increasing need for autonomous relative navigation and control capabilities. Many small satellites utilize non-impulsive low thrust propulsion or manipulation of perturbation forces such as differential drag for orbit control. These low-acceleration control technologies result in long time horizons over which the control actions must be planned and executed. Currently no dynamics model satisfies the computation, accuracy, and generalizability required for autonomous long-time-horizon control. This chapter presents a relative-dynamics model based on the Kustaanheimo-Stiefel transformation. We demonstrate that it achieves equivalent or better accuracy compared to existing relative-orbit models in the literature. In addition, our Kustaanheimo-Stiefel model requires a small number of timesteps per orbit and easily incorporates low-acceleration control inputs. These features make it easily adaptable to convex trajectory optimization methods; we demonstrate this by solving a low-thrust orbital rendezvous problem over a time horizon of 75 orbits with a maximum $20\,\mu\text{m/s}^2$ thrust constraint.

This work was presented at the 2023 IEEE Aerospace Conference [75].

Chapter 4: Propulsion-Free Cross-Track Control of a LEO Small-Satellite Constellation with Differential Drag

In this work, we achieve propellantless control of both cross-track and along-track separation of a satellite formation by manipulating atmospheric drag. Increasing the atmospheric drag of one satellite with respect to another directly introduces along-track separation, while cross-track separation can be achieved by taking advantage of

higher-order terms in the Earth's gravitational field that are functions of altitude. We present an algorithm for solving a multi-satellite formation flying problem based on linear programming. We demonstrate this algorithm in a receding-horizon control scheme in the presence of disturbances and modeling errors in a high-fidelity closed-loop orbital-dynamics simulation. Results show that separation distances of hundreds of kilometers can be achieved by a small-satellite formation in low-Earth orbit over a few months.

This work was performed in collaboration with Giusy Falcone and has been accepted to the 2023 Conference on Decision and Control [20].

Chapter 5: Building a Better B-Dot: Fast Detumbling with Non-Monotonic Lyapunov Functions

Spacecraft detumbling with magnetic torque is an inherently underactuated control problem. Contemporary and classical magnetorquer detumbling methods do not adequately consider this underactuation and suffer from poor performance as a result. These controllers can get stuck on an uncontrollable manifold, resulting in long detumbling times and high power consumption. This work presents a novel detumble controller based on a non-monotonic Lyapunov function that predicts the future magnetic field along the satellite's orbit and avoids uncontrollable configurations, resulting in detumble times that are less than half those of other controllers in the literature, while also converging to lower overall angular rates. We provide a derivation and proof of convergence for our controller, as well as Monte-Carlo simulation results demonstrating its performance in representative use cases.

This work was performed in collaboration with Paulo Fisch and Aleksei Seletskiy and has been accepted to the 2024 IEEE Aerospace Conference [76].

Chapter 6: Satellite Flight Hardware and Software

While development of novel control algorithms is academically interesting, it is only the first step in making them a contribution to the state of the art. To demonstrate the algorithms described in this thesis as well as other modern satellite control methods, we have developed flight hardware and software that falls under two unique

projects. Both projects are derived from the open-source PyCubed avionics [30]. The first is the PyCubed-mini project at Carnegie Mellon University. In this project we are developing a $5 \times 5 \times 5$ cm 1p Pocketqube satellite with powerful computing capabilities including a camera and dedicated embedded computer vision processor for low-cost demonstration of vision-in-the-loop satellite control. The second project is the Py4 project that is a collaboration between Carnegie Mellon University and NASA Ames. This project consists of four 1.5U cube satellites developed at NASA Ames. The satellites will be launched together and perform relative state estimation and maneuvering.

Chapter 2

Background

This chapter provides an introduction to the dynamics of spacecraft. Rather than provide a brief overview, verbose derivations and explanations are made to aid the reader's understanding. We begin by studying orbital dynamics, then proceed to attitude dynamics before combining the formulations into the full state dynamics for a rigid body satellite.

2.1 Orbital Dynamics

2.1.1 Two-Body Dynamics

For two point masses M and m where $M \gg m$, we define the position of m with respect to M as $\mathbf{r} \in \mathbb{R}^3$, and the velocity of m with respect to M as $\mathbf{v} \in \mathbb{R}^3$. We let the magnitude of the position and velocities be $r = \|\mathbf{r}\|$ and $v = \|\mathbf{v}\|$ respectively. The gravitational potential of m is

$$V(r) = -\frac{\mu}{r} \tag{2.1}$$

where $\mu = GM$ is the standard gravitational parameter, with G the gravitational constant. The acceleration of m due to the gravitational potential field is the negative

gradient of V with respect to $\mathbf{r}$,

$$\ddot{\mathbf{r}} = -\nabla_{\mathbf{r}} V = -\frac{\mu}{r^3}\mathbf{r}. \tag{2.2}$$

This is the two-body equation and when $M = M_{\text{Earth}}$ it governs the dynamics of a satellite in orbit around Earth [72].

The specific angular momentum of m is

$$\mathbf{h} = \mathbf{r} \times \mathbf{v}. \tag{2.3}$$

Taking the time derivative

$$\frac{d}{dt}\mathbf{h} = \mathbf{v} \times \mathbf{v} + \mathbf{r} \times \ddot{\mathbf{r}} \tag{2.4}$$

$$= \mathbf{v} \times \mathbf{v} - \frac{\mu}{r^3}\mathbf{r} \times \mathbf{r} = 0, \tag{2.5}$$

so the specific angular momentum of a two-body orbit is conserved. This also indicates that the angular momentum vector is constant for all $\mathbf{r}$ and $\mathbf{v}$, so the two-body orbital motion of a satellite is planar [53].

Defining the specific potential energy

$$\varepsilon_p = -\frac{\mu}{r} \tag{2.6}$$

and the specific kinetic energy

$$\varepsilon_k = \frac{1}{2}v^2 \tag{2.7}$$

the specific orbital energy is

$$\varepsilon = \varepsilon_k + \varepsilon_p = \frac{1}{2}v^2 - \frac{\mu}{r}. \tag{2.8}$$

Using the fact that

$$r\dot{r} = (\mathbf{r}^T\mathbf{r})^{1/2}\frac{d}{dt}(\mathbf{r}^T\mathbf{r})^{1/2} = (\mathbf{r}^T\mathbf{r})^{1/2}\frac{1}{2}(\mathbf{r}^T\mathbf{r})^{-1/2}(2\mathbf{r}^T\mathbf{v}) = \mathbf{r}^T\mathbf{v} \tag{2.9}$$

and taking the time derivative of ε,

$$\frac{d}{dt}\varepsilon = v\dot{v} + \frac{\mu}{r^2}\dot{r} \tag{2.10}$$

$$= \mathbf{v}^T\ddot{\mathbf{r}} + \frac{\mu}{r^2}\frac{\mathbf{r}^T\mathbf{v}}{r} \tag{2.11}$$

$$= \mathbf{v}^T\left(\ddot{\mathbf{r}} + \frac{\mu}{r^3}\mathbf{r}\right) \tag{2.12}$$

$$= \mathbf{v}^T(\ddot{\mathbf{r}} - \ddot{\mathbf{r}}) = 0 \tag{2.13}$$

we see that specific orbital energy is also conserved.

We will now derive the elliptical form of an orbit by finding a third conserved quantity, known as the Laplace-Runge-Lenz (LRL) vector. Taking the time derivative

$$\frac{d}{dt}(\mathbf{h}\times\mathbf{v}) = \dot{\mathbf{h}}\times\mathbf{v} + \mathbf{h}\times\ddot{\mathbf{r}} \tag{2.14}$$

$$= \mathbf{h}\times\ddot{\mathbf{r}} \tag{2.15}$$

$$= -\frac{\mu}{r^3}\mathbf{h}\times\mathbf{r} \tag{2.16}$$

$$= -\frac{\mu}{r^3}\mathbf{r}\times\mathbf{v}\times\mathbf{r}. \tag{2.17}$$

Now, recalling the vector triple product $a\times b\times c = b(a^Tc) - c(a^Tb)$ and that $r\dot{r} = \mathbf{r}^T\mathbf{v}$,

$$\frac{d}{dt}(\mathbf{h}\times\mathbf{v}) = -\frac{\mu}{r^3}\left(\mathbf{v}(\mathbf{r}^T\mathbf{r}) - \mathbf{r}(\mathbf{r}^T\mathbf{v})\right) \tag{2.18}$$

$$= -\frac{\mu}{r^3}\left(r^2\mathbf{v} - r\dot{r}\mathbf{r}\right) \tag{2.19}$$

$$= -\mu\left(\frac{\mathbf{v}}{r} - \frac{\dot{r}\mathbf{r}}{r^2}\right) \tag{2.20}$$

$$= -\mu\frac{d}{dt}\left(\frac{\mathbf{r}}{r}\right) \tag{2.21}$$

where in the last line we made the substitution

$$\frac{d}{dt}\left(\frac{\mathbf{r}}{r}\right) = \frac{\mathbf{v}}{r} - \frac{\dot{r}\mathbf{r}}{r^2}. \tag{2.22}$$

We've now shown that

$$\frac{d}{dt}\left(\mathbf{h} \times \mathbf{v} + \mu\frac{\mathbf{r}}{r}\right) = 0 \tag{2.23}$$

so we have another conserved quantity,

$$\mathbf{A} = -\mathbf{h} \times \mathbf{v} - \mu\frac{\mathbf{r}}{r}. \tag{2.24}$$

This is the LRL vector; it has magnitude $A = \|\mathbf{A}\|$. We define the true anomaly, θ, to be the angle between $\mathbf{A}$ and $\mathbf{r}$, so their dot product is

$$\begin{aligned} Ar\cos(\theta) &= \mathbf{A}^T\mathbf{r} && \text{(2.25a)} \\ &= -(\mathbf{h} \times \mathbf{v})^T\mathbf{r} - \mu\frac{\mathbf{r}^T\mathbf{r}}{r} && \text{(2.25b)} \\ &= (\mathbf{r} \times \mathbf{v})^T\mathbf{h} - \mu\frac{r^2}{r} && \text{(2.25c)} \\ &= h^2 - \mu r. && \text{(2.25d)} \end{aligned}$$

Solving for r, we find that

$$r = \frac{h^2}{\mu + A\cos(\theta)} = \frac{p}{1 + e\cos(\theta)} \tag{2.26}$$

where we define the semi-latus rectum $p = h^2/\mu$ and the eccentricity $e = A/\mu$. Equation (2.26) is the equation for a conic section. With $0 \leq e < 1$ it defines the radius of an ellipse, with $e = 1$ a parabola, and with $e > 1$ a hyperbola. When $0 \leq e < 1$, we define the size of an orbit by the average of its minimum and maximum radii, these occur at $\theta = 0$ and $\theta = \pi$ respectively. We refer to this quantity as the semi-major axis (SMA),

$$a = \frac{1}{2}(r_{\min} + r_{\max}) = \frac{p}{1 - e^2}. \tag{2.27}$$

Since $r_{\min}$ occurs at $\theta = 0$ and $\mathbf{A}$ is parallel to $\mathbf{r}_{\min}$, $\mathbf{A}$ is a vector pointing at periapsis, the minimum radius point of an orbit.

The true anomaly θ, semi-major axis a, and eccentricity e are three of the six

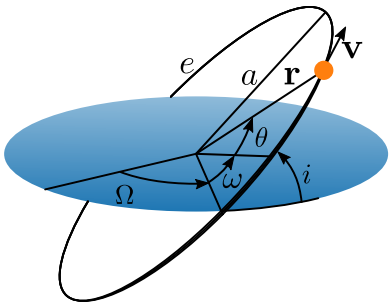


Figure 2.1: Notation used to describe a spacecraft orbital state.

orbital elements. The true anomaly specifies a spacecraft's position on the orbit ellipse, and the semi-major axis and eccentricity define the shape of the orbit ellipse. The three additional orbital elements specify the orientation of the orbit ellipse and require an external reference frame to be defined. For the Earth, this reference frame is the Earth-centered inertial (ECI) frame, with its x-axis aligned with the mean vernal equinox (which is fixed with respect to the stars), z-axis aligned with the Earth's rotation axis, and y-axis defined to complete a right-handed coordinate system. The right angle of the ascending node (RAAN), Ω, defines the angle from the ECI x-axis to the ascending node — the point where the orbit crosses up through the Earth's equatorial plane. The inclination i defines the angle between $\mathbf{h}$ and the ECI z-axis. And the argument of periapsis ω defines the angle from the ascending node to the orbit's periapsis. For circular orbits, the most common for Earth-orbiting satellites, the true anomaly and argument of periapsis are undefined, so the argument of latitude is used. We use θ to refer to both the argument of latitude and the true anomaly. A diagram of the orbital elements is shown in fig. 2.1.

2.1.2 Orbit Perturbations

The previous discussion applies to two point masses existing in a vacuum. In reality there are many perturbations that spacecraft experience. In low-Earth orbit, the perturbations are dominated by atmospheric drag and the non-spherical shape of the Earth.

The drag force experienced by a spacecraft is highly variable, depending on space weather as well as the attitude and surface finish of the spacecraft. For this reason, it

is difficult to produce an exact model. However an approximate model for acceleration due to atmospheric drag is given by

$$\mathbf{a}_D = -\frac{1}{2}\rho S C_D v_a \mathbf{v}_a \tag{2.28}$$

where ρ is the atmospheric density, C_D is the spacecraft's coefficient of drag, and $\mathbf{v}_a$ is the velocity of the satellite with respect to the atmosphere. The atmospheric density can vary by up to two orders of magnitude, but in general decreases exponentially with increased altitude.

The dominant non-spherical gravitational effects can be captured by including the J_2 acceleration [50],

$$\mathbf{a}_{J_2} = -\frac{3}{2}\frac{J_2 \mu R_E^2}{r^5}\begin{bmatrix}\left(1-5\left(\frac{x_3}{r}\right)^2\right)r_x\\ \left(1-5\left(\frac{x_3}{r}\right)^2\right)r_y\\ \left(3-5\left(\frac{x_3}{r}\right)^2\right)r_z\end{bmatrix}, \tag{2.29}$$

where J_2 is the normalized J_2 spherical-harmonic coefficient for the Earth's gravitational field, R_E is the radius of the Earth, and r_x, r_y, r_z are the components of $\mathbf{r}$ along the axes of the Earth-centered inertial (ECI) coordinate frame. Equation (2.29) captures the oblateness of the Earth, however the Earth has many additional gravitational deviations that it does not capture. These require a higher order gravity model based on a spherical harmonic expansion [53].

In addition to drag and non-spherical gravity, Earth-orbiting satellites experience perturbations from the Sun and Moon's gravity, and from the Sun's radiation pressure. In low-Earth orbit, these perturbations are orders of magnitude less than non-spherical gravity and drag [53].

2.2 Attitude Dynamics

The previous section described the position and velocity dynamics of an Earth-orbiting satellite. In this section we present the attitude dynamics. To do so, we must first define several reference frames and the notation used to represent quantities expressed in each frame. We let $\mathcal{B}$ denote the satellite's body-fixed frame, and $\mathcal{N}$ denote an

inertial frame - typically the Earth-centered inertial (ECI) frame.

2.2.1 Rotation Matrices

For a vector position ${}^{\mathcal{N}}\mathbf{r} \in \mathbb{R}^3$ expressed in the inertial frame, the expression in the body frame can be found with a rotation matrix ${}^{\mathcal{B}}Q^{\mathcal{N}} \in SO(3)$

$$ {}^{\mathcal{B}}\mathbf{r} = {}^{\mathcal{B}}Q^{\mathcal{N}}{}^{\mathcal{N}}\mathbf{r}. \tag{2.30} $$

Rotation matrices are orthogonal and have a unitary determinant, so

$$ I = ({}^{\mathcal{B}}Q^{\mathcal{N}})^{-1}{}^{\mathcal{B}}Q^{\mathcal{N}} = ({}^{\mathcal{B}}Q^{\mathcal{N}})^{T}{}^{\mathcal{B}}Q^{\mathcal{N}} = {}^{\mathcal{N}}Q^{\mathcal{B}}{}^{\mathcal{B}}Q^{\mathcal{N}}. \tag{2.31} $$

One way to view the rotation matrix ${}^{\mathcal{N}}Q^{\mathcal{B}}$ is that its rows are the $\mathcal{N}$ frame unit vectors expressed in the body frame, or equivalently its columns are the $\mathcal{B}$ frame unit vectors expressed in the $\mathcal{N}$ frame:

$$ {}^{\mathcal{N}}Q^{\mathcal{B}} = \begin{bmatrix} {}^{\mathcal{B}}n_1^T \\ {}^{\mathcal{B}}n_2^T \\ {}^{\mathcal{B}}n_3^T \end{bmatrix} = \begin{bmatrix} {}^{\mathcal{N}}b_1^T & {}^{\mathcal{N}}b_2^T & {}^{\mathcal{N}}b_3^T \\ & & \end{bmatrix} \tag{2.32} $$

With this in mind, multiplying a vector by a rotation matrix is taking the dot product of the vector with each frame direction in the frame the vector is being transformed into.

Rotation Matrix Kinematics

In this section we derive the kinematic relationship between a rotation matrix and the angular velocity of a reference frame. A visualization of the quantities described is shown in fig. 2.2. The $\mathcal{N}$ frame represents an inertially-fixed frame, the $\mathcal{B}$ frame represents a body-fixed frame that is rotating with some angular velocity ${}^{\mathcal{B}}\boldsymbol{\omega}$, ${}^{\mathcal{B}}\mathbf{t}^{\mathcal{N}}$ is the (potentially time varying) translation from the inertial frame to the body-fixed frame, and the vector ${}^{\mathcal{B}}\mathbf{x}$ represents a body-frame vector that has the inertial

representation

$$^{\mathcal{N}}\mathbf{x} = {}^{\mathcal{B}}\mathbf{t}^{\mathcal{N}} + {}^{\mathcal{N}}Q^{\mathcal{B}\mathcal{B}}\mathbf{x}. \tag{2.33}$$

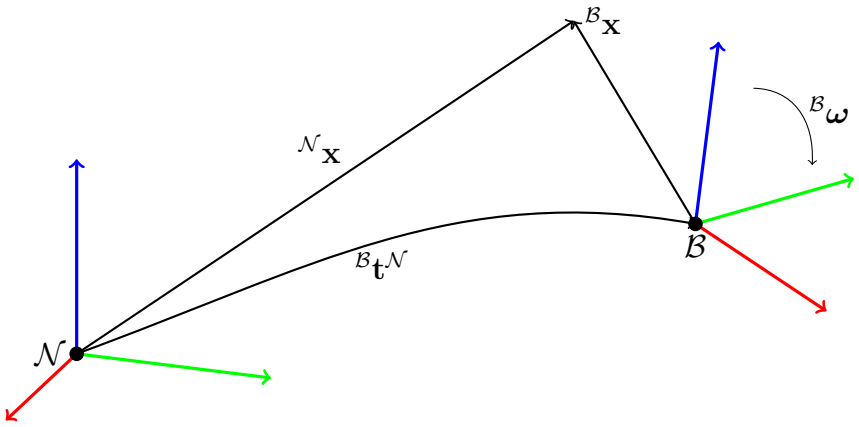


Figure 2.2: Visualization of coordinate frames for rotation matrix kinematics derivation.

By the transport theorem, if the $\mathcal{B}$ frame is rotating with angular velocity $^{\mathcal{B}}\boldsymbol{\omega}$, a body vector $^{\mathcal{B}}\mathbf{x}$ with body-referenced velocity $^{\mathcal{B}}\dot{\mathbf{x}}$ has inertial velocity

$$\begin{aligned} ^{\mathcal{N}}\dot{\mathbf{x}} &= {}^{\mathcal{B}}\dot{\mathbf{t}}^{\mathcal{N}} + {}^{\mathcal{N}}Q^{\mathcal{B}}\left({}^{\mathcal{B}}\dot{\mathbf{x}} + {}^{\mathcal{B}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\mathbf{x}\right) && \text{(2.34a)} \\ &= {}^{\mathcal{N}}Q^{\mathcal{B}\mathcal{B}}\dot{\mathbf{x}} + {}^{\mathcal{N}}Q^{\mathcal{B}\mathcal{B}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\mathbf{x}. && \text{(2.34b)} \end{aligned}$$

Now, by the chain rule, we also have

$$^{\mathcal{N}}\dot{\mathbf{x}} = \frac{d}{dt}\left({}^{\mathcal{N}}\mathbf{x}\right) = \frac{d}{dt}\left({}^{\mathcal{B}}\mathbf{t}^{\mathcal{N}} + {}^{\mathcal{N}}Q^{\mathcal{B}\mathcal{B}}\mathbf{x}\right) = {}^{\mathcal{B}}\dot{\mathbf{t}}^{\mathcal{N}} + {}^{\mathcal{N}}\dot{Q}^{\mathcal{B}\mathcal{B}}\mathbf{x} + {}^{\mathcal{N}}Q^{\mathcal{B}\mathcal{B}}\dot{\mathbf{x}}. \tag{2.35}$$

Equating eqs. (2.34) and (2.35) and eliminating terms,

$$\begin{aligned} {}^{\mathcal{B}}\dot{\mathbf{t}}^{\mathcal{N}} + {}^{\mathcal{N}}\dot{Q}^{\mathcal{B}\mathcal{B}}\mathbf{x} + {}^{\mathcal{N}}Q^{\mathcal{B}\mathcal{B}}\dot{\mathbf{x}} &= {}^{\mathcal{B}}\dot{\mathbf{t}}^{\mathcal{N}} + {}^{\mathcal{N}}Q^{\mathcal{B}\mathcal{B}}\dot{x} + {}^{\mathcal{N}}Q^{\mathcal{B}\mathcal{B}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\mathbf{x} && \text{(2.36a)} \\ {}^{\mathcal{N}}\dot{Q}^{\mathcal{B}\mathcal{B}}\mathbf{x} &= {}^{\mathcal{N}}Q^{\mathcal{B}\mathcal{B}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\mathbf{x} && \text{(2.36b)} \\ {}^{\mathcal{N}}\dot{Q}^{\mathcal{B}\mathcal{B}}\mathbf{x} &= {}^{\mathcal{N}}Q^{\mathcal{B}\mathcal{B}}\hat{\boldsymbol{\omega}}^{\mathcal{B}}\mathbf{x} && \text{(2.36c)} \end{aligned}$$

where we defined

$$\hat{\boldsymbol{\omega}} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}. \tag{2.37}$$

Since in eq. (2.36c) ${}^{\mathcal{B}}\mathbf{x}$ is an arbitrary vector the terms multiplying it define the rotation matrix derivative,

$${}^{\mathcal{N}}\dot{Q}^{\mathcal{B}} = {}^{\mathcal{N}}Q^{\mathcal{B}\mathcal{B}}\hat{\boldsymbol{\omega}}. \tag{2.38}$$

2.2.2 Unit Quaternions

While rotation matrices are easy to work with for hand calculations, unit quaternions (simply referred to as quaternions from now on) are the preferred attitude representation because they require fewer parameters and are easier to renormalize. Like rotation matrices, quaternions are a singularity-free attitude representation. A quaternion $q \in S^3 \in \mathbb{R}^4$ is a four-parameter vector with unit norm. Often quaternions are written as the concatenation of a scalar part $s \in \mathbb{R}$ and a vector part $\mathbf{v} \in \mathbb{R}^3$. We use the Hamiltonian convention of placing the scalar part first:

$$q = \begin{bmatrix} s \\ \mathbf{v} \end{bmatrix}. \tag{2.39}$$

For a rotation $\theta \in \mathbb{R}$ around an axis $\mathbf{a} \in \mathbb{R}^3$, the quaternion representing this rotation is

$$q = \begin{bmatrix} s \\ \mathbf{v} \end{bmatrix} = \begin{bmatrix} \cos(\theta/2) \\ \mathbf{a}\sin(\theta/2) \end{bmatrix}. \tag{2.40}$$

