Autonomous Management and Control of Multi-Spacecraft Operations Leveraging Atmospheric Forces

by

A. T. Harris

B.S., State University of New York at Buffalo, 2016

M.S., University of Colorado Boulder, 2018

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Ann and H.J. Smead Department of Aerospace Engineering Sciences

2021

Committee Members:

Prof. Hanspeter Schaub

Prof. Nisar Ahmed

Prof. Morteza Lahijanian

Prof. Marcin Pilinski

Dr. Islam Hussein

Harris, A. T. (Ph.D., Astrodynamics and Satellite Navigation)

Autonomous Management and Control of Multi-Spacecraft Operations Leveraging Atmospheric Forces

Thesis directed by Prof. Hanspeter Schaub

Future LEO missions consisting of tens to thousands of satellites will be well-positioned to benefit from technologies to automate mission operations and utilize atmospheric interactions to perform station-keeping. This dissertation aims to demonstrate novel technical methods for both differential drag and spacecraft autonomy that minimizes overall system impacts while remaining straightforward to implement, use, and test.

Two major areas of work are presented. In the first, a novel strategy for differential drag control using linearized attitude-orbit dynamics allows for the design of differential drag controllers that utilize small attitude variations. This control strategy is further examined and extended to deal with uncertainties in atmospheric density using a desensitized optimal control approach. The next area of work considers the applicability of deep reinforcement learning (DRL) to address challenges in spacecraft operations in a safe, performant, and scale-able manner. Frameworks and strategies for considering day-to-day spacecraft operations tasks as Markov Decision Processes are presented and discussed, culminating in the creation of several benchmark problems for spacecraft operations. The performance of DRL algorithms on each of these benchmark problems is presented and analyzed, demonstrating performance improvements against heuristic and black-box optimization approaches inspired by other strategies found in the literature.

The synergy of both approaches is further demonstrated on a representative challenge consisting of a spacecraft conducting science operations in Low Earth Orbit (LEO) while phasing using differential drag. DRL agents are shown to successfully learn to sequence mission activities, phasing campaigns, and health management tasks using novel strategies learned by experience alone.

Dedication

To my mother, who always wanted a doctor in the family.

Acknowledgements

Every PhD is the result of not only a single person's efforts, but the broader belief and striving of an entire civilization. This one in particular is the product of many things, so I'll try to keep it brief.

I would like to thank my parents, John and Lois Harris, whose enthusiasm, support, and belief in my abilities – despite having long been unable to understand the papers I send them – kept me going in rough times.

I would like to thank my significant other, Brandi McPherson, and her dog Chappie for their extensive support throughout the production of this dissertation.

There are many people in the CU community who were tremendously helpful in the production of this dissertation, too many to list. I would like to specifically thank each of the members of my PhD committee for volunteering their time and knowledge to aid in the writing of this dissertation. A special thanks goes to Dr. Hanspeter Schaub for adivising me, and for putting up with rather delayed sumission timelines.

Finally, I would like to thank the various organizations that have funded this work and my education – specifically AFRL and the AFOSR, which funded three years of this dissertation via the NDSEG program, and Anne Smead and Michael Byram, who practically adoped me for five years under the Smead Scholar program.

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

Technological trends including miniaturization and the consistent decline of launch costs has dramatically reshaped the space landscape. As spacecraft and launch costs shrink, new mission architectures which rely on coordination between multiple spacecraft have seen renewed interest from the Earth observation, telecommunications, and scientific communities. At the same time, there is growing interest in the application of artificial intelligence and machine learning techniques to the space domain by both government [2, 3] and commercial actors. The rise of multi-spacecraft mission architectures amplifies traditional objectives in both astrodynamics (such as decreasing mission fuel usage, increasing the robustness of trajectories, controlling nonlinear systems) and spacecraft operations (such as reducing operator workload and generating plans under constraints and uncertainty). Motivated by these two major challenges, **this dissertation aims to explore novel technologies to enable future missions to coordinate and cooperate in an intelligent, autonomous manner while minimizing fuel consumption.**

A specific motivator for this work is renewed interest in large, Low Earth Orbit (LEO) constellations (depicted in Figure 1.1) or formations that can leverage differential atmospheric drag to conduct in-plane maneuvers and phasing, such as the Planet Labs Flock [4]. Experiences in other atmospheric-assisted control techniques, such as aerobraking, suggest that operational uncertainty associated with using unpredictable atmospheric density for control can increase mission complexity and offset cost savings [5]. These studies have previously motivated the development of software suites to automate aerobraking campaign management [6]. In the same vein, this dissertation aims to provide contributions to mitigate operational uncertainty and complexity stemming from the operation of multiple spacecraft, including challenges arising from the use of differential drag. As such this dissertation aims to provide contributions to both the differential drag literature and the spacecraft operations literature relevant to future large, Earth-oriented constellations.

As a result, this dissertation consists of two central research thrusts. Thrust 1 aims to improve upon existing algorithms for differential drag formation flight by identifying novel architectures that minimize the systems and operational impacts of using differential drag. Thrust 2 aims to identify strategies and frameworks for the adaptation of modern machine learning approaches, specifically Deep Reinforcement Learning (DRL), to spacecraft operations problems with the aim of providing intelligent, adaptable solutions to day-to-day operations problems. Finally, both thrusts are combined to demonstrate the use of deep reinforcement learning to manage a representative drag-driven station keeping campaign while meeting other mission objectives.

1.1 Related Work

1.1.1 Differential Drag

Atmospheric forces on spacecraft have long been recognized as an avenue for coupling between attitude and orbital dynamics [7]. Owing to its dependence on atmospheric density, these forces and torques are small relative to gravity and are typically considered as perturbations in the context of orbital motion. However, at low-Earth Orbit (LEO) altitudes, forces from atmospheric interactions can have substantial impacts on spacecraft orbits [8]. For spacecraft that lack the volume or mass to mount thrusters (such as cubesats) or those whose thrusters are disabled but which maintain attitude control through other means, the coupling between attitude and orbital motion through drag presents one method of recovering mission utility. Additionally, there is rising interest in large LEO constellations for telecommunication and Earth-imaging. In this context, drag-enabled attitude-orbit coupling could provide a propellant-free method for formation constitution and maintenance, thereby extending mission lifetimes and reducing constellation costs. This

Figure 1.1: A rendering of a large, LEO constellation.

work aims to extend attitude-driven formation flight techniques to convex spacecraft geometries in a linear sense by exploiting attitude-orbit coupling under atmospheric drag.

In concept, the work presented here is related to the body of literature which focuses on ballistic-coefficient controlled differential-drag formation flight. These techniques focus on the control of one or more spacecraft's ballistic coefficient by means of actuated flaps [9] or panels, and treat either the ballistic coefficient or the spacecraft flow-wise projected area as the primary control input [10]. This class of differential drag-based control was flown by the AeroCube-4 technology demonstration mission [11]. The addition of actuated flaps and panels, while attractive for control purposes, unfortunately adds additional cost and system complexity that is undesirable for mission managers. Many spacecraft, including cubesats, have non-uniform geometries whose projected areas vary with attitude as demonstrated in Fig. 3.1; by adjusting the spacecraft's orientation with respect to the flow, accelerations from drag can be modulated and therefore potentially used for control.

Horsley et al [12] presents one method for incorporating the limitations of purely geometric-driven differential drag control as part of a two-step nonlinear planning and control routine. Discrete attitude configurations are selected to produce positive, negative, and zero relative accelerations, effectively using the spacecraft attitude to provide "bang-bang" orbit control. A similar approach based on discrete high- and low-drag attitude modes is used operationally by Planet Labs for constellation constitution and maintenance on their large-scale Earth-imaging cubesat constellation [13]. This approach does not require complex, on-line modeling of spacecraft geometries and provides the maximum possible differential drag for a pair of spacecraft. However, the "bang-bang" approach used by many discrete-attitude-mode controllers incurs substantial mission costs, due to both the time needed to conduct necessary attitude maneuvers and the potentially large attitude maneuvers needed to modulate the spacecraft attitude between configurations.

In addition, uncertainty surrounding both atmospheric neutral density models and atmosphere-surface interaction has hampered the practical application of differential drag techniques. Neutral atmospheric density in LEO can vary by orders of magnitude depending on solar forcing, geomagnetic activity, and diurnal variation [14]. This alone presents a substantial challenge to using differential drag for regular space operations, and is further compounded by the limited progress that has been made in predictive modeling for atmospheric density[8]. While higher accuracy models are potentially possible by incorporating live density estimates–for example, by measuring orbit variations in tracked orbital debris, as shown by [15]–these models rely on the availability of high-accuracy tracking data and spacecraft drag models, which are not widely available. This limitation severely constrains the types of missions and applications for differential-drag control to those that can tolerate substantial uncertainty in control accuracy, settling time, and other performance indices.

Of the variety of methods for leveraging atmospheric drag for spacecraft that have been proposed [9, 16, 11, 17, 4, 18], few have directly studied methods for mitigating the impact of mis-modeled atmospheric density. Time-optimal or bang-bang differential drag strategies can be naturally robust to these variations, as their control depends only upon the sign of the commanded acceleration rather than the magnitude [4]; however, these approaches require large attitude slews and can result in poor settling behavior. Prior work in formulating the attitude-driven differential-drag formation control problem as a linear regulation problem [19, 20] converges under small variations in density from the design quantity as a result of control feedback. While these methods are robust to variation, their performance can vary significantly as the atmospheric density changes, thereby hampering mission operations that depend on relatively precise timing or positioning. This work aims to examine the sensitivity of differential-drag control and develop techniques to minimize performance variation with respect to density changes.

1.1.2 Spacecraft Operations

One often overlooked concern in the field of differential drag is the induced operational burden associated with coupling spacecraft operations to the behavior of the upper atmosphere, which can result in large changes in maneuver duration, accuracy, or feasibility. Similar challenges for other types of aero-assisted spaceflight, such as aerobraking, have been recognized as reducing the utility and cost-efficiency of aero-assisted techniques due to increased operational overhead [5]. Other works outlining the complexity of mission operations in heavily drag-perturbed low-LEO orbits, such as [21], demonstrate the extreme challenge posed by day-to-day operations under a difficult to forecast drag regime. Just as the high cost and risk of aerobraking operations has spurned the development of autonomous tools to manage aerobraking campaigns, this dissertation considers the potential benefits and challenges of autonomous tools for assisting in day-to-day operations that leverage differential drag actuation.

Drag-based formation control poses many challenges for operators. For a variety of factors explored in Chapter 2.5, drag accelerations on spacecraft remain difficult to predict. Boshuizen et. al. [22] reported variation in actual versus predicted drag by 50% for PlanetLabs Flock 1A, requiring additional investment to increase mission lifespan; Planet also identified operations as a major challenge for their relatively large constellation and has invested into automation and decision-assistance tools to mitigate these challenges.

Operating spacecraft without human operators in the loop is also major enabler for future mission architectures ranging from deep-space asteroid sample return to large-scale Earth-orbiting constellations [23]. While decades of development have yielded notable successes in the development of decision support software for operators [24] or on-board observation planning [25, 26, 27], these approaches require substantial development efforts and may struggle to scale as the number of parameters considered increases. At the same time, the machine learning community has renewed its focus Reinforcement Learning techniques that leverage the capabilities of deep neural networks (termed "deep" reinforcement learning or DRL) to address similar high-dimensional decision problems, such as those presented by strategy games [28] or multi-agent coordination problems[29]. Successes in this field suggest that DRL may hold promise for future operational autonomy approaches.

Recent advances in machine learning may hold the key to these next-generation approaches for spacecraft autonomy and on-board decision-making, as they by definition allow agents to improve their behaviors as they gain experience. Contemporary reinforcement learning approaches, for example, do not require knowledge of system models and scale relatively well to large problems with multiple constraints or non-convex reward functions[30]. This work aims to explore the applications and frameworks necessary to apply deep reinforcement learning to the spacecraft decision-making problem.

At present, examples of spacecraft autonomy typically fall into two categories: rule-based autonomy and optimization-based autonomy. Rule-based autonomy treats a spacecraft as a state machine consisting of a set of mode behaviors and defined transitions between modes. Pioneered by missions like Deep Impact [31], and currently used by missions such as the PlanetLabs constellation [13], spacecraft using rule-based autonomy transition between operational and health-keeping modes (charging, momentum-exchange device desaturation) autonomously without ground contact. In contrast, optimization-based autonomy treats the spacecraft and its mission in the framework of constrained optimization, with the spacecraft's hardware and trajectory acting as constraints and metrics of mission return–images taken, communication link up-time, or other criteria–are the values being optimized. In contrast to rule-based autonomy, optimization-based autonomy typically requires large amounts of computing power throughout the mission life-cycle. Examples of this work include the Applied Physics Laboratory's SciBox software library (used to generate MESSENGER mode sequences) and the ASPEN mission planning suite developed by the Jet Propulsion Laboratory and applied to the Earth Observing-1 mission [27].

Owing to their successes in solving other high-dimensional, complex online decision-making problems, Deep RL strategies appear well-suited to the spacecraft operations management problem; the scaling properties of deep networks allows them to tackle high-dimensional, non-convex problems, while trained neural networks themselves are relatively quick to execute in comparison to optimization strategies. A small collection of other works in the application of machine learning techniques to spacecraft problems exists in the recent literature, mostly focusing on the application of learning approaches to control problems in uncertain environments. Several works, such as References [32] and [33], consider reinforcement learning in the context of autonomous aerobraking planners, with mixed results. Others explore machine learning techniques for asteroid proximity operations [34] or autonomous lunar landing[35]. This work builds on prior work in high-level spacecraft tasking and planning [20], creating a problem in the domain of attitude mode guidance that considers high-level mission objectives, traditional guidance considerations (such as mis-modeled dynamics), and spacecraft health constraints.

A small collection of other works in the application of machine learning techniques to spacecraft problems exists in the recent literature, mostly focusing on the application of learning approaches to control problems in uncertain environments. Several works such as References [32] and [33] have considered reinforcement learning in the context of autonomous aerobraking control, with mixed results. Others explore machine learning techniques for asteroid proximity operations [34] or autonomous lunar landing[35]. Importantly, these approaches have focused on low-level control with reinforcement learning, an area that has been traditionally been covered by conventional estimation and control techniques with great success. In contrast, this work explicitly examines applications of reinforcement learning to high-level spacecraft planning and decision-making problems that have traditionally been the domain of rigid expert policies or optimization-focused strategies.

This work considers the domain of spacecraft operations in the same vein as consisting of the active implementation of a mission design through the command and control of a spacecraft. As a core component of the space mission life cycle, a variety of techniques have been used to to generate and implement operational plans and concepts-of-operations for space missions. Several early spacecraft, including Explorer 1, performed their missions with virtually no ground input after launch as a form of extremely minimal autonomy. For relatively simple demonstration missions (such as [36]) or those that need to conduct precise maneuvers under the presence of light-speed delay (such as [31]), a common workflow involves the generation of detailed operational plans or schedules on the ground using human experts while relying on autonomous, closed-loop execution on-board. Owing to the complexity of the operations planning problem, a variety of tools have been developed or discussed to aid this process. JPL's ASPEN tool and its related developments [25, 26, 27] utilizes a constraint-driven job-shop scheduling approach which is amicable to on-board use, and has been demonstrated in flight for science observation tasking on-board the EO-1 mission. A variety of constraint-driven optimization approaches have been applied to various sub-problems in the spacecraft operations domain; for example, a variety of works deal with the scheduling of image collection events [37], communication links [38], or combinations of the two. While these approaches often produce acceptable results for small numbers of tasks or spacecraft, many have difficulties scaling as either the number of possible events, states, or spacecraft increases, especially those that rely on discrete representations of spacecraft states.

For spacecraft that are expected to conduct repetitive behavior, such as the nadir-staring cubesats of the PlanetLabs constellation [13], state-driven operations procedures can be generated that conceptualize the spacecraft as a hybrid system transitioning between discrete dynamical or operational conditions rather than a set of discrete tasks to be scheduled. These approaches are attractive from an implementation perspective, as they require relatively little computational power to execute on-board and can be rigorously verified and validated on the ground. However, developing state-driven rulesets that adequately meet mission criteria typically requires large amounts of engineering time to produce and verify; these challenges are amplified by changing hardware and mission parameters as a spacecraft's life progresses and the adaptation of complex models for system behavior that do not readily fit with conventional control techniques.

At the same time, Deep Reinforcement Learning approaches have been broadly studied in the context of autonomous decision-making and planning for large-scale domains, especially those that incorporate complex dynamics with few analytical models. Unlike other techniques that aim to solve MDPs, DRL techniques do not require explicit models of the environments which they are intended to solve, and can instead learn from pre-existing numerical simulators alone; in addition, their usage of deep neural networks for value and policy representation allows them to generalize from discrete to real-valued representations of state, greatly enhancing their usability and avoiding one aspect of the "curse of dimensionality." This flexibility has enabled DRL-based techniques to demonstrate human- or super-human performance in strategic decision-making tasks, ranging from real-time strategy games [39, 40, 41] to the command and control of autonomous vehicles [42, 43] to data-center power management [44]. Additional advances, such as the successful application of DRL to protein folding–a field that has remained the benchmark for computational challenges–demonstrate the potential for DRL-driven technologies to address extremely complex, high-dimensional, non-linear tasks that would otherwise be the domain of brute-force methods.

A small collection of other works in the application of machine learning techniques to spacecraft problems exists in the recent literature, mostly focusing on the application of learning approaches to control problems in uncertain environments. Several works such as References [32] and [33] consider reinforcement learning in the context of autonomous aerobraking planners, demonstrating the benefits of deep neural network architectures versus conventional tabular reinforcement learning for astrodynamics problems. Others explore machine learning techniques for asteroid proximity operations [34] or autonomous lunar landing[35]. Importantly, these approaches focus on replacing low-level controllers with control laws optimized via reinforcement learning, an area that is commonly addressed by conventional estimation and control techniques with great success. In contrast, this work explicitly examines applications of reinforcement learning to high-level spacecraft planning and decision-making problems that have traditionally been the domain of rigid expert-defined policies or optimization-focused strategies.

1.2 Summary of Objectives

Given the prior literature and areas of interest by the community, this dissertation aims to make the following concrete contributions to the state-of-the-art in spacecraft dynamics, control, and operations:

(1) **Exploit Attitude-Orbit Coupling via Drag:** Identify and implement strategies for differential drag control that directly consider and utilize

(2) **Robustness to Atmospheric Variation:** Identify and extend techniques in robust control to minimize the influence of density variation on differential drag trajectories and maneuver timelines.

(3) **Adaptation of RL Techniques for Space:** Future autonomous space missions will need to deal with high-dimensional operational spaces with complex dynamics which could be addressed with Deep Reinforcement Learning. This work will explore the challenges and benefits of adapting DRL to spacecraft operations problems, including problem formulation, enforcement of safety constraints, and generalizability.

(4) **Operational Management of Differential Drag:** In tandem with the development of new strategies for differential-drag maneuvering, this dissertation will apply DRL techniques to the higher-level operations management problem.

Chapter 2

Problem Formulation

Before engaging in further technical work, this chapter aims to identify relevant space mission archetypes, contemporary satellite hardware limitations, and relevant reference frames and dynamics inherent to the problems addressed in this work.

2.1 Motivating Mission Archetypes

This work intends to contribute to the growing field of guidance, navigation and control for resource-constrained small satellites operating in Low Earth Orbit that can benefit from leveraging atmospheric drag to conduct maneuvers. First, it is desirable to understand what specific types of missions are flow in LEO today and what relevant attributes or constraints they bring to the overall system engineering or operations engineering discussion.

Earth-observation missions are increasingly flown in LEO at altitudes that can make substantial use of differential drag by both government and commercial operators. The A-Train constellation, which consists of a heterogeneous set of Earth-observation satellites from various space agencies, uses a 705 kilometer altitude sun-synchronous orbit with a local solar time of 1:30PM; individual members pass over almost-identical parts of the Earth within minutes of one another to produce high-fidelity images of various environmental and physical phenomena. Cloudsat, an intermediate member of the A-Train, flew only 93 kilometers ahead of the next member of the constellation, allowing for essentially simultaneous measurements between the two spacecraft. This level of coordination has proven itself to be extremely valuable to scientists, as it enables new synergies between sensor types and minimizes the need for post-processing on the ground while combining sensor measurements. Companies such as PlanetLabs (now Planet) and Spire Global have pioneered the commercial use of cubesatellites in LEO for Earth observation, hosting either telescopes and radio occultation sensors respectively. Planet is particularly notable for their operational use of differential drag for phasing the Flock 2p constellation launched in June 2016 using a bang-bang methodology, doubly demonstrating the relevance of improving operational considerations for differential drag control.

In addition, there is growing government and commercial interest in the use of space-based space surveillance assets to monitor the debris environment and the behavior of hostile space assets. An early entrant into this domain is the accurately named Space-Based Space Surveillance (SBSS) constellation, a US Department of Defense constellation consisting of four spacecraft in Sun-Synchronous LEO orbits tasked with maintaining custody over objects in geosynchronous orbit. In addition, multiple academic works have examined the design space for LEO SSA/SDA constellations [45, 46], in orbits ranging from LEO to MEO. Demonstration missions for these technologies, such as the Glint Analyzing Data Observation Satellite (on which the author of this dissertation worked) [47], have also been designed and flown using LEO small satellites in ISS-like orbits.

Finally, there is considerable commercial interest in the use of LEO small satellites to enable world-wide broadband internet connectivity. The first practical commercial entrant to this domain was the Iridium constellation, a set of sixty-six communication satellites in LEO orbits with a launch mass of 689 kilograms per satellite. While initially a commercial failure, the Iridium constellation remains in commercial use and paved the way for follow-on LEO communications constellations. At present, at least three additional commercial constellations are in planning and development phases: SpaceX's Starlink, Amazon's Kuiper, and the OneWeb constellation. Virtually all of these constellations are intending to use orbit altitudes at or above the cutoff in which atmospheric drag dominates other perturbing forces, but may still benefit from novel strategies for operations management.

Despite their different objectives, each of these mission archetypes faces similar constraints due to the nature of constellation flight and the LEO environment. Each mission must:

(1) Conduct phasing and phase-keeping operations to ensure the stability of the constellation

(2) Maneuver to avoid conjunction events with debris or other constellation members

(3) Provide assurances about deorbit and maneuver capability to regulators

(4) Maintain per-spacecraft health under hardware and resource constraints (power, thermal limitations, momentum wheel limits, etc)

(5) Maximize time spent in mission-specific attitudes (nadir-facing for Earth-Observation or communication platforms; sidereal or target-tracking for SDA or space-based astronomy missions)

2.2 Assumed Spacecraft Capabilities

Improved technology has allowed the widespread use of small satellites – defined here as satellites with a launch mass of less than 500 kilograms – for the motivating mission archetypes described in Section 2.1. This section surveys the capabilities and dynamics of small satellites with the intention of further informing current constraints and capabilities.

2.2.1 Attitude Determination and Control

It is assumed that the spacecraft in question can be modeled as a rigid body whose rotational dynamics behave according to Euler's laws for a rigid body, i.e.:

$$[I]\dot{\boldsymbol{\omega}}_{B/N} = -[\boldsymbol{\omega}\times][I]\boldsymbol{\omega} + \boldsymbol{\tau}_{\text{ext}} \tag{2.1}$$

where $[I]$ represents the spacecraft inertia matrix, $\omega_{B/N}$ represents the angular rate between the body and inertial frames, and $\boldsymbol{\tau}_{\text{ext}}$ represents the external torques acting on the spacecraft. Small satellites use a similar array of attitude control devices to full-sized satellites, albeit at different scales; for example, cubesats make widespread use of magnetic torque rods or coils in place of reaction control thrusters due to mass and size constraints associated with thruster-based attitude control and the small moments of inertia associated with cubesats.

Most small satellites use momentum exchange devices, typically reaction wheels, as their primary attitude control device. Unlike other actuators which create torques through the reaction forces arising from magnetic field interactions or expelled gas, momentum exchange devices are notable for simply re-arranging the distribution of angular momentum within a spacecraft's body.

Attitude determination for small satellites is likewise accomplished in a similar manner to traditional spacecraft, albeit using smaller components. While traditional attitude determination approaches, such as bias-replacement multiplicative Extended Kalman Filters (mEKFs) assume extremely low-noise rate estimates produced by highly accurate laser-ring gyroscopes, small satellites must instead make due with low-cost, low-power MEMS devices for IMU functions. In a similar manner, cubesat- and small-satellite scale star trackers, sun sensors, and magnetometers are widely available and used for high-precision attitude control. Commercial solutions for small satellite attitude control, such as the Blue Canyon Technologies XACT family of 'ADC-in-a-box' systems, which claim 1-sigma pointing accuracies of ±0.003 degrees.

While small satellites present unique challenges for the ADCS community owing to their small size and usage of low-SWAP, low-precision sensors, solutions to these problems largely exist at the commercial scale. This work instead focuses on applications for spacecraft for which ADCS is treated as a servo-solved subsystem with implications for other on-board resources.

2.2.2 Orbit Determination and Control

Knowledge and control of a spacecraft's orbit is a critical part of spacecraft operations, especially when considering constellation-scale flight in which precise phasing or station-keeping is a mission requirement. In LEO, orbit determination is typically solved in two ways depending on the precise needs of the mission. For spacecraft that require low-fidelity position estimates in operations, ground-based orbit determination solutions can be constructed using ground-based optical and radar sensors to create local ephemerides. Examples of this approach include the Air Force Space Command's construction and distribution of Two-Line Element sets as a public service for satellite operators or commercial orbit determination providers like LEO Labs or STK. For missions that require precise on-board orbit determination, GPS-based solutions are extensively used in LEO.

Orbit maneuvering, on the other hand, is substantially more difficult for small satellites. Propellant-based systems that are scaled to small-satellite and even cube-satellite size are available, but are limited in terms of total Delta-V and thrust. In addition, there is increasing interest in the adaptation of low-thrust electric propulsion techniques for maneuverability, as contemporary electric propulsion thrusters have substantially higher Delta-V even when the reduced efficacy of low-thrust maneuvers is accounted for. However, as their name implies, low-thrust maneuvers are often more challenging to plan and implement than traditional impulsive maneuvers and complicate mission operations.

From this, it is straightforward to conclude that maneuverability remains a major concern for space missions in LEO, especially with regards to system impacts on overall mission operations.

2.2.3 Spacecraft Subsystems

Spacecraft rely on the proper functioning of a variety of subsystems to complete their mission objectives; as a result, the management of these subsystems forms a majority of the complexity behind spacecraft operations. This section details considerations behind major spacecraft subsystems and straightforward models for capturing the dominant behavior of those considerations.

2.2.3.1 Power

Power is utilized extensively to operate instruments, flight computers, sensors, actuators, and communication equipment, and can be considered as the lifeblood of spacecraft. Spacecraft power analyses are typically run at several fidelities depending on mission lifecycle, ranging from simple input-output models of power generation and consumption to full-system simulation of device voltages and currents using sophisticated circuit simulators. For the purposes of this work, a simple input-output model of spacecraft power:

$$P_{\text{net}} = P_{\text{gen}} - P_{\text{use}} \tag{2.2}$$

where $\P_{\text{gen}}$ and P_{use} represent the power generated and used on-board, respectively. Power storage, usually accomplished through the use of rechargable battery banks, is modeled simply by integrating P_{net}, with the option of adding a coefficient to represent battery efficiency.

A variety of technologies are used to generate power on-orbit. For satellites in Earth orbit, the most common is the use of solar panels, which can either be fixed to the spacecraft bus or deployed from it on adjustable hinges. A common model for solar panel power generation uses a cosine law:

$$P_{\text{Solar}} = P_{\text{sun}} C_{\text{eff}} A(\hat{n}^T \hat{s}) \tag{2.3}$$

where P_{sun} is the power of the sun's rays at the spacecraft's position, C_{eff} is the panel's efficiency coefficient, A is the panel area, $\hat{n}$ is the panel's surface normal, and $\hat{s}$ is the unit vector from the spacecraft towards the sun position. This cosine law for power captures the impact of both spacecraft attitude (which rotates $\hat{n}$ with respect to $\hat{s}$) and the spacecraft's distance from the sun in a compact and elegant model.

Power is consumed on-board by virtually all spacecraft devices. Flight computers, radios, sensors, payloads, and actuators all require various quantities of power to operate; as a result, spacecraft are frequently referred to by the amount of power they are capable of generating.

2.2.3.2 Communications

By definition, spacecraft require some means of relaying mission-relevant information from the point of collection on-orbit to human beings on the ground. A number of technologies have been developed to accomplish this goal, ranging from radio systems to still-speculative laser communications to the recording of relevant information onto physical medium, which is returned to Earth in drop-pods; the latter technology has been largely abandoned with advances in digitization and radio communication. As a result, most missions require periodic contact with a location on the ground to down-link information and receive ground commands. This contact usually requires the spacecraft to pass above a specific ground site with sufficient elevation and for a sufficient time to establish contact and transmit/receive enough data. From an astrodynamics perspective, communications requirements result in orbit requirements on the frequency and duration of passes over ground stations.