For two quaternions q_1 and q_2, quaternion multiplication is defined as

$$q_1 * q_2 = \begin{bmatrix} s_1 \\ \mathbf{v}_1 \end{bmatrix} * \begin{bmatrix} s_2 \\ \mathbf{v}_2 \end{bmatrix} \tag{2.41a}$$

$$= \begin{bmatrix} s_1 s_2 - \mathbf{v}_1^T \mathbf{v}_2 \\ s_1 \mathbf{v}_2 + s_2 \mathbf{v}_1 + \mathbf{v}_1 \times \mathbf{v}_2 \end{bmatrix} \tag{2.41b}$$

$$= \begin{bmatrix} s_1 & -\mathbf{v}_1^T \\ \mathbf{v}_1 & s_1 I + \hat{\mathbf{v}}_1 \end{bmatrix} \begin{bmatrix} s_2 \\ \mathbf{v}_2 \end{bmatrix} \triangleq L(q_1) q_2 \tag{2.41c}$$

$$= \begin{bmatrix} s_2 & -\mathbf{v}_2^T \\ \mathbf{v}_2 & s_2 I - \hat{\mathbf{v}}_2 \end{bmatrix} \begin{bmatrix} s_1 \\ \mathbf{v}_1 \end{bmatrix} \triangleq R(q_2) q_1. \tag{2.41d}$$

With quaternion multiplication defined we can see that the multiplicative identity quaternion is

$$q = \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix} \tag{2.42}$$

and the mulitplicative conjugate is

$$q^\dagger = \begin{bmatrix} s \\ -\mathbf{v} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -I \end{bmatrix} \begin{bmatrix} s \\ \mathbf{v} \end{bmatrix} \triangleq Tq. \tag{2.43a}$$

Note that the conjugate is equivalent to transposing $L(q)$ and $R(q)$:

$$q * q^\dagger = R(q^\dagger) q = R(q)^T q. \tag{2.43b}$$

For a body-to-inertial quaternion ${}^{\mathcal{N}}q^{\mathcal{B}}$, the rotation of a body-frame vector ${}^{\mathcal{B}}\mathbf{x}$ is

$$\begin{bmatrix} 0 \\ {}^{\mathcal{N}}\mathbf{x} \end{bmatrix} = {}^{\mathcal{N}}q^{\mathcal{B}} * \begin{bmatrix} 0 \\ {}^{\mathcal{B}}\mathbf{x} \end{bmatrix} * \left({}^{\mathcal{N}}q^{\mathcal{B}}\right)^\dagger \tag{2.44a}$$

$$= L\left({}^{\mathcal{N}}q^{\mathcal{B}}\right) R\left({}^{\mathcal{N}}q^{\mathcal{B}}\right)^T \begin{bmatrix} 0 \\ {}^{\mathcal{B}}\mathbf{x} \end{bmatrix} \tag{2.44b}$$

$$= R\left({}^{\mathcal{N}}q^{\mathcal{B}}\right)^T L\left({}^{\mathcal{N}}q^{\mathcal{B}}\right) \begin{bmatrix} 0 \\ {}^{\mathcal{B}}\mathbf{x} \end{bmatrix} \tag{2.44c}$$

Defining the matrix

$$H = \begin{bmatrix} 0 \\ I \end{bmatrix} \tag{2.45}$$

we have

$${}^{\mathcal{N}}\mathbf{x} = H^T L\left({}^{\mathcal{N}}q^{\mathcal{B}}\right) R\left({}^{\mathcal{N}}q^{\mathcal{B}}\right)^T H\,{}^{\mathcal{B}}\mathbf{x} \tag{2.46}$$

so the conversion from quaternion to rotation matrix is

$${}^{\mathcal{N}}Q^{\mathcal{B}} = H^T L\left({}^{\mathcal{N}}q^{\mathcal{B}}\right) R\left({}^{\mathcal{N}}q^{\mathcal{B}}\right)^T H. \tag{2.47}$$

And the quaternion kinematics are

$${}^{\mathcal{N}}\dot{q}^{\mathcal{B}} = \frac{1}{2}{}^{\mathcal{N}}q^{\mathcal{B}} * \begin{bmatrix} 0 \\ {}^{\mathcal{B}}\boldsymbol{\omega} \end{bmatrix} = \frac{1}{2}L({}^{\mathcal{N}}q^{\mathcal{B}})H\,{}^{\mathcal{B}}\boldsymbol{\omega}. \tag{2.48}$$

2.2.3 Angular Rate Dynamics

With a body-frame torque ${}^{\mathcal{B}}\boldsymbol{\tau}$, and inertia matrix J, the body-frame angular rate dynamics of a rigid body obey Euler's equation:

$${}^{\mathcal{B}}\dot{\boldsymbol{\omega}} = J^{-1}\left({}^{\mathcal{B}}\boldsymbol{\tau} - {}^{\mathcal{B}}\boldsymbol{\omega} \times J\,{}^{\mathcal{B}}\boldsymbol{\omega}\right). \tag{2.49}$$

2.3 Rigid Body Dynamics

With the terms defined above, the rigid-body dynamics for a satellite are

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{r}} \\ \dot{\mathbf{v}} \\ {}^{\mathcal{N}}\dot{q}^{\mathcal{B}} \\ {}^{\mathcal{B}}\dot{\boldsymbol{\omega}} \end{bmatrix} = \begin{bmatrix} \mathbf{v} \\ -\frac{\mu}{r^3}\mathbf{r} + \mathbf{a} \\ \frac{1}{2}L({}^{\mathcal{N}}q^{\mathcal{B}})H\,{}^{\mathcal{B}}\boldsymbol{\omega} \\ J^{-1}\left({}^{\mathcal{B}}\boldsymbol{\tau} - {}^{\mathcal{B}}\boldsymbol{\omega} \times J\,{}^{\mathcal{B}}\boldsymbol{\omega}\right) \end{bmatrix} \tag{2.50}$$

where $\mathbf{a}$ is any arbitrary perturbation in the inertial frame, and ${}^{\mathcal{B}}\boldsymbol{\tau}$ is any arbitrary torque in the body frame.

Chapter 3

Relative Kustaanheimo-Stiefel Dynamics

3.1 Introduction

Small-satellite constellations promise increased ground coverage, higher re-visit rates, and improved sensing resolution. However, a key enabling technology for these constellations is effective, autonomous, relative navigation and control. Because of the size, weight, and power constraints of small satellites, non-impulsive control methods using low-thrust propulsion systems [40] and manipulation of perturbation forces through techniques such as differential drag [21] and solar-sails [22] are gaining traction. A large number of models for the relative motion between spacecraft exist; however, these models aren't well-equipped for the constant low acceleration produced by non-impulsive control systems. In particular, the low accelerations result in long time horizons over which the control action must be planned and executed. When formulated using Earth-centered inertial coordinates, the resulting trajectory optimization problems require tens to hundreds of thousands of timesteps—much too large to perform autonomously on an embedded flight computer. In contrast, orbital-element-based models are not generalizable as they must be developed to handle the specific control inputs and perturbations a spacecraft encounters.

Recently, the size of nonlinear trajectory optimization problems for long-horizon

orbital maneuvers has been reduced by transforming the orbital dynamics using the Kustaanheimo-Stiefel (KS) transformation [69]. The KS transformation lifts the three inertial position coordinates of the spacecraft into a four-dimensional representation in which the unperturbed Keplerian dynamics become linear and time invariant (LTI). We refer to the four-dimensional KS-lifted position coordinates as the "KS space."

In this chapter, we apply the KS transformation to relative orbital maneuvers between spacecraft in low-Earth orbit. We modify the KS-transformed dynamics to include perturbation forces due to non-spherical gravity and low-thrust control inputs. These modifications result in nonlinear dynamics; however, since only the perturbation terms are nonlinear, the KS dynamics are better approximated by linearization than other relative dynamics formulations. This "near linearity" allows for significantly longer step sizes during numerical integration. We solve the relative orbital maneuver problem by linearizing the perturbed KS dynamics with respect to a reference orbit in KS space.

Our contributions include:

- A novel optimization-based method for smoothly transforming Cartesian states into lifted KS states
- A KS formulation of relative orbital dynamics that includes J2 perturbations and low-thrust control inputs
- Accuracy comparisons between the KS-based relative dynamics and several existing state-of-the-art relative orbital dynamics models
- A convex-optimization formulation of the orbital rendezvous problem using our KS-based relative-orbit dynamics
- An example computation of an optimal rendezvous trajectory for a small spacecraft in low-Earth orbit with very low thrust capability

This chapter proceeds as follows: We describe related work on relative-orbit models and previous applications of the KS transform to orbital maneuvers in section 3.2. In section 3.3 we provide a description of Cartesian and KS transformed orbit dynamics. Section 3.4 describes our method for transforming smooth trajectories from Cartesian to KS coordinates. In section 3.5 we derive a relative orbital dynamics model using the KS transform, and in section 3.6 we compare this model with other relative-orbit models found in the literature by computing the trajectory prediction error versus a

numerically integrated ground truth. In section 3.7 we incorporate the KS relative-orbit model into a convex trajectory optimization formulation, and solve a low-thrust orbital rendezvous problem. We conclude in section 3.8 by summarizing our results and suggesting future research directions.

3.2 Related Work

There is an extensive literature on relative-orbit models. Sullivan, Grimberg, and D'Amico [65] provide a survey of these models and perform extensive comparisons between them. In section 3.6, we compare our KS relative-orbital model with the Clohessy-Wiltshire (CW) [12]; Yamanaka-Ankersen (YA) [78]; and Koenig, Guffanti, and D'Amico (KGD) [36] relative-orbit models. These models are developed by linearizing and integrating either the nonlinear Cartesian equations of motion or the Gauss Variational Equations for the orbital elements.

The CW relative-orbit model has been used extensively since the 1960s. It assumes a Keplerian circular reference orbit, is linear-time-invariant, and is parameterized by time. The CW model has been developed for both Cartesian and curvilinear relative coordinate frames [14]; in our comparisons we use the Cartesian coordinates.

The YA relative-orbit model extends the CW model to Keplerian eccentric orbits. It is parameterized by the true anomaly and uses a normalized Cartesian relative state representation. It is considered the state of the art Cartesian relative state representation for arbitrary Keplerian orbits [65]. In the circular case, the YA model reduces to the CW model.

The KGD model uses relative orbital elements and reflects the state of the art in state transition matrices for perturbed elliptical orbits. It provides a significant increase in accuracy over the CW and YA models and has similar or better accuracy to other models in the literature [25, 65].

The Kustaanheimo-Stiefel transformation was originally introduced as a method for regularizing the numerical integration of perturbed two-body motion [38]. It extends the planar Levi-Civita transformation [43] to three dimensions, and provides exact linear-time-invariant equations of motion for unperturbed Keplerian orbits in three dimensions. To our knowledge, the first work applying the KS transform to the relative motion between spacecraft is by Eldin, who studied the KS transform in the

context of unconstrained planar rendezvous maneuvers [18]. Thorne and Hall [67] use the planar KS transform to develop analytic expressions for minimum-time continuous-thrust orbit transfers. Hernandez and Akella [28] use the Levi-Civita transformation to find a Lyapunov control policy for finite-thrust orbital rendezvous from arbitrary orbital positions, illustrating the power of working in the Levi-Civita or (more generally) the KS coordinates. Perturbation forces were not considered in these previous works.

Recently, Tracy and Manchester [69] used the KS dynamics and nonlinear trajectory optimization to perform low-thrust transfers from a geostationary transfer orbit (GTO) to a geostationary orbit (GEO). The difference between the approach we present here and the approach in [69] is that we linearize the relative KS dynamics in the presence of perturbations, and perform convex optimization to compute rendezvous maneuvers between multiple spacecraft. Liu and Lu [44] approach the satellite-rendezvous problem using successive convexification methods to approximate the nonlinear relative dynamics and to satisfy safety constraints. In contrast, the linear KS dynamics allow us to solve a single convex optimization problem.

3.3 Background

3.3.1 Cartesian Orbit Dynamics

As discussed in chapter 2, in inertial Cartesian coordinates the unperturbed Keplerian dynamics of a satellite orbiting a massive body are

$$\ddot{x} = -\frac{\mu}{r^3}x, \tag{3.1}$$

where we've changed notation to let $x \in \mathbb{R}^3$ be the position vector relative to an inertial frame centered on the massive body. The scalars $r = \|x\|_2$, and μ are as defined before. In low-Earth orbit, the perturbation of these dynamics is dominated by atmospheric drag and the non-spherical shape of the Earth. We capture the dominant non-spherical gravitational effects by including the J_2 acceleration from

eq. (2.29). The J_2 perturbed Cartesian dynamics are then

$$\ddot{x} = -\frac{\mu}{r^3}x + a_{J_2}. \tag{3.2}$$

In this work, we focus on the effects of eccentricity and the J_2 perturbation, so we neglect atmospheric drag. However, the model we present can be readily extended to include drag forces.

3.3.2 The Kustaanheimo-Stiefel Transform

We now consider the transformation of eq. (3.2) into the four-dimensional KS space [64]. The transformation is not unique when transforming from Cartesian ($\mathbb{R}^3$) to KS space ($\mathbb{R}^4$), so we first define the transform from KS space to Cartesian.

Let $y \in \mathbb{R}^4$ be the KS variable corresponding to the Cartesian position $x \in \mathbb{R}^3$, and define the matrix

$$L(y) = \begin{bmatrix} y_1 & -y_2 & -y_3 & y_4 \\ y_2 & y_1 & -y_4 & -y_3 \\ y_3 & y_4 & y_1 & y_2 \\ y_4 & -y_3 & y_2 & -y_1 \end{bmatrix}. \tag{3.3}$$

The KS transformation from y to x is then

$$\begin{bmatrix} x \\ 0 \end{bmatrix} = L(y)y. \tag{3.4}$$

The fourth row of $L(y)y$ is always zero. The matrix $L(y)$ has some useful properties. In particular,

$$L^T(y)L(y) = L(y)L^T(y) = (y^T y)I \tag{3.5}$$

where I is the identity matrix. It follows that

$$\begin{aligned} r^2 = x^T x &= \left(L(y)y\right)^T L(y)y \\ &= y^T L^T(y)L(y)y = \left(y^T y\right)^2. \end{aligned} \tag{3.6}$$

The KS transformation also introduces a scaled fictitious time s that relates to real time by the inverse of the radius,

$$dt = rds. \tag{3.7}$$

We denote variables differentiated with respect to real time with a dot, $\dot{x} = dx/dt$, and variables differentiated with respect to the fictitious time with a prime, $y' = dy/ds$.

Kustaanheimo-Stiefel Dynamics from Euler-Lagrange

For a system with configuration q, kinetic energy $T(q, \dot{q})$, and potential energy $V(q)$, we define the Lagrangian $\mathcal{L}(q, \dot{q}) = T(q, \dot{q}) - V(q)$. The dynamics of the system must obey the Euler-Lagrange equation (ELE),

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}}\right) - \frac{\partial \mathcal{L}}{\partial q} = 0. \tag{3.8}$$

One way to derive the equations of motion for a system is to find its Lagrangian and then compute its ELE.

Using the KS state y, we can use the ELE to derive the KS orbital dynamics. To do so, we compute the ELE in real time and then transform to fictitious time when it is convenient. We begin by writing the kinetic and potential energy in the KS coordinates. So the Kinetic energy is

$$T = \frac{1}{2}m\dot{x}^T\dot{x} = \frac{1}{2}m\left(2L(y)\dot{y}\right)^T\left(2L(y)\dot{y}\right) = 2y^T y\dot{y}^T\dot{y}. \tag{3.9}$$

and the potential energy is

$$V = -\frac{\mu m}{\sqrt{x^T x}} = -\frac{\mu m}{y^T y}. \tag{3.10}$$

The Lagrangian is then

$$\mathcal{L} = 2m(y^T y)(\dot{y}^T\dot{y}) + \frac{\mu m}{y^T y}. \tag{3.11}$$

Differentiating with respect to y, we have

$$\frac{\partial \mathcal{L}}{\partial y} = 4m(\dot{y}^T\dot{y})y^T - \frac{2\mu m}{(y^Ty)^2}y^T, \tag{3.12}$$

and differentiating with respect to $\dot{y}$, we have

$$\frac{\partial \mathcal{L}}{\partial \dot{y}} = 4m(y^Ty)\dot{y}^T, \tag{3.13}$$

which, differentiated with respect to t is

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{y}}\right) = 4m(y^Ty)\ddot{y}^T + 8m(y^T\dot{y})\dot{y}^T. \tag{3.14}$$

Transposing and combining the above, the Euler-Lagrange equation is

$$4m(\dot{y}^T\dot{y})y - \frac{2\mu m}{(y^Ty)^2}y - 4m(y^Ty)\ddot{y} - 8m(y^T\dot{y})\dot{y} = 0. \tag{3.15}$$

Eliminating $2m$ and collecting terms, we have

$$\left[2(\dot{y}^T\dot{y}) - \frac{\mu}{(y^Ty)^2}\right]y - 2(y^Ty)\ddot{y} - 4(y^T\dot{y})\dot{y} = 0. \tag{3.16}$$

Rearranging the term multiplying y,

$$\left[2(\dot{y}^T\dot{y})\frac{y^Ty}{y^Ty} - \frac{\mu}{(y^Ty)^2}\right]y - 2(y^Ty)\ddot{y} - 4(y^T\dot{y})\dot{y} \tag{3.17a}$$

$$= -\left[-2(y^Ty)(\dot{y}^T\dot{y}) + \frac{\mu}{(y^Ty)}\right]\frac{y}{y^Ty} - 2(y^Ty)\ddot{y} - 4(y^T\dot{y})\dot{y} = 0. \tag{3.17b}$$

The term in brackets in eq. (3.17b) is the negative specific orbital energy

$$h = -\frac{T+V}{m} = -2(y^Ty)(\dot{y}^T\dot{y}) + \frac{\mu}{y^Ty}. \tag{3.18}$$

This is constant for an unperturbed orbit. Substituting,

$$-\frac{1}{y^Ty}hy - 2(y^Ty)\ddot{y} - 4(y^T\dot{y})\dot{y} = 0, \tag{3.19}$$

and, solving for $\ddot{y}$, we have

$$\ddot{y} = -\frac{1}{2(y^T y)^2} h y - 2\frac{y^T \dot{y}}{y^T y}\dot{y}. \tag{3.20}$$

We now solve for y'' in terms of $\ddot{y}$ using the transform from real time to fictitious time,

$$\dot{y} = \frac{d}{dt}y = \frac{1}{r}\frac{d}{ds}y = \frac{1}{y^T y}y', \tag{3.21a}$$

$$\ddot{y} = \frac{1}{r}\frac{d}{ds}\left(\frac{y'}{y^T y}\right) = \frac{y''}{(y^T y)^2} - 2\frac{y^T y'}{(y^T y)^3}y' \tag{3.21b}$$

$$\implies y'' = (y^T y)^2 \ddot{y} + 2\frac{y^T y'}{y^T y}y'. \tag{3.21c}$$

Substituting in $\ddot{y}$ from eq. (3.20),

$$y'' = (y^T y)^2\left(-\frac{1}{2(y^T y)^2} h y - 2\frac{y^T \dot{y}}{y^T y}\dot{y}\right) + 2\frac{y^T y'}{y^T y}y' \tag{3.22a}$$

$$= (y^T y)^2\left(-\frac{1}{2(y^T y)^2} h y - 2\frac{y^T y'}{(y^T y)^3}y'\right) + 2\frac{y^T y'}{y^T y}y' \tag{3.22b}$$

$$= -\frac{h}{2}y - 2\frac{y^T y'}{y^T y}y' + 2\frac{y^T y'}{y^T y}y' \tag{3.22c}$$

$$= -\frac{h}{2}y. \tag{3.22d}$$

Under the KS transformation, the Keplerian two-body dynamics in eq. (3.1) become

$$y'' = -\frac{h}{2}y, \tag{3.23}$$

where

$$h = \frac{\mu}{r} - \frac{\dot{x}^T \dot{x}}{2} = \frac{\mu - 2y'^T y'}{y^T y} \tag{3.24}$$

is the negative specific energy of the orbit. Because h is constant for unperturbed orbits, eq. (3.23) is a four-dimensional simple-harmonic oscillator.

Disturbances and Control Inputs

In the KS space, the dynamics of any Keplerian orbit are linear and time invariant. Arbitrary Cartesian disturbance accelerations $d(x, \dot{x}) \in \mathbb{R}^3$ and control accelerations $u \in \mathbb{R}^3$, can be transformed to the KS dynamics. Under perturbation, eq. (3.23) becomes

$$y'' = -\frac{h}{2}y + \frac{y^T y}{2} L^T(y) \begin{bmatrix} u + d \\ 0 \end{bmatrix}. \tag{3.25}$$

With acceleration inputs, the energy h is no longer constant:

$$h' = -2y'^T L^T(y) \begin{bmatrix} u + d \\ 0 \end{bmatrix}. \tag{3.26}$$

To account for the energy dynamics, we define an augmented state:

$$z = \begin{bmatrix} y \\ y' \\ h \end{bmatrix} \in \mathbb{R}^9, \tag{3.27}$$

and write the perturbed dynamics

$$z' = f(z, u). \tag{3.28}$$

The disturbances $d(x, \dot{x})$ can be written as a function of y and y' as follows:

$$d(x, \dot{x}) = d(L(y)y, \frac{2}{y^T y} L(y)y') \triangleq d(y, y'). \tag{3.29}$$

For the remainder of this chapter, we let d be the J_2 acceleration in eq. (2.29).

3.4 Transforming from Cartesian to KS space

For each Cartesian position, $x \in \mathbb{R}^3$, there is a one-dimensional submanifold of $\mathbb{R}^4$ such that any point y on that manifold satisfies the KS transform in eq. (3.4). For this reason, the inverse of eq. (3.4), transforming from Cartesian to KS space, is not

unique. A single solution y can be found by algebraically inverting eq. (3.4) and arbitrarily choosing the value of one of the elements of y [64, 69]. Unfortunately, while the transformation between the Cartesian and KS spaces is smooth, this method exhibits singularities in the KS space and the transformation of a smooth trajectory into the KS space will not necessarily be smooth. Additionally, when computing the relative position between two KS states, this non-uniqueness leads to two degrees of freedom in the relative position and there may be a relative KS position with smaller norm.

To ensure that transformed trajectories are smooth, and to compute a relative position of minimum norm, we formulate the inverse KS transform as an optimization problem,

$$\begin{aligned} \underset{y}{\text{minimize}} \quad & (y-\bar{y})^T(y-\bar{y}) \\ \text{subject to} \quad & \begin{bmatrix} x \\ 0 \end{bmatrix} = L(y)y, \end{aligned} \tag{3.30}$$

where x is the Cartesian position we wish to convert and $\bar{y}$ is the KS position we desire y to be close to. The solution, y^*, of this optimization problem is the KS position closest to $\bar{y}$ that satisfies the KS transform. To convert points along a trajectory, we let $\bar{y}$ be the transform of the previous point. If there is no logical $\bar{y}$, we let $\bar{y} = [1, 0, 0, 0]^T$. We solve (3.30) efficiently using Newton's method.

Figure 3.1 shows the difference between our proposed nearest state method and the common method. The lines shown are the trajectory of an orbit with unit amplitude and period. The trajectory was originally computed in Cartesian space and has been transformed to the KS space using the common method of fixing an arbitrary element of the state vector and our nearest-state method. The plot shows the transformed KS coordinates of the trajectory. In this case, the common method arbitrarily assigns $y_4 = 0$ to invert eq. (3.4). This results in the discontinuity at $t = 0.5$. Our nearest-state method produces a smooth trajectory because it minimizes the difference between each point and the previous one.