To model these passes, a ground-centered frame–referred to as a 'topocentric' frame–is defined:

$$T = \{\boldsymbol{r}_{L/P}, \hat{\boldsymbol{t}}_1, \hat{\boldsymbol{n}}_2, \hat{\boldsymbol{r}}_3\} \tag{2.4}$$

2.2.3.3 On-Board Computing

This work is primarily concerned with the autonomous control of spacecraft, and as a result must consider the compute capabilities and constraints inherent to contemporary satellite systems. Due to the harsh radiation environment in space, most mission designers and operators preferred to fly radiation-hardened or radiation-tolerant systems; these are exemplified by the use of the RAD750, a radiation-hardened version of the 1997 PowerPC 750 family of CPUs, in space missions ranging from 2005's Deep Impact mission to the Perseverance lander. A similar component is the Mongoose-V CPU used to power the New Horizons deep-space probe, which is a radiation-hardened version of the MIPS R3000 which powered the original Sony PlayStation. As a result of the industry's preference for radiation-hardened compute platforms, the on-board computing capabilities of most spacecraft are severely limited in comparison to modern consumer PCs or even cell phones. As a result, solutions for autonomously conducting spacecraft control or operations without ground contact must be operable with low-levels of computational power if they are to be deployed on-board current spacecraft.

However, the explosion of interest and opportunities for launch to LEO and MEO and ensuing reduction of risk and cost for operators has encouraged more extensive flight usage of non-radiation hardened, consumer-grade computers for spacecraft near the protective magnetic field of the Earth.

SpaceX, for example, uses “a stripped-down Linux running on three ordinary dual-core x86 processors” to control the launch and autonomous landing systems on the Falcon 9 launch vehicle. In 2017, a Hewlett Packard Enterprise High-End Desktop using COTS components and custom software to mitigate hardware risks from radiation was deployed successfully on the International Space Station, demonstrating teraFLOP computing capabilities without unexpected interruptions for two years before being returned to Earth. Cubesats have also extensively utilized low-cost COTS computers to reduce costs; examples include the Nanoavionics OBC, which uses an Arm Cortex M7; the similar Pumpkin Space Systems Motherboard Module 2 is designed to support a BeagleBone Black SBC, which features an AM335x ARM Cortex A8. Planet’s first two constellations famously “[contained] no component directly sourced from the space industry” and used a low-power x86 processor with a conventional 500GB solid-state drive for storage. As a result of these successes, it is no longer inconceivable that future space missions will be constrained by decades-old compute capabilities and will have processing power approaching that of modern consumer devices which make extensive use of on-board machine learning and data fusion techniques.

The key takeaways from these trends are that techniques which minimize computational resources while providing good performance are critical for immediate applications, and generally desirable even as compute capabilities improve; however, it is also attractive to identify key technologies for on-board use that will take advantage of a growing recognition that modern computers can be flown in low-Earth orbit.

2.3 Frame Definitions

Before introducing dynamical models, it is important to define the reference frames which define the problem. First is the planet-centered inertial frame N, which is taken as the global origin of the system:

$$N = \{\boldsymbol{0}, \hat{\boldsymbol{n}}_1, \hat{\boldsymbol{n}}_2, \hat{\boldsymbol{n}}_3\} \tag{2.5}$$

in which the various unit vectors $\hat{\boldsymbol{n}}_i$ are stationary with respect to the inertial frame. Next is the Hill frame H, which is centered on the spacecraft at a given position $\boldsymbol{r}_{H/N}$ in orbit and consists of the following unit vectors:

$$H = \{\boldsymbol{r}_{H/N}, \hat{\boldsymbol{h}}_r, \hat{\boldsymbol{h}}_\theta, \hat{\boldsymbol{h}}_h\} \tag{2.6}$$

where $\boldsymbol{r}_{H/N}$ is the position vector of the spacecraft with respect to the center of the N frame and the unit vectors are defined as follows:

$$\hat{\boldsymbol{h}}_r = \frac{\boldsymbol{r}_{H/N}}{\|\boldsymbol{r}_{H/N}\|} \tag{2.7}$$

$$\hat{\boldsymbol{h}}_h = \frac{\boldsymbol{r}_{H/N} \times \dot{\boldsymbol{r}}_{H/N}}{\|\boldsymbol{r}_{H/N} \times \dot{\boldsymbol{r}}_{H/N}\|} \tag{2.8}$$

$$\hat{\boldsymbol{h}}_\theta = \hat{\boldsymbol{h}}_h \times \hat{\boldsymbol{h}}_r \tag{2.9}$$

The direction cosine matrix that maps vectors from H to N, denoted as $[HN]$, is expressed by:

$$[HN] = \begin{bmatrix} \hat{\boldsymbol{h}}_r^T \\ \hat{\boldsymbol{h}}_\theta^T \\ \hat{\boldsymbol{h}}_h^T \end{bmatrix} \tag{2.10}$$

The angular velocity of H with respect to N is given by the spacecraft's mean motion n, which forms the angular velocity vector ${}^{\mathcal{N}}\boldsymbol{\omega}_{H/N} = \dot{f}\hat{\boldsymbol{h}}_h$, where $\dot{f}$ is the orbit true anomaly rate. For circular orbits, the true anomaly rate is equal to the mean anomaly rate n.

Several components of this work focus on interactions between space and ground systems, necessitating the definition of an additional planet-fixed reference frame, denoted P:

$$P = \{\boldsymbol{r}_{P/N}, \hat{\boldsymbol{p}}_1, \hat{\boldsymbol{p}}_2, \hat{\boldsymbol{p}}_3\} \tag{2.11}$$

It is a common assumption to take $\hat{\boldsymbol{p}}_3 = \hat{\boldsymbol{n}}_3$ for planet-fixed environments where N represents a planet-centered inertial reference frame; under this assumption, the angular velocity of the P frame reduces to $\boldsymbol{\omega}_{P/N} = \begin{bmatrix} 0 & 0 & \omega_P \end{bmatrix}$, where ω_P is the planet's rotational velocity. This assumption neglects additional planetary attitude dynamics, such as the nutation and precession of the poles, which is typically of great importance for accurate ground-pass prediction.

Next, the spacecraft body frame B is defined, which is aligned with the spacecraft's principal inertia frame and written as the following:

$$B = \{\boldsymbol{r}_{H/N}, \hat{\boldsymbol{b}}_1, \hat{\boldsymbol{b}}_2, \hat{\boldsymbol{b}}_3\} \tag{2.12}$$

The angular velocity vector between the body and inertial frames is given generally as:

$$^{\mathcal{B}}\boldsymbol{\omega}_{B/N} = \begin{bmatrix} \omega_1 & \omega_2 & \omega_3 \end{bmatrix}^T \tag{2.13}$$

2.4 Orbit Regime

Analyses in different regions of space require the consideration of different dynamics and constraints. This work is primarily concerned with the behavior of spacecraft on Low Earth Orbits, with some consideration for spacecraft in low orbits about planets with atmospheres such that atmospheric effects are substantial, but not dominant over orbital dynamics. Using Newton's law of gravitation and assuming that the mass m of an orbiting body is far smaller than that of the dominant body (M), the acceleration felt by said satellite is calculated as:

$$\ddot{\boldsymbol{r}} = -\frac{\mu}{r^3}\boldsymbol{r} + \boldsymbol{a}_p \tag{2.14}$$

where $\boldsymbol{r}$ is the spacecraft position vector, $\ddot{\boldsymbol{r}}$ is the second derivative (acceleration) of the spacecraft position, μ is the gravitational parameter of the primary body (such that $\mu = GM$), and $\boldsymbol{a}_p$ is used to denote additional perturbing accelerations.

Figure 2.1 shows the relative magnitude of perturbations as a function of altitude for a spacecraft with an area-to-mass ratio of 0.01 . For spacecraft in Earth orbits with altitudes below 800 kilometers, the dominant source of perturbations from two-body motion arise from J_2 perturbations (i.e., gravitational effects of Earth's oblateness) and atmospheric drag, which is additionally strongly altitude-dependent. The accelerations arising from J_2 are computed from the spacecraft's position vector as:

$$^{\mathcal{N}}\boldsymbol{a}_{J2} = -\frac{3}{2}J_2\frac{\mu}{r^2}\frac{r_{\text{eq}}{}^2}{r}\begin{pmatrix} \left(1 - 5\left(\frac{z}{r}^2\right)\right)\frac{x}{r} \\ \left(1 - 5\left(\frac{z}{r}^2\right)\right)\frac{y}{r} \\ \left(3 - 5\left(\frac{z}{r}^2\right)\right)\frac{z}{r} \end{pmatrix} \tag{2.15}$$

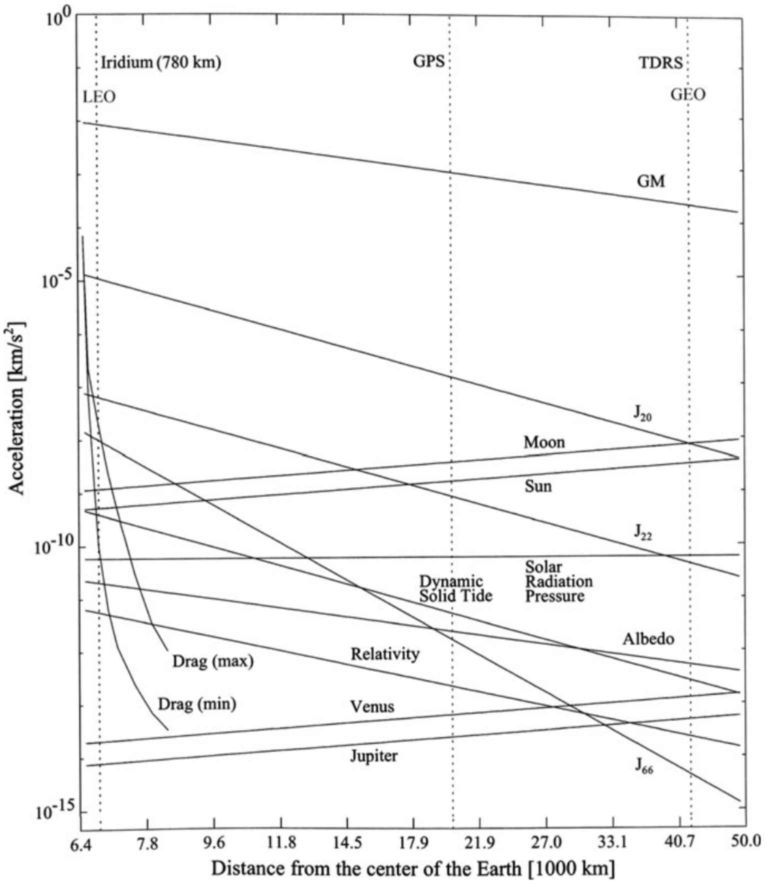


Figure 2.1: Orbit perturbation magnitudes versus altitude. From [1].

where x, y, and z represent the first, second, and third components of the spacecraft's position vector, r_{eq} is the planet equatorial radius, and r is the spacecraft radius. Because $\boldsymbol{a}_{J2}$'s arising from the Earth's oblateness is apparent in the latitude term that appears in the calculation of each force component.

2.5 Satellite Drag Modeling

Accurate predictions of atmospheric density remain a major challenge behind both orbit determination for LEO spacecraft and the adaptation of differential-drag control in LEO. As a result, the prediction and modeling of atmospheric density in the thermosphere and exosphere have been the subject of extensive study since the early days of spaceflight in the 1960s. This work focuses predominantly on spacecraft in strongly drag-perturbed orbits, which tend to fall in or below 500-800 kilometers for Earth orbiting spacecraft. These altitudes fall within the layer of the Earth's atmosphere known as the thermosphere. Unlike lower layers of the atmosphere, gasses in the thermosphere are largely ionized due to reduced pressure and exposure to solar radiation, resulting in a free molecular flow regime at LEO altitudes. Because of this, the term "drag" is perhaps a misnomer; the dynamics of such interactions are more akin to ballistic collisions than fluid-surface interactions. However, because these impacts still transfer momentum into spacecraft, the quadratic drag model is still widely used to model accelerations due to atmospheric interactions:

$$\boldsymbol{a}_D = -\frac{1}{2}\beta\rho(\boldsymbol{v}^T\boldsymbol{v})\hat{\boldsymbol{v}} \tag{2.16}$$

where β is the spacecraft ballistic coefficient and defined as $m^{-1}C_DA$ (where C_D is a non-dimensional drag coefficient and $\boldsymbol{v}$ is the relative velocity of the spacecraft with respect to the atmosphere. Faceted models for spacecraft drag, in which a spacecraft geometry is broken into a set of flat panels and drag forces are considered on a panel-by-panel basis, represent an intermediate fidelity model between full-scale particle simulation and low-fidelity, attitude-independent models of drag. Considering a spacecraft consisting of several flat faceted panels with individual areas A_i, individual drag coefficients $C_{D,i}$, and individual orientations in the body frame $\hat{n}_i$, the spacecraft ballistic coefficient due to a collection of n flow-exposed panels is written using a modified form of the expressions derived by Sutton [48]:

$$\beta = \frac{\sum_{i=1}^{n} C_{D,i}A_i({}^{\mathcal{B}}\hat{\boldsymbol{n}}_i \cdot [BN]^{\mathcal{N}}\hat{\boldsymbol{v}})}{m} \tag{2.17}$$

in which $\boldsymbol{v}$ is the flow-relative velocity of the spacecraft and $\hat{\boldsymbol{v}}$ is the unit direction of the flow-relative velocity, which is likewise modeled as consisting of both the spacecraft and atmospheric velocities:

$$\boldsymbol{v}_{atmo} = \boldsymbol{v}_{s/c} + \boldsymbol{v}_{\text{wind}} \tag{2.18}$$

The term $A_i({}^{\mathcal{B}}\hat{\boldsymbol{n}}_i \cdot [BN]^{\mathcal{N}}\hat{\boldsymbol{v}})$ is referred to as the projected area A_p. The drag coefficient, $C_{D,i}$, is a complex variable arising from interactions between rarefied atmosphere and a facet's material properties. Some analytical models of gas-surface interactions do include attitude dependence, such as those described by Bird [49] and used in a space context by Sutton [48]. Calculating the impacts of these effects requires detailed knowledge of both the spacecraft's material properties, which vary on orbit due to space weathering effects, and the specific temperature and composition of the local atmosphere. The combined uncertainty arising from these effects not only complicates the prediction of drag forces for an individual spacecraft, but further complicate the inverse problem of deducing atmospheric composition, density, and wind direction from satellite position and velocity measurements[50].

Those difficulties have in part led directly to the wide variety of empirical and analytical models of atmospheric density and wind used in the astrodynamics and aeronomy communities today. Thermospheric density models are typically grouped into static or dynamic categories, reflecting whether the model produces varying density profiles with time, space weather conditions, and other factors; models are further subdivided into empirical and analytical categories depending on the extent to which they use historical measurements of density versus computational models of atmospheric circulation. By far the most common atmospheric model used in astrodynamics is the simple exponential atmosphere, which arises as a consequence of the assumption of hydrostatic equilibrium for the entire atmosphere and has the following form:

$$\rho(r) = \rho_0 e^{\frac{r-r_0}{h}} \tag{2.19}$$

where ρ_0 is the density at r_0 and h is the atmospheric scale height. This model reflects gross trends in atmospheric density, but is generally inaccurate for point predictions of atmospheric density as a result of the impact of diurnal effects, variations in EUV, and geomagnetic interactions with solar wind, which add or remove energy from the thermosphere and can dramatically impact density [51, 52]. Empirical models attempt to overcome this theoretical weakness by fitting density profiles to observed density estimates in a variety of space weather conditions, resulting in models that map not only from position but also space weather indices to local densities and temperatures. Widely used contemporary models include the Jaccia-Bowman 2008 density model [53] and the

Naval Reconnaissance Laboratory Microwave Incoherent Scatter Experiment 2000 (NRLMSISE-00) model [54], which provide better fidelity for the impact of time-of-day and space environment impacts. Despite this, these models have been shown to produce 1-sigma prediction errors of 15-25% the total density prediction. First-principles circulation models, such as UCAR's TIE-GCM model[55], provide an alternative to empirical or semi-analytical models by attempting to directly simulate the upper atmosphere. While such models can match the accuracy of semi-analytical density predictions - a feat that speaks to the sophistication of our understanding of the upper atmosphere – circulation models typically require overwhelming compute capabilities that are challenging for use in on-line density prediction.

A note should also be made about thermospheric wind models, which follow similar trends to density models albeit with a smaller scope. Thermospheric winds tend to be small in comparison to spacecraft velocities; as a result, many analyses simply neglect the consideration of atmospheric winds at all. A common assumption in the astrodynamics community is the use of co-rotating atmosphere assumptions, which calculate the wind magnitude and velocity as a function of altitude and the Earth's rotational velocity $\boldsymbol{\omega}_E$:

$$\boldsymbol{v}_{\text{wind}} = \boldsymbol{r} \times \boldsymbol{\omega}_E \tag{2.20}$$

This simple analytical model is analogous to the widespread use of exponential models for atmospheric density; as an analytical, differentiable function of only the Earth's rotational velocity, this model is simple to implement and analyze. However, even early studies of thermospheric velocity have revealed deficiencies with this model at high altitudes [56], in addition to increasing errors as orbit inclination increases. As a result of these shortcomings, the empirical Horizontal Wind Model (HWM) [57] was developed, which provides predictions of zonal and meridian winds as a function of topographic position, altitude, time of day, day of year, and A_p index.

As a result of this literature survey, it is apparent that forecasting of atmospheric density and satellite drag coefficients remains challenging. At the same time, a large spread of both empirical and computational models exist for density forecasting with different respective strengths and weaknesses. As a result, a key takeaway from this survey is that density remains largely difficult to accurately predict and no individual model generalizes particularly well. Finally, as noted by Vallado [58], simulation tools used to incorporate the impacts of atmospheric density–and analyses of controllers that utilize density for orbit actuation – should be designed to incorporate as many different atmospheric density models as is feasible.

2.6 Conclusion

This section has established both contemporary capabilities and challenges in near-Earth spaceflight, as well as common dynamical and environmental models that will broadly inform the shape of this work. Rising demand for Earth-oriented missions for scientific, commercial, and defense purposes alongside dropping hardware and launch costs has renewed interest in the development and operations of constellations and formations consisting of low-cost satellites; it can be reasonably expected that these cost-constrained small satellites will face operational challenges arising from limited on-board resources, but enjoy the benefits of half a century's work on spacecraft guidance, navigation, and control alongside advanced levels of compute capability. Finally, models and challenges for predicting and modeling spacecraft drag were reviewed and summarized.

Chapter 3

Attitude-Driven Differential Drag

For missions that fly close to planets with atmospheres, atmospheric drag is not only a substantial perturbing force but also a source of coupling between spacecraft orbit and attitude motion. This chapter aims to derive new strategies and techniques for atmosphere-based orbit control with a specific focus on improving the usability of these approaches in the spacecraft operations pipeline.

3.1 Introduction

In concept, the work presented here is similar to an existing body of literature which focuses on ballistic-coefficient controlled differential-drag formation flight. These techniques focus on the control of one or more spacecraft's ballistic coefficient by means of actuated flaps[9] or panels, and treat either the ballistic coefficient or the spacecraft flow-wise projected area as the primary control input [10]. This class of differential drag-based control was flown by the AeroCube-4 technology demonstration mission [11]. The addition of actuated flaps and panels, while attractive for control purposes, unfortunately incurs additional cost and system complexity that is undesirable for mission managers. Many spacecraft, including cubesats, have non-uniform geometries whose projected areas vary with attitude as demonstrated in Fig. 3.1; by adjusting the spacecraft's orientation with respect to the flow, accelerations from drag can be modulated and therefore potentially used for control.

Horsley et al [12] presents one method for incorporating the limitations of purely geometric-driven differential drag control as part of a two-step nonlinear planning and control routine. Discrete

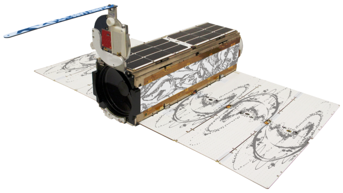


Figure 3.1: A PlanetLabs Dove spacecraft demonstrating non-uniform geometry

attitude configurations are selected to produce positive, negative, and zero relative accelerations, effectively using the spacecraft attitude to provide "bang-bang" orbit control. A similar approach based on discrete high- and low-drag attitude modes is used operationally by Planet Labs for constellation constitution and maintenance on their large-scale Earth-imaging cubesat constellation [13]. This approach does not require complex, on-line modeling of spacecraft geometries and provides the maximum possible differential drag for a pair of spacecraft. However, the "bang-bang" approach used by many discrete-attitude-mode controllers incurs substantial mission costs, due to both the time needed to conduct a maneuver and the potentially large attitude maneuvers needed to modulate the spacecraft attitude between configurations.

For cubesats (such as Planet's Dove spacecraft shown in Fig. 3.1), maneuvering between a minimum-drag and maximum-drag configuration requires a 90° slew. As such, spacecraft are not capable of conducting mission operations during orbit maintenance periods. Dell'Elce and Kerschen [17] present a method of single-axis attitude-driven orbit control for the QB50 constellation using an on-line optimizer and compensator. The computational intensiveness of this technique requires the use of approximations for on-line application, but nevertheless provides credibility to the concept of continuous differential-drag control using small attitude motions. Prior work [19] focused on the linearized dynamics of single-facet spacecraft; this work aims to extend this methodology to general spacecraft that can be modeled as collections of facets, allowing for the incorporation of higher-fidelity geometric models.

This work aims to improve upon computationally expensive optimization-based approaches by demonstrating a linear control approach for attitude-driven differential drag formation flight. To do so, the coupling between spacecraft geometries and experienced drag through attitude must be explored directly. The influence of geometry and surface material properties on spacecraft drag is a subject of intense research due to its importance in both space object tracking and aeronomy studies. For analytic insight, facet-based models such as those explored by Sutton [48] provide reasonable accuracy and insight into the dynamics of the "true" system. An alternative approach is the use of multi-particle Monte-Carlo (MPMC) or other MC-based methods to develop lookup tables or fitted analytical functions to approximate the real drag behavior of a given spacecraft. While potentially more accurate in the presence of concave geometries [59], simple convex geometries–such as those of cubesats– are reasonably well-modeled by analytical expressions. In contrast to previous work, this chapter explicitly considers models of multi-faceted spacecraft and examines the effect of additional substantial perturbation dynamics on the control's performance.

Coupling between translational and rotational motion has been treated extensively in the context of robotics, providing a beneficial framework for analyzing problems in astrodynamics. Filipe [60] develops a methodology for conducing coupled rotational-translational control for spacecraft rendezvous using a dual-quaternion representation of the attitude and orbit. Solar or electric sails, which also experience considerable attitude-orbit coupling, have served as the objects of study for coupled attitude-orbit control [61, 62] . A common issue with these approaches is the lack of additional intuition gained through the use of compact translation-rotation representations such as dual quaternions. For these reasons, a straightforward linear model of the underling relative dynamics is sought.

The work is organized as follows. First, a nonlinear model of coupled attitude-orbit motion is presented in Section 3.2.2. Next, this model is linearized about a selected reference orbit experiencing drag forces in Section 3.2.4. Section 3.2.6 describes the novel linearization of the system's geometric attitude dependence about a selected reference attitude. The linear controllability of this system is established in Section 3.3.1, which additionally describes necessary conditions for controllability. Finally, Section 3.3.2.1 demonstrates the implementation and performance of a linear-quadratic regulator based on the linearized system on both the linearized and nonlinear dynamics under realistic variations from the assumed linear system.

3.2 Problem Statement

3.2.1 Frame Definitions

Before addressing the system model, it is important to define the reference frames which define the problem. First is the planet-centered inertial frame N, which is taken as the global origin of the system:

$$N = \{\mathbf{0}, \hat{\boldsymbol{n}}_1, \hat{\boldsymbol{n}}_2, \hat{\boldsymbol{n}}_3\} \tag{3.1}$$

Next is the Hill frame H, which is centered on the spacecraft at a given position $\boldsymbol{r}_{H/N}$ in orbit and consists of the following unit vectors:

$$H = \{\boldsymbol{r}_{H/N}, \hat{\boldsymbol{h}}_r, \hat{\boldsymbol{h}}_\theta, \hat{\boldsymbol{h}}_h\} \tag{3.2}$$

where $\boldsymbol{r}_{H/N}$ is the position vector of the spacecraft with respect to the center of the N frame and the unit vectors are defined as follows:

$$\hat{\boldsymbol{h}}_r = \frac{\boldsymbol{r}_{H/N}}{\|\boldsymbol{r}_{H/N}\|} \tag{3.3}$$

$$\hat{\boldsymbol{h}}_h = \frac{\boldsymbol{r}_{H/N} \times \dot{\boldsymbol{r}}_{H/N}}{\|\boldsymbol{r}_{H/N} \times \dot{\boldsymbol{r}}_{H/N}\|} \tag{3.4}$$

$$\hat{\boldsymbol{h}}_\theta = \hat{\boldsymbol{h}}_h \times \hat{\boldsymbol{h}}_r \tag{3.5}$$

The direction cosines matrix that maps vectors from H to N, denoted as $[HN]$, is expressed by:

$$[HN] = \begin{bmatrix} \hat{\boldsymbol{h}}_r^T \\ \hat{\boldsymbol{h}}_\theta^T \\ \hat{\boldsymbol{h}}_h^T \end{bmatrix} \tag{3.6}$$

The angular velocity of H with respect to N is given by the spacecraft's mean motion n, which forms the angular velocity vector ${}^{\mathcal{N}}\boldsymbol{\omega}_{H/N} = \dot{f}\hat{\boldsymbol{h}}_h$, where $\dot{f}$ is the orbit true anomaly rate. For circular orbits, the true anomaly rate is equal to the mean anomaly rate n.

Finally, the spacecraft body frame B is defined, which is aligned with the spacecraft's principal inertia frame and written as the following:

$$B = \{\boldsymbol{r}_{H/N}, \hat{\boldsymbol{b}}_1, \hat{\boldsymbol{b}}_2, \hat{\boldsymbol{b}}_3\} \tag{3.7}$$

The angular velocity vector between the body and inertial frames is given generally as:

$$^{\mathcal{B}}\boldsymbol{\omega}_{B/N} = \begin{bmatrix} \omega_1 & \omega_2 & \omega_3 \end{bmatrix}^T \tag{3.8}$$

3.2.2 Nonlinear Dynamics

With the system reference frames established, the dynamics that underlie this work are next defined. A spacecraft experiencing spherical two-body gravity and other perturbation accelerations obeys the following equations of motion [52]:

$$\ddot{\boldsymbol{r}} = -\frac{\mu}{r^3}\boldsymbol{r} + \boldsymbol{a}_p \tag{3.9}$$

where $\boldsymbol{r}$ is the inertial spacecraft position vector, μ is the planet's gravitational parameter, and $\boldsymbol{a}_p$ is the inertial perturbing acceleration vector. It is assumed that drag is the sole perturbation force and follows a quadratic model [52]:

$$\boldsymbol{a}_p = \boldsymbol{a}_D = -\frac{1}{2}\beta P(\boldsymbol{v}^T\boldsymbol{v})\hat{\boldsymbol{v}} \tag{3.10}$$

in which β represents the spacecraft ballistic coefficient, P is used to represent the local atmospheric density, $\boldsymbol{v}$ is the flow-relative velocity of the spacecraft, and $\hat{\boldsymbol{v}}$ is the unit direction of the flow-relative velocity.

Attitude dependence enters into the system primarily through the ballistic coefficient β_d, which depends on the spacecraft's flow-wise projected area A_i. Considering a spacecraft consisting of several flat faceted panels with individual areas A_i, individual drag coefficients $C_{D,i}$, and individual orientations in the body frame $\hat{n}_i$, the spacecraft ballistic coefficient due to a collection of n flow-exposed panels is written using a modified form of the expressions derived by Sutton [48]:

$$\beta = \frac{\sum_{i=1}^{n} C_{D,i} A_i (^{\mathcal{B}}\hat{\boldsymbol{n}}_i \cdot [BN]^{\mathcal{N}}\hat{\boldsymbol{v}})}{m} \tag{3.11}$$

The term $A_i({}^{\mathcal{B}}\hat{\boldsymbol{n}}_i \cdot [BN]^{\mathcal{N}}\hat{\boldsymbol{v}})$ will be referred to as the projected area A_p. The drag coefficient, $C_{D,i}$, is a complex variable arising from interactions between rarefied atmosphere and a facet's material properties. Some analytical models of gas-surface interactions do include attitude dependence, such as those described by Bird [49] and used in a space context by Sutton [48]. Calculating the impacts of these effects requires detailed knowledge of both the spacecraft's material properties, which vary on orbit due to space weathering effects, and the specific temperature and composition of the local atmosphere. These parameters are difficult to determine in practice. For the purposes of this analysis, it is assumed that surface drag coefficients remain constant over the analysis time period.

3.2.3 Nonlinear Relative Dynamics

For spacecraft in LEO, drag forces alone can be used to achieve limited translational controllability. In general, they can only be used to reach orbits with equal inclinations (as drag acts primarily in the orbit plane) and lower energies. Instead of considering the case of general LEO orbital transfer, this paper's scope is restricted to consider only the relative motion between spacecraft experiencing atmospheric drag forces.

Classic relative motion equations describe the motion of a "deputy" spacecraft as seen by a "chief" spacecraft. The positions of these two spacecraft are related by the following expression:

$$\boldsymbol{r}_d = \boldsymbol{r}_c + \boldsymbol{\rho} \tag{3.12}$$

in which $\boldsymbol{\rho}$ is introduced to represent the relative position between the chief and deputy. Taking two inertial derivatives results in the following relationship between the accelerations:

$$\ddot{\boldsymbol{r}}_d = \ddot{\boldsymbol{r}}_c + \ddot{\boldsymbol{\rho}} \tag{3.13}$$

From this, the relative acceleration vector is solved for in terms of the chief and deputy accelerations given by Eq. 3.9:

$$\ddot{\boldsymbol{\rho}} = \ddot{\boldsymbol{r}}_d - \ddot{\boldsymbol{r}}_c \tag{3.14}$$

$$\ddot{\boldsymbol{\rho}} = -\frac{\mu}{r_d^3}\boldsymbol{r}_d + \boldsymbol{a}_{D,d} + \frac{\mu}{r_c^3}\boldsymbol{r}_c - \boldsymbol{a}_{D,c} \tag{3.15}$$

$$\tag{3.16}$$

in which $\mathbf{a}_{D,c}$ and $\mathbf{a}_{D,d}$ are used to represent the drag accelerations of the chief and deputy, respectively. Substituting in Eq. 3.12 results in:

$$\ddot{\boldsymbol{\rho}} = -\frac{\mu}{(r_c+\rho)^3}(\boldsymbol{r}_c + \boldsymbol{\rho}) + \frac{\mu}{r_c^3}\mathbf{r_c} + \boldsymbol{a}_{D,d} - \boldsymbol{a}_{D,c} \tag{3.17}$$

3.2.4 Linear Relative Dynamics with Drag

The nonlinear dynamics expressed in Eq. 3.17 provide little a-priori analytical insight into the behavior of relative spacecraft motion under drag. To this end, Silva [63] provides one set of analytical expressions for relative motion under the assumption of atmospheric drag, small relative positions and velocities, and circular chief orbits, effectively constituting a “Hill-Clohessy-Whitshire plus drag” formulation for differential drag motion. Rotated into the aforementioned Hill frame and taking $\boldsymbol{\rho} = \begin{bmatrix} x & y & z \end{bmatrix}^T$, these equations are:

$$\ddot{x} = 2\dot{y}n + 3n^2x - \frac{1}{2}\beta_d P_d n r_c \dot{x} \tag{3.18a}$$

$$\ddot{y} = -2\dot{x}n - n^2 r_c^2 \frac{1}{2}(\beta_c P_c - \beta_d P_d) - \beta_d P_d n r_c \dot{y} \tag{3.18b}$$

$$\ddot{z} = -zn^2 - \frac{1}{2}(\beta_d P_d r_c n)\dot{z} \tag{3.18c}$$

This model neglects the linearized effect of relative altitude variation on the atmospheric density. For an exponential atmosphere, the linearized deputy density would be

$$P_d = P_c e^{-x/H} \approx P_c(1 - x/H) \tag{3.19}$$

which is accurate within one atmospheric scale height of the chief’s position, or approximately 8 kilometers in LEO. However, introducing this linearization to Eq. 3.18a-3.18c creates a dependence on atmospheric scale height, which may be only coarsely known. Instead, the variable P_d will be retained.

For static relative equilibria to exist, both the first and second order derivatives must be zero. Setting the second order derivatives equal to zero yields the following expressions:

$$\frac{1}{2}\beta_d P_d r_c \dot{x} = 2\dot{y} + 3nx \tag{3.20a}$$

$$n^2 r_c^2 \frac{1}{2}(\beta_c P_c - \beta_d P_d) + \beta_d P_d n r_c \dot{y} = -2\dot{x}n \tag{3.20b}$$

$$\frac{1}{2}(\beta_d P_d r_c n)\dot{z} = -zn^2 \tag{3.20c}$$

A secular drift term exists in the y direction due to the differential drag force acting between the deputy and chief, without a dependence on the relative state components. At the same time, we see that conditions exist that permit stable modes in the x and y velocities for nonzero values of x. Additionally zeroing the first-order derivatives yields:

$$0 = 3nx \tag{3.21a}$$

$$0 = n^2 r_c^2 \frac{1}{2}(\beta_c P_c - \beta_d P_d) \tag{3.21b}$$

$$0 = -zn^2 \tag{3.21c}$$

which suggests that the system origin is a static equilibrium when the differential drag term $n^2 r_c^2 \frac{1}{2}(\beta_c P_c - \beta_d P_d)$ is zeroed.

As such, in addition to the classic HCW conditions for static equilibria, it is necessary to ensure that the deputy and chief values of the ballistic coefficient and local neutral density match. This condition could be achieved by either utilizing spacecraft with identical geometries and masses (i.e. formation constitution/maintenance), or by selecting different reference attitudes in which both spacecraft display identical ballistic coefficients. For the purposes of this work, the former assumption will be made for the remainder of the presented analysis.

3.2.5 Linear Relative Impact of Lift and Drag

3.2.6 Attitude Sensitivity

The linear approximations for both formation dynamics under drag and the effect of attitude on drag forces lend themselves to the application of linear controllability tools. To use these tools, it is necessary to restate the system dynamics in a linear form such that the system behavior is described by

$$\dot{\boldsymbol{x}} = [A]\boldsymbol{x} + [B]\boldsymbol{u} \tag{3.22}$$

where $[A]$ represents the linearized state dynamics and $[B]$ represents the linearized control effects matrix.

The second-order relative equations of motion given in Equations 3.18a-3.18c contain secular drift terms proportional to the deputy-chief differential drag $(\beta_c P_c - \beta_d P_D)$. Under the assumption of similar deputy and chief geometries, this term goes to zero, as $\beta_c = \beta_d$ for identical reference geometries and attitudes, and $P_D = P_c$ as $\boldsymbol{\rho}$ goes to $\mathbf{0}$. This assumption is reasonable for station-keeping within a formation of identical spacecraft, for which local variations in density are likely small and spacecraft are likely to have similar geometries. Applying this assumption yields the following state dynamics matrix:

$$[A] = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 3n^2 & 0 & 0 & -\frac{1}{2}\beta_d P_d n r_c & 2n & 0 \\ 0 & 0 & 0 & -2n & -\beta_d P_d n r_c & 0 \\ 0 & 0 & -n^2 & 0 & 0 & -\frac{1}{2}(\beta_d P_d r_c n) \end{bmatrix}, \quad \boldsymbol{x} = \begin{bmatrix} \boldsymbol{\rho} \\ \dot{\boldsymbol{\rho}} \end{bmatrix} \tag{3.23}$$

Denoting the sensitivity of the deputy ballistic coefficient on attitude as $\frac{\partial \beta_d}{\partial \boldsymbol{\sigma}_p}$, where $\boldsymbol{\sigma}_p$ is an arbitrary attitude variation, the sensitivities of the system dynamics to variation in attitude is described by:

$$\frac{\partial \ddot{x}}{\partial \boldsymbol{\sigma}_p} = -\frac{1}{2} P_d n r_c \dot{x}_0 \frac{\partial \beta_d}{\partial \boldsymbol{\sigma}_p} \tag{3.24a}$$

$$\frac{\partial \ddot{y}}{\partial \boldsymbol{\sigma}_p} = (\frac{1}{2} n^2 r_c^2 P_d - P_d n r_c \dot{y}_0) \frac{\partial \beta_d}{\partial \boldsymbol{\sigma}_p} \tag{3.24b}$$

$$\frac{\partial \ddot{z}}{\partial \boldsymbol{\sigma}_p} = -\frac{1}{2} (P_d r_c n) \dot{z}_0 \frac{\partial \beta_d}{\partial \boldsymbol{\sigma}_p} \tag{3.24c}$$

A consequence of this linearizion is that the relative acceleration partials are dependent upon the selection of initial or selected reference relative velocities. For the purposes of this work, $[B]$ will be evaluated at the desired equilibrium state where $\boldsymbol{\rho} = \mathbf{0}$, $\dot{\boldsymbol{\rho}} = \mathbf{0}$. By taking the variation from reference attitude $\boldsymbol{\sigma}_p$ as the control input to the system, the B matrix is stated as:

$$[B] = \begin{bmatrix} \mathbf{0}_{3\times 3} \\ \mathbf{0}_{1\times 3} \\ \frac{1}{2} n^2 r_c^2 P_d \frac{\partial \beta_d}{\partial \boldsymbol{\sigma}_p} \\ \mathbf{0}_{1\times 3} \end{bmatrix}, \ \boldsymbol{u} = \begin{bmatrix} \sigma_{p,1} \\ \sigma_{p,2} \\ \sigma_{p,3} \end{bmatrix} \tag{3.25}$$

For demonstrative purposes, the specific case of attitude-independent drag coefficients is considered. Attitude-independent drag coefficient models are commonly used throughout the formation flight literature. Under this assumption, all variation in the ballistic coefficient is due to attitude effects on the spacecraft's projected area. To examine these effects, an additional "Target" frame T is defined with a corresponding attitude matrix $[TB]$, allowing the expression of the projected attitude as

$$A_p = A_i(\hat{\boldsymbol{n}}^T [TB(\boldsymbol{\sigma}_p)][BN(\boldsymbol{\sigma}_r)]^{\mathcal{N}}\hat{\boldsymbol{v}}) \tag{3.26}$$

Modified Rodriguez Parameters (MRPs) are selected as the attitude parametrization for this linearization to improve the domain of linearity[7]. Without loss of generality, the inertial velocity direction is also rotated into the chief Hill reference frame. Under the assumption of circular orbits, the inertial direction of the velocity vector in the chief reference frame is simply the $\hat{\boldsymbol{h}}_\theta$ unit vector. The per-facet projected area is therefore:

$$A_p = A_i(\hat{\boldsymbol{n}}_i^T [TB(\boldsymbol{\sigma}_p)][BH(\boldsymbol{\sigma}_r)]^{\mathcal{H}}\hat{\boldsymbol{v}}) \tag{3.27}$$

if that σ_p is small such that second order terms can be neglected, Eq. 3.27:

$$A_p = A_i(\hat{\boldsymbol{n}}_i^T[BN(\boldsymbol{\sigma}_r)]\hat{\boldsymbol{v}} - 4\hat{\boldsymbol{n}}_i^T[\boldsymbol{\sigma}_p\times][BN(\boldsymbol{\sigma}_r)]\hat{\boldsymbol{v}}) \tag{3.28}$$

This expression contains two primary components: a constant term driven by the selected reference MRP, and a linearized rotational component based on the perturbing MRP. Treating this perturbing MRP as the control input to the system, it is apparent that the partials of the ballistic coefficient are dependent only on this small-angle rotational component:

$$\frac{\partial \beta_d}{\partial \boldsymbol{\sigma}_p} = \frac{1}{m_i}\sum_{i=1}^{n} -4C_{D,i}A_i\hat{\boldsymbol{n}}_i{}^T\frac{\partial}{\partial \sigma_p}([\boldsymbol{\sigma}_p\times][BN(\boldsymbol{\sigma}_r)]\hat{\boldsymbol{v}}) \tag{3.29}$$

Here, the properties of the cross product matrix are exploited to simplify the linearization. For arbitrary vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ and for an arbitrary matrix $[Z]$, the following properties hold:

$$[\boldsymbol{a}\times]\boldsymbol{b} = -[\boldsymbol{b}\times]\boldsymbol{a} \tag{3.30}$$

$$\frac{\partial}{\partial \boldsymbol{x}}[Z]\boldsymbol{x} = [Z] \tag{3.31}$$

To simplify the notation, the intermediate vector $\hat{\boldsymbol{q}} = [BN(\boldsymbol{\sigma}_r)]\hat{\boldsymbol{v}}$ is introduced. Applying these properties to the derivatives in Eq. 3.29 yields:

$$\frac{\partial \beta_d}{\partial \boldsymbol{\sigma}_p} = \frac{1}{m}\sum_{i=1}^{n} 4C_{D,i}A_i\hat{\boldsymbol{n}}_i^T[\hat{\boldsymbol{q}}\times] \tag{3.32}$$

which is entirely defined by the spacecraft mass, geometry, and reference attitude.

3.3 Controllability Analysis and Controller Implementation

3.3.1 Controllability Analysis

Equations 3.18a-3.18c and Eq. 3.24a-3.24c define a linear set of equations of motion for a deputy-chief pair with the deputy attitude as an input. While these equations of motion are general with regards to deputy and chief geometry, a restricted case dealing with identical deputy/chief geometries is used to demonstrate the controllability properties of this system. This can be considered to represent multiple use cases. One example is maneuvering to a predefined reference orbit and attitude during formation constitution (i.e., matching position and velocity with a fictitious chief). For rendezvous with a fictitious chief, it is desirable for the fictional chief orbit to have identical drag parameters to the real deputy. In any of these cases, the system aim is to drive both the relative position and relative velocity states to zero.

The linearized equations derived in Section 3.2.6 enables the use of straightforward linear analysis tools to demonstrate controllability. A classic approach to controllability for linear systems uses the controllability matrix, $[O]$, which is formed as[64]:

$$[O] = \begin{bmatrix} [B] & [A][B] & [A]^2[B] & ... & [A]^{n-1}[B] \end{bmatrix} \tag{3.33}$$

where n is the dimension of the state space. The column and null spaces of $[O]$ form bases for the controllable and uncontrollable subspaces for the system, respectively. Examining Equations 3.18a-3.18c shows that in-plane dynamics are coupled, but the out-of-plane z dynamics are independent. This suggests that the control effects matrix defined by Eq. 3.25 will allow for the control of both the x and y states and their derivatives.

Due to the symbolic complexity of these expressions, several numerical examples are provided to demonstrate the controllability properties of the linearized system. A reference system consisting of a single flat plate with dimensions, drag coefficient, and mass based upon those of a 3U cubesat with a three-meter by three-meter drag sail were used to numerically evaluate $[A]$ and $[B]$ for the purposes of forming $[O]$. The specific values used for these properties are listed in Table 3.1. Orbital elements for both the chief and deputy are given in Table 4.2. The matrix rank and QR factorization are computed using Numpy's `linalg` library.

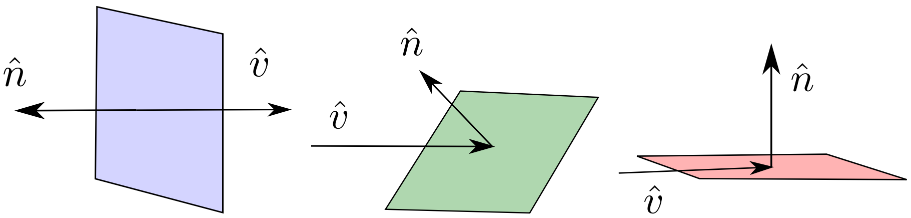
Table 3.1: Spacecraft geometric parameters used for numerical controllability analysis.

Parameter	Chief Value
P_i	$2.7346\times10^{-14}\ \frac{\text{kg}}{\text{m}^3}$
A_i	$9\ \text{m}^2$
m_i	$6\ \text{kg}$
$\hat{\boldsymbol{n}}_i$	$\begin{bmatrix}0 & 1 & 0\end{bmatrix}^T$
$C_{d,i}$	2.2

Three cases are examined: one in which the reference attitude is zero, representing a facet face-on into the flow; one in which the reference attitude is 90°, representing a facet edge-on into the flow; and one in which the reference attitude is equivalent to a 45° rotation about the HCW $\hat{\boldsymbol{h}}_h$ axis, representing an intermediate drag configuration. These configurations are visualized in Fig. 3.2. The rank of $[O]$ for each configuration, along with the controllable eigenvectors, are listed in Table 4.1.

Table 3.2: Controllability analysis results.

Bank Angle	`Rank`($[O]$)	Stabilizable State-Space Eigenvectors
0°	0	$[N/A]$
45°	4	$\begin{bmatrix}\hat{x} & \hat{y} & \hat{\dot{x}} & \hat{\dot{y}}\end{bmatrix}$
90°	4	$\begin{bmatrix}\hat{x} & \hat{y} & \hat{\dot{x}} & \hat{\dot{y}}\end{bmatrix}$



(a) Face-On Case (b) Banked Case (c) Edge-On Case

Figure 3.2: Visualization of face-on, intermediate, and edge-on attitude configurations for a single facet.

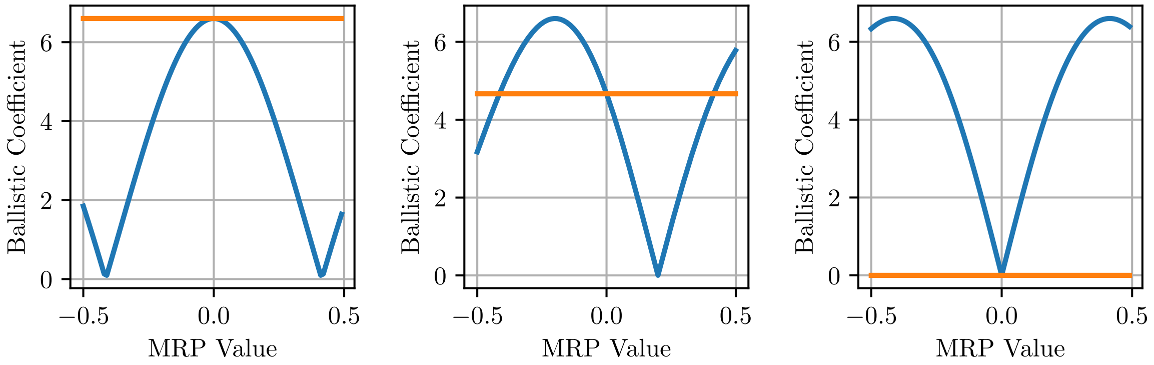
This analysis reveals multiple phenomenon relating to the system's controllability. First, the selected reference attitude can restore or prevent controllability. This is sensible when considering the nature of the small-attitude assumption as it relates to the area projection term. This dependence is more explicit when considering the area projection in terms of a single principle attitude angle θ, in which case the projection can be rewritten as:

$$\hat{\boldsymbol{n}}^T \hat{\boldsymbol{v}} = \cos(\theta) \tag{3.34}$$

Evaluating the cosine term about a reference angle θ_r and considering a small perturbation angle, θ_p, yields:

$$\hat{\boldsymbol{n}}^T \hat{\boldsymbol{v}} = \cos(\theta_r + \theta_p) \tag{3.35}$$

If $\theta_r = 0$, corresponding to the face-on case, the effect of the perturbation angle drops out, explaining the loss of linear controllability. However, the edge-on case also presents an issue. For a physical plate, rotation in either direction represents an increase in the projected area. If the chief is assumed to be uncooperative, this means that the effective control input can never produce a negative acceleration, and as such a linear model cannot effectively approximate its behavior. It is only in the banked case that control authority is provided about the described equilibrium condition in which the deputy and chief ballistic coefficients are equal. This is illustrated in Fig. 3.3, which demonstrates the variation of a single-plate deputy drag coefficient with attitude for reference attitude configurations facing into the flow, banked into the flow, and edge-on into the flow, and respectively. From this figure, it is apparent that the controllability of the system for $\theta_r = 90°$ is an artifact of the manner in which the system is linearized, due to the discontinuity at the facet edge.



(a) Face-On Case

(b) Banked Case

(c) Edge-On Case

Figure 3.3: Deputy (Blue) and Chief (Orange) ballistic coefficients using reference numbers from Table 3.1. Results shown for face-on, banked, and edge-on reference attitudes.

This analysis provides a framework to understand admissible conditions and geometries for differential drag control inside and outside the linear regime. Differential drag formation flight requires that the deputy-chief pair be able to achieve both positive and negative relative accelerations from drag. In the attitude-only non-cooperative rendezvous case considered here, this requires that the deputy geometry and attitude allow it to both increase and decrease its drag profile relative to the chief.

Using the intermediate case angle as the reference angle, this analysis reveals that the in-plane states and velocities are controllable from attitude-driven drag alone. This is consistent with both the well-known in-plane coupling expressed by the Hill-Clohessy Wiltshire equations for linear relative motion and the planar nature of drag forces on spacecraft. Additionally, these results agree with results found in the literature for this class of control [9]. In comparison to differential drag studies which utilize differential mean orbit elements (such as Reference [16]) and show that relative mean elements corresponding to component of in-plane motion are uncontrollable, these results can be considered as using short-period behavior (which is lost in the averaging analysis) to gain controllability in-plane at the expense of requiring small separation distances and maneuver time periods to remain in the linear regime.

3.3.2 Linear Control Performance

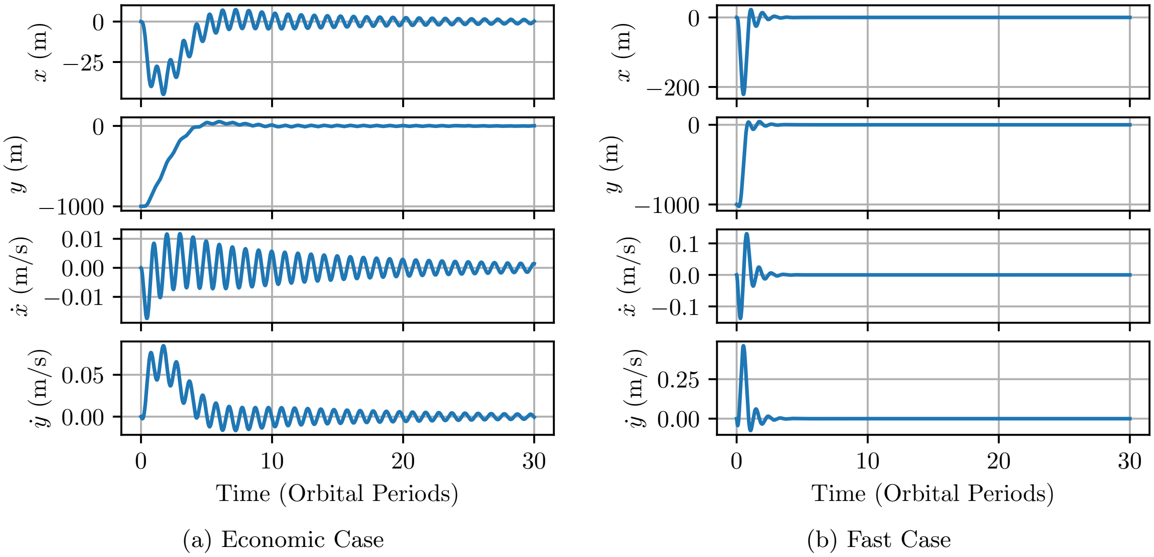
3.3.2.1 Single-Facet Control

Per the previous section, a subspace of the linearized system has been demonstrated to be linearly controllable. To this end, a straightforward linear control law based on LQR was developed and implemented for the sample linear system based on Table 3.1. Results are provided for two selected control objective weights — one which emphasized fast state performance ("Fast Case"), and one which emphasized economical use of the control input (the "Economic Case"). Both the state gains $[Q]$ and the control gains $[R]$ are selected to be diagonal with elements of the magnitude stated in Table 3.3. The latter can be analogously considered as minimizing the variance from the desired reference attitude. For demonstration purposes, the control objective is to drive the deputy spacecraft to the chief position and velocity.

With respect to the linearized system, both controls are found to be stabilizing, resulting in the state trajectories found in Figures 3.5 and 3.4. The commanded attitude MRPs are displayed in Fig. 3.6. Notably, the commanded attitudes in the fast-state case vary far outside the domain

Table 3.3: Selected control gains for Economic and Fast Case.

Control Design Variable	Economic Case Values	Fast Case Values
Q	0.1	1
R	1e7	1e4
dt	$5s$	$5s$



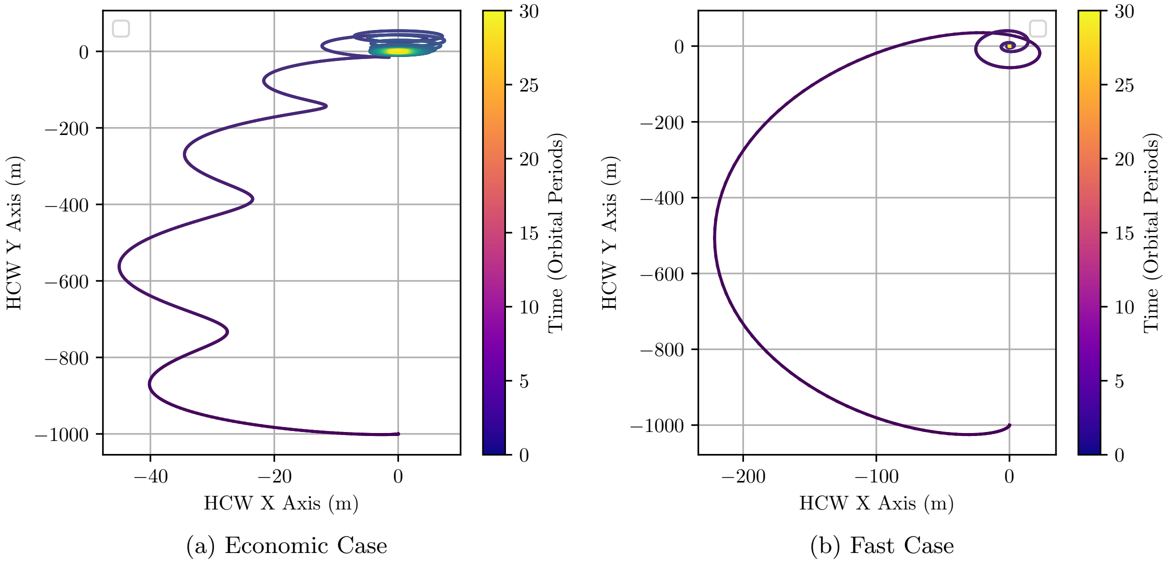
(a) Economic Case

(b) Fast Case

Figure 3.4: Deputy relative position and velocity state trajectories in the chief Hill frame under attitude-driven control simulated on the linear system

in which MRP switching would typically be utilized ($\sigma^2 = 1$), suggesting that the system violates the small angle assumption critical to the linearization. Additionally, it is noted that the fast-state case displays substantial oscillation even after notionally reaching the reference states, though it is apparent from Fig. 3.5a that its behavior is convergent towards the origin. Hill-Clohessy-Wiltshire dynamics are well-known to exhibit oscillatory modes in the form of a 2-by-1 ellipse in the planar states; as a result of the small control authority afforded by drag, it is difficult to completely eliminate this behavior.

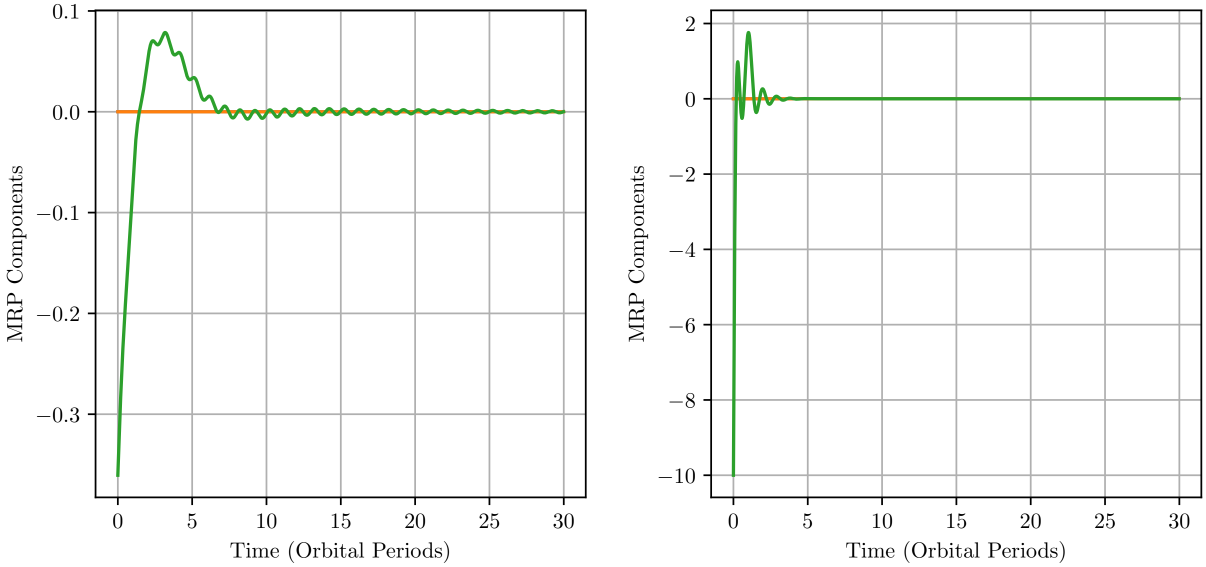
To provide further validation of this approach, the linear controller was implemented on a system following the full nonlinear equations of motion found in Eq. 3.15 under the same initial conditions and parameters used to generate the linearized system (i.e, those found in Tables 3.1-



(a) Economic Case

(b) Fast Case

Figure 3.5: HCW x and y state evolution under linear dynamics with LQR-derived controller.