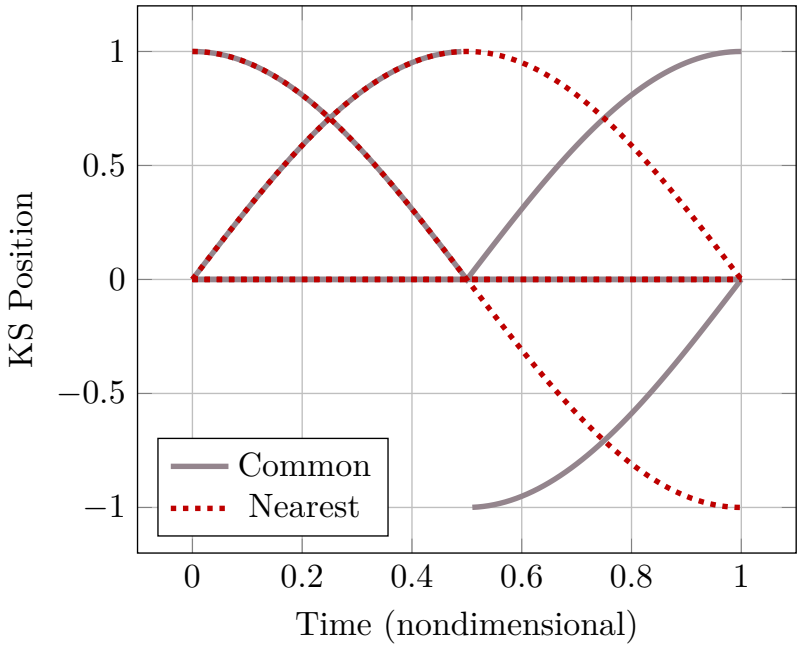


Figure 3.1: Cartesian to KS transformation of a unit orbit, comparing the non-smooth common method and our nearest state method.

3.5 Relative KS Dynamics

To define the relative KS dynamics, we let z and u be the state and control of the deputy satellite, and we define the chief state $\bar{z}$ and control $\bar{u}$. The relative state is $\delta z = z - \bar{z}$, the relative control is $\delta u = u - \bar{u}$, and the relative dynamics are:

$$\begin{aligned} \delta z' &= z' - \bar{z}' = f(z,u) - f(\bar{z},\bar{u}) \\ &= f(\bar{z} + \delta z, \bar{u} + \delta u) - f(\bar{z},\bar{u}) \\ &= \frac{\partial f}{\partial z}(\bar{z},\bar{u})\delta z + \frac{\partial f}{\partial u}(\bar{z},\bar{u})\delta u + \mathcal{O}(\|\delta z\|^2). \end{aligned} \tag{3.31}$$

Since the Keplerian dynamics for y'' are already linear, the higher-order terms in eq. (3.31) are due to the control inputs, perturbations, and the difference in energy between the deputy and chief orbits. The effect of these are orders of magnitude smaller than the Keplerian dynamics, so it is a very good approximation to drop the

higher-order terms. This yields the linear-time-varying relative dynamics

$$\delta z' = \frac{\partial f}{\partial z}(\bar{z}, \bar{u})\delta z + \frac{\partial f}{\partial u}(\bar{z}, \bar{u})\delta u. \tag{3.32}$$

To find the discrete-time linear-time-varying relative dynamics, we numerically integrate the controlled state-transition matrix dynamics,

$$\Phi' = \begin{bmatrix} \frac{\partial f}{\partial z}(\bar{z}, \bar{u}) & \frac{\partial f}{\partial u}(\bar{z}, \bar{u}) \\ 0_{3\times 9} & 0_{3\times 3} \end{bmatrix} \Phi, \tag{3.33}$$

along a given trajectory $\bar{z}$, $\bar{u}$. We then define the discrete-time linear-time-varying state-space system

$$\delta z_{k+1} = A_k \delta z_k + B_k \delta u_k \tag{3.34}$$

where $A_k \in \mathbb{R}^{9\times 9}$ contains the first nine rows and columns of $\Phi(s_{k+1}, s_k)$ and $B_k \in \mathbb{R}^{9\times 3}$ contains the first 9 rows and last 3 columns of $\Phi(s_{k+1}, s_k)$. Since Φ is computed with the Jacobian of the J2-perturbed KS dynamics, the LTV dynamics include both periodic and secular effects of the J2 perturbation.

3.6 Comparison of relative-orbit models

We now compare our linearized KS relative-orbit state transition matrix (eq. (3.33)) with the CW, YA, and KGD linear relative-orbit state transition matrices. Table 3.1 summarizes the differences between these models. The linearization procedure of section 3.5 is not unique to the KS dynamics, so we also use it to linearize the nonlinear Cartesian dynamics in eq. (3.2). We refer to the resulting linearized Cartesian model as the "LIN" model. The difference between the LIN model and the KS model shows the performance gained by using KS dynamics, eliminating any difference caused by computation methods used for the CW and YA models. In [36], the KGD model is developed for three different relative orbital element (ROE) states, the singular ROEs, quasi-nonsingualar ROEs, and the nonsingular ROEs; the nonsingular ROEs are the most general of the ROE states, so we use them for our comparisons.

Figure 3.2 shows the root-mean-square (RMS) position error measured along

trajectories propagated for one orbital period. The ground-truth orbits are a numerical integration of the J2-perturbed nonlinear equations of motion in eq. (3.2) using a high-accuracy adaptive Runge-Kutta method[58, 70]. The deputy initial conditions are perturbed with a range of offsets in mean anomaly, inclination, eccentricity, and semi-major axis while the other orbital parameters are held constant. We use the same reference orbit as in scenario 1 of Sullivan, et al. [65] for comparison with the relative-orbit models they present. Table 3.2 shows the reference orbit initial conditions used for the mean anomaly, inclination, and semi-major axis variation experiments.

As in [65], to compare performance on eccentric reference orbits, both the reference and deputy orbits are initialized with the same variation of eccentricity. All other initial orbital elements for the reference orbit match Table 3.2. The deputy orbit is offset from the reference orbit by 0.001 degrees in both mean anomaly and inclination. This corresponds to a distance of approximately 125 meters.

Table 3.1: Comparison of Relative-Orbit Models

	Linear	Independent Variable	Reference Orbit	J2	State Representation
Clohessy-Wiltshire (CW)	✓	Real time	Circular	✗	Cartesian
Yamanaka-Ankersen (YA)	✓	True anomaly	Elliptical	✗	Normalized Cartesian
Numerically Linearized Cartesian (LIN)	✓	Real time	Elliptical	✓	Cartesian
Koenig, Guffanti, and D'Amico (KGD)	✓	Real time	Elliptical	✓	Orbital Elements
Kustaanheimo-Stiefel (KS)	✓	Fictitious time	Elliptical	✓	Augmented Quaternion

Table 3.2: Reference Orbit Initial Conditions for RMS Trajectory Error Plots in Figure 3.2

a	e	i	Ω	ω	M
750 + 6378 km	0.0	98.2°	0°	0°	0°

Figure 3.2 shows that our KS model exhibits significantly less propagation error than any of the other relative-orbit models. Since the reference orbit is circular for the mean anomaly, inclination, and semi-major axis plots, the CW, YA, and LIN models perform identically. On the eccentricity plot, the CW model exhibits higher error than the YA model, which again matches the LIN model. On the mean anomaly and inclination models, the KGD model exhibits higher error than the KS model within the small angles shown on the plot. The astute reader will notice that extrapolating the KS and KGD mean anomaly and inclination plots, the KS error does grow faster

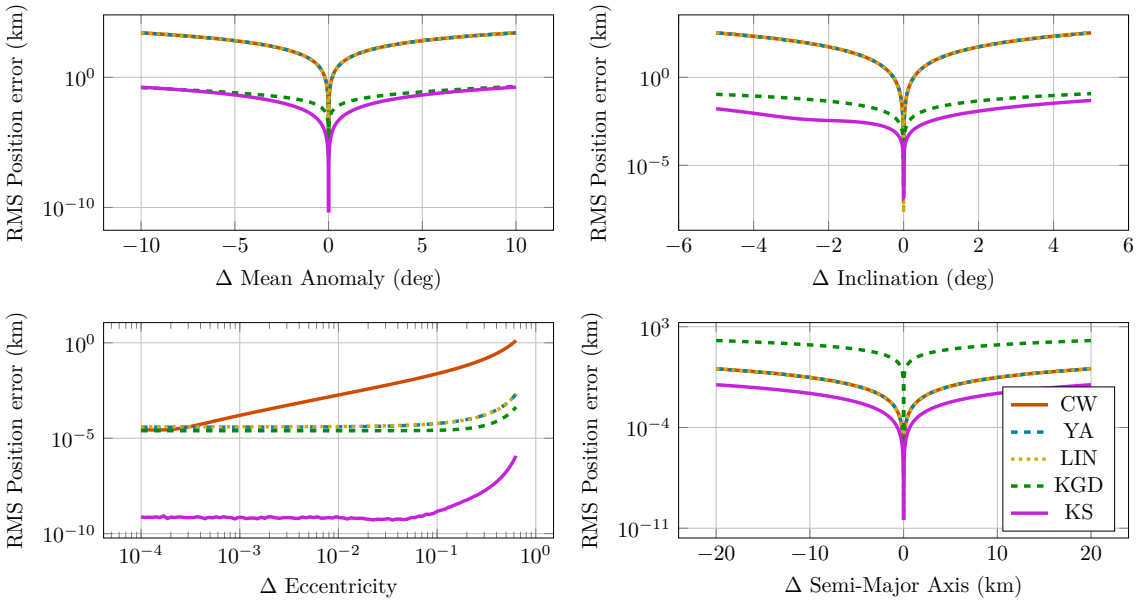


Figure 3.2: RMS position error along a single orbit with a range of initial variations in mean anomaly, inclination, eccentricity, and semi-major axis while the other orbital parameters are held constant. Lines correspond to the Clohessy-Wiltshire (CW), Yamanaka-Ankersen (YA), J2-perturbed linear-time-varying Kustaanheimo-Stiefel (KS) and the Koenig, Guffanti, and D'Amico (KGD) linear relative-orbit models. The ground truth is a numerical integration of eq. (3.2).

than the KGD error. While not shown, the difference between the KS and KGD error at large mean anomaly and inclination separation angles remains within an order of magnitude of each other. This should not detract from the excellent small-angle performance of the KS model, since relative-orbit models are most commonly used at deviations of less than 10 degrees in mean anomaly or inclination. For the semi-major axis variations, the KGD model exhibits higher error than any of the other models. As an additional note, the extensive use of orbital elements in the YA and KGD models leads to numerous degeneracies and singularities, significantly complicating their practical use. In contrast, the Cartesian and KS dynamics are globally smooth and well behaved.

3.7 Relative Orbital Maneuvers via Convex Optimization

The LTV relative dynamics given by eq. (3.33) allow us to construct a convex trajectory optimization formulation of the orbital rendezvous problem. With the discretized dynamics in eq. (3.34), trajectories of length N can be computed by solving a convex optimization problem over $\delta z_{1:N}$, and $\delta u_{1:N-1}$:

$$\begin{aligned} \underset{\delta z_{1:N},\, \delta u_{1:N-1}}{\text{minimize}} \quad & J(\delta z_{1:N}, \delta u_{1:N-1}) \\ \text{subject to} \quad & \delta z_{k+1} = A_k \delta z_k + B_k \delta u_k, \\ & \delta u_k \in \mathcal{U}_k, \\ & \delta z_k \in \mathcal{Z}_k, \end{aligned} \tag{3.35}$$

where $J(\delta z_{1:N}, \delta u_{1:N-1})$ is a convex cost function, and $\mathcal{Z}_k$ and $\mathcal{U}_k$ are convex sets.

The time steps in eq. (3.35) are scaled fictitious KS times. Once the optimal trajectory is found, the real times at which $\delta u^*_{1:N-1}$ should be applied are found by integrating eq. (3.7) along the chaser states,

$$t_k = \sum_{i=1}^{k} \|\bar{z}_k + \delta z^*_k\|^2 ds. \tag{3.36}$$

3.7.1 Low-Thrust Rendezvous Maneuver

We demonstrate the convex trajectory optimization with KS dynamics by solving a low-thrust rendezvous maneuver. The orbits and relative states for this scenario are similar to the International Space Station final approach performed by Soyuz and SpaceX Dragon spacecraft. The target and chaser initial orbital elements, as well as the initial RTN state of the chaser with respect to the target, are given in Table 3.3. To formulate this problem in the context of eq. (3.35), we assume the target spacecraft is not producing thrust, but is experiencing perturbations, and integrate the target states with eq. (3.28) to compute A_k and B_k. We additionally right-multiply B_k by a rotation matrix which maps vectors in the target spacecraft

RTN frame to Earth-centered inertial vectors. This allows us to compute the controls in the target RTN frame, which is a typical choice for formation flying problems [2]. The quadratic cost is,

$$\begin{aligned} J(\delta z_{1:N}, \delta u_{1:N-1}) = \delta z_N^T Q_N \delta z_N \\ + \sum_{k=1}^{N-1} \delta z_k^T Q_k \delta z_k + \delta u_k^T R_k \delta u_k, \end{aligned} \tag{3.37}$$

where $Q_* \succ 0$, $R_* \succ 0$. To demonstrate the long optimization horizon possible with KS dynamics, we use a maximum thrust acceleration constraint of $20\mu\text{m/s}^2$. This maximum thrust falls in the range of low-thrust, high specific impulse propulsion systems currently available for small satellites [40]. The optimization uses 20 timesteps per orbit and a 100 orbit horizon, resulting in 2000 knot points. A solution is computed once per orbit, and the controls from that solution are applied to the J2-perturbed nonlinear dynamics over the following orbit in a receeding-horizon fashion. We solve these trajectory optimization problems using the convex quadratic program solver OSQP [62]. It takes approximately 6 seconds to integrate the discrete dynamics, set up, and solve this trajectory optimization on a MacBook Pro with an Apple M1 Pro processor.

The results of this maneuver are shown in fig. 3.3. The top-left plot shows that the position and velocity errors do not converge monotonically, but do converge to zero over time. The top-right plot shows the thrust control inputs over time. The thrust constraints are active for much of the first 50 orbits. The bottom two plots show the chaser trajectories on the radial-tangential plane and radial-normal plane.

Table 3.3: Initial Conditions for Low-Thrust Rendezvous in Figure 3.3

Orbital Elements	a	e	i	Ω	ω	M
Target	417 + 6378 km	0.0003	51.64°	0°	300°	0°
Chaser	415 + 6378 km	0.0004	51.65°	0°	300°	−0.1°
RTN State	x_R	x_T	x_N	$\dot{x}_R$	$\dot{x}_T$	$\dot{x}_N$
	-2.7 km	-11.9 km	-1.0 km	0.0033 m/s	4.9 m/s	0.67 m/s

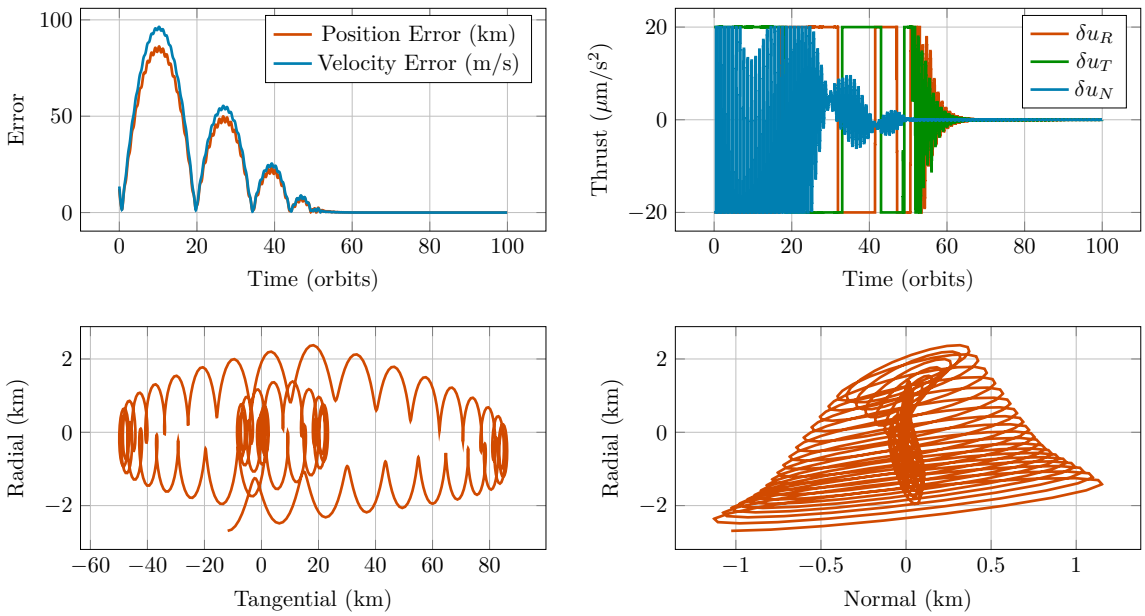


Figure 3.3: Position and velocity error, thrust control inputs, and radial-tangential-normal (RTN) trajectories for a low-thrust rendezvous maneuver.

3.8 Conclusions

We have shown that a relative-orbit-dynamics model based on the Kustaanheimo-Stiefel transformation that includes linearized J2 perturbations and control inputs is highly accurate and achieves higher accuracy than other state-of-the-art models in the literature. The KS relative dynamics model provides a linear-time-varying dynamics formulation that can be incorporated into standard estimation and control tools. Because the KS relative dynamics are very accurate, long-horizon prediction and trajectory-planning problems can be solved.

Our rendezvous demonstration provides one application of the KS relative dynamics. Many other scenarios are possible, including complex maneuvers with differential drag and solar sails. Additionally, safety constraints are an essential consideration in rendezvous or proximity operations problems that we will investigate in future work.

The code used to produce the results in this chapter is available at https://github.com/RoboticExplorationLab/KSRelativeOrbits.

Chapter 4

Drag-Based Formation Control

4.1 Introduction

Formations of multiple satellites are frequently used to perform tasks that a single satellite cannot accomplish alone. Examples include satellite navigation systems, like the global positioning system (GPS), and communications constellations like Iridium and Starlink. The ability to maneuver and control the relative positions of such satellites is key to establishing and maintaining a formation. However, satellites often rely on propulsion systems to maintain these formations, which may be prohibitively large or expensive, especially on smaller spacecraft. Instead, satellites can utilize external perturbation forces to adjust their orbits. In low-Earth orbit (LEO), there are primarily two such forces.

The first perturbation force is atmospheric drag [50], which influences a satellite's altitude and, consequently, its orbital velocity and position. As depicted in fig. 4.1, the drag area of a spacecraft can be changed by controlling the attitude of the spacecraft. By placing some spacecraft in a high-drag state and others in a low-drag state, a differential drag between satellites can be introduced and the relative along-track positions of satellites can be changed. This method has been used on orbit to establish and control the along-track positions for constellations of up to 100 satellites [21].

A second perturbation force on LEO satellites is nodal precession [50]. Nodal precession is due to Earth's non-spherical gravity field, and causes orbits to precess, or rotate, around the Earth's polar axis. This effect introduces a small cross-track

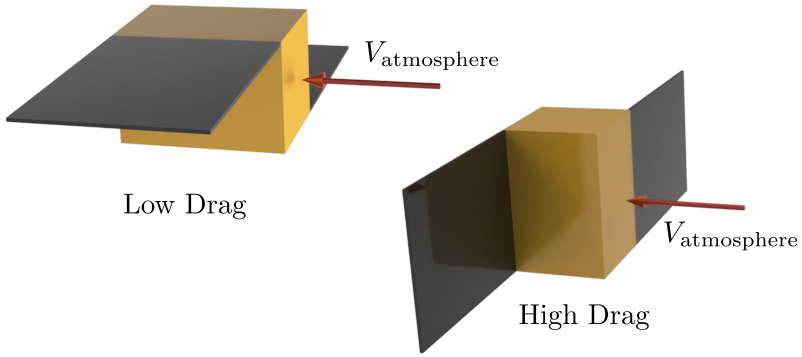


Figure 4.1: High- and low-drag configurations for a satellite with attitude-controlled drag modulation.

acceleration on a satellite that varies with altitude. By establishing a large differential altitude between spacecraft, the nodal precession of those spacecraft will occur at different rates, and cross-track orbital changes can be made.

Most differential-drag formation-flying methods ignore the cross-track influence of nodal precession because it is small compared to the along-track drift, requiring large altitude differences and long time horizons to have a significant effect. This chapter introduces a method to leverage nodal precession for long-term differential-drag maneuvers, simultaneously controlling both along-track and cross-track formation shifts. Our contributions include:

- A novel first-order analytical relationship between along-track and cross-track separation changes. This defines a fundamental limit on what along-track and cross-track separations are simultaneously achievable.
- A convex trajectory optimization formulation to compute differential-drag sequences that achieve desired formation configurations.
- A receding-horizon control strategy that re-plans maneuvers every few orbits to compensate for disturbances and modeling errors.
- Simulation results demonstrating our receding-horizon controller performing several different maneuvers in a high-fidelity orbital-dynamics simulation.

Throughout the chapter we assume the spacecraft is capable of controlling its attitude; this can be performed using a variety of methods including reaction wheels [50] and magnetorquers [24].

The chapter proceeds as follows: In section 4.2 we review previous research and on-orbit demonstrations of drag-based formation flying. Section 4.3 introduces background concepts that are used in section 4.4 to develop the along-track and cross-track formation flying linear trajectory optimization. The results of a single convex trajectory optimization and closed loop simulations with the trajectory optimization as a feedback controller are shown and discussed in section 4.5. We conclude in section 4.6.

4.2 Related Work

Many studies have explored drag-modulation techniques to enable formation control without propulsion systems. Leonard, et al. first proposed this technique for maintaining the relative separation of spacecraft already in formation [41]. Mathews, et al. [51] investigated a drag-propulsion combination to maintain a cyclical altitude and phase relationship between a spacecraft and a space station. Additional methods have been proposed since then for along-track formation keeping using drag [31, 37, 61, 73].

Differential drag control for along-track rendezvous has also been studied. Bevilacqua, et al. include J_2 perturbations in their model, which they solve with a two-step analytic method. They do not include cross-track separation in their relative state [6]. Harris and Açıkmeşe [27] propose an optimization approach to differential-drag rendezvous — they use a constrained linear program with minimum-time cost. Most differential-drag methods assume binary drag states where a satellite is in either a low- or high-drag configuration; Harris et al. investigate a continuous drag-modulation scheme based on the coupling of spacecraft attitude and drag [26]. We use a similar continuous-drag formulation, but with a one-norm cost to encourage binary or "bang-bang" drag states.

There have been multiple successful demonstrations of differential-drag control on orbit. The ORBCOMM communications constellation, launched in 1997-1999, used differential drag modulation, along with occasional propulsive maneuvers, to maintain the along-track separation for their network of thirty spacecraft [46]. A limited demonstration of differential drag modulation using deployable panels was performed on-orbit by the AeroCube-4 CubeSat mission in 2012 [23]. Perhaps the most complete on-orbit demonstration of differential drag was for the Planet Earth-

imaging constellation [21]. After deployment and initial contact, the slot-allocation and phasing problem was solved by a ground-control system using a genetic algorithm. The CYGNSS constellation also included differential-drag modulation in its mission design [10], and Millenium Space Systems recently demonstrated drag-based station keeping between two satellites on orbit [54].

The Planet differential drag system [21] spawned several derivative works. A continuous optimization of the Planet slot allocation and phasing problem was formulated by Blatner [8]. Repeated updates to handle perturbations, and continuous controls were presented by Sin et al. [60].

All the previously discussed works do not consider cross-track motion of the satellites, and all solutions were computed on the ground. In contrast, our work considers both along-track and cross-track motion, and our control formulation is amenable to autonomous on-orbit implementation.

Two works [39, 42] combine differential drag and nodal precession to modify the cross-track separation of satellites. These works are the most similar to ours. Leppinen [42] performs a feasibility study to demonstrate that differential drag can produce a sufficient altitude separation for nodal precession to change the RAAN of a satellite. No control methods are presented. Lee and Bang [39] present a method for modifying the ground-tracks of satellites in a constellation using differential drag and nodal precession. Synchronization of the along-track and cross-track state of the satellites is not investigated in either of these prior works; this is a key contribution of our work.

We formulate the differential-drag control problem as a convex trajectory optimization problem with a linear cost and linear constraints. Tillerson, et al. solved spacecraft formation flying problems with convex trajectory optimization over twenty years ago [68]. Since that time, convex trajectory optimization has gained popularity for solving many aerospace problems including orbital maneuvering, rocket soft landing, and planetary aerocapture [45, 48].