(a) Economic Case

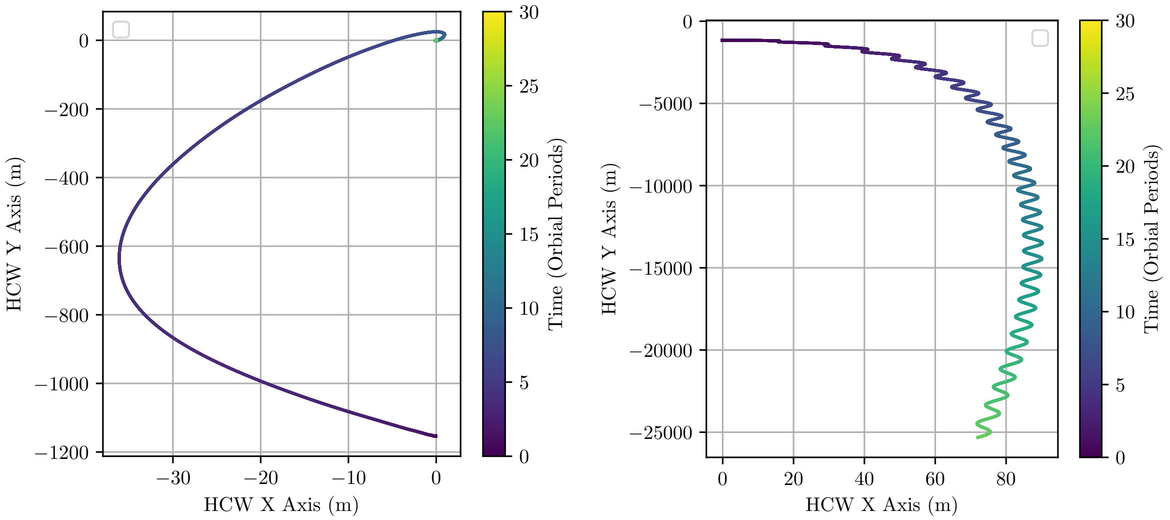
(b) Fast Case

Figure 3.6: Flow-relative attitude MRP component generated by the LQR-derived controller using linear dynamics under different control weights. Green represents the perturbed MRP value, while orange represents the reference value.

4.2). The selected scenario represents a slot-hoping maneuver, in which a spacecraft maneuvers to a selected reference location and attitude ahead of its current position on orbit. A small inclination difference is included to demonstrate the control's lack of influence on out-of-plane motion as expected. The results of these simulations under both control strategies can be found in Figures 3.7-3.8. Attitude trajectories for these cases are shown in Fig. 3.9.

Table 3.4: Orbital elements for both the deputy and chief spacecraft.

Orbital Element	Chief Value	Deputy Value
a	230km + r_{eq}	230km + r_{eq}
i	45°	45.01°
e	0	0
Ω	20.0°	20.0°
ω	30.0°	30.0 °
M_0	20.0°	19.99°

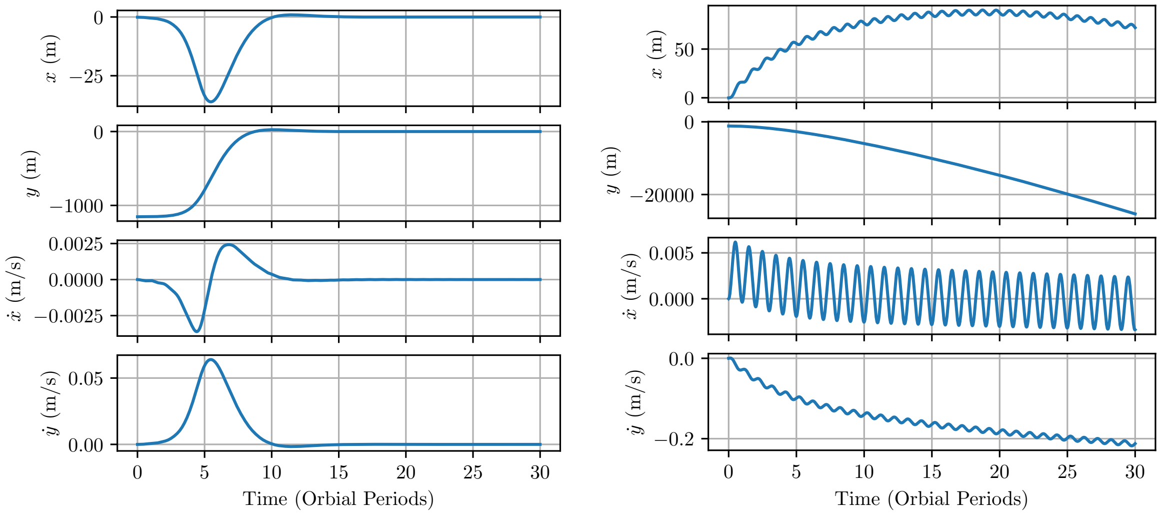


(a) Economic Case

(b) Fast Case

Figure 3.7: HCW x and y state evolution under the LQR-derived controller on the assumed nonlinear system

Notably, the LQR controller derived for the linear system provides similar performance for the nonlinear system using Economic Case's control gains. This both validates the linearizations used to derive the LQR controller and demonstrates the applicability of the linearized system to the "real" problem at hand. However, the results of the Fast Case, which display state divergence from the reference, demonstrate limitations of the linearization approach. In the Fast Case, the relatively large elements of $[B]$ cause the controller to request large attitudes outside the linear

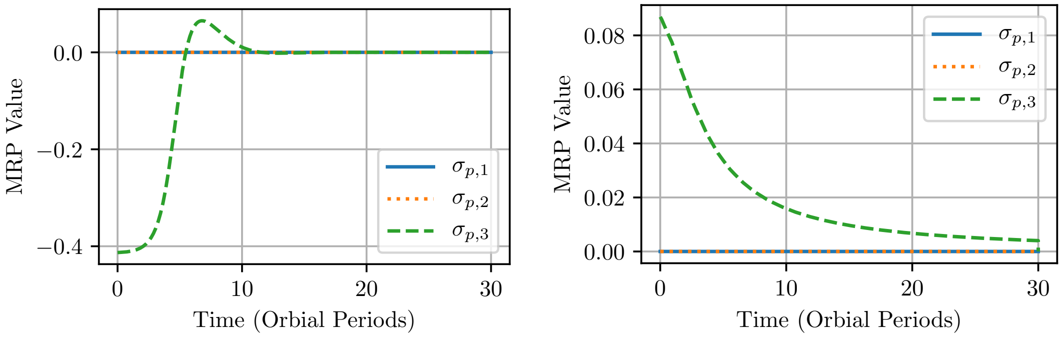


(a) Economic Case

(b) Fast Case

Figure 3.8: Deputy relative position and velocity state trajectories in the chief Hill frame under attitude-driven control

regime, a behavior shown in the linear system through Fig. 3.6b.



(a) Economic Case

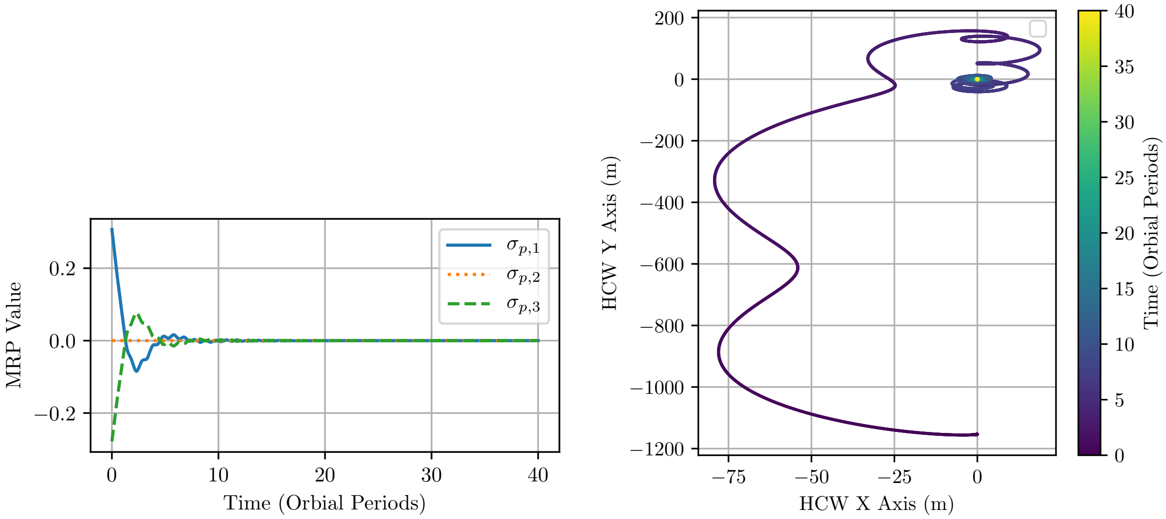
(b) Fast Case

Figure 3.9: Flow-relative attitude MRP component generated by the LQR-derived controller. Requested MRPs are transformed to the unit set.

These results demonstrate the need for large penalties for control use in the linearized attitude-driven case. Nonlinearities present in the assumed input – the spacecraft's attitude – are the dominant driver of non-convergence for the controlled system. These results were used as guidelines for the development of additional simulations to address other aspects of the system.

3.3.2.2 Multi-Faceted Performance

An expected benefit of this approach is the ability to add additional facets to the dynamic model to further approximate the geometry of a spacecraft. To demonstrate this advantage, the slot-hopping scenario described in Table 4.2 was repeated with a cuboid spacecraft representing a 3U cubesat flying obliquely into the flow. The resulting attitude and Hill-frame trajectory are shown in Fig. 3.10. The more complex, 3D geometry represented by the collection of facets results in additional nonzero terms along the row corresponding to $\dot{y}$ in the control matrix $[B]$; as such, the controller makes additional use of the corresponding component of the attitude MRP, resulting in a similar overall control magnitude but smaller axis-wise components to the single-panel case.



(a) Commanded MRP trajectories

(b) In-plane trajectories in the Hill frame

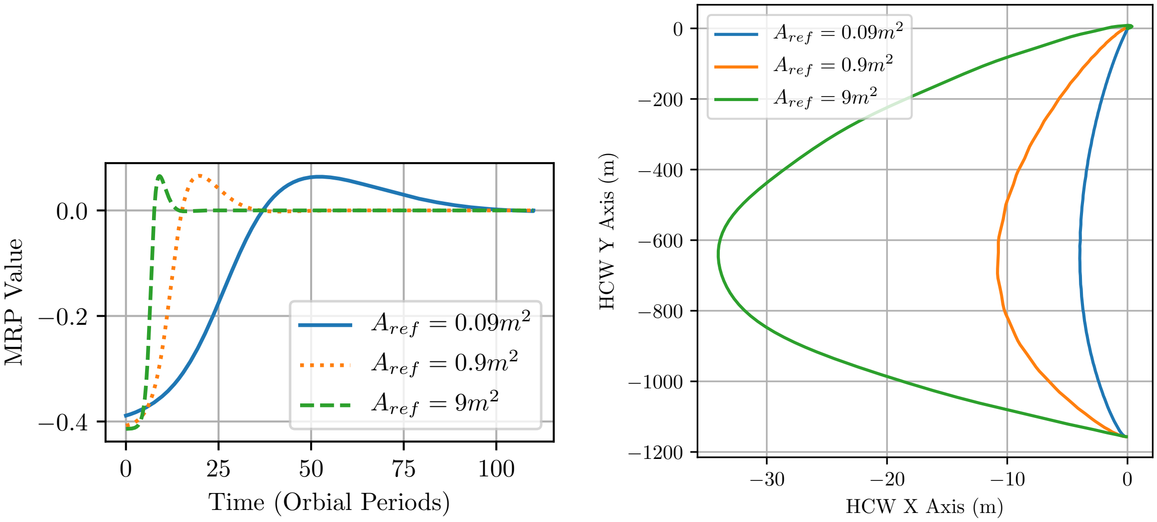
Figure 3.10: Control performance for a cubesat represented by 3 facets.

3.3.2.3 Ballistic Coefficient Variation

Large differences in maximum and minimum ballistic coefficient can produce large relative accelerations and therefore provide better relative motion control performance than small ones. To this end, the relative controllability of the scenario presented in Section 3.3.2.1 is studied under varied plate areas (of which ballistic coefficients are a linear function) with all other factors held constant.

Table 3.5: Low, Nominal, and High plate areas used to test the performance impact of ballistic coefficients

Case	Area
Low Area	0.09m^2
Nominal Area	0.9m^2
Large Area	9 m^2



(a) Commanded MRP trajectories

(b) In-plane trajectories in the Hill frame

Figure 3.11: Control performance for various area and resulting β values

Fig. 3.11 shows the attitude and in-plane trajectories for the nominal, high, and low-area cases. These results show that control convergence is maintained even with order-of-magnitude differences in ballistic coefficient. As expected, spacecraft with smaller facet areas require more time to converge than spacecraft with larger facets, reflecting the impact of area on the ballistic coefficient. Notably, the control uses larger deviations from the reference state for control when a larger panel area is available; this is a result of the in-plane coupling predicted by the linearized dynamics, as larger y-direction velocities would necessarily produce larger x-direction velocities and therefore deviations.

3.3.3 Performance under Mis-Modeled Dynamics

3.3.3.1 Impact of Mis-Modeled Atmosphere

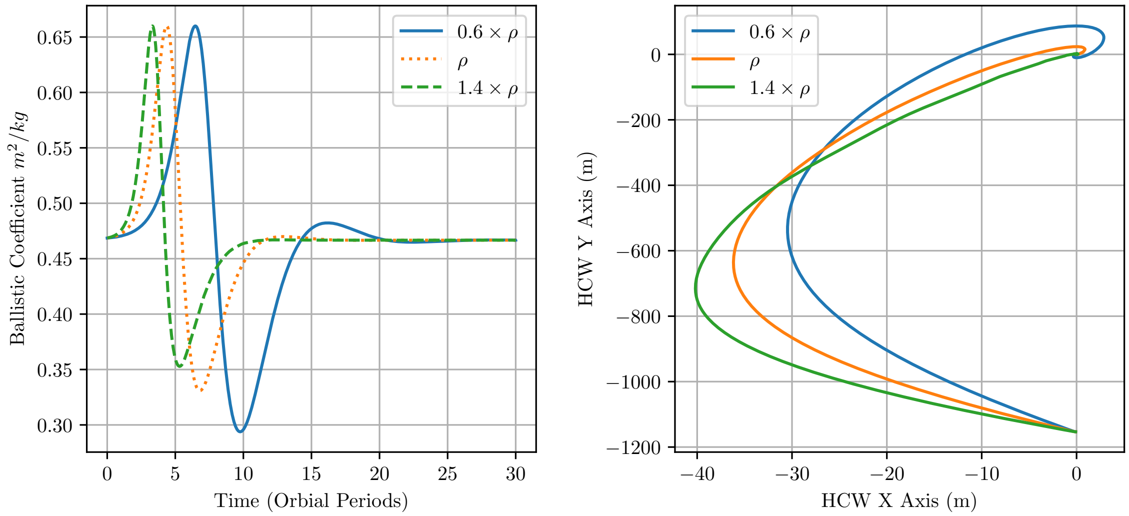
Even about the earth, neutral atmospheric density is notoriously difficult to predict. The structure of this approach to linear control necessitates the prediction of atmospheric density to formulate the linearized models which depend on estimates of a base atmospheric density. To identify the affect of mis-modeled atmospheric density on the controller's effectiveness, the same scenario used in Section 3.3.1 was run with the real density offset from the density used to construct the controller by 40% ($0.6\rho_0$ and $1.4\rho_0$, respectively).

While the model dynamics are linearly dependent on the exponentially-varying atmosphere, a degree of robustness is maintained by the assumption that the relative spacecraft-reference dynamics occur for nearby orbits (∼10 kilometers). Under the assumed exponential atmospheric model, this distance falls within one atmospheric scale-height of the reference orbit, which is beneath the point at which higher-order terms in the series expansion of an exponential atmospheric model become substantial. As shown in Fig. 3.12, variation in atmospheric density from the design value simply changes the rate of convergence of the controller, resulting in under or overshoot.

3.3.3.2 Convergence with Un-Modeled J_2

Accelerations from J2 are, alongside atmospheric drag, the dominant perturbations for spacecraft in LEO. While J2 is not included in the dynamical model used to construct the control scheme, the presence of feedback control suggests that the system may still be stable under the presence of un-modeled dynamics such as J2. To address this concern, the scenario defined by Table 4.2 was redone with an increased initial separation (∼10km of along-track separation) and the addition of un-modeled J2 accelerations using the following inertial expression:

$$^{\mathcal{N}}\boldsymbol{a}_{J2} = -\frac{3}{2}J_2\frac{\mu}{r^2}\frac{r_{\text{eq}}}{r}^2\begin{pmatrix}\left(1-5\left(\frac{z}{r}^2\right)\right)\frac{x}{r}\\\left(1-5\left(\frac{z}{r}^2\right)\right)\frac{y}{r}\\\left(3-5\left(\frac{z}{r}^2\right)\right)\frac{z}{r}\end{pmatrix} \tag{3.36}$$



(a) Ballistic coefficient variation. The red line represents the “reference” ballistic coefficient.

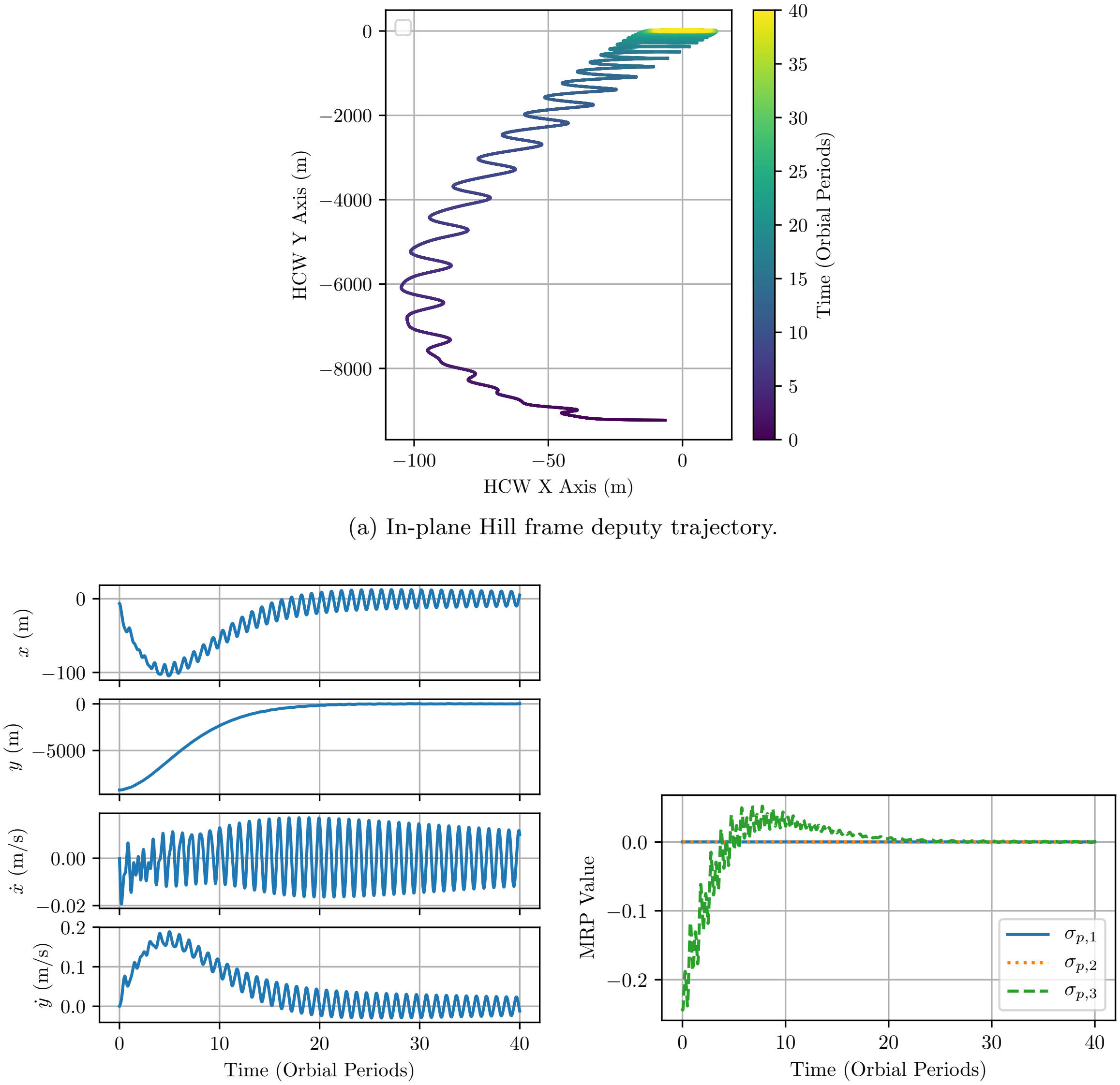
(b) In-plane HCW trajectories corresponding to different density cases.

Figure 3.12: Control performance under mis-modeled atmospheric density. Convergence is achieved with both higher and lower variation.

Importantly, the differential disturbance from J_2 goes to zero as the relative position goes to zero, and as such the inclusion of disturbances from J_2 will not affect the equilibria of the system.

The results of this analysis are shown in Fig. 3.13, which demonstrates that the resulting controller behavior drives the spacecraft into a neighborhood about the target position. Compared to Fig. 3.7a, more periodic oscillations are seen. These oscillations are consistent with the comparison of osculating-vs-mean controllability described in Section 3.3.1; the resulting oscillations are partially the result of attempting to control short-period variations caused by J_2.

From the demonstrated scenario, the controller still converges to a stable position near the designated position in the controlled axes in a time comparable to the unperturbed case; however, oscillations that are periodic with the orbit period resulting from J2 are clearly present early in the trajectory. Regardless, this suggests that the described attitude-only approach has merit for the control of realistically-sized spacecraft in LEO.



(a) In-plane Hill frame deputy trajectory.

(b) Deputy in-plane state trajectories.

(c) Deputy attitude components over time.

Figure 3.13: Control performance under un-modeled J2 dynamics.

3.4 Conclusion

A novel framework for the consideration of differential-drag formation flight as a **continuous linear formation control problem** has been presented and derived, meeting the first objective of this dissertation. For spacecraft pairs with intermediate drag geometry-attitude configurations, linear controllability is possible from small attitude motion alone without the assumption of additional drag surfaces. These results provide an alternative perspective on the controllability and stability of formation flight under the consideration of atmospheric drag. In the described lienarization, simulations show that the attitude linearization is a primary constraint for the design of controllers. Despite this, controllers based on this formulation of the differential drag formation flight problem show convergence in the presence of un-modeled variations in atmosphere and J_2 accelerations while providing physical insight into the problem structure. Work presented in this chapter has been published in the AIAA Journal of Guidance, Dynamics and Controls [20] and presented at the AIAA SciTech conference [65].

Chapter 4

Linear Desensitized Optimal Control for Robust Differential Drag

4.1 Introduction

Constellation- and formation-flight of spacecraft requires substantial on-board control effort and could benefit from the use of environmental forces, such as atmospheric drag, to conduct maneuvers in place of or as a supplement to propellant-consuming thrusters. At present, the unpredictable nature of atmospheric drag due to the difficulty of forecasting conditions in the upper atmosphere has restricted the precision and utility of drag-based maneuvering. This work aims to analyze the sensitivity of trajectories in differential drag formation flight to variations in density, including in the presence of control strategies meant to mitigate this variability.

The variability of exoatmospheric density in near-Earth orbits has led to a variety of theoretical and empirical studies to improve density predictions. Neutral atmospheric density in low-Earth orbit (LEO) can vary by orders of magnitude depending on solar forcing, geomagnetic activity, and diurnal variation [14, 66]. This alone presents a substantial challenge to using differential drag for regular space operations, and is further compounded by the limited progress in predictive modeling for atmospheric density [8]. While higher accuracy models are potentially possible by incorporating live density estimates–for example, by measuring orbit variations in tracked orbital debris, as shown by [15] – these models rely on the availability of high-accuracy tracking data and spacecraft drag models, which are not widely available. This limitation severely constrains the types of missions and applications for differential-drag control to those that can tolerate substantial uncertainty in control accuracy, settling time, and other performance measures.

Desensitized optimal control is a family of related techniques for minimizing the dependence of a given trajectory or control strategy to selected parameters. Originating in the 1960s with the sensitivity-vector approach described by Kahne[67], desensitized optimal control is successfully applied in other fields, such as optimal landing guidance [68] and the control of multi-body structures [69]. Seywald [70] presents a method for considering state sensitivities to system parameters by constructing a state transition matrix for those states and adding cost penalties for exciting those states, allowing the desensitized optimal control problem to be applied to general nonlinear problems. Makkapati [71] presents an alternative formulation using sensitivity functions, which are both similar to the original approach developed by Kahne and which offer improved computational efficiency versus the sensitivity matrix approach. This work develops an extended derivation of the sensitivity-vector approach for linear systems that can be considered a restricted case of Makkapati's sensitivity function control.

This chapter is arranged as follows. First, a brief overview of the drag-perturbed relative dynamics model and attitude effect is reviewed. Next, the theory of desensitized optimal control is reviewed and extended to consider sensitivities arising directly from control inputs, with an additional comment on the controllability of such a system and the relationship between sensitivities and observability for the states to be desensitized against. To demonstrate this approach, a simple linear mass-spring-damper is analyzed. This extended methodology is then used to analyze the sensitivity of the differential drag formation flight scenario. Following this analysis, strategies for desensitized optimal control are implemented and compared on the differential drag formation flight system under a range of atmospheric densities for short and long baseline maneuver distances.

4.2 Problem Statement

4.2.1 Linearized Differential Drag via Differential Attitude Dynamics

Prior work demonstrates that, given non-uniform geometries, attitude control alone is sufficient to achieve controllability between two spacecraft using differential drag [19, 72]. The derived attitude-dependent linearized equations of motion take the form of the Hill-Clohessy-Wiltshire equations plus several drag terms dependent on the spacecraft ballistic coefficient as derived by Silva [63] and refined in Reference [72]:

$$\ddot{x} = 2\dot{y}n + 3n^2x - \frac{1}{2}\beta_D\rho_D n r_C \dot{x} \tag{4.1a}$$

$$\ddot{y} = -2\dot{x}n - n^2r_C^2\frac{1}{2}(\beta_C\rho_C - \beta_D\rho_D) - \beta_D\rho_D n r_C \dot{y} \tag{4.1b}$$

$$\ddot{z} = -zn^2 - \frac{1}{2}(\beta_D\rho_D r_C n)\dot{z} \tag{4.1c}$$

where x,y, and z represent the relative cartesian Hill-frame position components, β_C and β_D represent the chief and deputy ballistic coefficients respectively, ρ_C represents the density at the chief location, and ρ_D represents the density at the deputy location. Note that Eqs. (4.1) neglect the kinematic effects of atmospheric drag, which are sub-dominant in LEO. The sensitivity of these equations with respect to a small relative variation in the deputy spacecraft's attitude represented as a Modified Rodriguez Parameter (MRP), $\boldsymbol{\sigma}_p$, is taken from Reference [72] as:

$$\frac{\partial\ddot{x}}{\partial\boldsymbol{\sigma}_p} = -\frac{1}{2}\rho_D n r_C \dot{x}_0 \frac{\partial\beta_D}{\partial\boldsymbol{\sigma}_p} \tag{4.2a}$$

$$\frac{\partial\ddot{y}}{\partial\boldsymbol{\sigma}_p} = (\frac{1}{2}n^2r_C^2\rho_D - \rho_D n r_C \dot{y}_0)\frac{\partial\beta_D}{\partial\boldsymbol{\sigma}_p} \tag{4.2b}$$

$$\frac{\partial\ddot{z}}{\partial\boldsymbol{\sigma}_p} = -\frac{1}{2}(\rho_D r_C n)\dot{z}_0 \frac{\partial\beta_D}{\partial\boldsymbol{\sigma}_p} \tag{4.2c}$$

The derivative of the ballistic coefficient with respect to attitude defined in MRP components for a faceted spacecraft with constant drag coefficients across each facet is taken from Reference [19] to be

$$\frac{\partial\beta_D}{\partial\boldsymbol{\sigma}_p} = \frac{1}{m_D}\sum_{i=1}^{n} -4C_{d,i}A_i\hat{\boldsymbol{n}}_i{}^T\frac{\partial}{\partial\sigma_p}([\boldsymbol{\sigma}_p\times][BN(\boldsymbol{\sigma}_r)]\hat{\boldsymbol{v}}_C) \tag{4.3}$$

where $C_{d,i}$ represents the drag coefficient of facet i, A_i represents the area of facet i, $\hat{\boldsymbol{n}}_i$ represents the unit vector from facet i expressed in the spacecraft body frame, $[BN(\boldsymbol{\sigma}_r)]$ represents the direction cosines matrix that maps from the inertial frame to the body reference frame, and $\hat{v}$ represents the inertial velocity heading.

While Eqs. 4.1-4.3 are linear in both the relative states and the linearized MRPs, both the state dynamics and the control effects contain an implicit dependence on the time-varying neutral density, ρ_D, and the chief radius and mean motion, r_C and n respectively. These parameters vary as drag acts to reduce the orbital radius of the chief and deputy alike, representing a source of modeling error within the dynamics.

4.2.2 Atmospheric Models

For the purposes of this work and without loss of generality, a simple exponential atmospheric model is used for the control design and analysis due to its analytical form. Exponential atmospheric models have the following form:

$$\rho(\boldsymbol{r}) = \rho_0 \mathrm{e}^{-\frac{|\boldsymbol{r}-\boldsymbol{r}_{\mathrm{ref}}|}{h_s}} \tag{4.4}$$

where $\boldsymbol{r}$ is the inertial, Earth-centric spacecraft position, $\boldsymbol{r}_{\mathrm{ref}}$ is a reference altitude, ρ_0 is the atmospheric density at the reference altitude, and h_s is the scale height of the atmosphere. In general, these properties are only coarsely known, and can vary substantially with changes in geomagnetic or solar weather; in addition, simple exponential atmospheric models misrepresent the actual atmospheric density at LEO altitudes. To remedy this, the NRLMSISE-00 model [1] was evaluated with historical space environment conditions at January 1st, 2000 across a range of altitudes. These run results were used to evaluate the base density ρ_0 at the chief spacecraft's altitude r_C to define the local density variation for numerical studies.