4.3 Background

4.3.1 Keplerian Motion

A satellite orbiting a perfectly spherical planet with no additional perturbations has dynamics given by the two-body equation:

$$\ddot{\mathbf{r}} = -\frac{\mu}{r^3}\mathbf{r} \tag{4.1}$$

where $\mathbf{r}$ is the position vector of the spacecraft in the planet-centered inertial frame, $r = \|\mathbf{r}\|$, $\ddot{\mathbf{r}}$ is the acceleration vector, and μ is the planet's standard gravitational parameter.

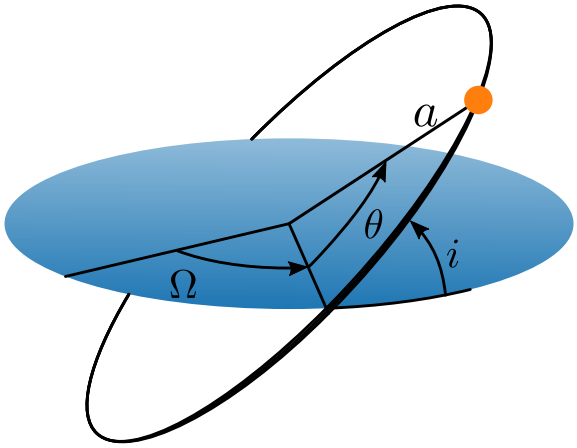


Figure 4.2: Notation used to describe the orbital state of a satellite in a circular orbit with the Earth's equatorial plane shown in blue and Ω referenced to an inertially fixed direction.

The orbital state of a satellite is commonly described using the six orbital elements [13]: a, the semi-major axis; e, the eccentricity of the orbit ellipse; i, the inclination; Ω, the right ascension of the ascending node (RAAN); ω, the argument of periapsis; and ν, the true anomaly. In this work, we consider circular orbits, so $e = 0$, and ω and ν are undefined. Instead of ω and ν, we use θ, the argument of latitude (AoL), which measures along-track orbital position from the equatorial plane. In the remainder of this work our focus will be on the dynamics of a, Ω, and θ shown in Figure 4.2.

A real spacecraft experiences a large number of secondary perturbation forces. The resulting dynamics are

$$\ddot{\mathbf{r}} = -\frac{\mu}{r^3}\mathbf{r} + \mathbf{p}, \tag{4.2}$$

where $\mathbf{p}$ is the perturbative acceleration vector. The largest perturbation forces on a satellite in LEO are due to atmospheric drag and the Earth's non-spherical gravitational field.

4.3.2 Atmospheric Drag

In LEO, atmospheric drag is modeled by

$$\mathbf{D} = -\frac{1}{2m}\rho A C_D v(\mathbf{v} - \mathbf{v_{atm}}) \tag{4.3}$$

where $\mathbf{D}$ is the drag acceleration, ρ is the atmospheric density, A is the satellite's incident cross-sectional area, C_D is the drag coefficient, m is the satellite mass, $\mathbf{v}$ is the inertial velocity vector of the satellite, $\mathbf{v_{atm}}$ is the velocity of the atmosphere, and $v = \|\mathbf{v} - \mathbf{v_{atm}}\|$ is the relative velocity vector magnitude [11, 57]. According to eq. (4.3), adjusting A through either deployable panels or by changing the spacecraft attitude can modulate drag [19, 21].

Drag always acts in the direction opposing velocity, and can only directly affect the motion of a spacecraft within the orbital plane, decreasing its eccentricity and semi-major axis [74]; since we are assuming circular orbits, we do not consider the eccentricity dynamics due to drag here. This is not a very limiting assumption since drag tends to naturally circularize orbits [9]. The semi-major axis dynamics due to drag are

$$\dot{a} = 2\sqrt{\frac{a^3}{\mu}}D \tag{4.4}$$

where $D = \|\mathbf{D}\|$ is the magnitude of the drag vector.

4.3.3 Nodal Precession and The Method of Averaging

Models of the Earth's gravitational field are typically expressed by a spherical harmonic expansion with coefficients J_n [11, 13]. The first non-spherical term, J_2, is several

orders of magnitude larger than all subsequent terms and captures the dominant effect of the Earth's oblateness. Since the J_2 acceleration is rotationally symmetric, it only depends on an orbit's inclination.

On short timescales, the J_2 perturbation impacts all of the orbital elements. However, many of these effects are periodic and average out over an orbit, and only variations on Ω persist over longer time scales. These long-term orbit-averaged dynamics with J_2 can be described by,

$$\dot{\bar{\Omega}} = -\left[\frac{3}{2}\frac{J_2\sqrt{\mu}R_E^2}{a^{7/2}}\right]\cos i \tag{4.5}$$

$$\dot{\bar{\theta}} = \sqrt{\mu/a^3} \tag{4.6}$$

where R_E is the Earth's equatorial radius.

4.4 Formation Flying

The nodal precession rate and the AoL rate, derived from eqs. (4.5) and (4.6), demonstrate a dependence on the semi-major axis. Modulating the semi-major axis using drag variation, as shown in eq. (4.3), facilitates the manipulation of a satellite's AoL and RAAN, enabling the establishment of satellite formations with both along-track and cross-track separations. This section details the linearized dynamics governing the separations between satellites in a formation, and the trajectory optimization approach employed for drag-based formation flying.

4.4.1 Linearized Dynamics

We linearize eqs. (4.5) and (4.6) around a reference semi-major axis a. Similarly, eq. (4.4) is linearized around a reference drag D, where the satellite's altitude, and thus its atmospheric density and velocity, have been fixed (see Eq. eq. (4.3)). The

resulting linearized equations are:

$$\Delta\dot{\bar{\theta}} = -\frac{3}{2}\sqrt{\frac{\mu}{a^5}}\Delta a \triangleq k_1 \Delta a, \tag{4.7a}$$

$$\Delta\dot{a} = 2\sqrt{\frac{a^3}{\mu}} D \Delta D \triangleq k_3 \Delta D, \tag{4.7b}$$

$$\Delta\dot{\bar{\Omega}} = \frac{21}{4} J_2 \sqrt{\frac{\mu}{a^9}} R_E^2 \cos i \Delta a \triangleq k_2 \Delta a, \tag{4.7c}$$

where $\Delta\bar{\theta}$, $\Delta\bar{\Omega}$, Δa, and ΔD represent the differences in AoL, RAAN, semi-major axis, and drag force between two satellites, respectively. Notably, from eq. (4.7a) and eq. (4.7c), it's evident that the rates of $\Delta\bar{\theta}$ and $\Delta\bar{\Omega}$ are both influenced by Δa, which implies they cannot be changed independently. Assuming $\Delta\bar{\theta} = \Delta\bar{\Omega} = 0$ initially, all achievable $\Delta\bar{\theta}$ must satisfy

$$\Delta\bar{\Omega} = \frac{k_2}{k_1}\Delta\bar{\theta} \triangleq k_4 \Delta\bar{\theta} \tag{4.8}$$

where $k_4 = k_2/k_1$ is a dimensionless constant that depends only on the reference orbit.

The linear equations eqs. (4.7a) to (4.7c) can be put in the standard form of a linear dynamical system,

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}, \tag{4.9}$$

where

$$\mathbf{x} = \begin{bmatrix} \Delta\bar{\theta} \\ \Delta a \end{bmatrix}, A = \begin{bmatrix} 0 & k_1 \\ 0 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 \\ k_3 \end{bmatrix}, \tag{4.10}$$

and $\mathbf{u} = \Delta D$. We omit $\Delta\bar{\Omega}$ from the state since eq. (4.8) establishes a relationship between $\Delta\bar{\Omega}$ and $\Delta\bar{\theta}$. The control action, ΔD, is a result of altering the satellite's cross-sectional area exposed to the oncoming atmosphere. In this work, it represents the difference in two spacecraft's attitude between their high- and low-drag configurations, as shown in fig. 4.1. When ΔD assumes maximum or minimum values, the two spacecraft have opposite attitude configurations. Conversely, when ΔD is null, it indicates that both satellites maintain identical configurations.

To extend this method to the case of $n > 2$ satellites, one satellite is arbitrarily chosen as the "chief" satellite, and all other satellite's Δ states are referenced to this

chief. We concatenate $n-1$ copies of eq. (4.10) to rewrite (4.9) as a $2(n-1)$ state system. When referring to the relative state between the chief and another satellite, we use the notation Δa^{1-p} and $\Delta\bar{\theta}^{1-p}$, where p is the index of the satellite.

4.4.2 Constraints on the Final Conditions of Drag-Based Formation Control

Given a pair of satellites deployed at the same initial orbit (i.e. $\mathbf{x}_0 = 0$), our goal is to manipulate the differential drag ΔD over time to achieve a final formation configuration $\mathbf{x}_f$ at some future time t_f. The control strategy involves lowering the orbital altitude of one satellite such that its nodal precession rate is larger than the other satellite. The satellites then remain in this configuration, with $\Delta D = 0$ until a desired $\Delta\bar{\theta}$, and therefore a desired $\Delta\bar{\Omega}$, is achieved. The higher satellite then lowers its altitude to match the first satellite. To maintain a fixed final formation configuration, we must have $\dot{\mathbf{x}}_f = 0$. To satisfy this, eqs. (4.7a) to (4.7c) show that Δa_f and ΔD_f must be zero — the satellites must be at the same final altitude and in the same drag configuration.

Modifying (4.8) to account for the fact that $\Delta\bar{\theta}$ is an angular quantity, the possible $\Delta\bar{\Omega}$ for a desired final $\Delta\bar{\theta}_f$ are given by

$$\Delta\bar{\Omega}_f = k_4(\Delta\bar{\theta}_f + 2\pi\ell) \tag{4.11}$$

where ℓ is any integer. To first order, eq. (4.11) defines the AoL and RAAN separations achievable using drag modulation. For differential-drag formation control to be feasible, eq. (4.11) is a fundamental limit that must be obeyed when selecting the final $\Delta\bar{\theta}$ and $\Delta\bar{\Omega}$ of a formation.

4.4.3 Optimization-Based Drag Maneuver Planning

Given n satellites deployed in the same orbit (i.e., $\mathbf{x}_0 = 0$), we seek to maneuver these satellites into a formation configuration at a final time t_f. To do so with differential drag, we must choose the final state $\mathbf{x}_f$ by choosing the desired value for either $\Delta\Omega_f$ or $\Delta\theta_f$ and selecting the other in accordance to eq. (4.11). The final altitude or final time are then a result of this choice. It remains to find the necessary control inputs

to achieve this formation.

A full trajectory of drag modulation inputs that drives the satellite formation from $\mathbf{x}_0$ to $\mathbf{x}_f$ can be planned by solving the convex optimization problem

$$\begin{aligned} \underset{\mathbf{x}_{1:N},\, \mathbf{u}_{1:N-1}}{\text{minimize}} \quad & g_f(\mathbf{x}_N) + \sum_{i=1}^{N-1} g(\mathbf{x}_i, \mathbf{u}_i) \\ \text{subject to} \quad & \mathbf{x}_{i+1} = A\mathbf{x}_i + B\mathbf{u}_i, \\ & \left[\Delta a_N^{1-2}, ..., \Delta a_N^{1-n}\right] = 0, \\ & \Delta a_{\min} \leq \left[\Delta a_i^{1-2}, ..., \Delta a_i^{1-n}\right] \leq \Delta a_{\max}, \\ & u_{\min} \leq \mathbf{u}_i \leq u_{\max} \end{aligned} \tag{4.12}$$

where $g(x, u)$ is a convex stage cost function, and $g_f(x)$ is a convex terminal cost function. The first constraint enforces the discrete form of the linear dynamics from eq. (4.9), the second constraint ensures the satellites end at the same final altitude, the third constraint restricts the minimum and maximum altitude differences for each pair of satellites to be within $\Delta a_{\min}$ and $\Delta a_{\max}$, and the final constraint enforces $u_{\min}$ and $u_{\max}$ as lower and upper bounds on the drag achievable by each satellite. In this work, meeting the $\Delta\theta$ final conditions is not treated as a constraint but included in the cost function; this relaxes the problem and avoids infeasibility.

The cost functions g and g_f can be chosen to shape the overall system behavior. To produce minimum-time bang-bang control commands, one-norm costs are used [79]:

$$\begin{aligned} g_f(\mathbf{x}_N) &= \left\|\Delta\bar{\theta}_N^{1-2} - \Delta\bar{\theta}_f^{1-2}\right\|_1 + ... + \left\|\Delta\bar{\theta}_N^{1-n} - \Delta\bar{\theta}_f^{1-n}\right\|_1 \\ g(\mathbf{x}, \mathbf{u}) &= \left\|\Delta\bar{\theta}^{1-2} - \Delta\bar{\theta}_f^{1-2}\right\|_1 + ... + \left\|\Delta\bar{\theta}^{1-n} - \Delta\bar{\theta}_f^{1-n}\right\|_1 \\ &\quad + \|u_1\|_1 + \|u_2\|_1 + ... + \|u_n\|_1 . \end{aligned} \tag{4.13}$$

Other convex cost functions, such as a quadratic costs, are also possible.

4.5 Simulation Experiments

In our simulation experiments, the linear program in eq. (4.12) uses eq. (4.13) as the cost function. Various solvers such as ECOS [17], GLPK [47], or MOSEK [4] can solve it. This work implements eq. (4.12) and eq. (4.13) using Julia's Convex.jl

modeling toolbox [71] and the MOSEK solver.

The satellite constellation considered consists of identical 1.5kg CubeSats with a 15cm × 10cm × 10cm chassis and equipped with two deployable solar panels each with dimension 20cm × 15cm × 0.3cm. The satellite's achievable drag ratio is 7.5:1, defined by the equation:

$$D_{ratio} = \frac{D_{max}}{D_{min}} = \frac{A_{max}}{A_{min}}. \tag{4.14}$$

This ratio quantifies the ability of the satellite to modify drag by adjusting its attitude. Consequently, the input constraints for eq. (4.12), namely $u_{\min}$ and $u_{\max}$, are set to values of $1/7.5 \approx 0.13$ and 1, respectively.

4.5.1 Trajectory Optimization

In this experiment we solve eq. (4.12) once for a pair of satellites deployed at 440 km altitude and with an inclination of 51.5° — conditions that approximate deployment from the International Space Station (ISS). The final conditions are set to $\Delta\bar{\theta}_f = 0$ and $\ell = 2$. From eq. (4.11), this results in $\Delta\bar{\Omega}_f = 1.4°$, for a maximum cross-track distance of 165 km. In this scenario, the altitude limits, $\Delta a_{\max}$ and $\Delta a_{\min}$, are set to ±10km.

The optimization solution for a 1500 orbit time horizon, is in fig. 4.3. The top plot shows the drag control trajectory. The bottom three plots show the change in Δa, $\Delta\bar{\theta}$, and $\Delta\bar{\Omega}$ respectively. To increase the relative AoL and RAAN, the orbital altitude of the second satellite is decreased first. The relative AoL increases by 720°, or two full orbits, and at the end the first satellite lowers its altitude to exactly reach $\Delta\bar{\theta}_f = 0$. The 10km altitude constraint was also satisfied. This optimization took 0.8s to solve on a MacBook Pro with an Apple M1 Pro processor.

4.5.2 Closed-Loop Simulation Results

This experiment explores the impacts of realistic modeling errors and disturbances on spacecraft through closed-loop simulations. These simulations integrate additional perturbations not included in (4.9), including the effects of Earth's rotation on drag, the influence of the first five zonal harmonics (J_1-J_6) for gravity, and an initial orbit eccentricity of $e = 0.005$.

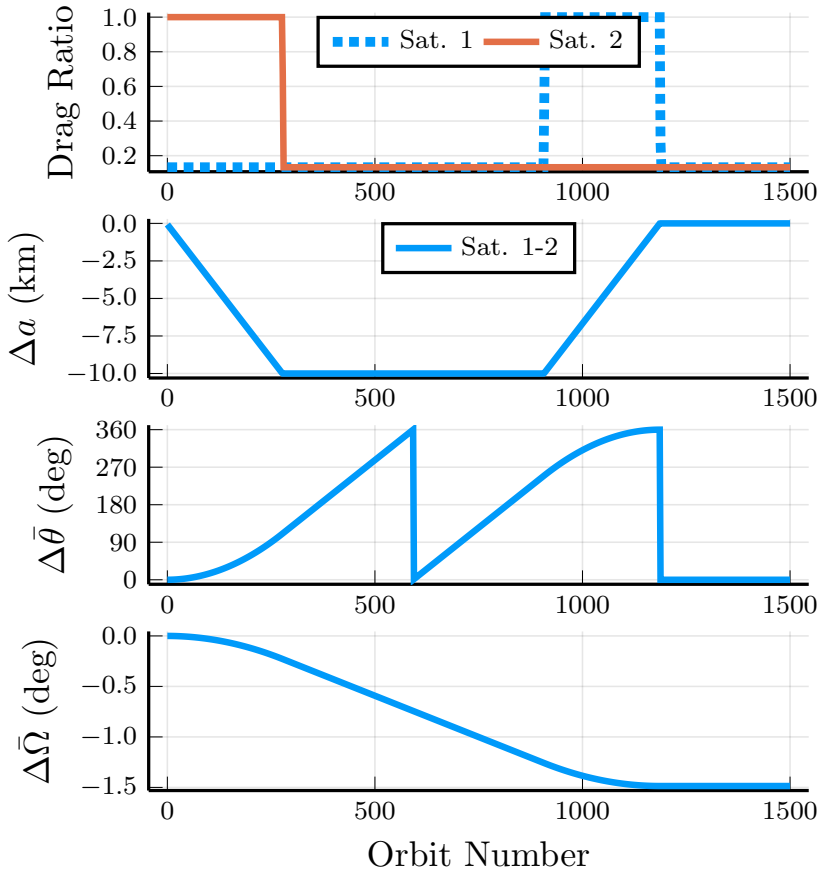
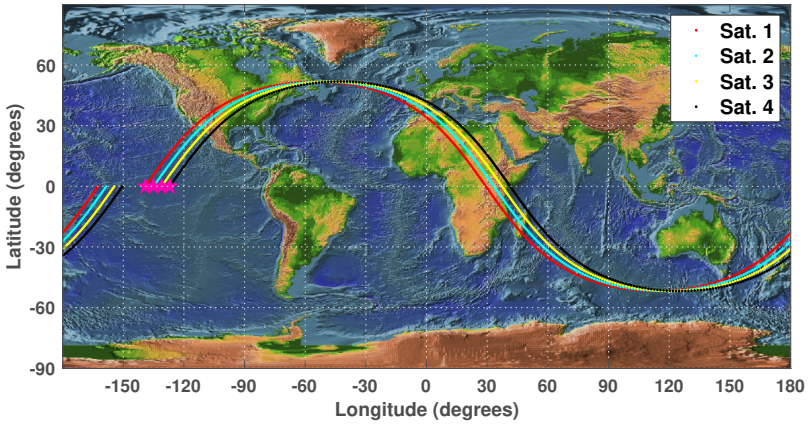


Figure 4.3: Optimized trajectory for a two-satellite formation. Top: the drag ratios. Second: the relative altitude. Third: the AoL between the two satellites. Bottom: the RAAN between the two satellites. The satellites end at the same altitude, resulting in a constant final AoL and RAAN.

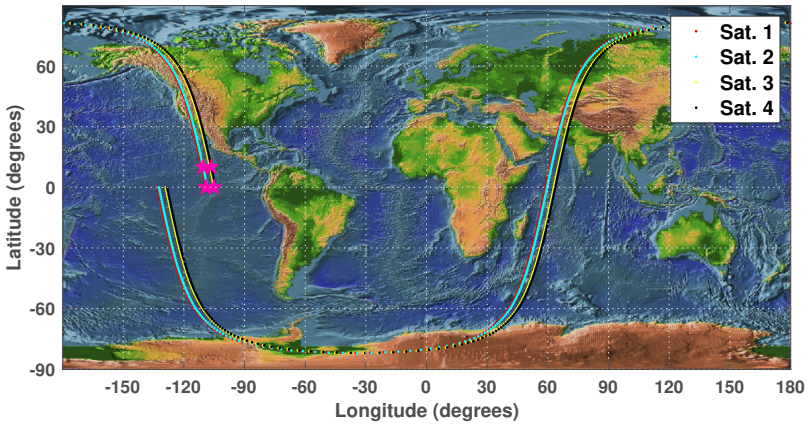
To address the challenges posed by modeling errors and disturbances, a model-predictive control (MPC) methodology was employed. This method operates as a receding-horizon loop where, during each iteration, the optimization problem is re-solved using the spacecraft's current measured state. The updated control inputs are then applied until the next iteration. The update frequency of this control loop is once per orbital period. The terminal cost and terminal constraints in eq. (4.12) ensure the stability of this MPC approach [52].

The receding-horizon control algorithm was applied in two scenarios, depicted in figs. 4.4a and 4.4b and detailed in section 4.5.2. The initial state for these scenarios

was chosen to emulate common orbits CubeSats are deployed in; due to the ISS and SpaceX Transporter deployments. A constant assumption across these scenarios is that all satellites maintain the same drag ratio and start from a uniform state.



(a)



(b)

Figure 4.4: Ground tracks of desired final formation configuration scenarios. (a) Scenario 1: formation of four satellites in a line with equally distributed RAAN. (b) Scenario 2: formation of four satellites distributed in AoL and RAAN to form the vertices of a square.

Scenario 1 — Line Formation

Scenario 1 assumes that four satellites are deployed from the ISS, with an altitude of 440 km, $e = 0.005$, and $i = 51.5°$. The goal is to maneuver the satellites to be equally distributed in the cross-track direction with zero change in AoL, so they pass

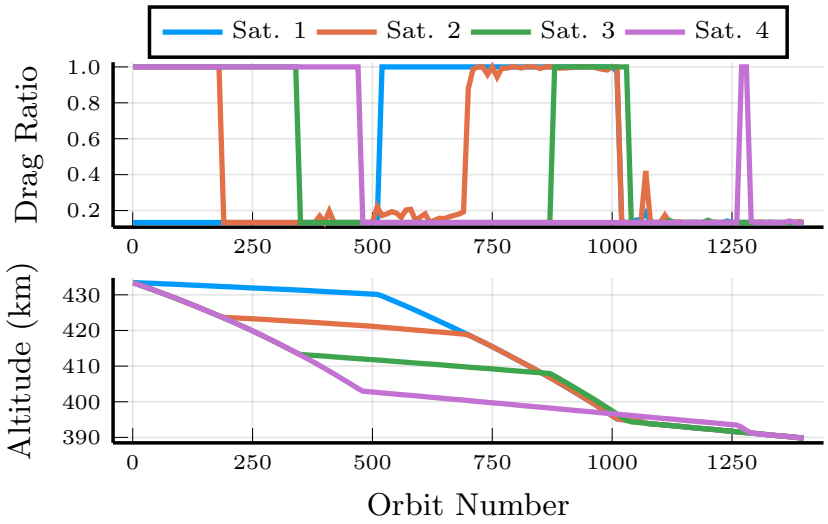


Figure 4.5: Scenario 1. Top: The control trajectories for the four satellites. Bottom: The altitude variation of the four satellites. Unlike in fig. 4.3, the control trajectories are not piecewise constant due to the on-line correction of disturbances.

over the equator in a line, as depicted in fig. 4.4a. This corresponds to $\Delta\bar{\theta}_f = 0$ and $\Delta\bar{\Omega}_f = k_4 2\pi\ell$ with $\ell = 1, 2, 3$. In this scenario, the receding-horizon control policy is re-solved once per orbit over a time horizon of 1400 orbits and the altitude limits $\Delta a_{\max}$ and $\Delta a_{\min}$ were set to ± 100 km.

The results of the first scenario are presented in figs. 4.5 and 4.6 and table 4.1. The final orbit has an altitude of 389.73 km, $e = 0.003$, and $i = 51.477°$. The top plot of fig. 4.5 shows the control trajectories for the four satellites. The fourth satellite, the satellite that aims to reach the largest $\Delta\bar{\Omega}$, drives the overall differential drag required for the formation. The bottom plot shows the altitude variation for the four satellites; when the altitude rate is steeper, the satellite is in a high drag configuration. Contrarily, where the altitude rate is shallower, the satellite is in a low drag configuration. Figure 4.6 shows the AoL and RAAN difference for the three satellite pairs. The difference is calculated with respect to the chief satellite.

Table 4.1 reports the overall maneuver time, the final difference in the AoL and the RAAN, and the spherical distance between the chief satellite and the other satellite. It takes three months to reach the final configuration, and the maximum final distance between two satellites is 268.3 km.

On average, each optimization took 0.88s to solve on a MacBook Pro with an

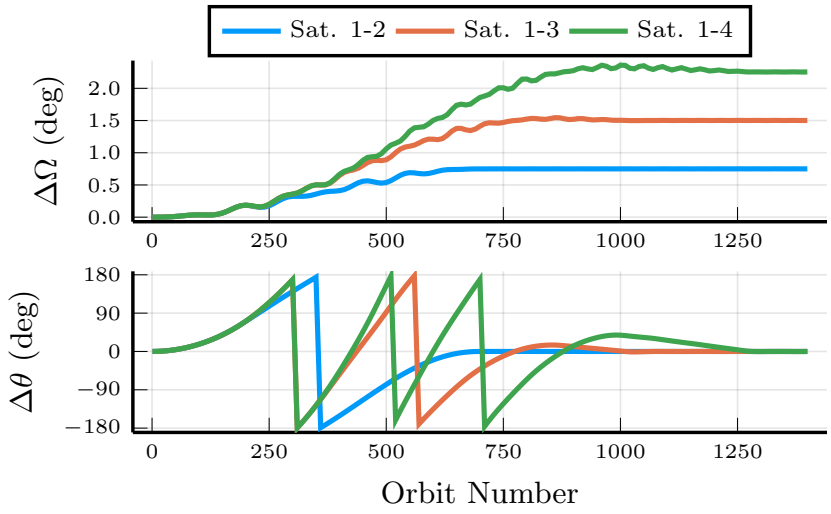


Figure 4.6: Scenario 1. Top: RAAN difference with respect to the chief satellite. Bottom: AoL difference with respect to the chief satellite. All the satellite reach the same final AoL.

Apple M1 Pro processor.

Pair	t_f, months	$\Delta\theta_f$, deg	$\Delta\Omega_f$, deg	Spherical Distance, km
Sat. 1 - 2	3	-0.007	-0.75	89.28
Sat. 1 - 3		0.005	-1.5	178.7
Sat. 1 - 4		-0.05	-2.25	268.3

Table 4.1: Results for Scenario 1

Scenario 2 — Square Formation

The second scenario assumes that four satellites are deployed from an approximately sun-synchronous SpaceX Transporter launch, corresponding to an altitude of 550 km, an $e = 0.005$, and $i = 98°$. The goal is to maneuver the satellites to be distributed in AoL and RAAN to form the vertices of a square, as depicted in fig. 4.4b. For this scenario, the ℓ values are 0, 4, and 4, while the $\Delta\bar{\theta}_f$ are 0.03, 0, and 0.03. The receding-horizon control policy is re-solved every orbit over a time horizon of 4100 orbits.

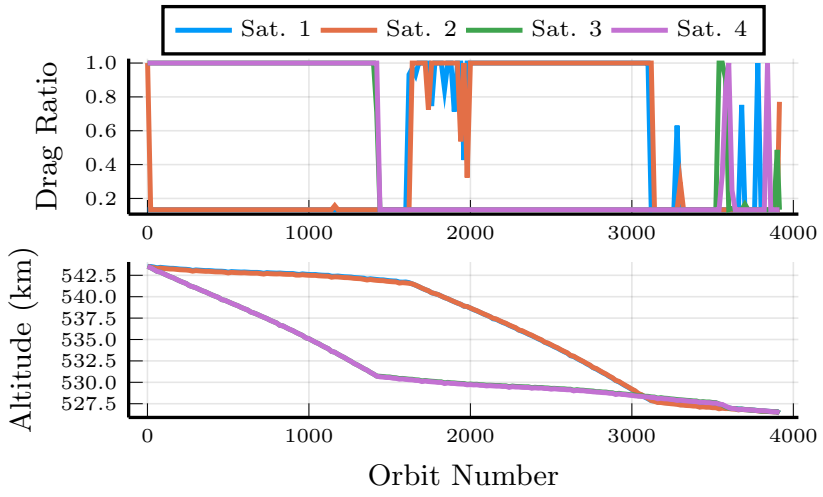


Figure 4.7: Scenario 2. Top: The control trajectories for the four satellites. Bottom: The altitude variation of the four satellites. Notice that unlike in fig. 4.5, the satellites here change altitude in pairs, with only slight deviations to adjust for the desired AoL difference.

Pair	t_f, months	$\Delta\theta_f$, deg	$\Delta\Omega_f$, deg	Spherical Distance, km
Sat. 1 - 2	8.6	10.74	-0.006	1299
Sat. 1 - 3		0.005	-0.63	76
Sat. 1 - 4		10.84	-0.64	1313.5

Table 4.2: Results for Scenario 2

The results of scenario 2 are presented in fig. 4.7, fig. 4.8, and table 4.2. The final orbit has a 514.1km altitude, e of 0.0043, and i of 98°. The top plot of fig. 4.7 shows the control input, and the bottom plot shows the altitude change for the four satellites. The plots in fig. 4.8 report the AoL and RAAN difference for the three pairs. As before, the difference is evaluated with respect to the chief satellite. Table 4.2 reports the overall maneuver time, the AoL and RAAN final differences, and the spherical distance between the chief satellite and every other satellite. This scenario takes longer than the first scenario; however, the results show that in less advantageous initial conditions a spacecraft formation with both along-track and cross-track separations can be established using our presented drag-based method. Furthermore, the algorithm is able to define the control trajectory in the presence of

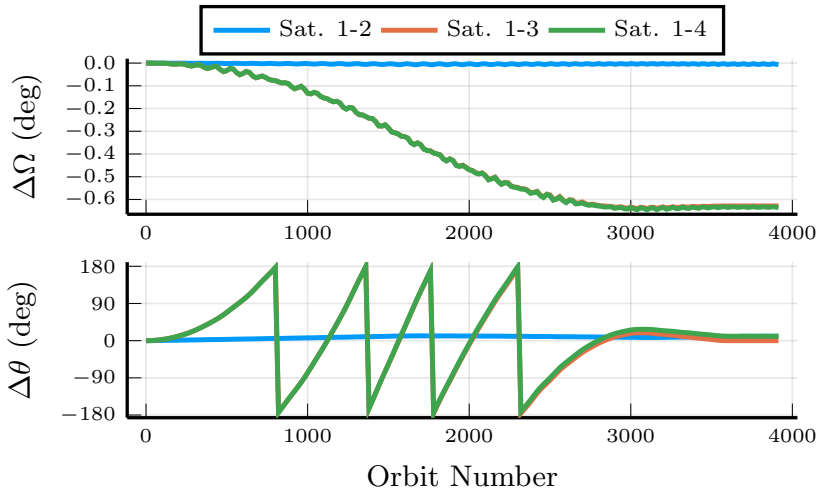


Figure 4.8: Scenario 2. Top: RAAN difference with respect to the chief satellite. Bottom: AoL difference with respect to the chief satellite. Satellites 1-2 and satellites 3-4 reach the same RAAN, while satellites 1-4 and satellites 2-3 reach a comparable AoL.

disturbances and modeling errors. The optimizations took an average of 4.2s each to solve on an Apple M1 Pro MacBook Pro. This took longer than the first scenario due to the extended time horizon.

The two scenarios have interesting differences from a mission-design viewpoint. Lower orbits, like the ISS orbit, result in faster natural orbital decay due to drag, reducing the possible altitude change. However, the lower inclination for the ISS results in a larger k_4 and faster $\Delta\bar{\Omega}$ rate of 0.745° per 2π revolution of$\Delta\bar{\theta}$. Contrarily, deployment from the SpaceX Transporter allows a larger available overall altitude change but a smaller $\Delta\bar{\Omega}$ rate of 0.16° per 2π revolution of $\Delta\bar{\theta}$. This is why scenario 2 takes longer to complete than scenario 1.

4.6 Conclusions

We have presented a novel control scheme that is able to maneuver a low-Earth orbit satellite formation in both along-track and cross-track directions without expending propellant. The drag-based formation control is formulated as a linear program, with solution times of less than one second. This allows it to be used in a receding-horizon

manner, updating the control inputs and trajectory for a satellite once per orbit. Simulation results confirm the robustness of the proposed method to disturbances and viability for autonomous on-orbit implementation. While the scheme assumes known atmospheric density, the actual atmospheric density in low-Earth orbit is widely varying. Future extensions will focus on accurate atmospheric drag estimation, which can be integrated into the trajectory optimization, ensuring robust performance. The proposed approach has the capability to significantly reduce the cost and complexity of deploying multi-plane satellite formations by eliminating the need for propulsion systems onboard.

Chapter 5

Magnetorquer Detumbling

5.1 Introduction

After a spacecraft is deployed on orbit, a common first phase of operation is detumbling. In this phase, the angular velocity of the satellite is reduced from tens of degrees per second to rates that are tolerated by the satellite mission or managed by other onboard control systems. To perform detumbling, the spacecraft must reduce its total angular momentum by one to two orders of magnitude. This is only accomplished by generating external torques, either through expending propellant, or, in low-Earth orbit, with magnetic torque coils (magnetorquers) that exchange momentum with the Earth's magnetic field.

Magnetorquers are appealing because they do not require expending propellant. However, at any instant in time, they only generate torque in a two-dimensional subspace perpendicular to the Earth's local magnetic-field vector. To prove convergence, most common magnetorquer detumbling controllers, including the classic B-dot and B-cross controllers [49, 50], condition their convergence guarantee on the motion of the satellite through the Earth's magnetic field. For all but equatorial orbits, this makes the magnetic field time varying in the orbit frame. Over the spacecraft's full orbit complete control authority is achieved. In this work, we demonstrate that these classic controllers, and their modern variants, can take many hours to detumble a spacecraft, despite it being possible to detumble much faster and with much less total control effort. These controllers are, therefore, very inefficient and waste precious

energy and time during the early stages of satellite operation.

To mitigate the inefficiencies of the classic magnetic detumbling controllers, we present a novel controller that uses a prediction of the future magnetic field vector to dramatically improve convergence time. The magnetic field prediction is done using only gyroscope and magnetometer sensor measurements; no inertial attitude or position reference is required. The controller is based on a discrete-time non-monotonic Lyapunov function [1], which is able to temporarily increase the angular momentum of the spacecraft, allowing the system to move away from control singularities. We demonstrate detumble times less than half those of other controllers in the literature across 100 Monte-Carlo simulation runs with randomly sampled initial conditions.

Our contributions include:

- A survey and unified presentation of the numerous magnetorquer detumbling controllers that exist in the literature
- A derivation and analytic proof of convergence of our predictive detumbling controller based on a discrete-time non-monotonic Lyapunov function
- Monte-Carlo simulation experiments showing the performance of our predictive controller in comparison to five other controllers from the literature

The chapter proceeds as follows: In section 5.2 we discuss prior work on magnetorquer detumbling. Then, in section 5.3 we present the attitude dynamics of a spacecraft and provide a unified derivation of five detumbling controllers from the literature that we compare ours to. We also provide a brief introduction to non-monotonic Lyapunov functions. Section 5.4 derives our discrete non-monotonic detumbling controller, and section 5.5 presents our Monte-Carlo simulation results. Finally, we summarize our conclusions and directions for future work in section 5.6.

5.2 Related Work

Magnetorquer detumbling has a long history dating back to the earliest days of space exploration [49, 63]. In general, magnetorquer detumbling controllers come in two categories with many variants: B-dot and B-cross. B-dot controllers assume only magnetometer measurements are available. B-cross [5, 50] controllers assume both magnetometer and gyroscope measurements are available onboard the spacecraft. As

we will show in section 5.3.2, these two categories are related by a simple approximation, and the many variations in the literature reduce to a selection of gains and saturation methods for handling control limits [15, 16, 33].

In addition to the magnetic detumbling methods discussed here, there has been significant work on full magnetic attitude control, including the work by Wisniewski [77] which models the magnetic field as a periodic system, and more recent work that utilizes numerical optimal control to perform three-axis magnetorquer attitude control [24, 59]. Ovchinnikov presents a recent survey of both magnetorquer detumbling and attitude control [56].

5.3 Background

5.3.1 Attitude Dynamics

Let $h \in \mathbb{R}^3$ be the angular momentum of a spacecraft, $B \in \mathbb{R}^3$ be the Earth's local geomagnetic field vector at the spacecraft's location, and $\mu \in \mathbb{R}^3$ be the dipole moment produced by the magnetorquers.

With a magnetic dipole moment as input, a spacecraft's angular momentum dynamics expressed in an inertial reference frame are,

$$\dot{h} = \tau = -B \times \mu = -\hat{B}\mu, \tag{5.1}$$

where $\tau \in \mathbb{R}^3$ is the torque on the spacecraft and $\hat{B}$ is the skew-symmetric cross product matrix,

$$\hat{B} = \begin{bmatrix} 0 & -B_z & B_y \\ B_z & 0 & -B_x \\ -B_y & B_x & 0 \end{bmatrix}. \tag{5.2}$$

With inertia matrix $J \in \mathbb{R}^{3\times3}$, the angular momentum relates to the angular velocity as

$$h = J\omega. \tag{5.3}$$

We make use of the time derivative of the geomagnetic field vector with respect to the body frame, $\dot{B}^{\mathcal{B}}$, and with respect to the inertial frame, $\dot{B}^{\mathcal{N}}$. Both are expressed in body-fixed coordinates. The relationship between these quantities is

$$\dot{B}^{\mathcal{N}} = \hat{\omega} B + \dot{B}^{\mathcal{B}}. \tag{5.4}$$

5.3.2 Detumbling Control

Many of the detumbling control laws found in the literature are variations of a single control law that is derived from the Lyapunov function

$$V = \frac{1}{2} h^T h. \tag{5.5}$$

Taking the time derivative,

$$\dot{V} = h^T \dot{h} = -h^T \hat{B} \mu. \tag{5.6}$$

We desire to find μ that minimizes $\dot{V}$ at every instant in time. To do so, we formulate this as an optimization problem with bound constraints that limit the maximum dipole moment the satellite can produce:

$$\begin{aligned} \underset{\mu}{\text{minimize}} \quad & \dot{V} = -h^T \hat{B} \mu \\ \text{subject to} \quad & -\mu_{\max} \leq \bar{\mu} \leq \mu_{\max}. \end{aligned} \tag{5.7}$$

This optimization problem in eq. (5.7) is a linear program with a closed-form solution in the form of a bang-bang control law:

$$\mu = \text{sign}(\hat{h} B) \mu_{\max}, \tag{5.8}$$

where the sign function is interpreted element-wise. Bang-bang controllers like eq. (5.8) are prone to chattering in the presence of noise, so we replace the sign hard saturation with a soft saturation,

$$\mu = \tanh(k \hat{h} B) \mu_{\max}, \tag{5.9}$$

where the tanh function is, again, interpreted element-wise and k is a tuning parameter. We refer to this control law as the Lyapunov momentum control law.

The control law in eq. (5.8) is closely related to the classical B-dot and B-cross control laws [50]. The B-cross law replaces h with ω and relaxes the bang-bang saturation to a linear feedback law with gain k,

$$\mu = k\hat{\omega}B. \tag{5.10}$$

Avanzini and Giulietti [5] propose selecting the B-cross controller gain

$$k = 2\frac{1}{\sqrt{a^3/GM}}(1 + \sin(\xi_m))\lambda_{\min} \tag{5.11}$$

where a is the orbit semi-major axis, GM is the Earth's gravitational parameter, ξ_m is the orbit's geomagnetic inclination and $\lambda_{\min}$ is the minimum eigenvalue of the spacecraft's inertia matrix J.

The B-dot law [49, 63] modifies eq. (5.10) by making the assumption that

$$\hat{\omega}B \approx -\dot{B}^{\mathcal{B}}, \tag{5.12}$$

resulting in

$$\mu = -k\dot{B}^{\mathcal{B}}. \tag{5.13}$$

The B-dot law has the advantage that $\dot{B}^{\mathcal{B}}$ is readily estimated from a magnetometer only, so no gyroscope measurements are required for its implementation. However, as $\omega \to 0$, the approximation in eq. (5.12) becomes less accurate and the B-dot law suffers from long convergence times.

Desouky [15, 16] presented two control laws: Their "time-optimal" control law is equivalent to eq. (5.8) and their "B-dot Variant" control law inverts eq. (5.12) with a regularizing term to solve for ω and substitutes the result into eq. (5.10) to obtain the control law,

$$\mu = -k\hat{B}(\epsilon I + \hat{B})^{-1}\dot{B}^{\mathcal{B}} \tag{5.14}$$

where $0 < \epsilon \ll 1$, and $\epsilon = 1 \times 10^{-6}$ in practice.

Invernizzi and Lovera [33] use a projection-based method to compute a time-varying gain for an unsaturated version of eq. (5.8),

$$\begin{aligned} k &= -k_1 \exp\left(-k_2 \frac{B^T J h}{\|B\|(\|h\| + \epsilon)}\right) \\ \mu &= k \hat{h} B. \end{aligned} \tag{5.15}$$

All of the previously discussed methods are essentially the same and suffer from the same fundamental limitation: when h (or ω) and B are aligned, their cross product is zero, and the commanded control input goes to zero. The controllers are convergent on long time scales because B is time varying in the orbital frame, so eventually $\hat{h}B$ will no longer be zero. However, they are prone to getting stuck in this uncontrollable subspace, resulting in long convergence times.

Consider the B-cross controller in eq. (5.10). If we decompose $\omega = \omega^{\parallel} + \omega^{\perp}$, where $\omega^{\parallel}$ and $\omega^{\perp}$ are the components of ω parallel and perpendicular to B,

$$\mu = k(\omega^{\parallel} + \omega^{\perp}) \times B = k\hat{\omega}^{\perp} B \tag{5.16}$$

and

$$\dot{h} = -\hat{B}\mu = -k\hat{B}(\hat{\omega}^{\perp} B) = -\lambda \omega^{\perp} \tag{5.17}$$

for some $\lambda > 0$. So, h is only reduced in the $\omega^{\perp}$ direction with no change in the $\omega^{\parallel}$ direction. Figure 5.1 shows this effect in two simulation runs of the B-cross controller with the same initial conditions and two different gains. At the beginning, the smaller-gain controller decreases $\|h\|$ at a slower rate, but ultimately converges sooner because the larger gain causes the controller to get stuck on the uncontrollable subspace where ω and B are parallel. The smaller gain was chosen based on eq. (5.11) and the larger gain is a factor of 100 larger. The gain sweep results in fig. 5.2 show a similar phenomena occurring with the other controllers. While this can be partially avoided with appropriate tuning, this does not guarantee the controller will converge without getting stuck.

To address this issue, we relax a fundamental constraint on the controllers presented

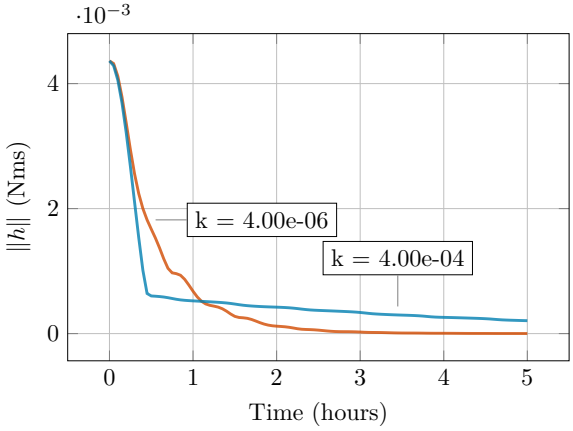


Figure 5.1: Two simulation runs of the B-cross controller in eq. (5.10) with different gains. As the gain increases, the controller convergence gets worse because it gets stuck in the uncontrollable subspace where ω and B are parallel.

so far: we derive a controller that does not decrease the angular momentum of the spacecraft monotonically, but still maintains a Lyapunov convergence guarantee. This allows the controller to trade off decreasing the angular momentum while moving towards the uncontrollable subspace or increasing the angular momentum while making the angular momentum more controllable in the future.

5.4 Non-monotonic Control Derivation

We begin by introducing discrete-time monotonic Lyapunov analysis, then extend it to non-monotonic Lyapunov analysis. The discrete-time dynamical system

$$x_{k+1} = f(x_k) \tag{5.18}$$

with $x \in \mathbb{R}^n$ has a globally asymptotically stable (GAS) equilibrium at $x = 0$ if there exists a Lyapunov function $V(x) : \mathbb{R}^n \to \mathbb{R}$ such that

$$V(x) > 0 \qquad \forall x \neq 0 \tag{5.19a}$$

$$V(0) = 0 \tag{5.19b}$$

$$V_{k+1} < V_k \qquad \forall k \tag{5.19c}$$

where we use the notation $V_k = V(x_k)$. It is well known that in general there is no guaranteed method of finding a Lyapunov function that satisfies eq. (5.19), even if the system is GAS. Ahmadi and Parrilo suggest that the monotonic decrease condition in eq. (5.19c) may be too restrictive, and present several alternative stability theorems that only require V to decrease on average [1]. We rely on Theorem 2.1 from their work. It modifies the conditions in eq. (5.19) so that eq. (5.18) is GAS at $x = 0$ if there exists a scalar $\alpha \geq 0$ and a Lyapunov function $V : \mathbb{R}^n \to \mathbb{R}$ such that

$$V(x) > 0 \qquad \forall x \neq 0 \tag{5.20a}$$

$$V(0) = 0 \tag{5.20b}$$

$$\alpha(V_{k+2} - V_k) + (V_{k+1} - V_k) < 0 \qquad \forall k. \tag{5.20c}$$

The condition in eq. (5.20c) relaxes eq. (5.19c) to allow V_k to decrease on average between two timesteps.

5.4.1 Non-Monotonic Detumbling

Now, to derive the non-monotonic detumbling controller, we begin with the discrete-time Lyapunov function

$$V_k = \frac{1}{2} h_k^T h_k. \tag{5.21}$$

This trivially satisfies eqs. (5.20a) and (5.20b), so it remains to design the control input μ such that the non-monotonic Lyapunov condition

$$\Delta V = \alpha(V_{k+2} - V_k) + (V_{k+1} - V_k) < 0 \tag{5.22}$$

from eq. (5.20c) is satisfied for $\alpha \geq 0$.

Through the rest of this section we drop the subscript k and use $[\cdot]_0 = [\cdot]_k$, $[\cdot]_1 = [\cdot]_{k+1}$, $[\cdot]_2 = [\cdot]_{k+2}$ for clarity.