4.3 Sensitivity Analysis and Mitigation

4.3.1 Sensitivity Dynamics

Desensitized optimal control is a type of optimal control that attempts to generate control solutions or trajectories under the presence of perturbations in non-state parameters. The methodology of Kahne [67] is briefly summarized here for reference, with an additional extension to sensitivities in the control matrix B.

[1] https://ccmc.gsfc.nasa.gov/modelweb/models/nrlmsise00.php

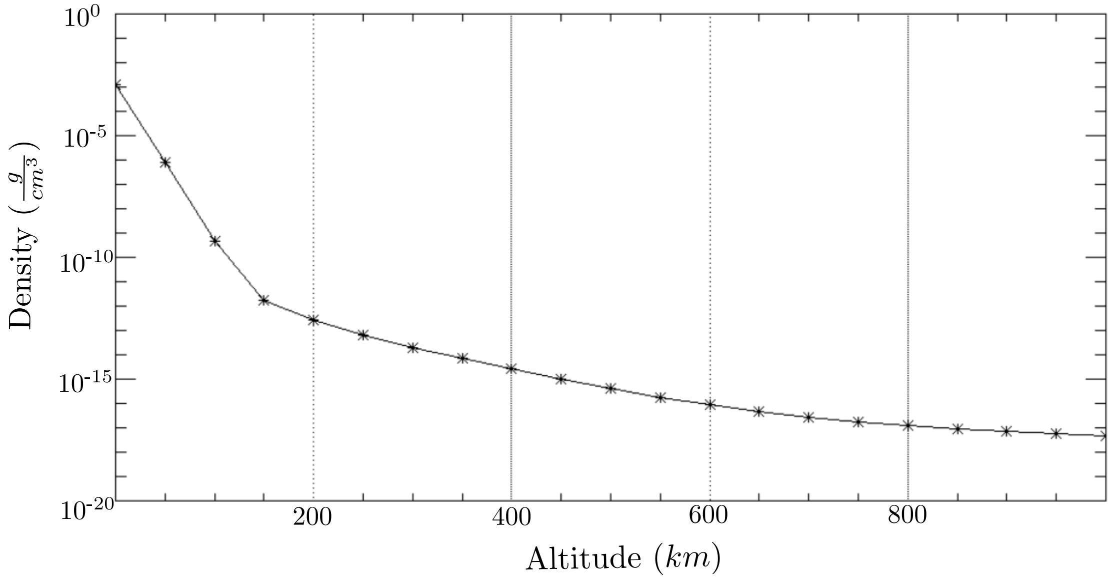


Figure 4.1: NRLMSISE-00 profile of mass density versus altitude for January 1st, 2000.

The sensitivities of a linear system $\dot{\boldsymbol{x}} = A\boldsymbol{x} + B\boldsymbol{u}$ are best understood as the gradient of the system's state dynamics with respect to a parameter α:

$$\dot{\boldsymbol{s}} = \frac{\partial \dot{\boldsymbol{x}}}{\partial \alpha} = A\boldsymbol{s} + C\boldsymbol{x} + D\boldsymbol{u}, \;\; \boldsymbol{s}(0) = \boldsymbol{0} \tag{4.5}$$

where A represents the linear state dynamics matrix, C represents the state sensitivity matrix defined by $C_{ij} = \frac{\partial A_{ij}}{\partial \alpha}$, and D represents the control sensitivity matrix defined as $D_{ij} = \frac{\partial B_{ij}}{\partial \alpha}$. When B does not depend on α, this expression reduces to the sensitivity dynamics produced by Reference [67].

Using this definition of the sensitivities, the objective of minimizing the overall system sensitivity is defined by the sensitivity cost

$$J_s = \frac{1}{2}\int_{t_0}^{t_f} \boldsymbol{s}(t)^T E \boldsymbol{s}(t) \mathrm{d}t \tag{4.6}$$

which is readily combined with the classical LQR cost function to yield

$$J = \frac{1}{2}\boldsymbol{x}_f^T N \boldsymbol{x}_f + \frac{1}{2}\int_{t_0}^{t_f} \Big(\boldsymbol{x}(t)^T Q \boldsymbol{x}(t) + \boldsymbol{u}(t)^T R \boldsymbol{u}(t) + \boldsymbol{s}(t)^T E \boldsymbol{s}(t)\Big) \mathrm{d}t \tag{4.7}$$

From this, the formal statement of the desensitized optimal control problem is stated as:

$$\begin{aligned} \underset{u}{\text{minimize}} \quad & J = \frac{1}{2}\boldsymbol{x}_f^T N \boldsymbol{x}_f + \frac{1}{2}\int_{t_0}^{t_f} \Big(\boldsymbol{x}(t)^T Q \boldsymbol{x}(t) + \boldsymbol{u}(t)^T R \boldsymbol{u}(t) + \boldsymbol{s}(t)^T E \boldsymbol{s}(t)\Big) \mathrm{d}t \\ \text{subject to} \quad & \dot{\boldsymbol{x}} = A\boldsymbol{x}(t) + B\boldsymbol{u}(t), \ \dot{\boldsymbol{s}} = A\boldsymbol{s}(t) + C\boldsymbol{x}(t) + D\boldsymbol{u}(t) \end{aligned}$$

The Hamiltonian for this problem is written using a separate set of co-states for each constraint, denoted $\boldsymbol{p}$ for the state dynamic constraint and $\boldsymbol{\lambda}$ for the sensitivity dynamics:

$$H = \frac{1}{2}\Big(\boldsymbol{x}(t)^T Q \boldsymbol{x}(t) + \boldsymbol{u}(t)^T R \boldsymbol{u}(t) + \boldsymbol{s}(t)^T E \boldsymbol{s}(t)\Big) + \boldsymbol{p}(t)^T (A\boldsymbol{x}(t) + B\boldsymbol{u}(t)) + \boldsymbol{\lambda}^T (A\boldsymbol{s}(t) + C\boldsymbol{x}(t) + D\boldsymbol{u}(t)) \tag{4.8}$$

The canonical equations of this system are:

$$\dot{\boldsymbol{x}} = \frac{\partial H}{\partial \boldsymbol{p}} = A\boldsymbol{x} + B\boldsymbol{u} \tag{4.9a}$$

$$\dot{\boldsymbol{s}} = \frac{\partial H}{\partial \boldsymbol{\lambda}} = A\boldsymbol{s} + C\boldsymbol{x} + D\boldsymbol{u} \tag{4.9b}$$

$$\dot{\boldsymbol{p}} = -\frac{\partial H}{\partial \boldsymbol{x}} = -Q\boldsymbol{x} - A^T\boldsymbol{p} - C^T\boldsymbol{\lambda} \tag{4.9c}$$

$$\dot{\boldsymbol{\lambda}} = -\frac{\partial H}{\partial \boldsymbol{s}} = -E\boldsymbol{s} - A^T\boldsymbol{\lambda} \tag{4.9d}$$

The control parameter, $\boldsymbol{u}$, is solved for using the additional property

$$\frac{\partial H}{\partial \boldsymbol{u}} = \boldsymbol{0} \tag{4.10}$$

which yields

$$0 = R\boldsymbol{u} + B^T\boldsymbol{p} + D^T\boldsymbol{\lambda} \tag{4.11}$$

$$\boldsymbol{u} = -R^{-1}(B^T\boldsymbol{p} + D^T\boldsymbol{\lambda}) \tag{4.12}$$

Substituting this back into Eqn. 4.9 and collecting the terms yields the following total system dynamics matrix:

$$\begin{bmatrix} \dot{\boldsymbol{x}} \\ \dot{\boldsymbol{s}} \\ \dot{\boldsymbol{p}} \\ \dot{\boldsymbol{\lambda}} \end{bmatrix} = \begin{bmatrix} A & \boldsymbol{0} & -BR^{-1}B^T & -BR^{-1}D^T \\ C & A & -DR^{-1}B^T & -DR^{-1}D^T \\ -Q & \boldsymbol{0} & -A^T & -C^T \\ \boldsymbol{0} & -E & \boldsymbol{0} & -A^T \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{s} \\ \boldsymbol{p} \\ \boldsymbol{\lambda} \end{bmatrix} \tag{4.13}$$

The initial conditions of this equation are given by:

$$\boldsymbol{x}(0) = \boldsymbol{x}_0 \tag{4.14a}$$

$$\boldsymbol{s}(0) = \boldsymbol{0} \tag{4.14b}$$

$$\boldsymbol{p}(t_f) = F\boldsymbol{x}(t_f) \tag{4.14c}$$

$$\boldsymbol{\lambda}(t_f) = \boldsymbol{0} \tag{4.14d}$$

These equations form a linear system in time whose state transition matrix, Φ, is solved for. The elements of this matrix are written in terms of the super-state, $\boldsymbol{z} = \left[\boldsymbol{x}(t)^\top,\ \boldsymbol{s}(t)^\top\right]^\top$, and the adjoint super-state, $\boldsymbol{\psi}(t) = \left[\boldsymbol{p}(t)^\top, \boldsymbol{\lambda}(t)^\top\right]^\top$:

$$\begin{bmatrix} \boldsymbol{z} \\ \boldsymbol{\psi} \end{bmatrix} = \begin{bmatrix} \phi_{11}(t, t_0) & \phi_{12}(t, t_0) \\ \phi_{21}(t, t_0) & \phi_{22}(t, t_0) \end{bmatrix} \begin{bmatrix} \boldsymbol{z}_0 \\ \boldsymbol{\psi}_0 \end{bmatrix} \tag{4.15}$$

From Kahne [67], this matrix and its submatrices are invertible, allowing us to solve for the dynamics of the costates over time, evaluating at $t = T$ and noting t_0 can be any time t:

$$\boldsymbol{\psi}(t) = [\phi_{22}(T, t) - G\phi_{12}(T, t)]^{-1}[G\phi_{11}(T, t) - \phi_{21}(T, t)]\boldsymbol{z}(t) = K(t)\boldsymbol{z}(t) \tag{4.16}$$

where $K(t)$ denotes the optimal linear feedback gain. Differentiating this equation with respect to time and substituting in the state and co-state dynamics yields a modified version of the Matrix Ricatti equation:

$$\dot{K}(t) + K(t)L(t) + P(t)K(t) - K(t)M(t)K(t) + N(t) = 0 \tag{4.17}$$

$$L(t) = \begin{bmatrix} A & \mathbf{0} \\ C & A \end{bmatrix} \tag{4.18}$$

$$P(t) = \begin{bmatrix} A^T & C^T \\ \mathbf{0} & A^T \end{bmatrix} \tag{4.19}$$

$$M(t) = -\begin{bmatrix} -BR^{-1}B^T & -BR^{-1}D^T \\ -DR^{-1}B^T & -DR^{-1}D^T \end{bmatrix} \tag{4.20}$$

$$N(t) = \begin{bmatrix} Q & \mathbf{0} \\ \mathbf{0} & E \end{bmatrix} \tag{4.21}$$

$$\tag{4.22}$$

The optimal gain matrices are found by integrating this equation backwards in time from the terminal condition $K_{11}(t_f) = F\boldsymbol{x}(t_f)$; in terms of the gain matrix elements, this is rewritten as a set of coupled ordinary differential equations:

$$\begin{aligned} \dot{K}_{11} = & -K_{11}A - A^TK_{11} - K_{12}C - C^TK_{21} + K_{11}BR^{-1}B^TK_{11} + K_{11}BR^{-1}D^TK_{21} \\ & + K_{12}DR^{-1}B^TK_{11} + K_{12}DR^{-1}D^TK_{21} - Q \end{aligned} \tag{4.23a}$$

$$\begin{aligned} \dot{K}_{12} = & -K_{12}A - A^TK_{12} - C^TK_{22} + K_{11}BR^{-1}B^TK_{12} + K_{11}BR^{-1}D^TK_{22} \\ & + K_{12}DR^{-1}B^TK_{12} + K_{12}DR^{-1}D^TK_{22} \end{aligned} \tag{4.23b}$$

$$\dot{K}_{21} = \dot{K}_{12}^T \tag{4.23c}$$

$$\begin{aligned} \dot{K}_{22} = & -K_{22}A - A^TK_{22} + K_{21}BR^{-1}B^TK_{12} \\ & + K_{21}BR^{-1}D^TK_{22} + K_{22}DR^{-1}B^TK_{12} + K_{22}DR^{-1}D^TK_{22} - E \end{aligned} \tag{4.23d}$$

$$\tag{4.23e}$$

Finally, the optimal closed-loop control trajectory is found using

$$\boldsymbol{u}(t) = -R^{-1}B^T\Big(K_{11}(t)\boldsymbol{x}(t) + K_{12}(t)\boldsymbol{s}(t)\Big) - R^{-1}D^T\Big(K_{21}(t)\boldsymbol{x}(t) + K_{22}(t)\boldsymbol{s}(t)\Big) \tag{4.24}$$

which is equivalent to the finite-time LQR control trajectory when $E = \mathbf{0}$ and the control sensitivity matrix D is a zero matrix.

4.3.2 Reachability and Controllability of Sensitivities

A critical consideration for the application of desensitized control to a system is whether the system sensitivities are adequately coupled to the states and control inputs such that they can be affected. Because both the state and sensitivity dynamics are linear and coupled, it is possible to construct an augmented linear system consisting of both the states and their corresponding sensitivities:

$$\begin{bmatrix} \dot{\boldsymbol{x}} \\ \dot{\boldsymbol{s}} \end{bmatrix} = \begin{bmatrix} A & \mathbf{0}_{n\times n} \\ C & A \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{s} \end{bmatrix} + \begin{bmatrix} B \\ D \end{bmatrix} \boldsymbol{u} \tag{4.25}$$

This augmented system is itself a linear system and can therefore be analyzed with standard tools in linear controls. The reachable subspace of the system given the inputs is computed by analyzing the system controllability matrix, $[O]$, for its rank (which reflects the number of controllable eigen-directions). Likewise, a basis for the controllable subspace is found by analyzing the QR decomposition of $[O]$ and examining the first $\texttt{rank(}O\texttt{)}$ columns [64].

4.4 Applications

4.4.1 Illustrative Example: Mass-Spring-Damper with Force Control

The simplest example of a dynamical system with sensitivities to a coarsely-known parameter is a mass-spring-damper system controlled by an external force with a variable mass. The equations of motion for this system assuming a linear spring and damper are given in state-space form as

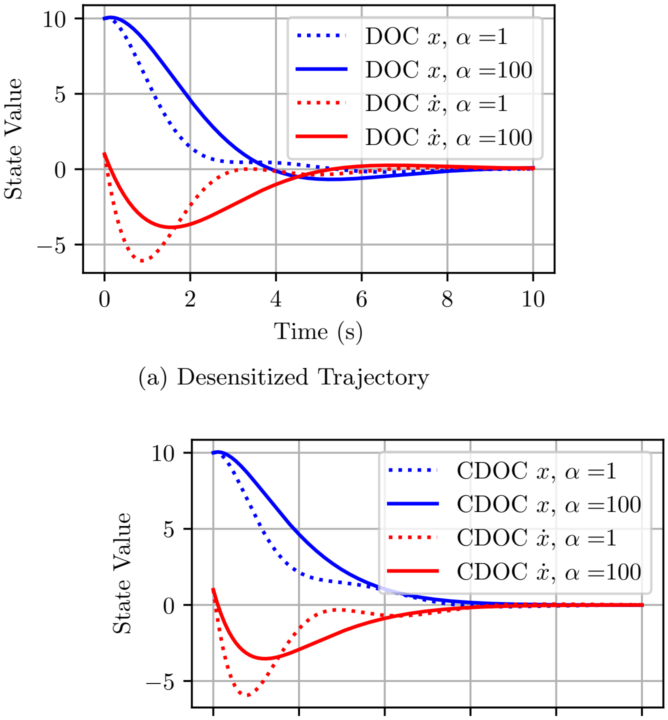
$$A = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{c}{m} \end{bmatrix}, \ \boldsymbol{x} = \begin{bmatrix} r \\ \dot{r} \end{bmatrix} \tag{4.26}$$

$$B = \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix}, \ \boldsymbol{u} = \begin{bmatrix} F \end{bmatrix} \tag{4.27}$$

The sensitivity matrices are given by

$$C = \begin{bmatrix} 0 & 0 \\ \frac{k}{m^2} & \frac{c}{m^2} \end{bmatrix}, \ D = \begin{bmatrix} 0 \\ -\frac{1}{m^2} \end{bmatrix} \tag{4.28}$$

To demonstrate the efficacy of the control-desensitized approach in comparison to prior methods, simulations were run across a range of mass values with 1 kilogram as the reference value. Fig. 4.2 demonstrates how the control-desensitized and desensitized approach compare to a finite-time LQR solution at the design value and at an extreme mass value; the finite-time LQR approach produces much different behavior at the extremes, while the desensitized and control-desensitized approaches produce similar outcomes at varying values of the system mass.



(a) Desensitized Trajectory

(b) Control-Desensitized Trajectory

Figure 4.2: State values versus time for nominal ($m = 1$) and off-nominal $m = 100$ LQR, DOC, and CDOC controllers for the spring-mass-damper system.

To extend this comparison, the energy-like quantity $\boldsymbol{x}^T\boldsymbol{x}$ was integrated over time at various values of mass; in these cases, reduced trajectory energy corresponds to faster system settling times, and is used as a comparative performance metric. The results, shown in Fig. 4.3, show that the control-desensitized trajectories vary less with the uncertain parameter than the finite-time LQR or state-desensitized LQR approaches.

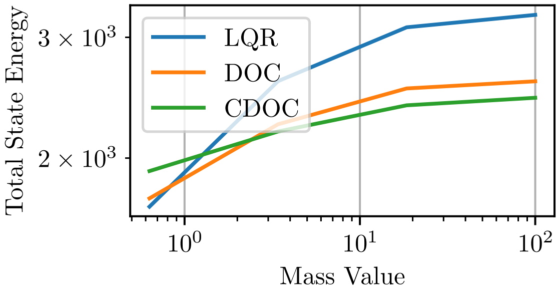


Figure 4.3: Trajectory energy vs. system mass for LQR, DOC, and CDOC controllers.

4.4.2 Differential Drag Formation Flight

Controlling relative spacecraft position and velocity with differential drag introduces a coupling between the system controllability and the local atmospheric density, the latter of which is often only coarsely known. In addition, formulations that depend on knowledge of other orbital parameters, such as the Hill-Clohessy-Wiltshire derived differential drag formation flight system described by Eqn. (4.1), which are also coarsely-known. As such, these systems are prime candidates for the application of desensitized optimal control.

To maintain numerical conditioning of the resulting matrices, it is desirable for the magnitude of the sensitivity matrix to be on the same order as the state dynamics. Noting that the system is linear in many uncertain parameters–namely n, ρ_D, r_C, and the reference deputy ballistic coefficient β_D– we introduce an additional scaling parameter α on these quantities, and seek to minimize our sensitivity to variations in α. This is equivalent to minimizing the sensitivity of the system to any of the designated values, but without the potential for numerical conditioning issues. Incorporating this additional parameter and reducing the state dynamics to the in-plane controllable states yields

$$A = \frac{\partial F}{\partial \boldsymbol{x}} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 3n^2 & 0 & -\frac{1}{2}\alpha\beta_D\rho_D n r_C & 2n \\ 0 & 0 & -2n & -\alpha\beta_D\rho_D n r_C \end{bmatrix}, \; C = \frac{\partial A}{\partial \alpha} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{2}\beta_D\rho_D n r_C & 0 \\ 0 & 0 & 0 & -\beta_D\rho_D n r_C \end{bmatrix} \tag{4.29}$$

$$D = \frac{\partial B}{\partial \alpha} = \begin{bmatrix} \mathbf{0}_{2\times 3} \\ 0 \\ \frac{1}{2} n^2 r_C^2 \rho_D \frac{\partial \beta_D}{\partial \boldsymbol{\sigma}_p} \\ 0 \end{bmatrix}, \quad \boldsymbol{u} = \begin{bmatrix} \sigma_{p,1} \\ \sigma_{p,2} \\ \sigma_{p,3} \end{bmatrix} \tag{4.30}$$

where $\alpha_{\mathrm{nom}} = 1$ is used to maintain numerical conditioning.

From these expressions, it is apparent that the sensitivity of the system to variations from the nominal environmental conditions arises directly from relative velocities and relative attitudes, which form the non-zero entries of C and D respectively. C, which represents the mapping from states to sensitivity rates, has two marginally stable eigenvalues, one stable one, and one unstable eigenvalue. In this sense, the derived C and D matrices provide analytical insight into the sources of uncertainty within the system, thereby providing designers with additional information for heuristic maneuver planning or formation design.

4.4.3 Controllability and Stabilizability of the Sensitivities and States

While the in-plane relative states are known to be controllable, the controllability or lack thereof of the sensitivity states should be understood before applying controllers to the sensitivity-augmented system. To do so, the augmented linear system described by Eqn. 4.25 is constructed for the desensitized differential drag system described by Eqns. 4.29-4.30. The results of this analysis are shown in Table 4.1, and show that while the in-plane states remain controllable as expected, only the radial position sensitivity (s_1) is controllable given the assumed input. Uncontrollable eigendirections exist in both the y-position sensitivity s_2 and the planar velocity sensitivities s_3 and s_4, though these states are partially controlled through a coupling to the planar velocities. Examining the sensitivity dynamics matrix for the differential drag system suggests that the sensitivities share an equilibrium position at the origin with the states. Examining the uncontrolled system poles shows that small, unstable modes exist in the sensitivity states; under the presence of a feedback controller, these poles are shifted closer the origin but remain unstable. Unfortunately, these uncontrollable modes are coupled to the same sensitivity states as the controllable ones, and as a result compromise the performance of explicitly desensitized controllers for the drag-driven formation flight system. In addition, the fundamental instability in these states suggests that small relative planar velocities will eventually produce large sensitivities, further compromising control that reacts to those states.

Table 4.1: Controllability analysis results.

System Definition	$\texttt{Rank}([O])$
States	4
States and Sensitivities	5
States and Sensitivities w/ Control Effect	5

4.5 Performance Characterization

Prior work has shown control results for this system using infinite-time LQR with static gains selected without considering the impact of density variation. This approach is used as a baseline with which the optimal control techniques described in Section 4.3 are compared. To this end, four specific guidance approaches are demonstrated: the baseline infinite-time LQR approach, two finite-time desensitized optimal controllers with and without the sensitivity impact arising from attitude control inputs, and an infinite-time LQR approach with identical position and control weights, but greatly increased weights on the planar relative velocities. The tuned LQR approach represents an attempt at using the analytical insights gained from the linear sensitivity analysis presented in Section 4.4.2 without encountering the numerical controllability issues identified in 4.4.3.

These strategies are compared in a semi-realistic design environment, in which the control gains are designed on the linear differential drag model described by Eqns. 4.29–4.30 but simulated using the nonlinear equations for two-body dynamics plus a facet-based drag model; for more details on the propagation environment, see Reference [72]. To model atmospheric variation, NRLMSISE-00 was consulted for an approximate reference density at the initial Chief's orbit; this value was used as the basis for an exponential model surrounding the chief orbit using a scale height of eight kilometers. Density variation is modeled by multiplying this base density value by an offset and holding the modified density constant across a simulation period.

To provide a direct comparison, solution-specific results such as the overall cost are not presented; due to differences in the scale of the DOC vs CDOC sensitivities alongside the fact that behavior which is optimal under one cost function is likely sub-optimal under another, these direct comparisons are unlikely to provide useful feedback. Each controller is tuned individually using a grid-search over all combinations of state, control, sensitivity, and final error weights. The final values used for each controller are listed in Table 4.4; gain matrices are set diagonally, except for the Tuned LQR approach, which multiplies the state weights for velocity states by a factor of $1{\times}10^7$. Additionally, the initial conditions for the control scenario–designed to represent a 500 meter along-track maneuver in LEO – are listed in Table 4.2. The environmental and spacecraft parameters used are listed in Table 4.3, and are intended to reflect the use of the largest face of a 6U cubesatellite in a 300km, circular orbit.

Table 4.2: Orbital elements for both the deputy and chief spacecraft.

Orbital Element	Chief Value	Deputy Value
a	300 km + r_E	300 km + r_E
i	45°	45.°
e	0	0
Ω	20.0°	20.0°
ω	30.0°	30.0 °
M_0	20.0°	19.995°

Table 4.3: Spacecraft and environment parameters.

Parameter	Value
ρ_0	$2.2{\times}10^{-11}\ \frac{\mathrm{kg}}{\mathrm{m}^3}$
r_C	300 km
h	$8,000$m
A_i	0.06 m^2
m_i	6 kg
$C_{d,i}$	2.2

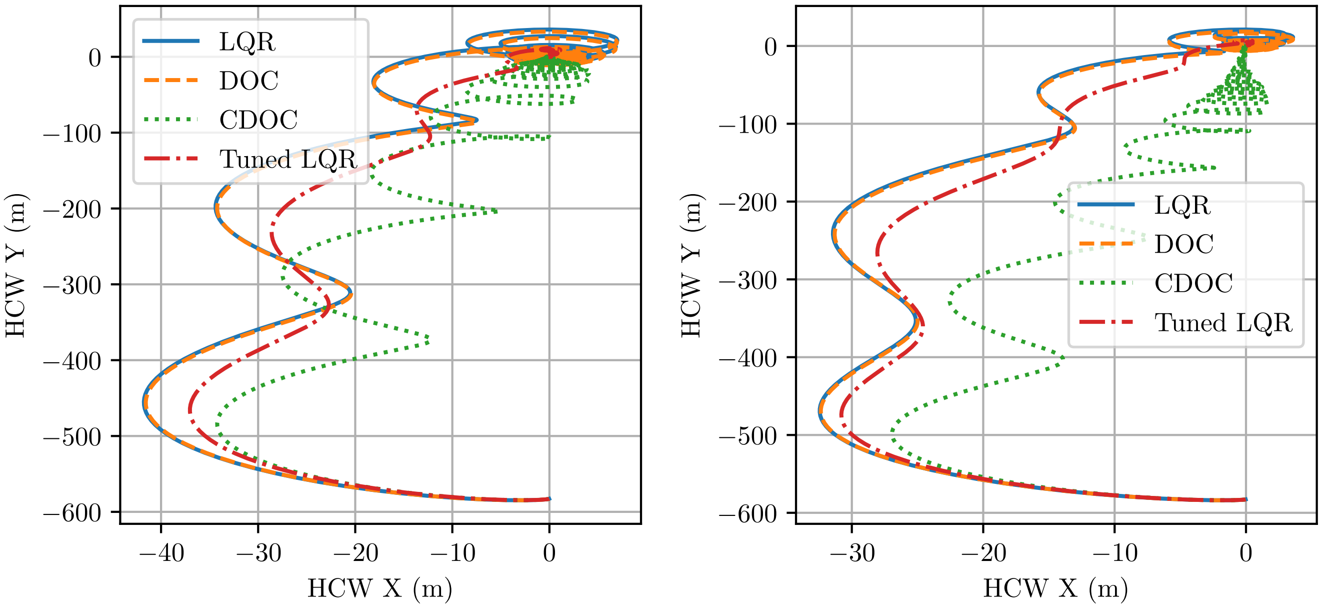
Table 4.4: Selected control weights for each strategy.

Control Design Variable	LQR	Desensitized Optimal	Control-Desensitized Optimal
Q	1.4	1.4	1.4
R	$1{\times}10^7$	$1{\times}10^7$	$1{\times}10^7$
U	0	$1.1{\times}10^{-4}$	$1.1{\times}10^4$

4.6 Linear and Nonlinear Simulation Results

To demonstrate these results, controllers were evaluated for a 500m slot-hopping maneuver using the linear dynamics. A comparison of the Hill-frame trajectories, shown in Fig. 4.4, demonstrates that each controller successfully drives the spacecraft to the origin; this is further confirmed in the time domain by Fig. 4.5a. Additionally, these results show that the control-desensitized and desensitized simulations do indeed produce smaller relative velocities in comparison to the LQR approach, representing the impact of desensitizing the trajectories. While the state trajectories are convergent, the sensitivities shown in Fig. 4.6a demonstrate the impact of the lack of controllability predicted by Section 4.5; while both oscillations and non-convergence are apparent in s_2, s_3, and s_4, these states to stabilize as the states converge. Due to the inclusion of sensitivities from the control input, the scale of the CDOC results is orders of magnitude larger than the LQR or DOC results; however, their relative qualitative behavior still demonstrates the effects of the relative controllability of the sensitivities.