We approximate the discrete time dynamics in eq. (5.1) using Euler integration,

so that

$$h_1 \approx h_0 + \Delta t \dot{h}_0 \tag{5.23a}$$
$$= h_0 + \Delta t \tau_0 \tag{5.23b}$$
$$= h_0 + \Delta t(\mu_0 \times B_0) \tag{5.23c}$$

and

$$h_2 \approx h_1 + \Delta t \dot{h}_1 \tag{5.24a}$$
$$= h_0 + \Delta t \tau_0 + \Delta t \tau_1 \tag{5.24b}$$
$$= h_0 + \Delta t(\mu_0 \times B_0) + \Delta t(\mu_1 \times B_1). \tag{5.24c}$$

Substituting,

$$V_1 = \frac{1}{2} h_1^T h_1 \tag{5.25a}$$
$$= \frac{1}{2} \left(h_0 + \Delta t \tau_0\right)^T \left(h_0 + \Delta t \tau_0\right) \tag{5.25b}$$
$$= \frac{1}{2} \left(h_0^T h_0 + \Delta t h_0^T \tau_0 + \Delta t \tau_0^T h_0 + \Delta t^2 \tau_0^T \tau_0\right) \tag{5.25c}$$
$$= \frac{1}{2} \left(h_0^T h_0 + 2\Delta t h_0^T \tau_0 + \Delta t^2 \tau_0^T \tau_0\right) \tag{5.25d}$$

and

$$V_2 = \frac{1}{2} h_2^T h_2 \tag{5.26a}$$
$$= \frac{1}{2} \left(h_0 + \Delta t \tau_0 + \Delta t \tau_1\right)^T \left(h_0 + \Delta t \tau_0 + \Delta t \tau_1\right) \tag{5.26b}$$
$$\begin{aligned} = \frac{1}{2} \big(&h_0^T h_0 + \Delta t h_0^T \tau_0 + \Delta t h_0^T \tau_1 \\ &+ \Delta t \tau_0^T h_0 + \Delta t^2 \tau_0^T \tau_0 + \Delta t^2 \tau_0^T \tau_1 \\ &+ \Delta t \tau_1^T h_0 + \Delta t^2 \tau_1^T \tau_0 + \Delta t^2 \tau_1^T \tau_1\big) \end{aligned} \tag{5.26c}$$
$$\begin{aligned} = \frac{1}{2} \big(&h_0^T h_0 + 2\Delta t h_0^T \tau_0 + 2\Delta t h_0^T \tau_1 + \Delta t^2 \tau_0^T \tau_0 \\ &+ 2\Delta t^2 \tau_0^T \tau_1 + \Delta t^2 \tau_1^T \tau_1\big) \end{aligned} \tag{5.26d}$$

so

$$V_1 - V_0 = \frac{1}{2}\left(h_0^T h_0 + 2\Delta t h_0^T \tau_0 + \Delta t^2 \tau_0^T \tau_0\right) - \frac{1}{2} h_0^T h_0 \tag{5.27a}$$

$$= \Delta t h_0^T \tau_0 + \frac{1}{2}\Delta t^2 \tau_0^T \tau_0 \tag{5.27b}$$

$$= -\Delta t h_0^T \hat{B}_0 \mu_0 + \frac{1}{2}\Delta t^2 \mu_0^T \hat{B}_0^T \hat{B}_0 \mu_0 \tag{5.27c}$$

and

$$V_2 - V_0 = \frac{1}{2}\big(h_0^T h_0 + 2\Delta t h_0^T \tau_0 + 2\Delta t h_0^T \tau_1 + \Delta t^2 \tau_0^T \tau_0 + 2\Delta t^2 \tau_0^T \tau_1 + \Delta t^2 \tau_1^T \tau_1\big) - \frac{1}{2} h_0^T h_0 \tag{5.28a}$$

$$= \Delta t h_0^T \tau_0 + \Delta t h_0^T \tau_1 + \frac{1}{2}\Delta t^2 \tau_0^T \tau_0 + \Delta t^2 \tau_0^T \tau_1 + \frac{1}{2}\Delta t^2 \tau_1^T \tau_1 \tag{5.28b}$$

$$= -\Delta t h_0^T \hat{B}_0 \mu_0 - \Delta t h_0^T \hat{B}_1 \mu_1 + \frac{1}{2}\Delta t^2 \mu_0^T \hat{B}_0^T \hat{B}_0 \mu_0 + \Delta t^2 \mu_0^T \hat{B}_0^T \hat{B}_1 \mu_1 + \frac{1}{2}\Delta t^2 \mu_1^T \hat{B}_1^T \hat{B}_1 \mu_1 \tag{5.28c}$$

$$\tag{5.28d}$$

where we used the identities

$$\tau = \mu \times B = -B \times \mu = -\hat{B}\mu \tag{5.29a}$$

$$\tau^T \tau = (\mu \times B)^T (\mu \times B) = (-B \times \mu)^T (-B \times \mu) = (-\hat{B}\mu)^T (-\hat{B}\mu) = \mu^T \hat{B}^T \hat{B} \mu. \tag{5.29b}$$

Bringing these terms together, we have

$$\Delta V \triangleq \alpha(V_2 - V_0) + (V_1 - V_0) \tag{5.30a}$$

$$= \alpha\big(- \Delta t h_0^T \hat{B}_0 \mu_0 - \Delta t h_0^T \hat{B}_1 \mu_1 + \frac{1}{2}\Delta t^2 \mu_0^T \hat{B}_0^T \hat{B}_0 \mu_0 + \Delta t^2 \mu_0^T \hat{B}_0^T \hat{B}_1 \mu_1 + \frac{1}{2}\Delta t^2 \mu_1^T \hat{B}_1^T \hat{B}_1 \mu_1\big) \tag{5.30b}$$

$$- \Delta t h_0^T \hat{B}_0 \mu_0 + \frac{1}{2}\Delta t^2 \mu_0^T \hat{B}_0^T \hat{B}_0 \mu_0$$

$$= -(\alpha+1)\Delta t h_0^T \hat{B}_0 \mu_0 - \alpha \Delta t h_0^T \hat{B}_1 \mu_1 + (\alpha+1)\Delta t^2 \mu_0^T \hat{B}_0^T \hat{B}_0 \mu_0 + \alpha \Delta t^2 \mu_0^T \hat{B}_0^T \hat{B}_1 \mu_1 + \alpha \frac{1}{2}\Delta t^2 \mu_1^T \hat{B}_1^T \hat{B}_1 \mu_1 \tag{5.30c}$$

$$= \frac{1}{2}\alpha \Delta t^2 \begin{bmatrix} \mu_0^T & \mu_1^T \end{bmatrix} \begin{bmatrix} \hat{B}_0^T \hat{B}_0 & \hat{B}_0^T \hat{B}_1 \\ \hat{B}_1^T \hat{B}_0 & \hat{B}_1^T \hat{B}_1 \end{bmatrix} \begin{bmatrix} \mu_0 \\ \mu_1 \end{bmatrix} - \Delta t h_0^T \begin{bmatrix} \hat{B}_0^T & \hat{B}_1^T \end{bmatrix} \begin{bmatrix} \mu_0 \\ \mu_1 \end{bmatrix} + \frac{1}{2}\Delta t^2 \begin{bmatrix} \mu_0^T & \mu_1^T \end{bmatrix} \begin{bmatrix} \hat{B}_0^T \hat{B}_0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \mu_0 \\ \mu_1 \end{bmatrix} - \Delta t h_0^T \begin{bmatrix} \hat{B}_0 & 0 \end{bmatrix} \begin{bmatrix} \mu_0 \\ \mu_1 \end{bmatrix} \tag{5.30d}$$

$$= \alpha \frac{1}{2}\Delta t^2 \bar{\mu}^T \bar{B} \bar{B}^T \bar{\mu} - \alpha \Delta t h_0^T \bar{B}^T \bar{\mu} + \frac{1}{2}\Delta t^2 \bar{\mu}^T Z \bar{B} \bar{B}^T Z \bar{\mu} - \Delta t h_0^T \bar{B}^T Z \bar{\mu} \tag{5.30e}$$

$$= \frac{1}{2}\bar{\mu}^T Q_1 \bar{\mu} + \frac{1}{2}\alpha \bar{\mu}^T Q_2 \bar{\mu} - q_1^T \bar{\mu} - \alpha q_2^T \bar{\mu} \tag{5.30f}$$

$$= \frac{1}{2}\bar{\mu}^T (Q_1 + \alpha Q_2)\bar{\mu} - (q_1 + \alpha q_2)^T \bar{\mu} \tag{5.30g}$$

where we defined

$$\bar{\mu} = \begin{bmatrix} \mu_0 \\ \mu_1 \end{bmatrix} \in \mathbb{R}^6, \tag{5.31a}$$

$$\bar{B} = \begin{bmatrix} \hat{B}_0^T \\ \hat{B}_1^T \end{bmatrix} \in \mathbb{R}^{6\times 3} \tag{5.31b}$$

$$Z = \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{3\times 3} \tag{5.31c}$$

$$Q_1 = \Delta t^2 Z \bar{B} \bar{B}^T Z \tag{5.31d}$$

$$Q_2 = \alpha \Delta t^2 \bar{B} \bar{B}^T \tag{5.31e}$$

$$q_1 = \Delta t (h_0^T \bar{B}^T Z)^T \tag{5.31f}$$

$$q_2 = \alpha \Delta t (h_0^T \bar{B}^T)^T. \tag{5.31g}$$

Since $\bar{B}\bar{B}^T$ is symmetric and $\text{rank}(\bar{B}\bar{B}^T) = \text{rank}(\bar{B}) \leq 3 < 6$, Q_1 and Q_2 are symmetric and positive semi-definite.

This means that ΔV is convex, and, as we will see in the following, its minimum is less than zero. So, it is possible to find $\bar{\mu}$ such that $\Delta V < 0$, and the non-monotonic Lyapunov conditions of eq. (5.20c) are satisfied.

We wish to find a control law for $\bar{\mu}$ such that ΔV is minimized:

$$\begin{aligned} \underset{\bar{\mu}}{\text{minimize}} \quad & \Delta V = \frac{1}{2}\bar{\mu}^T (Q_1 + \alpha Q_2)\bar{\mu} - (q_1 + \alpha q_2)^T \bar{\mu} \\ \text{subject to} \quad & -\mu_{\max} \leq \bar{\mu} \leq \mu_{\max}. \end{aligned} \tag{5.32}$$

Since Q_1, and Q_2 are positive semi-definite, ΔV is not strictly convex and eq. (5.32) has multiple minima. We add $\beta\mu^T\mu/2$ with $1 \gg \beta > 0$ as a regularizing term to make the objective strictly convex. The result is a convex quadratic program that is reliably and quickly solved with a numerical solver. Alternatively, we can make the same simplification as in eq. (5.9) and solve the unconstrained minimization problem, enforcing a soft saturation constraint on the result. The optimization is then

$$\begin{aligned} \underset{\bar{\mu}}{\text{minimize}} \quad & F = \frac{1}{2}\beta\bar{\mu}^T\bar{\mu} + \frac{1}{2}\bar{\mu}^T(Q_1 + \alpha Q_2)\bar{\mu} \\ & \qquad - (q_1 + \alpha q_2)^T\bar{\mu} \end{aligned}. \tag{5.33}$$

We find the analytic solution by taking the gradient of F with respect to $\bar{\mu}$ and setting it to zero. The gradient of F is

$$\nabla F = \beta\bar{\mu} + (Q_1 + \alpha Q_2)\bar{\mu} - (q_1 + \alpha q_2). \tag{5.34}$$

Setting to zero and solving for $\bar{\mu}$,

$$\bar{\mu}^* = (\beta I + Q_1 + \alpha Q_2)^{-1}(q_1 + \alpha q_2). \tag{5.35}$$

Plugging $\bar{\mu}^*$ into F and recalling that $(I + Q_1 + Q_2)$ is symmetric, we have

$$\begin{aligned} F^* &= \frac{1}{2}(q_1 + \alpha q_2)^T(\beta I + Q_1 + \alpha Q_2)^{-1}(\beta I + Q_1 + \alpha Q_2)(\beta I + Q_1 + \alpha Q_2)^{-1}(q_1 + \alpha q_2) \\ &\quad - (q_1 + \alpha q_2)^T(\beta I + Q_1 + \alpha Q_2)^{-1}(q_1 + \alpha Q_2) \end{aligned} \tag{5.36a}$$

$$\begin{aligned} &= \frac{1}{2}(q_1 + \alpha q_2)^T(\beta I + Q_1 + \alpha Q_2)^{-1}(q_1 + \alpha q_2) \\ &\quad - (q_1 + \alpha q_2)^T(\beta I + Q_1 + \alpha Q_2)^{-1}(q_1 + \alpha q_2) \end{aligned} \tag{5.36b}$$

$$F^* = -\frac{1}{2}(q_1 + \alpha q_2)^T(\beta I + Q_1 + \alpha Q_2)^{-1}(q_1 + \alpha q_2). \tag{5.36c}$$

Since $\beta I + Q_1 + \alpha Q_2$ is positive definite, $(\beta I + Q_1 + \alpha Q_2)^{-1}$ is also positive definite, and $F^* < 0$ for all $\alpha, \beta > 0$. The regularizing term in F is always positive in $\bar{\mu}$, so we can conclude that $\Delta V < 0$, which satisfies eq. (5.20c).

5.4.2 Causal Implementation

Examining eq. (5.31b), we see that the controller formulation in section 5.4 relies on knowledge of B_{k+1}. This is not causal. However, B_{k+1} can be predicted using knowledge of the satellite's orbit and a model of the geomagnetic field. Detumbling is often executed during early operations of a satellite, so orbit knowledge and a computationally expensive geomagnetic field model may not be available. An alternative is to approximate B_{k+1} as

$$B_{k+1} \approx B_k + \Delta t \dot{B}_k^{\mathcal{N}}. \tag{5.37}$$

We cannot directly measure $\dot{B}^{\mathcal{N}}$; however using eq. (5.4) it can be estimated from magnetometer and gyro measurements. To do so, a high-quality estimate of $\dot{B}^{\mathcal{B}}$ can be made using multiple measurements of B. For ease of implementation, we currently use a derivative of the magnetic field model computed using automatic differentiation to get $\dot{B}^{\mathcal{B}}$.

5.4.3 Complete Controller

Bringing together the development from the last two sections, the discrete non-monotonic controller is given by algorithm 1. The input B_{k+1} is computed using the approximation in eq. (5.37). On lines 1 and 2 we normalize the values of B_* to avoid numerical issues and ensure consistency of performance across the wide range of geomagnetic field magnitudes a spacecraft will experience. Lines 3 to 9 set up the problem components and follow from eq. (5.31). We solve for $\bar{\mu}$ on line 10, where $\backslash$ indicates numerically solving the linear system. Finally, on line 11 we perform a soft saturation of the computed control output and rescale it to satisfy the limits of the satellite's magnetorquers.

Algorithm 1: Discrete Non-monotonic controller

Data: Given B_k, B_{k+1}, J, ω, k, α, β
Result: μ

1 $b_1 = B_k / \|B_k\|$
2 $b_2 = B_{k+1} / \|B_{k+1}\|$
3 $\bar{b} = \begin{bmatrix} \hat{b_1} & \hat{b_2} \end{bmatrix}$
4 $Z = \beta \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix}$
5 $Q_1 = Z\bar{b}\bar{b}^T Z$
6 $Q_2 = \alpha \bar{b}\bar{b}^T$
7 $h = J\omega$
8 $q_1 = Z\bar{b}h$
9 $q_2 = \alpha \bar{b}h$
10 $\bar{\mu} = (I + Q_1 + Q_2) \backslash (q_1 + q_2)$
11 $\mu = \tanh(k\bar{\mu}[1:3]) * \mu_{\max}$

5.5 Simulation Experiments

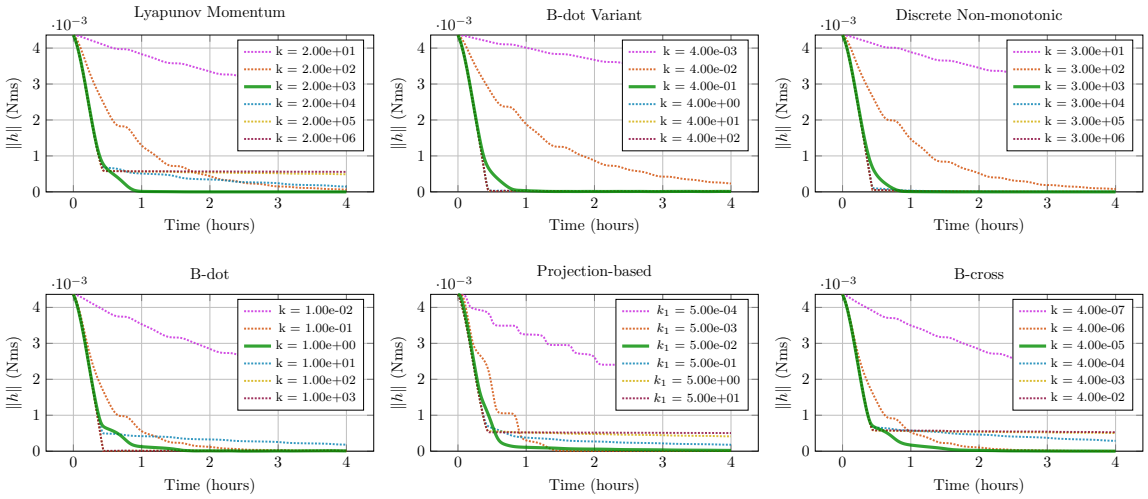


Figure 5.2: Gain sweep study showing the effect each controller's gain has on its detumbling performance for a single initial condition. The solid green line is the gain that was used for the Monte-Carlo simulation experiment shown in figs. 5.3 to 5.5. The other lines are for gains varying from two orders of magnitude lower to three orders of magnitude higher than the chosen gain.

All simulations are performed in a 12 degree-of-freedom orbital and attitude dynamics simulation environment that we developed and is available as an open-source repository[1]. The simulation environment relies on the open-source SatelliteDynamics.jl[2] orbital dynamics package and includes perturbations due to J2 and atmospheric drag. The attitude dynamics include orbit-coupled drag torques, and to accurately model the geomagnetic field, a differentiable implementation of the International Geomagnetic Reference Field (IGRF) [3]. The spacecraft properties used for the simulations are given in table 5.1, and reflect the properties for a 1.5U CubeSat with printed circuit board magnetorquers embedded in the solar panels. Noisy sensor measurements and a randomly initialized constant gyro bias are also included in the simulation; the noise parameters are representative of low-cost micro-electromechanical (MEMS) IMU and magnetometer hardware.

[1] github.com/RoboticExplorationLab/non-monotonic-detumbling
[2] sisl.github.io/SatelliteDynamics.jl

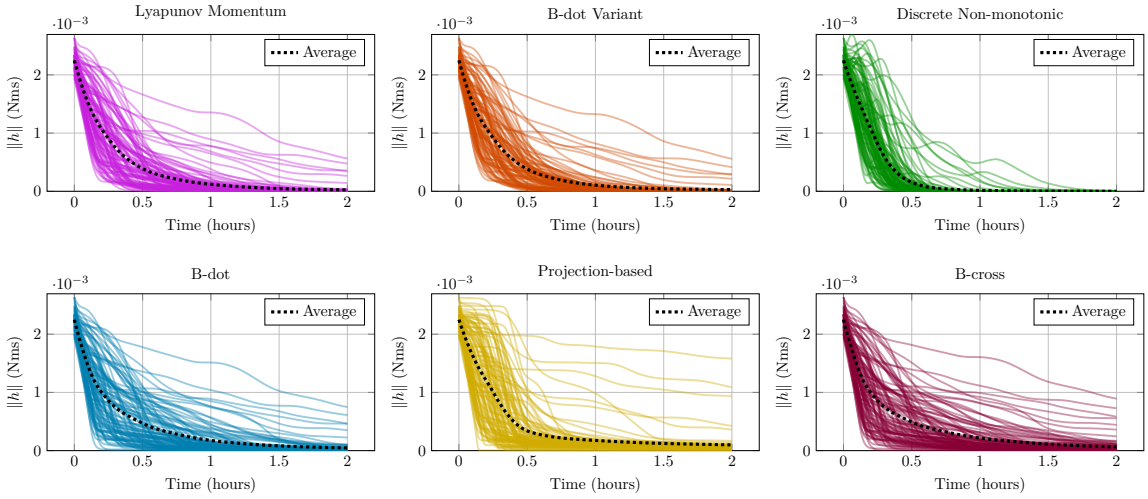


Figure 5.3: Momentum magnitude versus time plot for each of the controllers discussed in this chapter. The Discrete Non-monotonic controller differs from the other controllers in that the system momentum increases before decreasing and converging to zero. This is the key distinction of this controller and allows it to have faster convergence times than other detumbling controllers.

Each of the controllers is sensitive to its tuning parameter, k. To have meaningful comparison between controllers, each controller needs to be tuned to perform in the best possible manner. To do so, we simulate the performance impact of each controller's gain, sweeping it over several orders of magnitude. The results of this study are shown in fig. 5.2. The solid green line is the gain that was used for the Monte-Carlo simulation experiment shown in figs. 5.3 to 5.5; this gain was chosen as a tradeoff between fast convergence and avoiding high gains that lead to the controller getting stuck in their uncontrollable subspace.

The Monte-Carlo results are simulations of the satellite dynamics starting from 100 randomly sampled initial states for each of the six controllers discussed in this chapter. The random initial states are the same for each controller for fair comparison. The ranges and values of the Monte-Carlo initial conditions are sampled uniformly across the ranges given in table 5.2. They represent random circular orbits at a fixed altitude and random vehicle axis of rotation with a fixed initial angular velocity magnitude. Since most satellites in low Earth orbit operate at high inclinations, and to avoid the lack of controllability all magnetorquer control systems experience at near equatorial inclinations [7], we restricted the orbital inclinations to $[20°, 160°)$. The

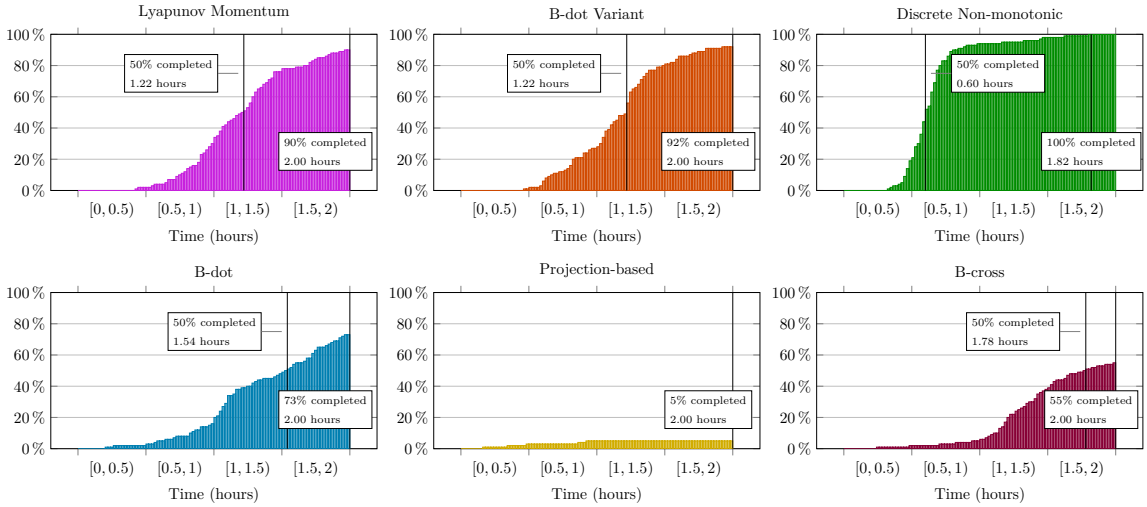


Figure 5.4: Cumulative distribution of detumble times for each of the controllers discussed in this chapter. Detumble times are defined as the time when the satellite first reaches 1% of its initial angular momentum. The simulation ended at two hours, so only detumble times less than two hours are counted. Our Discrete Non-monotonic controller has the lowest average detumble time. It is also the only controller to detumble for all simulation runs.

controller parameters and corresponding equation reference are shown in table 5.3. In developing this comparison we attempted to tune all controllers to achieve their best performance.

The simulation results are shown in figs. 5.3 to 5.5. Figure 5.4 shows a histogram of the time it takes for the spaceraft momentum to be reduced to 1% of its initial value. As can be seen, many of the common control methods do not converge to this threshold within two hours. However, our discrete non-monotonic controller converges within the two hour simulation period for all initial conditions tested and the majority of the initial conditions converge within one hour. The reason for this can be seen in fig. 5.3; it shows the time history of the momentum magnitude for the 100 Monte-Carlo simulation runs, as well as an average time history of these runs. The discrete non-monotonic controller exhibits significantly different behavior than the five other controllers, increasing in momentum one or more times before finally converging to zero. By relaxing the monotonic decrease requirement for the controller, we have developed a controller that reduces detumble times by up to 50%.