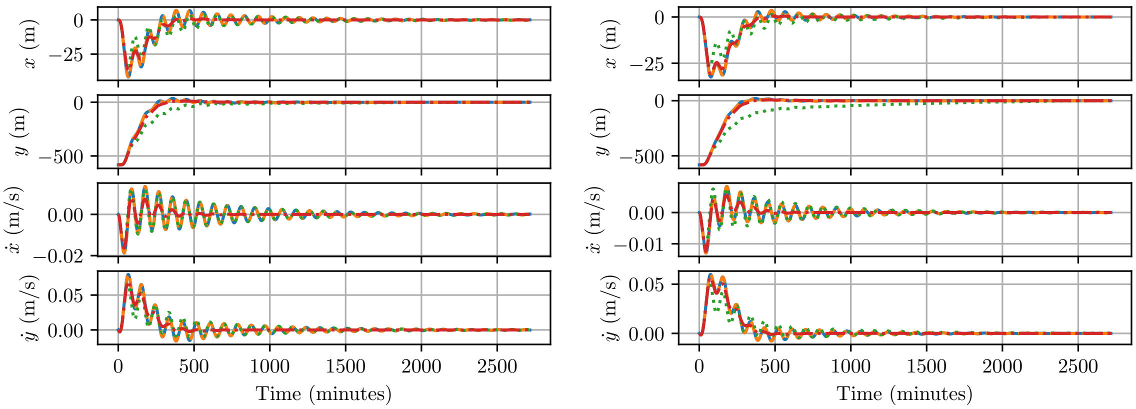
Next, this trajectory is repeated using the nonlinear dynamics from which the linear results were derived. Hill-frame trajectories are displayed in Fig. 4.4, with time histories of both the relative position and velocity trajectories shown in Fig. 4.5b. From these, note that the LQR and Control-Desensitized approaches both appear to successfully drive the deputy spacecraft to the origin in roughly 10-15 orbits, while the Desensitized approach merely moves the spacecraft towards the origin and prefers to minimize the initial motion of the spacecraft. Additionally, these results show that the controllers do not encounter attitude saturation, as shown in Fig. 4.7. Once again, Fig. 4.7 demonstrates the impact of the control desensitized approach, which reduces the



(a) Linear Dynamics

(b) Nonlinear Dynamics

Figure 4.4: Planar hill-frame trajectories of LQR, Desensitized, and Control-Desensitized Trajectories under linear and nonlinear dynamics.



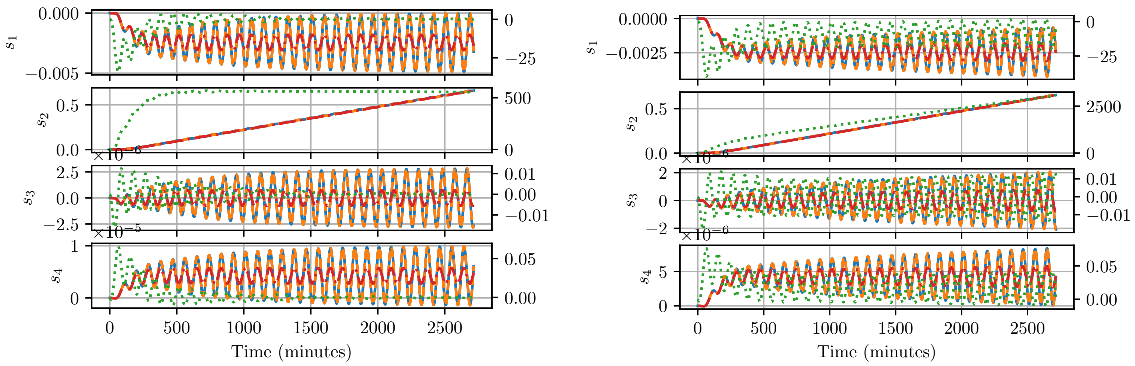
(a) Linear Dynamics

(b) Nonlinear Dynamics

Figure 4.5: Comparison of system state trajectories over the simulation period under linear dynamics.

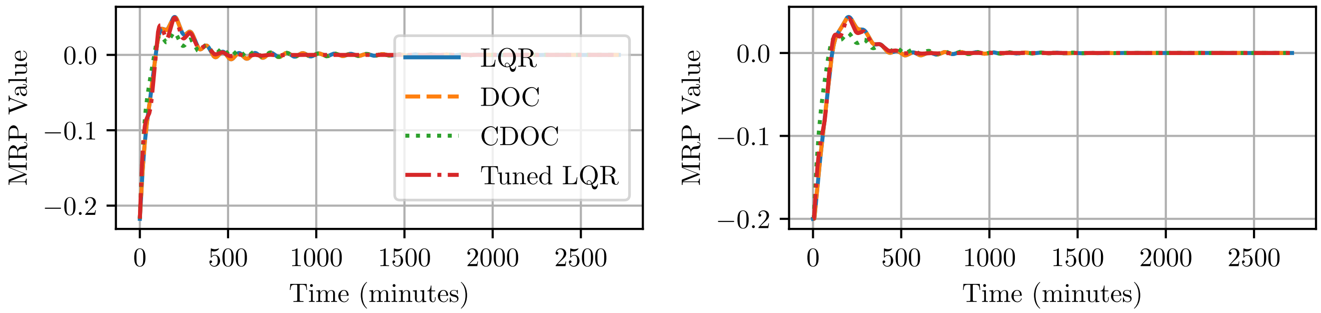
commanded MRP in comparison to the LQR and pure desensitized approach due to the impact of the control input on the system sensitivities.

In comparing the LQR and Control-Desensitized approaches, it is demonstrated by Fig. 4.5b



(a) Linear Dynamics (b) Nonlinear Dynamics

Figure 4.6: Sensitivity trajectories for each control strategy over the simulation period under linear dynamics; LQR and DOC results use the scale on the left, while CDOC results are shown using the scale on the right.



(a) Linear Dynamics (b) Nonlinear Dynamics

Figure 4.7: Attitude trajectories for each control over the maneuver.

that the control-desensitized approach produces much smaller relative velocities than the LQR-derived approach while achieving the same control objective in similar time; however, oscillations remain present in the control-desensitized trajectory that are damped out in the pure-LQR guidance approach. These results reflect a fundamental trade-off in the application of desensitized optimal control; in cases where control authority is limited, including penalties for sensitivities in the cost function J necessarily reduces the control's performance in the states. In addition, comparing the linear and nonlinear results suggests that the control-desensitized optimal controller and to a lesser extent the standard desensitized optimal controller produce trajectories that resemble those generated by the linear system. To quantify this, the following nonlinearity index is utilized:

$$\nu(t, t_0) = \frac{1}{t - t_0} \int_{t_0}^{t} ||\boldsymbol{x}_{nl}(t) - \boldsymbol{x}_l(t)|| dt \tag{4.31}$$

where $\boldsymbol{x}_{nl}(t)$ is the state propagated by the true nonlinear system and $\boldsymbol{x}_l(t)$ is the state propagated by the linear system. This metric is evaluated for all three controllers throughout the simulated maneuver; the resulting plot is shown in Fig. 4.8. Here, it is apparent that the control-desensitized approach reduces the impact of nonlinearities on the system trajectory arising from the sensitivities.

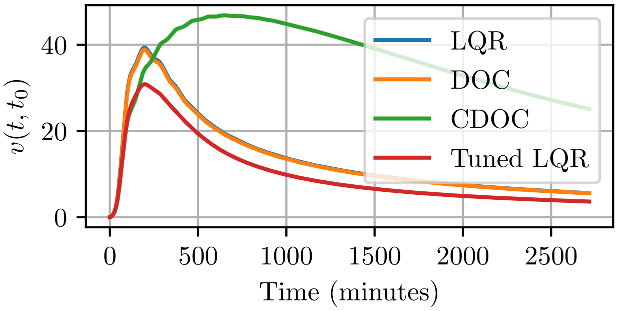


Figure 4.8: Nonlinearity indices for each control type over the 500m slot-hopping maneuver.

4.7 Robustness Characterization

4.7.1 Robustness Metrics and Methodology

The use of different cost functions for each control strategy necessitates the use of additional figures of merit as a basis for comparison. One straightforward figure of merit is the terminal miss distance and velocity achieved by each controller at a given density. Assuming the control target is the origin allows these values to be computed as the norm of the relative position and velocity states at the final time t_f, thereby representing the achievable accuracy of the controller in a given situation.

In addition to accuracy, it is useful for practical operations planning to understand how variable given trajectories are as atmospheric density varies directly. To evaluate this, the percent difference between the terminal miss values at each density and the miss values at the reference density is calculated:

$$\%\text{Departure} = \frac{\text{abs}(d(t_f) - d^*(t_f))}{d^*(t_f)} \times 100\% \tag{4.32}$$

where $d^*(t_f)$ is the miss distance at the nominal density ρ^*.

Finally, the time to reach a satisfactory position, t_{sat}, is used to represent the responsiveness of the control strategies. It is defined as:

$$t_{\text{sat}} = t \text{ s.t. } d(t) < d_{\text{sat}} \tag{4.33}$$

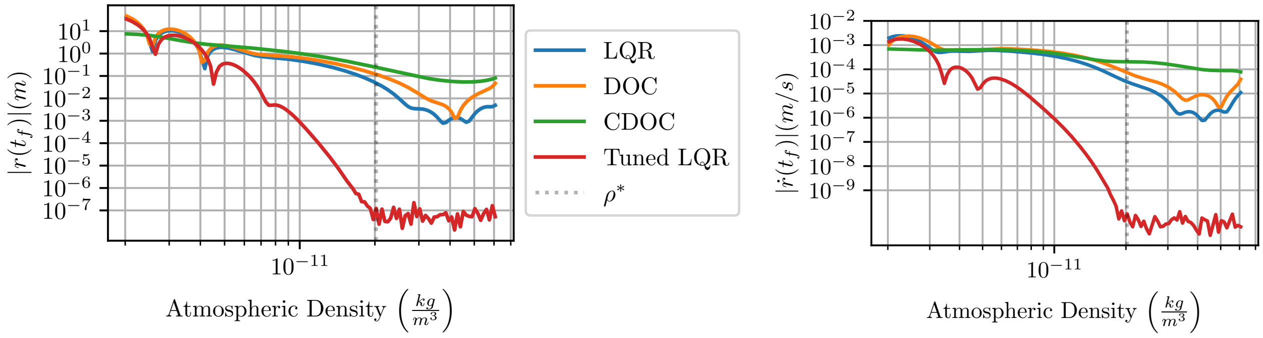
where d_{sat} is a specified value; for the purposes of this work, d_{sat} is chosen as ten meters for all control strategies. The satisfaction time is useful as a point of comparison for the settling time of each approach as density varies; because it is expected that higher densities produce faster responses due to increased control authority, satisfaction time allows users to identify how not only accuracy but settling time are impacted by density variation.

Two scenarios are examined under density variation: a 600m along-track maneuver of the previous section, wherein each control strategy remains within the linear region of both the attitude linearization and the orbit linearization, and a 12,000m along-track maneuver with an additional 100 meter radial offset which acts as a limiting case for the linearized orbital and attitude dynamics.

4.7.2 600m Along-Track Maneuver

The 600m maneuver falls well within the linearization regime for both the orbital mechanics and the linearized control effects. As a result, each of the control strategies is able to bring the maneuvering spacecraft within 10 meters of the target across a range of densities. Figure 4.9 shows that the terminal position and velocity errors for each control strategy tend to vary inversely with atmospheric density. While changes in density do impact the control effectiveness, this impact is not symmetric; increases in density tend to reduce terminal errors, while decreases in density reduce terminal accuracy. From Eqn. (4.1), it is apparent that increasing density directly increases the magnitude of the control effects matrix, thereby improving the control authority available to the controller and therefore the achievable accuracy. This is especially apparent with the tuned LQR approach, which reaches the simulation noise floor at density values at and above the design value.

Both desensitized control approaches are notably less accurate at and above the reference density value; however, their performance is more consistent across the range of density values than the LQR or tuned LQR approaches, with the CDOC approach varying by a maximum of 200% from the miss distance achieved at ρ^* at the lower boundary of the sampled densities; by comparison, the tuned LQR approach differs by 1,000,000,000% at the same boundary. While this metric is biased by the extremely high accuracy of the tuned LQR approach at the nominal density, the differential speaks to the rapid degradation in performance for the tuned LQR approach as the density is varied from the nominal value as shown by Fig. 4.9. The loss of absolute accuracy in exchange for reduced variation as density changes for the DOC and CDOC strategies reflects the fundamental trade-off between state and sensitivity performance described in Section 4.3.

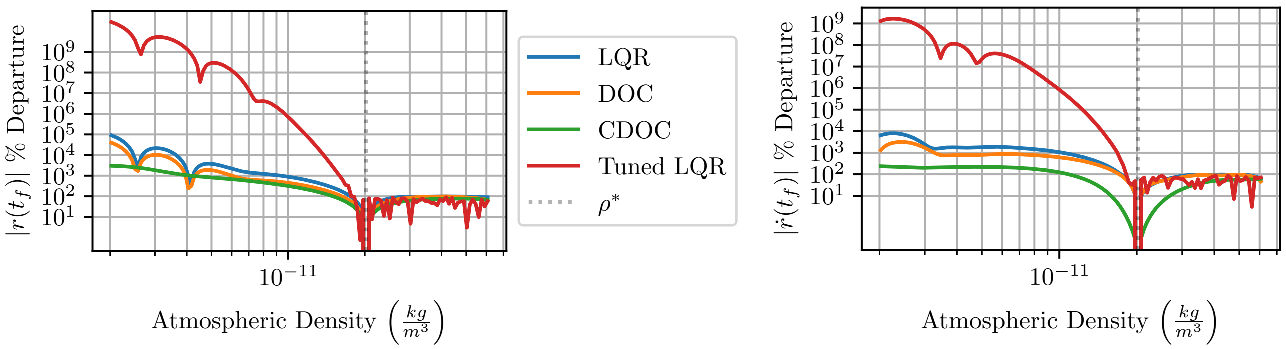


(a) Position Accuracy vs. Density

(b) Velocity Accuracy vs. Density

Figure 4.9: Terminal position and velocity errors versus atmospheric density for the 600m maneuver with various multipliers on the base exponential density from the design reference.

From these analyses, it is evident that the control-desensitized approach produces slightly less accurate terminal position and velocity accuracy while remaining more consistent with its performance at the design density than other approaches as evidenced by Fig 4.10, which shows that the worst-case relative percent difference from the nominal condition is orders of magnitude smaller for the control desensitized approach than for either LQR controller. The consistency of the CDOC approach is further demonstrated in examining the satisfaction time for each controller shown in



(a) Position Accuracy vs. Density

(b) Velocity Accuracy vs. Density

Figure 4.10: Variation in terminal performance relative to performance at the nominal density for the 600m maneuver.

Figure 4.11, wherein the CDOC approach has larger but more consistent settling times than other approaches at a range of densities. These results are consistent with the theory of desensitized optimal control presented in Section 4.3 and demonstrates the trade-off between control consistency and performance. Notably, the control accuracy still produces sub-meter positioning accuracy and sub-millimeter-per-second velocity accuracy for the maneuver. This simulation represents a best-case scenario for each controller, which remains far from the linearity constraints imposed by the assumed attitude guidance input and converges relatively quickly given the allotted time; under these circumstances, the natural robustness of LQR-based control is apparent and the need for desensitization is reduced.

4.7.3 12,000m Along-Track Maneuver:

To better display the benefits of the CDOC approach, a larger maneuver based on the previous example was constructed by increasing the along-track separation to 12,000 meters and adding an additional radial offset. As a result, Fig. 4.14 shows that each controller commands a the attitude trajectory to saturate at the maximum and minimum command-able attitudes for the initial part of the maneuver; this behavior leads to each trajectory initially following the same path, as shown in Fig. 4.12 in the planar positions and Fig. 4.13 across all states in time. In addition to pushing

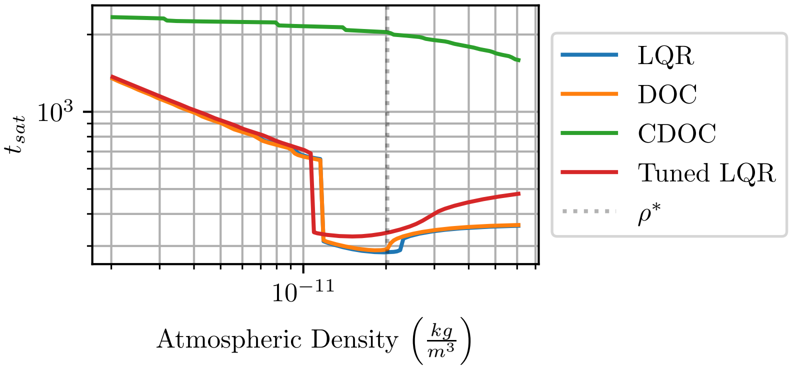


Figure 4.11: Satisfaction time for each controller versus density for the 1500m maneuver.

the assumptions made during the linearization of both the states and control, this maneuver shows additional differences in the trajectories produced by each controller.

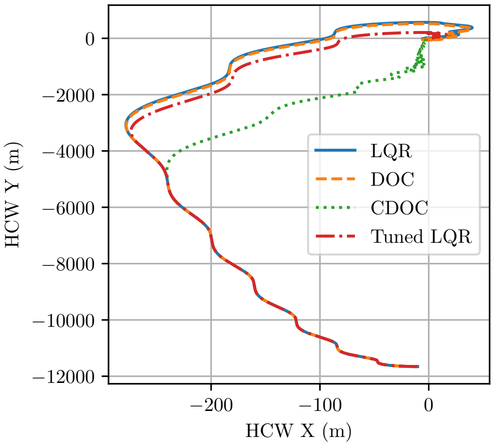


Figure 4.12: Hill-frame performance of LQR, Desensitized, and Control-Desensitized Trajectories under nonlinear dynamics for a 1500m maneuver

This more strenuous case was run over the same density range as the previous example, and the results are shown in Figs. 4.15-4.17. The results of these cases broadly echo those of the 600m maneuver, with control performance improving across the board as density increases and degrading rapidly as density decreases. However, the increased separation distance and control saturation implies that the impact of sensitivities on each trajectory would be larger, especially as the impact of feedback is limited by density-reduced control authority. Fig. 4.15 shows that for values below $1 \times 10^{-11} \frac{\mathrm{kg}}{\mathrm{m}^3}$, the control-desensitized optimal controller produces broadly comparable

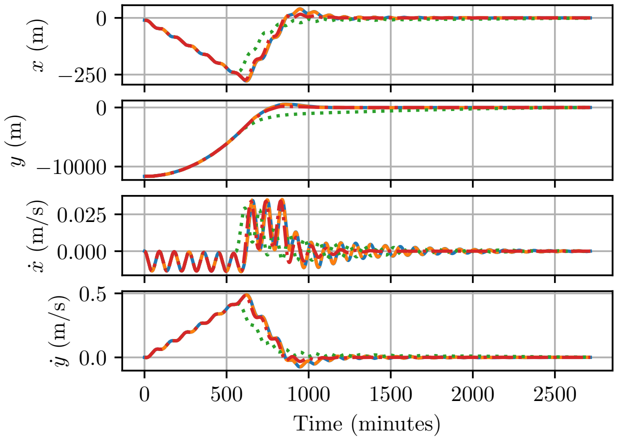


Figure 4.13: Comparison of Hill-frame relative states versus time for the 12 kilometer maneuver; colors and line styles correspond to the legend in Fig 4.12.

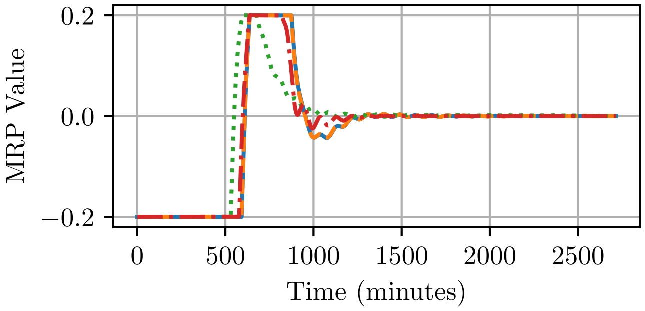
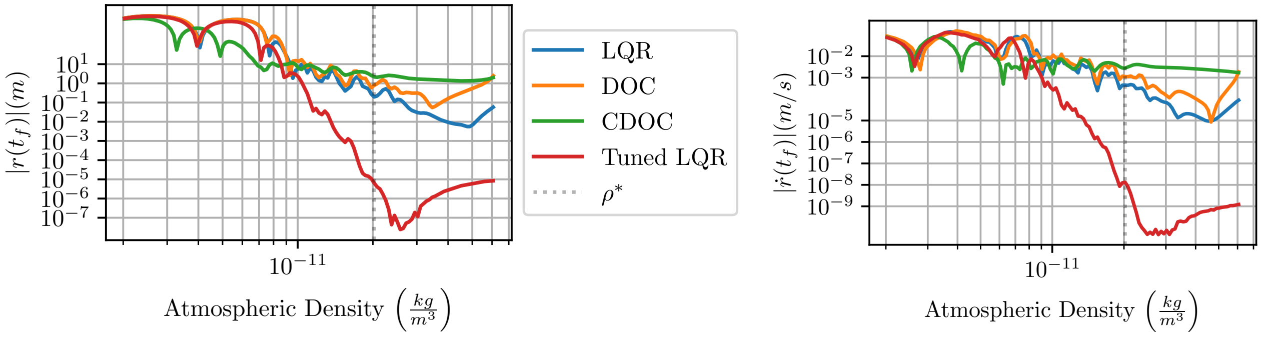


Figure 4.14: Attitude trajectories for each controller versus time for the 12 kilometer maneuver; colors and line styles correspond to the legend in Fig 4.12.

accuracies in the miss distance and velocity to its performance at the design density *and* better accuracy than the other approaches, suggesting that desensitized control does remain effective in a broader range of atmospheric conditions than controllers that do not consider the impact of density variation. This assertion is further backed by the increased density span for which t_{sat} is non-zero (indicating control convergence) shown in Fig. 4.17.

4.8 Conclusions

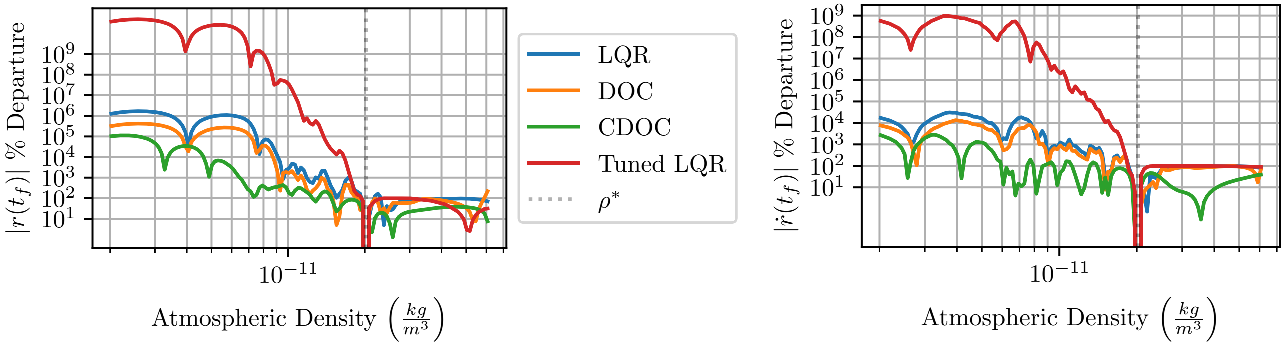
The performance of differential drag orbit control systems is tightly coupled to the latent atmospheric neutral density, which drives both the plant dynamics and the control authority of



(a) Position Accuracy vs. Density

(b) Velocity Accuracy vs. Density

Figure 4.15: Comparison of terminal position/velocity accuracy versus atmospheric density for the 12 kilometer maneuver.



(a) Position Departure from Nominal vs. Density

(b) Velocity Departure from Nominal vs. Density

Figure 4.16: Comparison of percent difference from nominal value for miss distance and velocity versus density for the 12 kilometer maneuver.

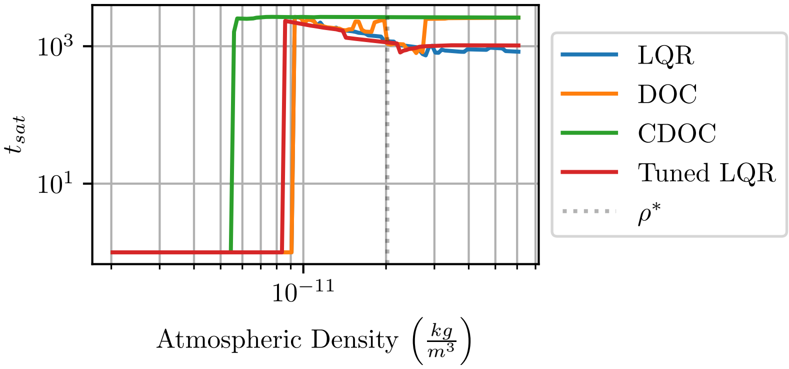


Figure 4.17: Satisfaction time for each controller versus density for the 12 kilometer maneuver.

said systems. Linear sensitivity tools provide a powerful avenue for providing analytical insight into how these couplings impact overall system behavior and potentially provide an avenue for direct mitigation through desensitized optimal control or indirect mitigation through the application of sensitivity insights to traditional control design; in addition, an extension to linear differential drag control that incorporates the effects of parametric variation for systems in which the control effect is dependent on uncertain parameters is derived. While the linear sensitivities of the linearized attitude-driven differential drag system are shown to be only partly controllable, desensitized control techniques that consider the direct impact of control inputs on the sensitivity states are still found to be beneficial for producing control trajectories that vary less with changes in density than other approaches at the cost of nominal performance and accuracy. In addition, applying insights gained from the linear sensitivity analysis allows for substantial impacts in both nominal and off-nominal control performance as well as closer agreement to linear predictions of control performance in simulation. In general, the benefits of desensitized control are greater in cases where density is decreased from the nominal value, which reduces control authority and therefore the ability of a feedback controller to stabilize the system via control, providing an alternative for differential drag control in practices where predictability under density variation is more important than absolute accuracy.

Chapter 5

Machine Learning for Spacecraft Operations

Having established the challenges associated with both LEO spacecraft operations alone and additional challenges arising from the use of atmospheric drag as a control force, this chapter considers the use of contemporary machine learning techniques to safely automate the day-to-day operations of spacecraft in an extensible and scalable manner. Building off of the literature survey presented in Section 1.1.2, this chapter specifically focuses on the adaptation of Deep Reinforcement Learning (DRL) to general spacecraft operations problems. To demonstrate this, several key questions must be addressed:

- How can spacecraft operations be modeled, and which aspects of that model most benefit from automation?
- What advantages does DRL offer over other approaches to spacecraft autonomy?
- Can domain-specific drawbacks for DRL, such as enforcement of safety properties and sample efficiency, be overcome?

While comprehensive questions to each of these problems are beyond the scope of this dissertation, this chapter aims to demonstrate the viability of DRL techniques for addressing future spacecraft operations problems in a safe and scalable manner.

This chapter is organized as follows. First, a description of a general high-level spacecraft mission operations problem is presented and contextualized in the language of partially-observable Markov Decision Processes (POMDPs), considering specific common attributes of these problems that can be exploited to improve the efficiency of learning techniques. Next, a brief description of deep reinforcement learning and its potential benefits for designing operations procedures is presented alongside representative pipelines for implementation as a remote or on-board decision-making engine while addressing safety constraints. Finally, the recommendations of this work are put into practice for two representative operational challenges, outlining the technique's adaptability and merit against a heuristic agent and a timeline-optimizing genetic algorithm to provide points of comparison against rule-based and timeline-based solutions for the presented problems.

5.1 Spacecraft Operations Components and Frameworks

Traditional spacecraft operations planning and execution is a complex, multi-step process with many stakeholders which relies heavily on expert knowledge. For reference, a generic version of this paradigm is presented here. First, mission stakeholders specify mission objectives and a reference mission trajectory. Given this trajectory and a set of desired tasks, a set of activities are defined and scheduled as spacecraft resources (power, fuel, compute time) and mission resources (observation/maneuver/communication windows) permit. Finally, these activities are converted into an action sequence, up-linked to a spacecraft, and executed by on-board software. In parallel to these planning activities, teams of human operators typically monitor mission execution and spacecraft health parameters and intervene when parameters fall outside of a defined specification, either directly by changing the current action sequence or indirectly by initiating a re-planning sequence. Uhlig [73] identifies several key aspects of the mission operations lifecycle:

(1) **Downlink/Uplink scheduling:** Communicating results and telemetry is almost always a critical aspect of space mission operations. To this end, most operational design processes emphasize the design and management of communication opportunities.

(2) **Orbit and Attitude Maneuver Design**: Most missions will require regular attitude slews or station-keeping maneuvers throughout their lifetime; the design of these maneuvers and the conditions that trigger them is a core component of spacecraft operations.

(3) **Operations mode design:** Owing to fundamental physical or electronic constraints, it is almost always necessary to specify multiple operating states for the spacecraft's hardware and software which can satisfy both mission goals and said constraints.

(4) **Mission-Driven Tasking:** Some missions, such as those focused on surveillance or targeted ground observation, involve the active assignment of spacecraft tasks to mission-relevant domains.

(5) **Operational Plan Development and Execution:** The above actions must be combined at a high level to meet a diverse set of mission goals while satisfying hardware and software constraints.

The first three components are typically shared between missions, and as a result can leverage a large body of work describing ground access prediction, orbit determination and maneuvering, and attitude control. However, no comparable body of standardized, generalized approaches exists for the development of operational plans across a variety of mission types. While important sub-problems have been automated or assisted using various techniques, other important aspects of the spacecraft operations lifecycle - such as spacecraft health management - are not typically considered or would render such techniques computationally infeasible.