Figure 5.5 shows the final angular momentum magnitudes. The discrete non-

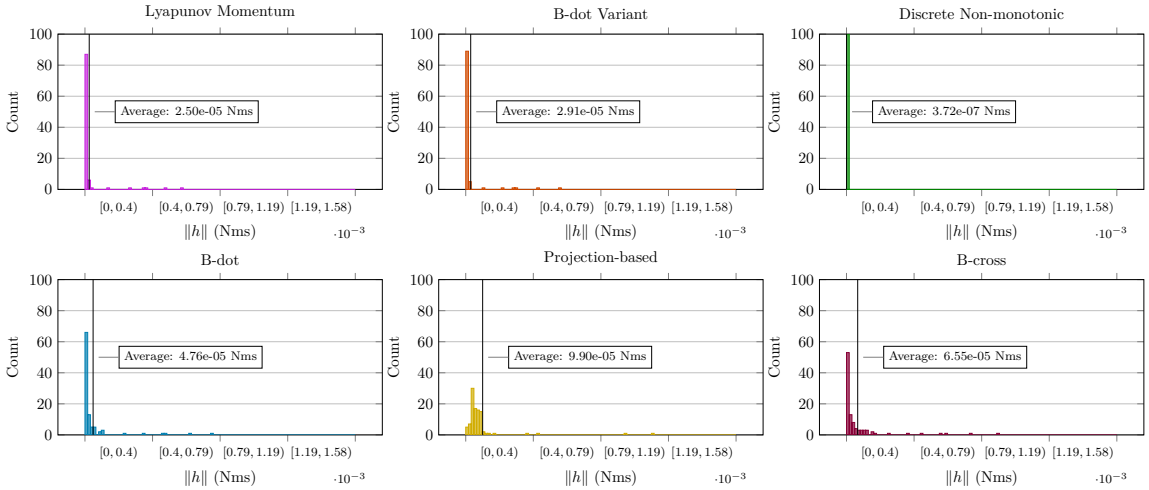


Figure 5.5: Histogram of final momentum magnitudes for each of the controllers discussed in this chapter. Our Discrete Non-monotonic controller is the only controller to have all of its simulation runs in the leftmost bin, with an average angular momentum two orders of magnitude lower than the other controllers.

monotonic controller has converged to a lower angular momentum than the other controllers in all cases. The projection-based controller is particularly susceptible to getting stuck in the uncontrollable subspace and has the worst performance of all of the controllers. The Lyapunov momentum, B-cross, B-dot variant, and B-dot controllers all perform similarly at the beginning of their simulation runs, but the B-cross controller converges significantly slower than the other controllers.

5.6 Conclusion

The many variants of B-dot and B-cross controllers in the literature differ primarily in how the gains and saturation are selected. Their performance is similar, with each having the potential to get stuck in the uncontrollable subspace where B and h are aligned. Even recent magnetorquer detumbling controllers, such as the projection-based controller [33] suffer from this failure point. The novel non-monotonic Lyapunov magnetorquer detumbling control law we have presented is a more significant departure from the classical B-dot and B-cross control laws: our control law implicitly predicts the future controllability of the system and avoids putting the satellite in

Table 5.1: Simulated spacecraft properties

Name	Value
Inertia J_{xx}	4.5×10^{-3} kg m^2
J_{xy}	-3.2 $\times 10^{-4}$ kg m^2
J_{xz}	0.0 kg m^2
J_{yy}	5.1×10^{-3} kg m^2
J_{yz}	0.0 kg m^2
J_{zz}	3.7×10^{-3} kg m^2
Mass	1.6 kg
Dimensions	75 cm $\times$ 10 cm $\times$ 15 cm
Drag Coefficient	2.2
$\mu_{\max}$	$\begin{bmatrix}0.070 & 0.053 & 0.070\end{bmatrix}$ A m^2
Magnetometer noise	15 nT
Gyro noise	$0.005^\circ/\text{s}/\sqrt{\text{Hz}}$
Gyro bias	1 $^\circ$ s^{-1}

Table 5.2: Monte-Carlo Initial Condition Distribution

Satellite State	Value
Altitude	400 km
Eccentricity	0
Inclination	$[20^\circ, 160^\circ)$
RAAN	$[0^\circ, 360^\circ)$
Arg. of Latitude	$[0^\circ, 360^\circ)$
$\|\omega\|$	30° s^{-1}

an uncontrollable state. In our Monte-Carlo simulation, it results in detumbling times that are more than twice as fast as the other controllers while operating with representative sensor noise and gyro bias. In addition, the tuning of our control law is straightforward and less sensitive to tuning errors than other control laws.

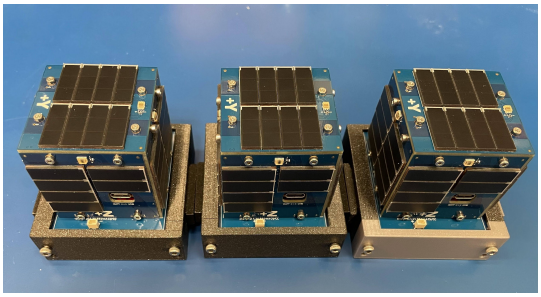
To put this control law into practical use a high-quality estimate of the geomagnetic field time derivative is needed. Future work will focus on generating this estimate and analyzing the full closed-loop performance of the magnetic field estimator and control law in combination.

Table 5.3: Controller Parameters

Controller	Parameters
Lyapunonv Momentum eq. (5.9)	$k = 2.0 \times 10^3$
B-cross eq. (5.10)	$k = 4.0 \times 10^{-6}$
Discrete Non-monotonic (algorithm 1)	$k = 3.0 \times 10^3$, $\alpha = 100$, $\beta = 1$, $\Delta t = 10$min
B-dot Variant eq. (5.14)	$k = 0.4$, $\epsilon = 1 \times 10^{-6}$
Projection-based eq. (5.15)	$k_1 = 10.0$, $k_2 = 10.0$, $\epsilon = 1 \times 10^{-6}$
B-dot eq. (5.13)	$k = 1.0$

Chapter 6

Satellite Flight Hardware and Software



(a) Three PyCubed-mini satellites.

(b) Four Py4 satellites.

Figure 6.1: PyCubed-mini and Py4 satellites.

While the previous two chapters focused on theoretical work, this chapter discusses practical hardware and software development for two different satellite platforms. The first is a $5 \times 5 \times 5$ cm 1p pocketqube, referred to as "PyCubed-mini" and shown in fig. 6.1a; the second is a set of four $10 \times 10 \times 15$ cm 1.5U cubesats, referred to as "Py4" and shown in fig. 6.1b. Both satellite platforms are derivatives of the open-source PyCubed flight computer [30]. PyCubed relies on the CircuitPython language that enables Python code to run on an embedded microcontroller. This enables rapid development as well as access to the many open-source CircuitPython libraries. The Py4 satellites also incorporate a Raspberry Pi Zero W as a secondary processor for

high-computation tasks. The following sections provide an overview of the hardware and describe the flight software for the PyCubed-mini and Py4 projects.

6.1 PyCubed-mini

The overall goal of the PyCubed-mini project is to develop a low-cost and full-functioning satellite in the 1p ($5 \times 5 \times 5$ cm and 250 g) pocketqube form factor [55]. The development began with a Stanford capstone project in 2019-2020 and has continued by a team of graduate and undergraduate students at Carnegie Mellon University. While the project is a derivative of the PyCubed project, it involved many hardware and software design modifications to create an easy to use system and to meet the volume requirements of the 1p form factor. In the end every hardware component and the full flight software system has been redesigned.

6.1.1 Flight Hardware

The PyCubed-mini platform includes the typical satellite subsystems: power, communications, computation, guidance and control, and a configurable payload. An extensive description of the PyCubed-mini hardware design can be found in Neil Khera's thesis [35]. A summary of each subsystem is given here to provide context.

Power

The power subsystem consists of solar cells, solar charging circuits, lithium-ion batteries, and power regulators. The solar cells are located on each face of the satellite, with a total of 42 cells overall. A distributed solar charging system is used, with each face having its own solar charging circuit so solar charging can happen at maximum efficiency regardless of the satellite's orientation with respect to the sun. The lithium-ion batteries provide 1700 mAh of charge capacity and a bus voltage of $3.6 - 4.2$ V.

Communications

The communications subsystem relies on the HopeRF RFM98W 433 MHz long range (LoRa) radio transceiver. This transceiver is capable of transmitting at up to 1 W of output power. The transceiver output is connected to a dipole tape-spring antenna tuned to 50 Ω at 433 Mhz. With the LoRa spread-spectrum protocol this compact and low power radio is sufficient to close the link between the satellite and a groundstation.

Computation

The computation subsystem consists of a single ATSAMD51 microcontroller with an ARM Cortex-M4 core. Relevant peripheral hardware it connects to include a watchdog timer, a real time clock, and an SD card that acts as the primary data storage device.

Guidance and Control

One of the primary objectives for the PyCubed-mini is to accomplish three-axis attitude control in a pocketqube. To do so in such a small volume requires novel control methods. We developed printed-circuit-board magnetic coils and embedded them in each of the six solar panel faces. A depiction of the coils embedded in a solar panel is shown in fig. 6.2. The coils in parallel faces of the satellite are tied together in pairs and are connected to full H-bridge motor drivers that provide pulse-width-modulated voltage control. When driven by a nonlinear optimal control algorithm these coils can provide full three-axis attitude control. Simulation results show they can complete 180 degree slews over 10-15 minute time horizons [24].

For attitude estimation, the PyCubed-mini includes a nine degree of freedom inertial measurement unit; it measures acceleration, angular velocity, and magnetic field in three axes. In addition, a single digital light sensor is placed on each solar panel face to provide coarse sun vector measurements. All of the sensors are connected to the primary microcontroller.

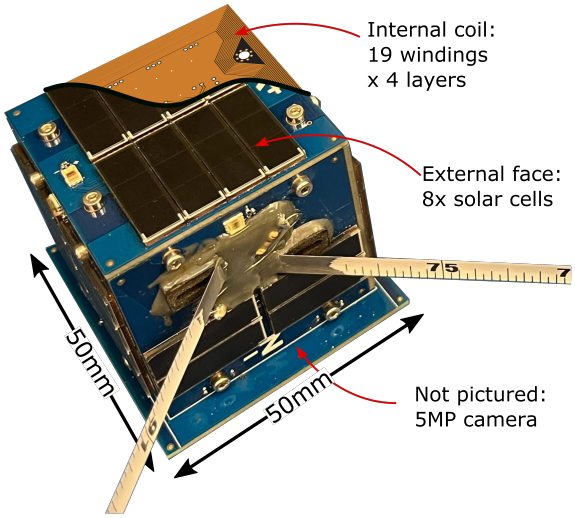


Figure 6.2: PyCubed-mini pocketcube with cutaway showing the coil traces embedded in a solar panel face.

Payload

As pointed out in fig. 6.2, PyCubed-mini also includes a camera payload. The camera is a 5 MP OV5640 and includes a secondary processor that is designed specifically for image processing and is supported by the OpenMV open-source machine vision software. This allows for real time processing of camera data to use it as an additional sensor for orbit estimation and orbit determination. The payload is a standalone PCB, allowing it to easily be swapped for a sensor that measures a different wavelength of light, such as infrared, or an entirely different payload such as a hardware radiation experiment.

6.1.2 Flight Software

All of the flight software for PyCubed-mini[1] is written in CircuitPython and runs on the ATSAMD51 primary microcontroller. The PyCubed-mini flight software is responsible for managing the full operational state of the satellite. These tasks include interfacing with each of the subsystem components, recording telemetry, performing communication operations including packetizing and depacketizing data, and performing real-time state estimation and control.

[1]Available at www.github.com/pycubed-mini/flight_software

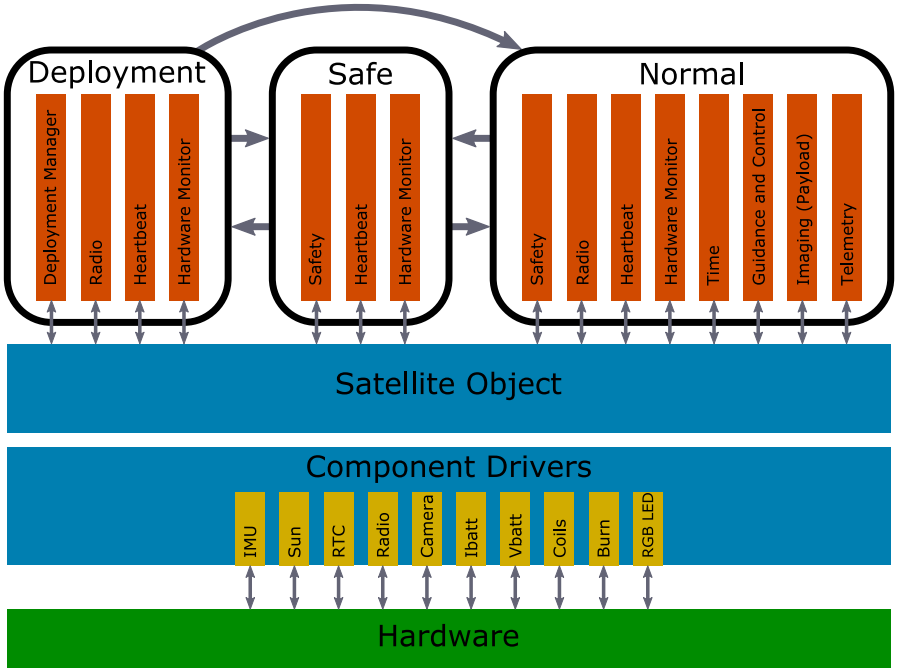


Figure 6.3: PyCubed-mini flight software architecture. The three boxes at the top are the software states: Deployment, Safe, and Normal. Transitions between the software states occur when power or thermal thresholds are crossed, and when communication with a groundstation occurs. The boxes within each state are the individual tasks that state executes. All tasks communicate through a common satellite object that provides an abstraction of the component drivers and hardware.

Figure 6.3 shows the architecture used for the PyCubed-mini flight software. At the highest level, the flight software consists of a three-state state machine. The three states reflect the three major modes it operates in: Deployment, Safe, and Normal. The boxes within each state are the individual tasks executed by that state. While many of the tasks have internal state, it was determined to keep the high-level state of the satellite simple. These tasks operate in a round-robin fashion, with each task yielding its execution regularly so other tasks can complete. There are a total of nine distinct tasks in the PyCubed-mini flight software: Deployment Manager, Radio, Heartbeat, Hardware Monitor, Safety, Time, Guidance and Control, Imaging (or Payload), and Telemetry. The states and the behavior of tasks that operate in each state are discussed subsequently.

The tasks all interface with a common abstraction layer that we refer to as the

"Satellite Object." The Satellite Object provides easy access to all of the system data a task would need without the task directly accessing the component device drivers. This also allows for an emulated satellite to be used for software-in-the-loop simulation and unit testing of the tasks.

6.1.3 Software States

Deployment

The deployment state represents the early operation of the satellite and is the first state that it enters. The primary objective in the deployment state is to run a timer and activate the antenna burn wires when the timer completes. The satellite remains in the deployment state after the burn wires are activated until a radio packet is received by the satellite. If a packet is not received after a long duration, it is assumed that the antennas failed to deploy and the satellite activates the burn wires again. When a packet is received the satellite transitions to the normal state.

Safe

The safe state is entered when the satellite either experiences a low battery or high temperature. In this state all non-essential components are shut down. In particular we shut down the highest power-draw components: the magnetic torque coils, the payload, and the radio. The satellite then waits for nominal conditions to return and transitions to the previous state it was in (deployment or normal).

Normal

The normal state encompasses the majority of satellite operations. All tasks except for the deployment manager operate in the normal state.

6.1.4 Software Tasks

Deployment Manager Task

The deployment manager is responsible for executing the logic of the deployment state described in section 6.1.3. It also replaces the safety task. During burn wire

activation the battery voltage can drop below the safe state transition threshold, so this transition is suspended until burn wire activation ends. Otherwise the battery and temperature monitoring are the same as in the safety task.

Safety Task

This task is responsible for regularly checking the battery state of charge and system temperature. If either are off-nominal the task initiates a transition to the safe state.

Radio Task

The radio task is one of the more complex tasks. It is responsible for sending data in the transmission queue, and it monitors the radio's status, waiting for it to indicate that an incoming message has been received. Messages can be one of three types: memory buffered, disk buffered, and command. Memory buffered messages are multi-packet messages that are relatively short, so they are buffered in RAM. Disk buffered messages are multi-packet messages that are buffered on the disk. Commands are messages that cause the satellite to execute some action. The commands the satellite can execute are given in table 6.1.

Table 6.1: PyCubed-mini software commands

Command	Argument	Action
NO_OP	N	No operation, used for testing
HARD_RESET	N	Perform a hard reset of the flight computer
RELOAD	N	Restart the flight software
QUERY	Y	Execute the argument with Python, result returned
EXEC_PY	Y	Execute the argument with Python, result not returned
REQUEST_FILE	Y	Place a file on the transmission queue
LIST_DIR	Y	Transmit the contents of a directory
TQ_SIZE	N	Transmit the length of the transmission queue
CLEAR_TX_QUEUE	N	Clear the contents of the transmission queue
MOVE_FILE	Y	Move a file from one location to another
COPY_FILE	Y	Copy a file in one location to another
DELETE_FILE	Y	Remove a file from the filesystem
REQUEST_BEACON	N	Transmit the most recent beacon data
GET_RTC	N	Transmit the RTC time as a tuple of YMDHMS
GET_RTC_UTIME	N	Transmit the RTC time as a Unix timestamp
SET_RTC	Y	Set the RTC with a tuple of YMDHMS
SET_RTC_UTIME	Y	Set the RTC with a Unix timestamp

The transmission queue is a priority queue that packetizes and transmits data when it is loaded onto the queue. The most recent mission requirements for PyCubed-mini stipulated that transmission could only occur over Germany. Because of this, the transmission queue only transmits after it receives a command and stops transmitting 5 minutes after the last received command.

Heartbeat Task

The heartbeat task is for convenient debugging of the satellite when it is on the bench. It blinks an LED at a customizable rate and color depending on the state the satellite is in. It is the lowest priority task, so if it is blinking regularly it is a good indication that all tasks are completing on time.

Hardware Monitor Task

The hardware monitor keeps track of hardware peripherals and logs an error when hardware becomes unavailable.

Time

The time task is simply responsible for logging the time since boot at a regular interval to give a baseline time if nothing else makes log entries. It also performs garbage collection each time it is executed.

Guidance and Control

The guidance and control task is not yet complete, so the purpose of the task is described here. It uses time information and a previous orbit location to approximately compute its orbital position. It then uses the magnetometer, gyro, and sun sensor data to compute an estimate of the vehicle attitude with respect to the Earth-centered inertial frame. This attitude estimate is then tracked by an attitude controller. In future software versions this task may also operate in a reduced manner to perform detumbling in the deployment state.

Imaging Task

This task controls the operation of the camera payload and sends image capture commands. The camera payload operates asynchronously from the main flight controller and has its own data storage. However it does not have a direct connection with the radio, so the imaging task is responsible for transferring image data from the camera payload to the main flight controller data storage.

Telemetry Task

A large number of measurements are made by the satellite as it is operating. The telemetry task is responsible for regularly capturing and saving these measurements to make them easy to recover. The following telemetry items are logged:

- System time
- State machine state
- System flags:
  - RTC time valid
  - Groundstation contact has occurred
  - Burn wires have been activated
- Software error count
- Boot count
- Battery voltage
- CPU temperature
- IMU temperature
- 3-axis gyro measurement
- 3-axis magnetometer measurement
- Most recent received signal strength indicator (RSSI) for the radio
- Most recent frequency error for the radio
- Sun sensor measurements from all six faces

6.2 Py4

The Py4 system consists of four 1.5U CubeSats (shown in fig. 6.1b) that use the PyCubed satellite avionics and that were constructed at NASA Ames. It is a follow-on to the VR3X project with the goal of demonstrating "spacecraft-to-spacecraft ranging, on-orbit relative navigation, and coordinated simultaneous multi-point radiation measurements" [32]. The four 1.5U CubeSats will all be launched simultaneously and will immediately begin performing range measurements between the spacecraft. In addition, once the primary ranging experiments are completed, the satellites will become research platforms allowing for individual satellite and formation estimation and control experiments.

Each Py4 satellite is equipped with a LoRa radio module that allows for networked communication between satellites and for time-of-flight range measurements between the satellites. These radio modules work at a range of up to approximately 20 km. The satellites are also equipped with global positioning system (GPS) units that allow for precise global localization of the satellites. By including both time-of-flight ranging and GPS, formation estimation experiments using the ranging measurements can be performed and compared to the GPS data to verify their performance. The GNC sensors and actuators are similar to those on the PyCubed-mini. For attitude estimation, each Py4 satellite includes sun sensors on each face, as well as an inertial measurement unit containing a gyro and magnetometer. For attitude control, each face of the Py4 satellite incorporates PCB magnetic torque coils.

All of the sensors and the coil drivers are connected to the PyCubed avionics microcontroller. However the Py4 hardware also includes a secondary Raspberry Pi Zero 2 W that allows for higher performance computation onboard the satellite. The Raspberry Pi is connected to the PyCubed microcontroller via UART; all of the sensor data and control commands are communicated through this interface. This allows for the development of the more computationally expensive guidance and control software on the Raspberry Pi.

6.2.1 Guidance and Control Software

Since hardware development is performed by NASA Ames, the primary contribution of CMU (and this thesis) is the guidance and control flight software. In keeping with the spirit of PyCubed, and for ease of development, the guidance and control software is developed in Python. We refer to the guidance and control software as PyGNC[2]. A block diagram outlining the various PyGNC software components is shown in fig. 6.4. We have already discussed the PyCubed microcontroller and other satellite hardware; in the subsequent sections we will discuss the remaining blocks.

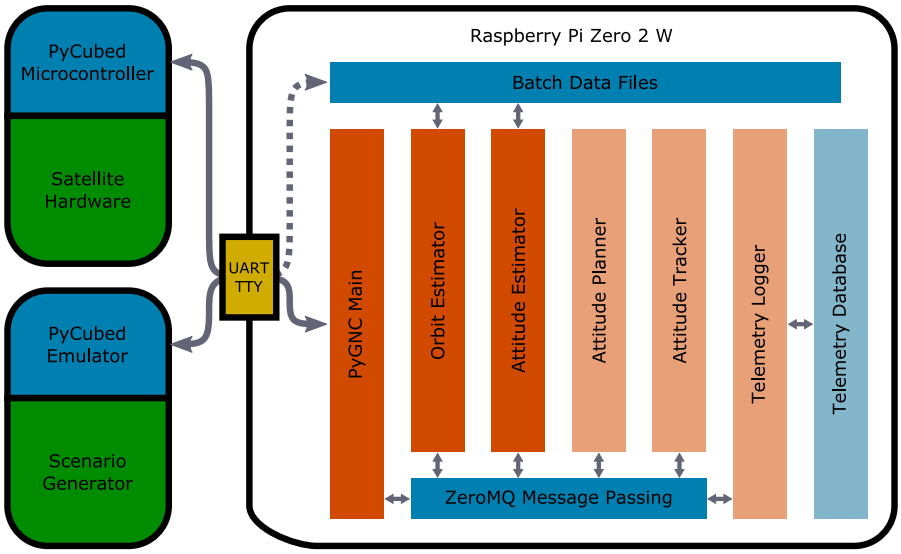


Figure 6.4: PyGNC software architecture. The Raspberry Pi runs the PyGNC algorithms that start by performing batch state estimation updates and then proceed to incremental state estimation and control updates. The PyGNC software includes the PyCubed emulator and scenario generator: a software-in-the-loop emulator that simulates orbit and attitude dynamics, computes realistic sensor measurements, and packetizes the data into the same format as the PyCubed flight software. This allows for straightforward software-in-the-loop simulation without modifying the flight software on the Raspberry Pi. Tasks with a lighter colored background (Attitude Planner, Attitude Tracker, Telemetry Logger and Telemetry Database) are yet to be implemented.

[2] www.github.com/pygnc/pygnc

PyCubed Emulator and Scenario Generator

All of the PyGNC software operates on the Raspberry Pi. Not only is this a straightforward decoupling of responsibilities, it also makes it easy to operate the Raspberry Pi in a realistic software-in-the-loop simulation with a personal computer acting in place of the PyCubed microcontroller. This functionality is captured by the PyCubed emulator and scenario generator blocks of fig. 6.4. The scenario generator simulates orbit and attitude dynamics, computes realistic sensor measurements, and packetizes the data into the same format as the PyCubed flight software. The emulator interfaces with the Raspberry Pi via a UART connection, transfers batch data files, and then starts the PyGNC main task. It also provides sequential update data when prompted by the PyGNC main task.

Batch Data Files and Sequential Updates

To reduce the power impact of running the Raspberry Pi all the time, the PyCubed microcontroller gathers a batch of sensor data and passes it to the Raspberry Pi before starting PyGNC. PyGNC then uses this data to perform batch orbital and attitude state estimates. Starting with a batch update allows PyGNC to converge on an up-to-date state estimate within a few seconds of the Raspberry Pi booting, rather than requiring minutes or more of data. After the batch update completes, PyGNC continues to operate with sequential measurement updates from the PyCubed microcontroller.

PyGNC Main and Message Passing

PyGNC Main is the entry point to the PyGNC software. It is a Python process that is responsible for starting and monitoring the other PyGNC tasks. It is also responsible for sending controller updates to the PyCubed microcontroller and for receiver sensor data updates. To allow for true parallelism of tasks, PyGNC Main uses the Python multiprocessing library to start each task as a standalone Python process. For this reason, a reentrant and thread-safe message passing system is needed for tasks to communicate. We use ZeroMQ[3] for message passing and MsgPack[4] for serializing the

[3] www.zeromq.org
[4] www.msgpack.org

messages.

State Estimation

Three state estimation problems need to be solved for Py4: orbit estimation, attitude estimation, and formation estimation.

The orbit estimator uses an extended Kalman filter to compute an estimate of the satellite's position and velocity in the Earth-centered inertial frame. Its measurement inputs are GPS position and velocity measurements. It estimates the position and velocity of the satellite as well as three acceleration perturbation parameters that capture drag and other unmodelled accelerations. During the batch estimation phase the GPS measurements occur at a 25 second time interval; once the sequential estimation phase begins, GPS measurements occur at a 1 second time interval.

The attitude estimator uses a multiplicative extended Kalman filter to compute an estimate of the satellite's orientation with respect to the Earth-centered inertial frame. Its measurement inputs are sun sensor, gyro, and magnetometer measurements. It also relies on the orbital estimate to determine the expected sun vector and Earth magnetic field vector. In addition to an attitude estimate, it also estimates several sensor bias parameters [34].

In our present design, the formation estimation problem is not solved onboard the satellite. This is due to two factors: the computational cost of solving a large batch least squares problem, and limitations on the ranging measurement topology. Each satellite only gets range measurements in a hub-and-spoke or star pattern, so it lacks ranges between the other pairs of satellites. In addition, each satellite pair has an unknown offset and scale factor in their range measurements [29]. By collecting data on the ground, these offsets and scale factors can be calibrated before the formation estimation problem is solved.

Attitude Planner and Tracker

The attitude planner and tracker will follow [24] to perform 3-axis magnetorquer-only attitude control. The attitude planner is based on an iterative linear quadratic regulator (iLQR) nonlinear trajectory optimization algorithm. It outputs the optimal attitude trajectory for the satellite to follow. These trajectories have an approximately

15 minute time horizon. The trajectory is then passed to the attitude tracker that uses feedback control to compute the magnetorquer control inputs needed to track the trajectory. Prototypes of these algorithms exist, but they are not yet part of PyGNC.

Telemetry Logger and Database

All of the tasks produce data that needs to be logged over time. This includes the raw sensor measurements coming from PyGNC main, as well as the state estimates and control commands computed by the other tasks. These are all available on separate ZeroMQ message channels. The telemetry logger subscribes to all of the messages and saves each one in the telemetry database.

6.2.2 Detumbling

In addition to the previously discussed PyGNC software, the Py4 satellites also include a detumbling controller that operates natively on the PyCubed microcontroller. This allows the detumbling controller to operate at a faster update rate of 10-20 Hz. The detumbling controller is computationally inexpensive in comparison to the PyGNC algorithms, so there is not a concern with it operating on the resource-constrained PyCubed microcontroller.

Chapter 7

Conclusions

In all of the research results published here, we exceeded the current state-of-the-art. The Kustaanheimo-Stiefel linear relative-orbital-dynamics model developed in chapter 3 achieves higher accuracy than other models in the literature while also representing the relative orbital state in a singularity-free space. It enables convex trajectory optimization of low-thrust relative orbital maneuvers. The drag-based formation control scheme in chapter 4 relies on a novel first-principles relationship between along-track and cross-track position to find out-of-plane orbital maneuvers with drag modulation as the only control input. The non-monotonic Lyapunov detumbling controller derived in chapter 5 results in a 50% reduction in detumbling time and drives the momentum of the system to a lower overall angular momentum than existing detumbling controllers. The flight hardware and software development discussed in chapter 6 will enable these and other advanced spacecraft control algorithms to be validated on orbit.

At the conclusion of any research project, there are more questions than answers. This research is the same. Each chapter includes a discussion of the future work related to that project. For any of them to operate successfully on orbit there are many more details to be worked out than can be enumerated here. Rather than discussing those details, the following are some high-level ideas that could drive future research directions:

- All of the algorithms discussed rely on large amounts of computation. Efficient implementation many not be enough to get the algorithms running reliably on

hardware. When computational bottlenecks are reached, spacecraft avionics designers turn to FPGAs. One future research direction would be to implement core components of these algorithms, such as a linear system solver or matrix factorizaton on FPGAs.

- Utilization of low-cost imaging sensors. CMOS imaging sensors like those found in cell phone cameras are inexpensive and ubiquitous. Many of the challenging aspects of satellite attitude determination could be solved by replacing low-information sensors such as sun-sensors and Earth-horizon sensors with high-information CMOS imaging sensors. The challenge then becomes developing image processing algorithms that provide useful information on a computation-constrained satellite. The PyCubed-mini satellite would be an ideal platform for developing these capabilities.
- Distributed state estimation and control. For higher performance satellites radiation tolerance is of utmost importance. The proliferation of low-cost small satellites has pushed back on the need for radiation tolerant systems at a constellation level. What if we took that approach at a satellite level, placing many low-cost attitude determination and control modules on a single satellite and letting them work together? Fault tolerance would then be a result of extreme redundancy, allowing a much more gradual performance degradation than the loss of a single centralized system.

Bibliography

[1] Amir Ali Ahmadi and Pablo A Parrilo. Non-monotonic Lyapunov functions for stability of discrete time nonlinear and switched systems. In *2008 47th IEEE conference on decision and control*, pages 614–621. IEEE, 2008. 5.1, 5.4

[2] Kyle T Alfriend, Srinivas R Vadali, Pini Gurfil, Jonathan P How, and Louis Breger. *Spacecraft formation flying: Dynamics, control and navigation*, volume 2. Elsevier, 2009. 3.7.1

[3] Patrick Alken, Erwan Thébault, Ciarán D Beggan, Hagay Amit, J Aubert, J Baerenzung, TN Bondar, WJ Brown, S Califf, A Chambodut, et al. International geomagnetic reference field: the thirteenth generation. *Earth, Planets and Space*, 73(1):1–25, 2021. 5.5

[4] ApS MOSEK. The MOSEK optimization software. `https://github.com/MOSEK/Mosek.jl`, 2021. 4.5

[5] Giulio Avanzini and Fabrizio Giulietti. Magnetic detumbling of a rigid spacecraft. *Journal of guidance, control, and dynamics*, 35(4):1326–1334, 2012. 5.2, 5.3.2

[6] Riccardo Bevilacqua and Marcello Romano. Rendezvous maneuvers of multiple spacecraft using differential drag under J2 perturbation. *Journal of Guidance, Control, and Dynamics*, 31(6):1595–1607, November 2008. ISSN 0731-5090, 1533-3884. doi: 10.2514/1.36362. URL `https://arc.aiaa.org/doi/10.2514/1.36362`. 4.2

[7] Sanjay P. Bhat. Controllability of nonlinear time-varying systems: applications to spacecraft attitude control using magnetic actuation. *IEEE Transactions on Automatic Control*, 50(11):1725–1735, 2005. 5.5

[8] Andrew Blatner. *Optimal Differential Drag Control of Small Satellite Constellations*. PhD thesis, University of California at Berkeley, August 2018. URL `https://www2.eecs.berkeley.edu/Pubs/TechRpts/2018/EECS-2018-121.pdf`. 4.2

[9] Joseph A. Burns. Elementary derivation of the perturbation equations of celestial mechanics. *American Journal of Physics*, 44(10):944–949, October 1976. ISSN 0002-9505, 1943-2909. doi: 10.1119/1.10237. URL `http://aapt.scitation.org/doi/10.1119/1.10237`. 4.3.2

[10] Charles D. Bussy-Virat, Aaron J. Ridley, Abhay Masher, Kyle Nave, and Marissa Intelisano. Assessment of the differential drag maneuver operations on the CYGNSS constellation. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(1):7–15, January 2019. ISSN 2151-1535. doi: 10.1109/JSTARS.2018.2878158. 4.2

[11] Vladimir A. Chobotov. *Orbital Mechanics*, chapter 8. Introduction to Orbit Perturbations, page 185–213. American Institute of Aeronautics and Astronautics, Inc., 2002. 4.3.2, 4.3.3

[12] WH Clohessy and RS Wiltshire. Terminal guidance system for satellite rendezvous. *Journal of the Aerospace Sciences*, 27(9):653–658, 1960. 3.2

[13] Howard D. Curtis. *Orbital Mechanics for Engineering Students*, page 652–715. Elsevier Butterworth Heinemann, third edition, 2005. 4.3.1, 4.3.3

[14] F deBruijn, E Gill, and J How. Comparative analysis of Cartesian and curvilinear Clohessy-Wiltshire equations. *Journal of Aerospace Engineering*, 3(2):1, 2011. 3.2

[15] Mohammed AA Desouky and Ossama Abdelkhalik. A new variant of the B-dot control for spacecraft magnetic detumbling. *Acta Astronautica*, 171:14–22, 2020. 5.2, 5.3.2

[16] Mohammed AA Desouky and Ossama Abdelkhalik. Time-optimal magnetic attitude detumbling. *Journal of Spacecraft and Rockets*, 57(3):549–564, 2020. 5.2, 5.3.2

[17] Alexander Domahidi, Eric Chu, and Stephen Boyd. ECOS: An SOCP solver for embedded systems. In *2013 European control conference (ECC)*, pages 3071–3076. IEEE, 2013. 4.5

[18] HA Nour Eldin. Trajectory control in rendezvous problems using the regularization techniques. *IFAC Proceedings Volumes*, 3(1):101–115, 1970. 3.2

[19] Giusy Falcone and Zachary R Putnam. Energy depletion guidance for aerobraking atmospheric passes. *Journal of Guidance, Control, and Dynamics*, 45(4):651–668, 2022. doi: 10.2514/1.G006171. 4.3.2

[20] Giusy Falcone, Jacob B Willis, and Zachary Manchester. Propulsion-free cross-track control of a leo small-satellite constellation with differential drag. *arXiv preprint arXiv:2306.13844*, 2023. 1

[21] Cyrus Foster, James Mason, Vivek Vittaldev, Lawrence Leung, Vincent Beukelaers, Leon Stepan, and Rob Zimmerman. Constellation phasing with differential drag on Planet Labs satellites. *Journal of Spacecraft and Rockets*, 55(2):473–483, March 2018. ISSN 0022-4650, 1533-6794. doi: 10.2514/1.A33927. URL https://arc.aiaa.org/doi/10.2514/1.A33927. 3.1, 4.1, 4.2, 4.3.2

[22] Bo Fu, Evan Sperber, and Fidelis Eke. Solar sail technology—A state of the art review. *Progress in Aerospace Sciences*, 86:1–19, 2016. 3.1

[23] Joseph Gangestad, Brian Hardy, and David Hinkley. Operations, orbit determination, and formation control of the AeroCube-4 CubeSats. *Small Satellite Conference*, 2013. 4.2

[24] Andrew Gatherer and Zac Manchester. Magnetorquer-only attitude control of small satellites using trajectory optimization. In *Proceedings of AAS/AIAA Astrodynamics Specialist Conference*, 2019. 4.1, 5.2, 6.1.1, 6.2.1

[25] Dong-Woo Gim and Kyle T Alfriend. State transition matrix of relative motion for the perturbed noncircular reference orbit. *Journal of Guidance, Control, and Dynamics*, 26(6):956–971, 2003. 3.2

[26] Andrew T Harris, Christopher D Petersen, and Hanspeter Schaub. Linear coupled attitude–orbit control through aerodynamic drag. *Journal of Guidance, Control, and Dynamics*, 43(1):122–131, 2020. 4.2

[27] Matthew W. Harris and Behçet Açıkmeşe. Minimum time rendezvous of multiple spacecraft using differential drag. *Journal of Guidance, Control, and Dynamics*, 37(2):365–373, March 2014. ISSN 0731-5090. doi: 10.2514/1.61505. URL https://arc.aiaa.org/doi/10.2514/1.61505. 4.2

[28] Sonia Hernandez and Maruthi R Akella. Lyapunov-based guidance for orbit transfers and rendezvous in Levi-Civita coordinates. *Journal of Guidance, Control, and Dynamics*, 37(4):1170–1181, 2014. 3.2

[29] Max Holliday, Kevin Tracy, Zachary Manchester, and Anh Nguyen. The V-R3X mission: Towards autonomous networking and navigation for cubesat swarms. In *The 4S Symposium 2022*, 2022. 6.2.1

[30] Maximillian Holliday, Andrea Ramirez, Connor Settle, Tane Tatum, Debbie Senesky, and Zachary Manchester. PyCubed: An open-source, radiation-tested cubesat platform programmable entirely in Python. *33rd Annual AIAA/USU Conference on Small Satellites*, 2019. 1, 6

[31] Matthew Hunter and Simone D'Amico. Closed-form optimal solutions for propulsive-differential drag control of spacecraft swarms. In *AAS/AIAA Astrodynamics Specialist Conference*, 2022. 4.2

[32] Roger C. Hunter. NASA Small Spacecraft Technology Program Town Hall, 2023. URL https://www.nasa.gov/wp-content/uploads/2023/09/12.hunter-2023-nasa-town-hall-august-7-final.pdf. 6.2

[33] Davide Invernizzi and Marco Lovera. A projection-based controller for fast spacecraft detumbling using magnetic actuation. *Automatica*, 113:108779, 2020. 5.2, 5.3.2, 5.6

[34] Benjamin Jensen. A low-cost attitude determination and control system and hardware-in-the-loop testbed for cubesats. Master's thesis, Carnegie Mellon University, Pittsburgh, PA, August 2022. 6.2.1

[35] Neil Khera. PyCubed-mini: A low-cost, open-source satellite research platform. Master's thesis, Carnegie Mellon University, Pittsburgh, PA, August 2023. 6.1.1

[36] Adam W Koenig, Tommaso Guffanti, and Simone D'Amico. New state transition matrices for spacecraft relative motion in perturbed orbits. *Journal of Guidance, Control, and Dynamics*, 40(7):1749–1768, 2017. 3.2, 3.6

[37] Balaji Shankar Kumar, Alfred Ng, Keisuke Yoshihara, and Anton De Ruiter. Differential drag as a means of spacecraft formation control. *IEEE Transactions on Aerospace and Electronic Systems*, 47(2):1125–1135, April 2011. ISSN 1557-9603. doi: 10.1109/TAES.2011.5751247. 4.2

[38] P. Kustaanheimo, A. Schinzel, H. Davenport, and E. Stiefel. Perturbation theory of kepler motion based on spinor regularization. *Journal für die reine und angewandte Mathematik*, 1965(218):204–219, 1965. doi: doi:10.1515/crll.1965.218.204. URL https://doi.org/10.1515/crll.1965.218.204. 3.2

[39] Juyoung Lee and Hyochoong Bang. Ground track control using differential drag for small earth observation satellite constellations. *Journal of Spacecraft and Rockets*, 59(5):1552–1564, 2022. 4.2

[40] Kristina Lemmer. Propulsion for cubesats. *Acta Astronautica*, 134:231–243, 2017. 3.1, 3.7.1

[41] C. L. Leonard, W. M. Hollister, and E. V. Bergmann. Orbital formation keeping with differential drag. *Journal of Guidance, Control, and Dynamics*, 12(1): 108–113, January 1989. ISSN 0731-5090, 1533-3884. doi: 10.2514/3.20374. URL https://arc.aiaa.org/doi/10.2514/3.20374. 4.2

[42] Hannu Leppinen. Deploying a single-launch nanosatellite constellation to several orbital planes using drag maneuvers. *Acta Astronautica*, 121:23–28, April 2016. ISSN 00945765. doi: 10.1016/j.actaastro.2015.12.036. 4.2

[43] T Levi-Civita. Sur la résolution qualitative du problème restreint. *Acta Mathematica*, 30:305, 1906. 3.2

[44] Xinfu Liu and Ping Lu. Robust trajectory optimization for highly constrained rendezvous and proximity operations. In *AIAA Guidance, Navigation, and Control (GNC) Conference*, page 4720, 2013. 3.2

[45] Xinfu Liu, Ping Lu, and Binfeng Pan. Survey of convex optimization for aerospace applications. *Astrodynamics*, 1:23–40, 2017. 4.2

[46] TD Maclay and Christopher Tuttle. Satellite stationkeeping of the ORBCOMM constellation via active control of atmospheric drag: operations, constraints, and

performance (AAS 05-152). *Advances in the Astronautical Sciences*, 120(1):763, 2005. 4.2

[47] Andrew Makhorin. GLPK (GNU linear programming kit). https://github.com/jump-dev/GLPK.jl, 2023. 4.5

[48] Danylo Malyuta, Yue Yu, Purnanand Elango, and Behçet Açıkmeşe. Advances in trajectory optimization for space vehicle control. *Annual Reviews in Control*, 52:282–315, 2021. 4.2

[49] F. Markley. Attitude control algorithms for the solar maximum mission. In *Guidance and control conference*, page 1247, 1978. 5.1, 5.2, 5.3.2

[50] F. Landis Markley and John L. Crassidis. *Fundamentals of spacecraft attitude determination and control*, volume 1286. Springer, 2014. 2.1.2, 4.1, 4.1, 5.1, 5.2, 5.3.2

[51] Michael Mathews and Suzan Leszkiewicz. Efficient spacecraft formationkeeping with consideration of ballistic coefficient control. In *26th Aerospace Sciences Meeting*. American Institute of Aeronautics and Astronautics, 1988. doi: 10.2514/6.1988-375. URL https://arc.aiaa.org/doi/abs/10.2514/6.1988-375. 4.2

[52] David Q. Mayne, James B. Rawlings, Christopher V. Rao, and Pierre O.M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000. doi: 10.1016/S0005-1098(99)00214-9. 4.5.2

[53] Oliver Montenbruck, Eberhard Gill, and Fh Lutze. Satellite orbits: models, methods, and applications. *Appl. Mech. Rev.*, 55(2):B27–B28, 2002. 2.1.1, 2.1.2

[54] Sanny Omar. Drag-based formation control of Millennium Space Systems satellites. *2023 Small Satellite Conference*, 2023. 4.2

[55] Alba Orbital, Delft University of Technology, and GAUSS Srl. The PocketQube Standard, 2018. URL https://www.albaorbital.com/pocketqube-standard. 6.1

[56] M. Yu Ovchinnikov and D.S. Roldugin. A survey on active magnetic attitude control algorithms for small satellites. *Progress in Aerospace Sciences*, 109: 100546, 2019. 5.2

[57] John E. Prussing and Bruce A. Conway. *Orbital Mechanics*, chapter 9. Perturbation, page 155–168. Oxford University Press, 2013. 4.3.2

[58] Christopher Rackauckas and Qing Nie. DifferentialEquations.jl — A performant and feature-rich ecosystem for solving differential equations in Julia. *Journal of Open Research Software*, 5(1):15, 2017. 3.6

[59] Enrico Silani and Marco Lovera. Magnetic spacecraft attitude control: a survey and some new results. *Control engineering practice*, 13(3):357–371, 2005. 5.2

[60] Emmanuel Sin, Murat Arcak, and Andrew Packard. Small satellite constellation

separation using linear programming based differential drag commands. In *2018 Annual American Control Conference (ACC)*, pages 4951–4956, June 2018. doi: 10.23919/ACC.2018.8431408. 4.2

[61] Dario Spiller, Fabio Curti, and Christian Circi. Minimum-time reconfiguration maneuvers of satellite formations using perturbation forces. *Journal of Guidance, Control, and Dynamics*, 40(5):1130–1143, May 2017. ISSN 0731-5090, 1533-3884. doi: 10.2514/1.G002382. URL https://arc.aiaa.org/doi/10.2514/1.G002382. 4.2

[62] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. OSQP: An operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 12(4):637–672, 2020. doi: 10.1007/s12532-020-00179-2. URL https://doi.org/10.1007/s12532-020-00179-2. 3.7.1

[63] A. Craig Stickler and K.T. Alfriend. Elementary magnetic attitude control system. *Journal of spacecraft and rockets*, 13(5):282–287, 1976. 5.2, 5.3.2

[64] Eduard L Stiefel and Gerhard Scheifele. *Linear and regular celestial mechanics: Perturbed two-body motion, numerical methods, canonical theory*, volume 174. Springer, 1971. 3.3.2, 3.4

[65] Joshua Sullivan, Sebastian Grimberg, and Simone D'Amico. Comprehensive survey and assessment of spacecraft relative motion dynamics models. *Journal of Guidance, Control, and Dynamics*, 40(8):1837–1859, 2017. 3.2, 3.6

[66] Michael Swartwout. The internet ruins everything: The sixth age of small satellites. In *2023 IEEE Aerospace Conference*, pages 1–8. IEEE, 2023. 1

[67] James D Thorne and Christopher D Hall. Minimum-time continuous-thrust orbit transfers using the Kustaanheimo-Stiefel transformation. *Journal of guidance, control, and dynamics*, 20(4):836–838, 1997. 3.2

[68] Michael Tillerson, Gokhan Inalhan, and Jonathan P How. Co-ordination and control of distributed spacecraft systems using convex optimization techniques. *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal*, 12(2-3):207–242, 2002. 4.2

[69] Kevin Tracy and Zachary Manchester. Low-thrust trajectory optimization using the Kustaanheimo-Stiefel transformation. In *AAS/AIAA Astrodynamics Specialist Conference*, pages 1–12, 2021. 3.1, 3.2, 3.4

[70] Ch Tsitouras. Runge–kutta pairs of order 5 (4) satisfying only the first column simplifying assumption. *Computers & Mathematics with Applications*, 62(2): 770–775, 2011. 3.6

[71] Madeleine Udell, Karanveer Mohan, David Zeng, Jenny Hong, Steven Diamond, and Stephen Boyd. Convex optimization in Julia. *SC14 Workshop on High*

Performance Technical Computing in Dynamic Languages, 2014. 4.5

[72] David A Vallado. *Fundamentals of astrodynamics and applications*, volume 12. Springer Science & Business Media, 2001. 2.1.1

[73] Surjit Varma and Krishna Dev Kumar. Multiple satellite formation flying using differential aerodynamic drag. *Journal of Spacecraft and Rockets*, August 2012. doi: 10.2514/1.52395. URL https://arc.aiaa.org/doi/abs/10.2514/1.52395. 4.2

[74] James R. Wertz. *Space mission analysis and design*, chapter 6. Introduction to Astrodynamics, page 131–159. Microcosm Press, 2005. 4.3.2

[75] Jacob B Willis and Zachary Manchester. Convex optimization of relative orbit maneuvers using the Kustaanheimo-Stiefel transformation. In *2023 IEEE Aerospace Conference*, pages 1–7. IEEE, 2023. 1

[76] Jacob B. Willis, Paulo R.M. Fisch, Aleksei Seletskiy, and Zachary Manchester. Building a better B-dot: Fast detumbling with non-monotonic Lyapunov functions. *2024 IEEE Aerospace Conference (accepted)*, 2024. 1

[77] Rafal Wisniewski and F. Landis Markley. Optimal magnetic attitude control. *IFAC Proceedings Volumes*, 32(2):7991–7996, 1999. 5.2

[78] Koji Yamanaka and Finn Ankersen. New state transition matrix for relative motion on an arbitrary elliptical orbit. *Journal of guidance, control, and dynamics*, 25(1):60–66, 2002. 3.2

[79] L.A. Zadeh and B.H. Whalen. On optimal control and linear programming. *IRE Transactions on Automatic Control*, 7(4):45 – 46, 1962. 4.4.3