5.1.1 Spacecraft Operations As Control

This work is primarily concerned with mission operations that abstract collections of relevant low-level behaviors and states into operational modes that can be readily composed by operators as part of a general trend towards the formalization of such design practices. Mode-based operations planning is common in the small satellite domain for both Earth-oriented and deep-space missions. The use of operational modes as the basic primitives for mission planning greatly simplifies the overall learning problem and allows the use of existing tools and processes to address low-level problems, such as attitude determination and control. As with all abstractions, the application of operational modes also hides true subsystem behaviors that can impact missions on a high level.

Specifically, this work conceptualizes spacecraft operations as a hybrid system consisting of discrete operational modes $q_i \in Q$ that affect the evolution of a constant set of continuous states $\boldsymbol{x} \in X$. The aim of tasking process is to select discrete modes to maximize performance with respect to a mission objective function $R = R(q_i, \boldsymbol{x})$ while satisfying a set of constraints corresponding to system hardware and software limitations.

A key benefit of this approach is the natural manner in which it can be translated into a partially-observable Markov Decision Process (POMDP), allowing the use of contemporary solution algorithms. POMDPs are formally defined as a tuples of states S, actions A, observations O, rewards R, and functions which map between states ($s' = T(s, a)$) and between states and observations ($o = H(s)$). As their name implies, MDPs are Markovian such that system trajectories can be predicted or inferred given the system state at a single time; however, POMDPs can break the Markov property through partial observability, necessitating the use of belief or memory functions to infer the status of un-observed states. The hybrid systems model of spacecraft mission operations can be interpreted as a POMDP in which the action set A is the set of discrete actions Q and the observation space is a subset or transformation of the true system states, which may be unobserved or partially observed by the planning process.

For a spacecraft, the general high-level autonomy POMDP can be stated as follows. Given the constraints of orbital dynamics, on-board hardware, and pre-defined software behaviors, select the sequence of behaviors that best satisfies mission objectives. This framework situates the operational procedure as an "agent" that reacts to given circumstances in the state space using a set of predefined actions. Under this definition, there are multiple issues in translating from the real-world problems of spacecraft operations to the Markov framework that are common to other real-world examples [74]. In seeking an MDP formulation for spacecraft decision-making, three major questions must be addressed:

(1) How is time represented?

(2) Which states/actions should we select? Should we consider discrete or continuous spaces?

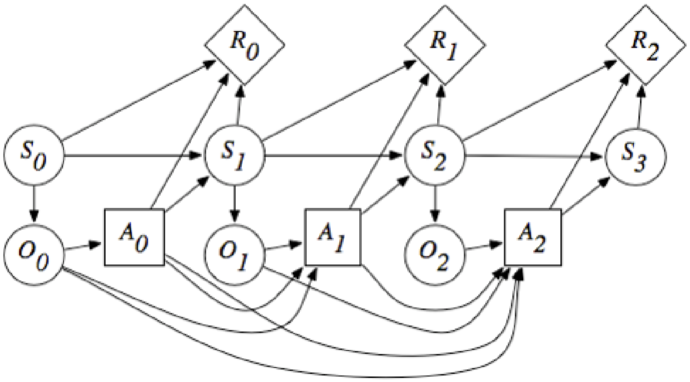


Figure 5.1: Sequential Partially Observable Markov Decision process framework for representing decision problems.

(3) How do we define reward functions and therefore agent objectives?

While the POMDP framework places no restrictions on the nature of any of the transition functions or states, the consideration of infinite-dimensional, continuous state and action spaces can be extremely computationally intensive. Given the limited computational resources of both research and development efforts in aerospace, it is desirable to identify strategies for reducing the dimensionality of the state and action environment without losing representative information about the problem. Additionally, it is noted that POMDPs attempt to describe holistic, system-level problems within a unified framework that is theoretically related to but practically divorced from traditional estimation and control approaches. For these reasons, POMDP-based approaches to autonomy are most frequently studied in cases where traditional estimation and controls approaches are not readily tractable, including human-assisted machine decision-making [29] or multi-vehicle coordination problems [75].

5.1.1.1 State-Action Models

State, action, and transition-space modeling is a critical method for encoding known information into the decision-space for a learning agent. At present, it is common in the reinforcement learning space to include a wide variety of "raw" information from a system as the input to an agent (AtariNet, for example, attempts to map directly from pixels on a screen to button inputs).

Shortcomings in this approach have spurned further research in the domain of world modeling and intermediate representation learning, wherein an agent learns a model of the world and intermediate representations of actions or observations in addition to policy-defining behaviors. Given the range of prior work in spacecraft state estimation and control, it is desirable instead to leverage existing state and action representations, such as the hybrid systems representation of spacecraft operations suggested in Section 5.1.1, which provides a straightforward way to reduce the space of actions to a finite set of discrete operational modes and the observations to a subset of continuously- or discrete-valued system states.

To further simplify the observation model, assumptions can be applied based on prior knowledge of other control strategies for hybrid systems. One well-known result to demonstrate stability of switching strategies for hybrid systems is the theory of multiple Lyapunov functions (MLF) [76]. MLF theory demonstrates that, for a switched hybrid system, the stability of switching sequences on said system can be shown by constructing candidate Lyapunov functions for each subsystem V_i and demonstrating that said functions remain Lyapunov-like for each switching time:

$$V_i = \boldsymbol{s}_i^T P_i \boldsymbol{s}_i \text{for } \boldsymbol{s}_i \in \boldsymbol{s}(k); \dot{V}_i < 0 \text{ if } a(k) = a_i \tag{5.1}$$

Inspired by this approach, this work proposes 'Lyapunov Dimensionality Reduction' to simplify MDP construction for switched hybrid systems. Rather than reporting the entire system state to the agent, LDR proposes that it is sufficient to learn switching sequences by observing the value of candidate Lyapunov functions for subsets of the system state that are stabilized by each operational mode, alongside other information necessary to ensure proper subsystem functionality which would otherwise break the hybrid system abstraction.

5.1.1.2 Objective Functions

A major issue in the application of MDP solution methods is the difficulty of specifying agent reward or objective functions, which face conflicting requirements of both reflecting desired behavior and providing a learn-able sequence of returns. In the space domain, several considerations are present. At a minimum, reward functions must be specified such that desirable results produce large rewards (or result in small penalties). When possible, it is desirable for rewards to be *shaped* such that agents can determine more- and less-desirable behaviors. For deep learning agents specifically, there is considerable debate surrounding the use of “reward engineering” to encourage exploration of alternate strategies by assigning smaller rewards to intermediate actions; however, agents that learn with extremely structured reward may have difficulty generalizing or discover unintended behaviors that are reward-optimal due to the complex relationship between problem dynamics, state representations, and reward functions. As a result, this work identifies two major archetypes for future objective functions, both based primarily on achieving mission objectives:

Discrete events: A common mission archetype studied broadly in the literature involves obtaining access to specific points on a planet under specific constraints (time, local solar time, etc.) Agents receive a reward for accomplishing specific mission events under the provided constraints. This structure is likely desirable for operations policies that are attempting to replicate or extend the behavior of discrete-event scheduling algorithms such as those presented by Chien et al [25, 26, 27, 77].

Abstracted events: As referenced in [25], mission-specific algorithms for scheduling science events may already exist; in this case, these behaviors may be abstracted to a “mission mode” that provides the agent a reward for entering that mode with desirable system conditions to maximize the likelihood of success of those lower-level schedulers.

For space missions, reward functions can be readily specified given mission-level success criteria to the degree that such criteria are known. For example, an earth-observation mission might search for plans that maximize the amount of data down-linked to the ground, with no specifications for intermediate behavior. This degree of freedom in the planning process additionally introduces the possibility of algorithms discovering additional desirable behaviors without spending engineering time.

5.1.2 Survey of Reinforcement Learning Strategies

Reinforcement learning techniques are a class of machine learning techniques that learn behaviors for interacting with unknown environments through repeated interactions with said environments. Directly inspired by research into learning in humans and animals, reinforcement learning combines insights from both nature-inspired learning processes and optimal control. Several surveys have been written summarizing the history of the field [78], general concepts [30], and best practices for RL and DRL research [79]. This section will briefly review the mathematics behind deep learning concepts and specifically the design of algorithms used within this paper.

Deep Reinforcement Learning (DRL) algorithms seek to optimize the behavior of decision-making agents as they interact with environments that can be represented as MDPs. These behaviors are represented by policies, π, which map from states or state observations to actions or action probabilities. As a differentiator from classical reinforcement learning, these policies are parameterized by the weights and biases of one or more deep neural networks. Following with the description of Markov Decision Processes, the objective of virtually all DRL approaches is to maximize the expected reward achieved by an agent interacting with an MDP:

$$R = \sum_{t=0}^{T} r_t \tag{5.2}$$

For cases where $T = \infty$, a discount factor $\gamma \in [0, 1]$ is considered and the return is instead calculated as the discounted return:

$$R = \sum_{t=0}^{\infty} \gamma^t r_t \tag{5.3}$$

Given a policy, it is possible to compute the value function V for a given state, which is taken as the expectation of the future return of a policy given the current state:

$$V^{\pi}(s) = E\big[\sum_{t=0}^{t=\infty} \gamma^t r_t | s_t = s, \pi(\theta)\big] \tag{5.4}$$

where $V^{\pi}(s)$ represents the value of state s under policy π, γ represents the reward discounting factor (chosen to be between 0 and 1), r_t is the reward value at step t, s_t is the state at step t, and

$\pi(\theta)$ is a policy parameterized by a vector of parameters θ. A similar function, the state-action value function Q, can be similarly defined by conditioning future returns on states and actions:

$$Q^{\pi}(s,a) = E\big[\sum_{t=0}^{t=\infty} \gamma^t r_t | s_t = s, a_t = a\big] \tag{5.5}$$

Two archetypal reinforcement learning approaches, value iteration and Q-learning, learn the optimal values of V^{π} or Q^{π} by drawing samples from the environment and applying one-step Bellman backups to their V or Q functions respectively:

$$V(s_t) \leftarrow V(s_t) + \alpha(r_t + \gamma V(s_{t+1}) - V(s_t)) \tag{5.6}$$

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha(r_t + \gamma \max_{a'} Q(s_{t+1},a') - Q(s_t,a_t)) \tag{5.7}$$

Given these value functions, it is possible to predict the best-known actions by selecting actions that correspond to the largest values of V and Q; if these functions have converged to the optimal functions on an environment, such a policy is also optimal. A variety of additional tweaks to these formulas, such as TD(λ) learning, have also been developed to speed convergence for environments with sparse rewards by performing backups over sampled trajectories instead of individual samples. Notably, the policies learned by value iteration and Q-learning do not include explicit models of environment transition dynamics, leading to the use of the term 'Model-Free RL' to describe them. In contrast, model-based RL approaches attempt to explicitly build a model of environmental dynamics while learning a policy. An archetypal example of model-based RL is the Dyna architecture proposed by Sutton [30] for discrete state and action spaces. Dyna uses a frequentist approach to construct transition probabilities for state-action pairs to build a model of the transition function, $\hat{T}$, and reward function $\hat{R}$. These reward and transition models are used to update the policy's Q function at a state:

$$Q(s,a) = \hat{R}(s,a) + \gamma \sum_{s'} \hat{T}(s,a,s') \max_{a'} Q(s',a') \tag{5.8}$$

In addition, for each sample drawn from the 'real' experience, k additional state-action pairs are randomly selected and used to update other parts of the Q function using the updated transition dynamics, thereby ensuring that other parts of the policy are updated with every additional sample drawn from the environment. This approach has obvious benefits if drawing samples from the environment is expensive, such as for agents learning on real systems; as a result, Dyna-style approaches are often shown to be more sample efficient in producing policies with respect to samples drawn from an environment. Unfortunately, the learning of policies and state models in parallel can produce unstable behavior in training, especially if policies in an environment are relatively simple but the environment dynamics are difficult to predict; in these cases the parallel updates to transition models and policies can prevent convergence altogether. As a result, despite their supposed advantages, model-based RL approaches remain infrequently used for cutting-edge results.

A key challenge in all of these techniques is the requirement to store intermediate functions of arbitrary complexity and dimensionality; this constraint restricted RL approaches to discrete state and action spaces (so-called 'tabular' reinforcement learning) for most purposes until relatively recently due to the resultant scaling issues. Deep Reinforcement Learning (DRL) addresses these limitations by utilizing deep neural networks to approximate V and Q functions (or, for policy-gradient methods, π directly) instead of probability tables. Neural networks are widely known for their capabilities as universal function approximators [80]; this property, combined with the discovery that gradient descent and back-propagation could be used to rapidly train large networks on modern hardware [81], has led to an explosion of practical applications of deep learning. Techniques based in deep learning currently represent the state-of-the-art in several fields, most notably image and natural language processing, both of which represented long-standing challenges in artificial intelligence that were thought to be unsolvable without extensive human intervention. The success of deep learning for practical tasks is generally poorly understood, but is thought to be at least partially a result of the viability of local minima for addressing practical problems. Initial work in contemporary DRL [82] applied deep networks for Q-function approximation ('Deep-Q' learning) and demonstrated human-level performance on various Atari games after learning directly from pixel information provided by each game (an artifact of the legacy of DRL's roots in image processing successes.)

5.1.3 Policy Gradient Techniques

To meet the aim of having a broadly applicable approach with few limitations on the structure of the action or observation space, Proximal Policy Optimization [83] – a recently developed model-free policy gradient algorithm – is used as the algorithm of choice. Empirical results have shown that PPO provides a robust mix of performance, relative insensitivity to hyperparameter selection, and applicability to a variety of problems owing to its use of a probabilistic policy. At the same time, because the resulting policy is not deterministic but instead a conditional probability distribution over actions given an observation, the behavior of PPO-derived agents is non-deterministic, which has important implications for safety and verification. For the reader's convenience, a brief review of policy gradient methods and PPO specifically is provided.

Policy gradient methods are so-called because they directly optimize the agent's policy using results from the policy gradient theorem [30]:

$$\nabla_\theta J(\theta) = E\big[Q^\pi(s,a)\nabla_\theta \ln(\pi_\theta)(a|s)\big] \tag{5.9}$$

PPO is a simplified version of Trust-Region Policy Optimization (TRPO) by limiting the size of steps in the policy space. First, the probability ratio between a new policy and an old one, $r(\theta)$, is defined as:

$$r(\theta) = \frac{\pi_\theta(a|s)}{\pi_{\theta \text{ old}}(a|s)} \tag{5.10}$$

PPO enforces a step-size constraint on the size of gradient updates by clipping θ updates to remain within $1 \pm \epsilon$ of the previous policy. PPO therefore adjusts the TRPO objective function to include clipping to constrain the size of a given update:

$$J^{\text{CLIP}}(\theta) = \mathbb{E}\left[\min\Big(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\Big)\right] \tag{5.11}$$

where $\hat{A}_t$ is an estimate of the advantage function $A_\pi = Q_\pi(s,a) - V(s)$ and ϵ is a tuneable hyperparameter described as the clipping fraction. When implementing PPO2 with a single neural network for both the policy and value functions, the full objective function is typically augmented

Algorithm 1: Proximal Policy Optimization Algorithm

```
Result: $\pi(\theta)$
for $k$¡$k_{max}$ do
    Collect sampled transitions $D_k^{\pi}$ using current policy $\pi_k$
    Compute advantage estimate $\hat{A_k}^{\pi}$ for each sampled transition
    Compute new policy parameters using minibatch SGD with Eqn. 5.12
end
$a$ = SampleDistribution($\pi(o)$)  return $a$
```

to include both a value target and entropy term:

$$J^{\text{Full}}(\theta) = \mathbb{E}\left[J^{\text{CLIP}}(\theta) - c_1(V_\theta(s) - V_{\text{target}})^2 + c_2 H(\pi_\theta(s))\right] \tag{5.12}$$

where $H(\pi_\theta(s))$ represents the entropy of the probabilistic policy π in the state s, $V_\theta(s)$ represents the current value prediction in the current state, V_{target} represents the value target at the current state.

A representative implementation of PPO is described in Alg. 1. First, a set of partial trajectories is sampled from the environment using the current policy. Next, advantage estimates $\hat{A}^{\pi_k}$ are calculated from those samples using an advantage estimation algorithm such as the Generalized Advantage Estimation method presented in Ref [84]. Finally, minibatch stochastic gradient descent is used with the loss function described in Eqn. 5.12 to compute the policy improvement.

5.1.4 Intrinsic Dimensionality and Problem Complexity

Reference [85] identifies a random subspace growth methodology for identifying the intrinsic dimension of arbitrary machine learning objective problems, providing a scalar figure of merit to compare the difficulty of learning in different classification and reinforcement learning problems. This work is briefly summarized in the context of policy-based learning agents for the reader's convenience.

Given a learning agent parameterized by a set of parameters $\theta^D \in \Re^D$, the intrinsic dimensionality d_{int} of a given problem is defined as the codimension of the solution set inside of $\Re^D$:

$$D = d_{\text{int}} + s \tag{5.13}$$

In general, the dimensionality of this space is non-trivial to determine analytically. To determine these dimensions empirically, an iterative process wherein the learning agent is trained with successively larger random subspaces drawn from the overall policy space is used by defining θ^D as:

$$\theta^D = \theta_0^D + P\theta^d \tag{5.14}$$

where P is a randomly generated, orthonormalized $D \times d$ projection matrix, θ^d is a parameter vector in a subspace of D such that $d \leq D$ and θ_0^D is an initial vector suited to the problem at hand. Gradients are taken with respect to θ^d; the training process is repeated for a specified number of iterations or samples, and at some point d is incremented; when $d < d_{\text{int}}$, it is by definition not possible for the learning agent to adequately solve the problem at hand. As a result, sweeping across a range of values for d and identifying the value at which solutions appear provides an estimate for the real value of d_{int}. Due to the numerical challenge of obtaining "100%" solutions, the intrinsic dimension of an agent with 90% of the baseline solution, d_{int90}, is used as the figure of comparison for problems.

This technique is particularly attractive as it allows direct comparisons between the number of parameters required for a given neural network to sufficiently address a given problem; moreover, this technique allows for comparisons of difficulty across different problems and problem types in terms of network requirements. Given the explosion of DNN-driven techniques in other fields, it is desirable to understand exactly how problems in spacecraft tasking and planning relate in terms of difficulty and learnability.

5.1.5 Agent Implementation Frameworks

A major assumption in our formulation of the spacecraft control problem as a (PO)MDP shown in Eqn. 5.32 is the discretization of time, which–when combined with the mechanics of learning as described in Section 5.1.2–results in decision-making agents that can only *react* to current observations, as shown in Fig. 5.2. Rather than utilizing a specific plan or strategy, all relevant planning and strategy information is encoded in the deep network utilized by the agent. In practice, evaluating neural networks is nearly constant-time and can be readily hardware-accelerated, making this implementation attractive for future on-board use where system information is readily available and humans are already out of the loop.

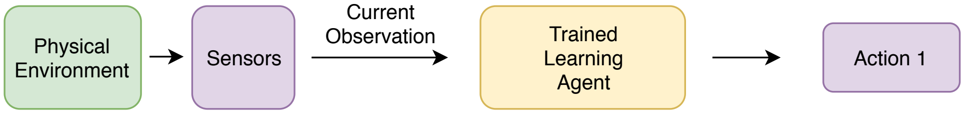


Figure 5.2: Sequential decision-making agent architecture.

At the same time, many existing systems assume that discrete sets of actions will be periodically up-linked from the ground and lack the on-board processing power to evaluate a neural network. For these systems, an architecture which uses a ground-side simulator to propagate forward existing observations and actions is proposed as shown in Fig. 5.3. The incorporation of a simulator allows for the agent to make "future" decisions based on current knowledge and plan ahead. This architecture is also attractive for near-term implementation, as it allows human operators to verify and validate action sequences in advance of execution.

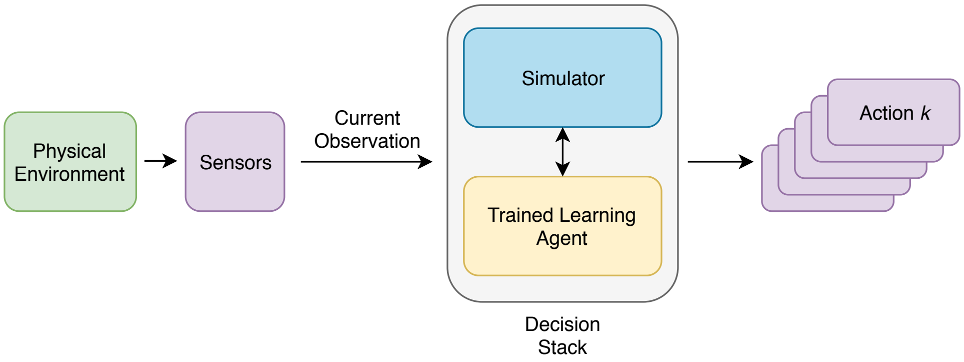


Figure 5.3: Planning architecture using a sequential decision-making agent.

Examination of the properties and benefits of planning versus reactive agents is left outside the scope of this work, which focuses on establishing training and safety properties for DRL-based sequential decision-making agents for spacecraft command and control.

5.1.6 Safety Guarantees with Shielded Learning

Safety in the face of uncertain spacecraft performance, environmental parameters, and operating sequences is a critical requirement for future spacecraft autonomy architectures. While some reinforcement learning techniques can bound their performance with respect to a reward function within an MDP, these weak guarantees often do not generalize, especially when moving from simulated data to real-world application. In practice, this is dealt with through reward engineering; unsafe action or state combinations are given large costs or penalties to achieved reward. This approach has several key disadvantages: many problems for which reinforcement learning is well-suited have complex environment/reward interactions, which makes manual reward engineering difficult. When reward engineering is feasible, it does not prevent the agent from taking unsafe actions in conditions outside the training set presented by its environment, especially when considering agents that utilize stochastic policies such as PPO2. Finally, there is no quantifiable boundary or degree of safety provided through reward engineering. These shortcomings have motivated the search for alternative approaches to safety that can be combined with common DRL approaches.

Reactive synthesis is one category of techniques that can provide performance bounds and guarantees for controllers on specified systems. In general, reactive synthesis algorithms operate on discrete, known, finite systems and attempt to produce behavior on such systems that satisfies a specification written in a temporal logic language, such as Linear Temporal Logic (LTL). Also described as "correct-by-construction" approaches, reactive synthesis algorithms only produce control policies that meet a given specification; if the specification cannot be met on the current system, no policy will be produced, allowing for designers to check feasibility before implementation. While powerful for addressing systems with discrete, finite, known dynamics, reactive synthesis approaches scale poorly with system and specification complexity. These characteristics limit their applicability in solving general spacecraft planning problems, which are difficult to discretize to sufficient fidelity[32].

Shielded learning techniques [86] combine common DRL approaches with reactive synthesis-based shields to combine the power of black-box optimization with formal guarantees of safety. Shielded learning depends on the construction of a coarse, finite-state safety MDP from the original MDP the learning agent is intended to solve that is conservative with respect to the original environment's dynamics and the safety specification, yet limited enough that reactive synthesis can be applied to it. Next, a safety specification is created using Linear Temporal Logic which encapsulates all desired safety conditions and provided as an input to a reactive synthesis algorithm, such as a two-player game, which produces a discrete, state-dependent strategy. Finally, this strategy is implemented alongside the learning agent as shown in Figure 5.5; in this implementation, the shield accepts observations of the current system state and the action attempted by the learning agent, and permits the action only if it aligns with the shield's strategy. This implementation architecture is applicable to both training and on-line use of the sequential decision agent, allowing it to provide safety boundaries during mission execution.

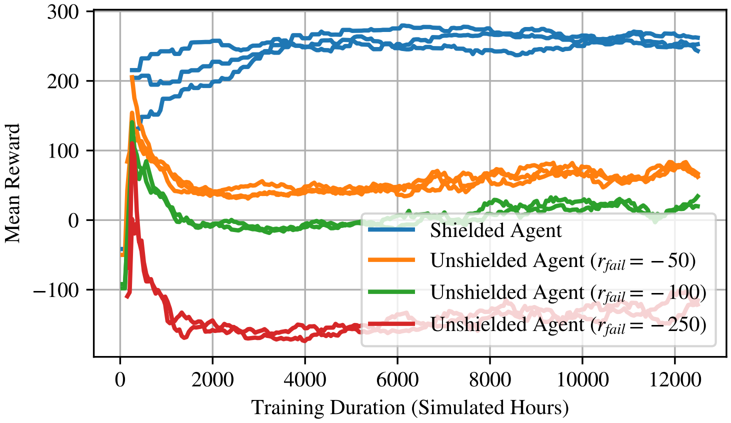


Figure 5.4: Comparison of shielded vs. unshielded agent performance with erroneous reward returns

One concrete example of the practical benefits of off-loading safety properties to shields rather than enforcing them through reward engineering occurred early in the development of the LEO attitude-health mode management simulator, wherein agents are penalized for failing to maintain adequate power or for allowing reaction wheels to saturate. Due to an error in the environment implementation, negative reward signals for failure states were never provided to the agent; as a result, agents trained using PPO2 were unable to succeed in the environment. Because the efficacy of the safety shields prevented agents from entering into unsafe regions of the state space, agents trained using SPPO2 were still able to converge and produce stable results despite the environment implementation error; the resulting training curves are displayed in Figure 5.4. Due to the oft-derided complexity of both detailed simulation environments deep learning frameworks themselves, this type of error is endemic and difficult to diagnose; however, the adaptation of shielding can guard against the impact of such bugs, if not identify them directly.

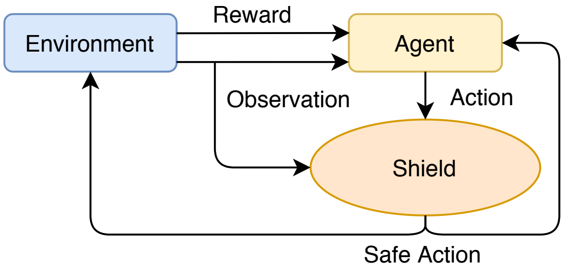


Figure 5.5: Post-Posed shielded reinforcement learning framework.

An example of this transformation in practice is shown for a system with two safety-critical dimensions in Fig. 5.6. Mission designers first identify state combinations that represent mission failure, such as depleting the spacecraft's battery or allowing reaction wheels to spin up beyond manufacturer's specifications. In addition to the hard safety constraints, operators and mission planners typically incorporate additional boundaries to act as margins of safety against actual failure; these are represented by the dashed lines labeled "operational boundary," which are used to define "warning states." While in this boundary, operators typically take immediate action to return the system to safe, nominal operating conditions. In this view, the system's behavior can be plotted on a phase-plot, where individual samples of the system's true trajectory are represented as curves in the observation variable space. The continuous but bounded system creates a natural framework for the construction of a safety MDP, wherein each warning state becomes a discrete state, including products of warning states. It is important that the safety MDP contain all information necessary for the system to operate safely, which may require the inclusion of states

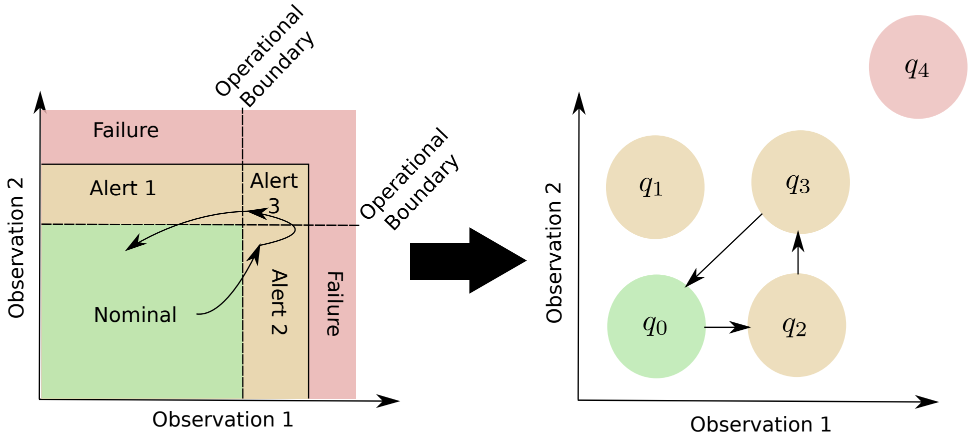


Figure 5.6: Conversion from continuous states to a discrete safety MDP.

which are not themselves safety risks but which affect the performance of actions necessary for the safety of the system. This process results in a discrete "safety" MDP which exists in parallel with the continuous POMDP.

5.1.6.1 Shield Construction

To apply the shielded learning technique to space mission operations, a simplified version of the mission POMDP is first constructed using a-priori knowledge. Here, alert states are defined using the operational limits found in Table 5.1. These limits are applied to transform the continuous-time, continuous-state system described by Equation 5.32 into a simplified, discrete MDP in the observed variables, represented graphically in Fig. 5.10. This MDP is stated as P_{disc}:

$$P = \begin{cases} s & = \{\boldsymbol{\omega}_{BN} \in \{\text{nominal, high}\},\ |\omega_{RW}| \in \{\text{nominal, alert, failure}\},\ \text{J}\in \{\text{nominal, low, failure}\} \\ o & = \{q \in \{q_0, q_1, ...q_7, q_8\} \\ a & = \{\text{Mission}, \text{Sun Pointing},\ \text{Desaturation}\} \\ T & = \{f_{\text{Mission}},\quad f_{\text{Sun Pointing}},\ f_{\text{Desaturation}}\} \\ R & = \{\emptyset\} \end{cases} \tag{5.15}$$

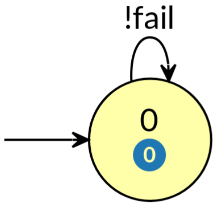


Figure 5.7: The one-state Büchi automaton representing the safety specification for the system.

While substantially smaller than the continuous state POMDP, the safety MDP encodes important information; for example, desaturation events are only feasible when the spacecraft is not in a tumbling state, and tumbling states themselves do not lead to failure unless the battery charge or wheel speed are already near the failure criteria. In addition, the various state combinations that lead to failure are lumped into q_8 for brevity; this permits the use of the simple LTL specification

$$\varphi = G(\neg\text{“fail”}) \tag{5.16}$$

which is represented using the Büchi automaton shown in Fig. 5.7, and can be understood in English as “globally never allow the state to reach the failure state.”

5.1.7 Safety Game Solutions

To solve this safety game, the game itself was implemented as a stochastic Markov game (`smg`) within the PRISM-games solver. In this single-objective case, PRISM-games solves the safety game using value iteration [87] to identify optimal strategies for both the environment and shield. Following the definition of a safety game, the shield player is specified to minimize the probability of a transition into the failure state, transforming the LTL specification described in Eqn. 5.16 into PRISM's rPATL modeling language as

$$P_{\text{min}} =?\ [G!"fail"] \tag{5.17}$$

After solving for the failure-minimizing strategy, PRISM-games then saves the shield strategy as a `.adv` file, which encodes the state-action strategy which maximizes the probability of remaining safe. For this work, the resulting strategy is memoryless and state-based, making it especially amicable to on-line implementation. Following the assumptions presented by Alshiekh et. al. [86], these solutions guarantee that the risk of failure over trajectories for the system is minimized so long as the safety MDP is correct in a conservative manner to the behavior of the true MDP.

5.2 Reference Problems

To demonstrate the guidelines in the previous section, two reference operations problems are presented and transformed into POMDPs using the advice provided in the previous section. Implementations of these environments as used in this work are publicly available from the `basilisk-env` git repository [1] .

5.2.1 Simulation Framework

Both the training process for DRL-based operations agents and the verification framework require the ability to simulate a space mission to high fidelity. DRL techniques in particular can struggle when transferring from simulated to real experiences due to the "simulation gap," as DRL agents can over-fit on specific attributes of low-fidelity simulators which do not generalize to the real world. At the same time, a key benefit of DRL is the ability to learn on complex simulators without the introduction of intermediate approximations or simplifications of system behaviors. Verification techniques for autonomous agents also require the existence of high-fidelity, trusted simulation capability which adequately captures the behavior of the real system. For spacecraft, this requires the ability to simulate not only traditional astrodynamics components (orbital and attitude dynamics), but also the behavior of flight software components.

The Basilisk astrodynamics simulation package represents an ideal toolset for both of these applications. Specifically, Basilisk provides:

(1) **High Fidelity Astrodynamics:** The Basilisk dynamics engine can simulate fully-coupled multi-body dynamics in tandem with GPU-accelerated orbital dynamics [88], allowing for

[1] https://github.com/atharris/basilisk_env

the simulation of second- and third-order effects like attitude/orbit coupling, fuel slosh, and flexing panels.

(2) **Flight Software Simulation/Integration:** Developed as a tool to aid flight software development by providing a flight-like environment for testing, Basilisk provides first-class support for the integration of flight software components alongside a library of algorithms with flight heritage.

(3) **Computational Performance:** Compute-heavy code is written in C/C++ and is highly performant as a result; even with tasks like image generation in the loop, BSK-based simulations are thousands of times faster than real-time, allowing for rapid generation of samples for both DRL and verification algorithms.

(4) **Integration with common ML/RL frameworks:** Basilisk is wrapped with SWIG and provides a Python API for setting up, executing, and analyzing simulations, which allows it to be integrated with other common ML/RL packages (Tensorflow, Keras, gym, `scikit-learn`).

To facilitate the integration of Basilisk with other machine learning tools, a library of OpenAI `gym` environments which utilize Basilisk for spacecraft simulation has been created and opened to the public. This library supports common DRL frameworks such as OpenAI's `baselines` and the `stable-baselines` fork. A public version is available on GitHub.

In addition, this work has motivated the development of several extensions to and within the Basilisk framework to include simulation components necessary for the holistic, systems-driven operational problems this work aims to address. These extensions include the incorporation of high-fidelity models for atmospheric neutral density, attitude-dependent atmospheric drag, models of on-board power generation and storage, models of on-board data handling, and models for interactions between spacecraft and fixed locations on the ground (such as ground stations or imaging targets). Parts of the Basilisk simulation and messaging system were also overhauled, providing usability improvements for the community writ large.

5.2.1.1 Reference Problems

At present, three baseline mission operations problems have been identified and implemented using the aforementioned advice. These problems are briefly summarized here with associated results.

(1) **Mars Science Operations:** This is a hybrid systems regulation problem in which the learning agent must choose between conducting orbit determination, maneuvering based on their orbit knowledge to a target orbit, or collecting science data while in the target orbit. These operational modes are implemented as a set of linear dynamical modes; the agent is rewarded based on their proximity to the target orbit while collecting science data.

(2) **LEO Earth Observation:** This scenario considers a spacecraft operating in LEO that must maintain it's health status (battery charge, wheel speed) while maximizing its time spent observing the Earth. This scenario incorporates safety constraints.

(3) **LEO Coordinated Earth Observation**: This scenario considers a set of spacecraft in LEO that must image a ground target using a heterogeneous combination of sensors while maintaining spacecraft health.

5.2.2 Mars Station Keeping Task

5.2.2.1 Problem Description

Spacecraft conducting science operations typically need to maintain specific orbital parameters to achieve location-specific mission objectives. While trajectory designers seek to minimize the impact of perturbations on such trajectories, mis-modeling of these perturbations is inevitable and spacecraft typically conduct station-keeping burns at regular intervals. Because station keeping performance is coupled to the spacecraft's navigation accuracy and navigation processes are not run or updated constantly, orbit determination activities must be considered when sequencing station-keeping burns. This scenario simulates the high-level trade-offs between estimation, station-keeping burns, and science operations for a spacecraft which can only accomplish one mode at a time.

Due to hardware constraints, the spacecraft is capable of entering either estimation mode or control mode but not both at the same time; as a result, the spacecraft operations challenge is centered around managing the (unknown) true state error while maximizing observation time. To match the hybrid system assumption for dimensionality reduction described in Section 5.1.1.1, the estimation and control modes are implemented using piecewise-Hurwitz matrices that are stable in their respective states (i.e., estimation error decays exponentially in the estimation mode and control error decays exponentially in the control mode). To represent safety constraints, these modes fail to operate if the respective error state falls outside of a specified bound; this is representative of challenges presented by linear or linearized estimation and control approaches, which face challenges when operating outside of their linear regime.

The “true” non-linear dynamics resulting from gravity interactions are taken to follow the two-body equations of motion in the presence of perturbing accelerations:

$$\ddot{\boldsymbol{r}} = \frac{-\mu}{r^3}\boldsymbol{r} + \boldsymbol{a}_p \tag{5.18}$$

At the same time, a pre-defined reference trajectory obeying two-body dynamics without perturbing accelerations is used to define the desired mission:

$$\ddot{\boldsymbol{r}}^* = f^*(\boldsymbol{r}^*) = \frac{-\mu}{r^{*3}}\boldsymbol{r}^* \tag{5.19}$$

The erroneous propagator in Equation (5.19) is also used to propagate forward the spacecraft's current orbital state estimate, $\hat{x}$. The resulting state, estimate, and control errors are defined as

$$\boldsymbol{e}_s = \boldsymbol{x} - \boldsymbol{x}^*,\ \boldsymbol{e}_{\text{est}} = \boldsymbol{x} - \hat{\boldsymbol{x}},\ \boldsymbol{e}_c = \hat{\boldsymbol{x}} - \boldsymbol{x}^* \tag{5.20}$$

The asymptotically stabilizing Cartesian continuous feedback control law for orbits defined in [7] is used in the control mode to define control accelerations that will lead back to the reference trajectory:

$$\boldsymbol{u} = -(f^*(\hat{\boldsymbol{x}}) - f^*(\boldsymbol{x}^*)) - [K_1](\hat{\boldsymbol{x}} - \boldsymbol{x}^*) - [K_2](\dot{\hat{\boldsymbol{x}}} - \dot{\boldsymbol{x}}^*) \tag{5.21}$$

where $\boldsymbol{u}$ is the control acceleration in the planet-centered inertial frame, f^* is the two-body equations of motion, and $[K_1], [K_2]$ are positive definite 3×3 matrices. This control law is chosen due to its amicable convergence properties, which allow $\hat{\boldsymbol{x}}$ to converge to $\boldsymbol{x}^*$ from arbitrary orbits (albeit at the cost of excessive fuel usage, which is not considered in this environment). The estimation process is modeled by approximating the dynamics of a well-tuned and robust Kalman Filter by stable and unstable estimate error vector and covariance matrix dynamics. When not in the estimation mode, the estimated state $\hat{\boldsymbol{r}}$ and covariance matrix $[P]$ are propagated by:

$$\ddot{\hat{\boldsymbol{r}}} = f^*(\hat{\boldsymbol{r}}),\ [P] = [P] + [Q] \tag{5.22}$$

which reflects the drift of the mean estimate due to mis-modeled dynamics and the steady growth of the covariance matrix due to process noise. In the estimation mode, the error vector explicitly computed and propagated separately with exponentially decaying dynamics across all states, with some noise added to the estimate to represent additional sensor noise:

$$\boldsymbol{e}_{\text{est}} = \boldsymbol{x} - \hat{\boldsymbol{x}},\ \dot{\boldsymbol{e}}_{\text{est}} = [A_{\text{est}}]\boldsymbol{e}_{\text{est}} + \boldsymbol{q},\ \hat{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{e}_{\text{est}} \tag{5.23}$$

where $[A_{\text{est}}]$ is a diagonal Hurwitz matrix and $\boldsymbol{q}$ is a normally distributed random vector. The full MDP statement for this problem is therefore:

$$P = \begin{cases} s & = \{\boldsymbol{r} \in \mathbb{R}^3,\ \dot{\boldsymbol{r}} \in \mathbb{R}^3, \boldsymbol{r}^* \in \mathbb{R}^3,\ \dot{\boldsymbol{r}}^* \in \mathbb{R}^3\} \\ o & = \{\boldsymbol{e}_s \in \mathbb{R}^6, \boldsymbol{e}_{\boldsymbol{c}} \in \mathbb{R}^6, \boldsymbol{\sigma} \in \mathbb{R}^6\} \\ a & = \{\text{Mission}, \text{Orbit Determination},\ \text{Orbit Control}\} \\ T & = \{f_{\text{Mission}},\ \ f_{\text{Orbit Determination}},\ f_{\text{Orbit Control}}\} \\ R & = \{R_s, -1 \text{ if } \boldsymbol{e}_s > \boldsymbol{e}_{s,crit} | \boldsymbol{e}_c > \boldsymbol{e}_{c,crit}\} \end{cases} \tag{5.24}$$

While simple in its dynamics and implementation of spacecraft estimation and control constraints, this problem reflects real-world challenges in managing couplings between state estimation and control for real spacecraft. Because the spacecraft can only control with respect to its current state estimate, failing to reduce its estimation error can cause divergence in the true state error as the spacecraft computes burns that inaccurately reflect the current state error. As a result, this problem tasks an agent with managing both mis-modeled dynamics, coupling of estimation and control processes, and optimization of total mission science time given an orbit accuracy constraint while remaining computationally quick to execute.

5.2.2.2 Shield Construction

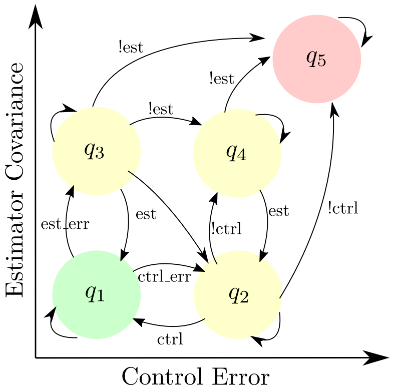


Figure 5.8: Safety MDP constructed for the station keeping task.

The primary challenge for safety properties in this environment is ensuring that the agent has enough knowledge of its true state–obtained by entering the estimation mode–to correctly understand which state it should enter. The discretized system used to represent the safety game is shown in Figure 5.8. If the agent's estimator covariance is low and its state error is close to the linearity constraint, the shield will force the agent to conduct a station-keeping burn; if both errors are high, the graph is constructed to bias the agent towards conducting estimation modes, to ensure that the estimated errors are actually as large as they believe.

5.2.3 LEO Attitude and Health Management Task

5.2.3.1 Problem Description

To represent the feasibility of applying DRL techniques to spacecraft health-keeping as well as station keeping, a scenario reflecting the challenges of day to day operations in Low Earth Orbit

(LEO) is presented. A spacecraft on a pre-determined trajectory around the Earth is tasked with maximizing its time spent conducting an Earth observation task (represented by a nadir pointing attitude) while maintaining on-board power and dumping excess reaction wheel momentum. This scenario is implemented in the Basilisk software framework, and uses existing flight heritage control laws and hardware models. A representative block diagram of the simulation components is shown in Figure 5.9; further documentation on these components and their functionality can be found in the Basilisk documentation[2] . As a result, this environment represents a more realistic challenge for prospective planning and scheduling approaches, as agents interact directly with a simulation stack intended for ADCS design and verification rather than an approximation of those systems made more amicable to planning approaches.

Stated formally, the full-system POMDP under the assumption of full observation of system states on-board is provided by Eqn. 5.25:

$$P = \begin{cases} s & = \{\boldsymbol{r} \in \mathbb{R}^3,\ \dot{\boldsymbol{r}} \in \mathbb{R}^3,\ \boldsymbol{\sigma}_{BN} \in \mathbb{O}^3, \boldsymbol{\omega}_{BN} \in \mathbb{R}^3, \boldsymbol{\omega}_{RW} \in \mathbb{R}^3,\ \mathrm{J}\in \mathbb{R}^1\} \\ o & = \{\sigma_{BN} \in \mathbb{O}^3, \boldsymbol{\omega}_{BN} \in \mathbb{R}^3, \boldsymbol{\omega}_{RW} \in \mathbb{R}^3,\ \mathrm{J}\in \mathbb{R}^1\} \\ a & = \{\text{Science}, \text{Charge Mode},\ \text{Desaturation Mode}\} \\ T & = \{f_{\text{Nadir Pointing}},\quad f_{\text{Sun Pointing}},\ f_{\text{Desaturation}}\} \\ R & = \{r_s, -1 \text{ if } J = 0 \text{ or } |\boldsymbol{\omega}_R W| > 250\frac{rad}{s}\} \end{cases} \tag{5.25}$$

In this case, it is assumed that an operations agent would observe all relevant on-board dynamic information as they are observed or estimated. The reward function is engineered to provide the agent with a positive reward that is inversely proportional to the attitude error σ_{BR} when in the science mode:

$$r_s = \frac{1}{\sigma_{BR}^T \sigma_B R + 1} \tag{5.26}$$

In general attitude transients may not settle out within one timestep, even with proper time-step size selection. In addition, there is a tradeoff between maintaining the Markov property and the granularity of decision-making intervals that can be challenging to resolve; longer timesteps are

[2] http://hanspeterschaub.info/basilisk/index.html

more likely to resolve transient behavior, but result in fewer planning intervals over a set period of time and therefore less flexibility for agent responses in other non-attitude system domains. By rewarding agents for entering science mode with small attitude errors, this reward function ensures that agents that must take multiple steps in the science mode to settle out transient behavior are properly rewarded. Rewards are scaled such that the maximum achievable reward over one environment run is 1; this simplifies reward engineering for failure states, which are defined as providing a reward of -1 and ending the scenario, ensuring that the maximum reward for a failed run is 0. This strategy has several appealing properties: it simplifies analysis of agent performance, as best- and worst-case scores are knowable; it clarifies cases wherein agents fail; it is unit norm, and therefore unlikely to cause numerical issues; rewards are continuously shaped and convex around a desired objective, but also simple to implement.

In addition to this structured reward, feature engineering inspired by the LDR approach described in Section 5.1.1.1 is also applied to the base POMDP described by Eqn. 5.25. Rather than observing the current attitude and reference states separately, the agent is instead provided with the magnitude of the error MRP state, which compactly represents the overall attitude error. Similarly, the overall body to inertial angular velocity is reduced to the norm angular velocity, as is the reaction wheel speed vector. These modifications reflect the intended behavior of each mode and summaries relevant to the reward function. In this problem, applying LDR reduces the system dimensionality from 13 individual elements to 5 elements:

$$P = \begin{cases} s & = \{\boldsymbol{r} \in \mathbb{R}^3,\ \dot{\boldsymbol{r}} \in \mathbb{R}^3,\ \boldsymbol{\sigma}_{BN} \in \mathbb{O}^3, \boldsymbol{\omega}_{BN} \in \mathbb{R}^3, \boldsymbol{\omega}_{RW} \in \mathbb{R}^3,\ \mathrm{J}\in \mathbb{R}^1\} \\ o & = \{|\sigma_{BN}| \in \mathbb{R}^1, |\boldsymbol{\omega}_{BN}| \in \mathbb{R}^1, |\boldsymbol{\omega}_{RW}| \in \mathbb{R}^1,\ \mathrm{J}\in \mathbb{R}^1\} \\ a & = \{\text{Science}, \text{Charge Mode},\ \text{Desaturation Mode}\} \\ T & = \{f_{\text{Nadir Pointing},}\quad f_{\text{Sun Pointing},}\ f_{\text{Desaturation}}\} \\ R & = \{r_s, -1 \text{ if } J = 0 \text{ or } |\boldsymbol{\omega}_R W| > 250\frac{rad}{s}\} \end{cases} \tag{5.27}$$

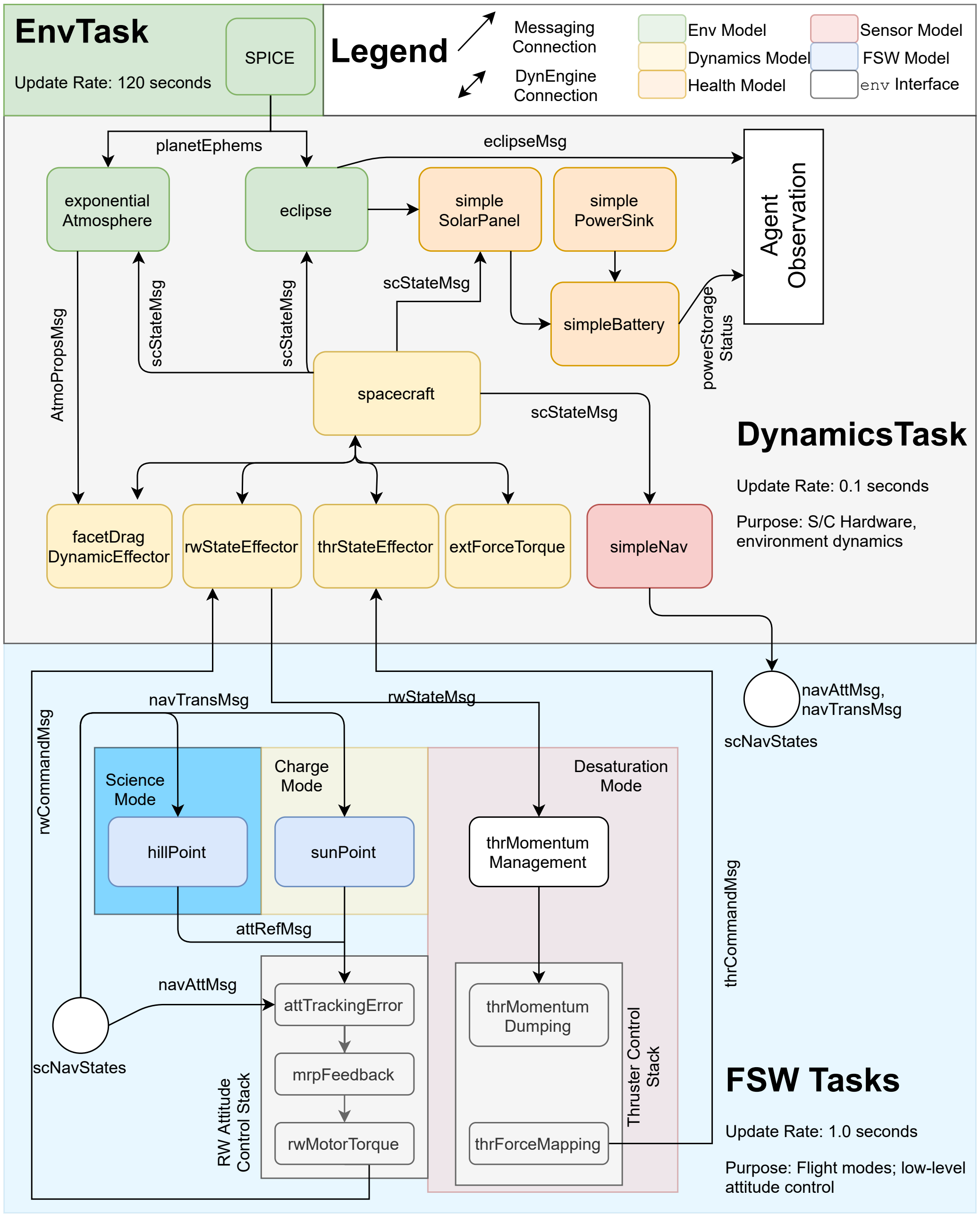


Figure 5.9: Simulation block diagram for the LEO attitude and health management task.

5.2.4 Shield Design

Owing to its higher complexity, the shield design problem for the LEO attitude mode selection problem is substantially more complex. In this case, the system is again discretized in accordance to safety-relevant states along expert-defined operational thresholds listed in 5.1, resulting in the safety MDP shown in Figure 5.10. However, the reaction wheel desaturation controller computes the momentum to be removed by thruster impulses under the assumption that the spacecraft is near-stationary; as a result, triggering this mode (action 2 in Figure 5.10) will destabilize the spacecraft, potentially *increasing* the momentum in the reaction wheels and triggering a failure state. To prevent this, additional states representing a combination of one or more of the safety conditions *and* tumbling above the body rate specified in Table 5.1 are added to the safety MDP; it is assumed that these states can be exited by entering the sun-pointing mode, which shares a reference attitude with the desaturation mode for simplicity.

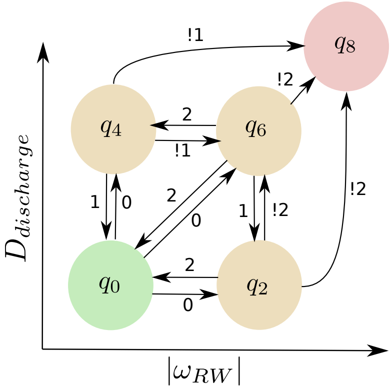


Figure 5.10: Safety MDP constructed for the LEO attitude mode planning simulator. $D_{\text{discharge}}$ represents the depth of discharge (i.e., $1 - J$) Modes relating to “tumble” states with large body rates are omitted for clarity.

Table 5.1: Safety MDP labeling parameters

Observed Variable	Operational Limit	Safety Limit
$\lvert\omega_{BN}\rvert$	0.05 rad/s	N/A
$\lvert\omega_{RW}\rvert$	1,000 RPM	1,500 RPM
J_{stored}	5 W-Hr	0 W-Hr

5.2.5 LEO Coordinated Sensing Task

Coordinating sensor information from heterogeneous sensors located on multiple heterogeneous satellites represents a prototypical multi-agent coordination problem faced by future space missions. The LEO coordinated sensing task models this challenge by considering observations of

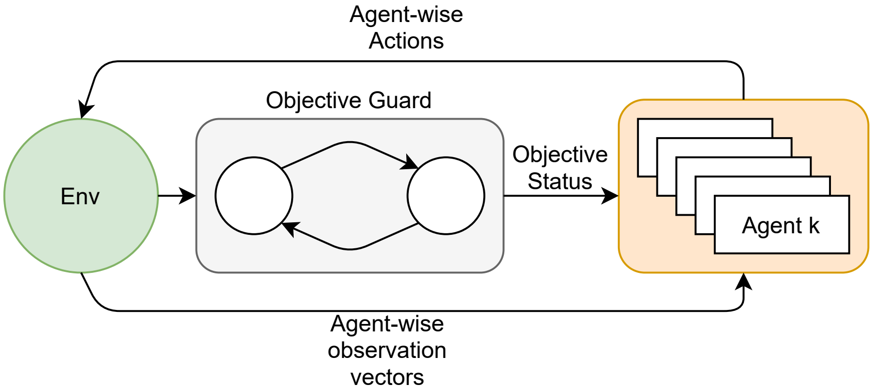


Figure 5.11: Diagram of how image requests are simulated for the LEO Coordinated Sensing task.

a ground target by a set of spacecraft in different orbits; each spacecraft is assumed to have both sensors on-board, but is limited to activating only one at a time. In addition, the hardware and attitude constraints enforced by the LEO attitude mode management problem are present for each spacecraft in the constellation. In general, the objective of this mission is to ensure the relevant image type is taken at an appropriate time while the spacecraft has access to the target. In a true operational scenario, these image type requests would be specified by operators on the ground, and will vary with both incoming information and changing weather or environmental conditions, factors which are challenging to simulate in general. For the purposes of this reference task, image requests are updated each time any agent takes an image by an objective guard function as shown in Figure 5.11. The guard behavior is selected for the reference task to point towards the observation type with the longest latency, i.e. the image type that was not taken at the last image opportunity. This simple behavior nevertheless results in a challenging problem in which agents must learn to weight the objective coordination information relayed by the environment.

To demonstrate additional capabilities, each spacecraft is also granted a small maneuver budget and the ability to plan and execute maneuvers that attempt to minimize the longitude miss over the ground target latitude. These maneuvers are calculated to eliminate the longitude miss distance at the next latitude pass:

$$\lambda_{\text{miss}} = \lambda_f - \lambda_{\text{target}} \tag{5.28}$$

where λ_f is the longitude of the next pass at the target latitude and λ_{target} is the target longitude. The spacecraft true latitude, θ, at the target latitude can be computed by:

$$\theta^* = \arcsin\big(\frac{\sin(\phi_{\text{target}})}{\sin(i)}\big) \tag{5.29}$$

where θ^* is the true latitude (defined as $f + \omega$) and ϕ_{target} is the target latitude. Under the assumption of circular orbits $\dot{f} = n$, the difference between the target true latitude and current true latitude is used to compute the time to the next pass:

$$\Delta t = (\theta - \theta^*)/n \tag{5.30}$$

This time is used to propagate the current state forward; at the critical time, the pass longitude is computed by rotating the inertial spacecraft position into the ECEF frame and calculating the longitude:

$$\lambda_f = \arctan\big(\frac{{}^{\mathcal{F}}r_2^*}{{}^{\mathcal{F}}r_1^*}\big) \tag{5.31}$$

Maneuvers are specified to occur at the point of maximum or minimum true latitude, as this is the ideal point for performing pure RAAN change maneuvers [7]. While the magnitude of the burn can be found in a straightforward manner from the Gaussian Variational Equations, the burn direction must be specified to rotate the current velocity vector without impacting its magnitude (i.e., the current and final velocity vectors retain the same magnitude but have different directions).

The resulting POMDP resembles an augmented version of the LEO attitude health management problem, but agents observe additional information about their inertial position, the target inertial position, whether they currently have access to the target, their remaining delta-v budget, the delta-v of a maneuver, and the current longitude miss distance; other dynamics and spacecraft parameters are unchanged from the LEO attitude health management problem. As a result, the safety constraints and shield definition are re-used from the LEO attitude health management problem. These factors result in the following POMDP specification:

$$P = \begin{cases} s & = \{\boldsymbol{r} \in \mathbb{R}^3,\ \dot{\boldsymbol{r}} \in \mathbb{R}^3,\ \boldsymbol{\sigma}_{BN} \in \mathbb{O}^3, \boldsymbol{\omega}_{BN} \in \mathbb{R}^3, \boldsymbol{\omega}_{RW} \in \mathbb{R}^3,\ \text{J}\in \mathbb{R}^1\} \\ o & = \{\boldsymbol{\sigma}_{BN} \in \mathbb{O}^3, \boldsymbol{\omega}_{BN} \in \mathbb{R}^3, \boldsymbol{\omega}_{RW} \in \mathbb{R}^3,\ J \in \mathbb{R}^1, \Delta v_{\text{budget}} \in \mathbb{R}^1, \lambda_{\text{miss}} \in \mathbb{R}^1, a_{\text{ground}} \in \mathbb{R}^1, s_{\text{type}} \in 0, 1\} \\ a & = \{\text{Mission}, \text{Orbit Determination},\ \text{Orbit Control},\ \text{Maneuver Planning}\} \\ T & = \{f_{\text{Mission}},\ \ f_{\text{Orbit Determination}}, f_{\text{Orbit Control}}\} \\ R & = \{R_s, -1 \text{ if } J = 0 \text{ or } |\boldsymbol{\omega}_{RW}| > 314\frac{rad}{s}\} \end{cases} \tag{5.32}$$

5.3 Feasibility Analysis

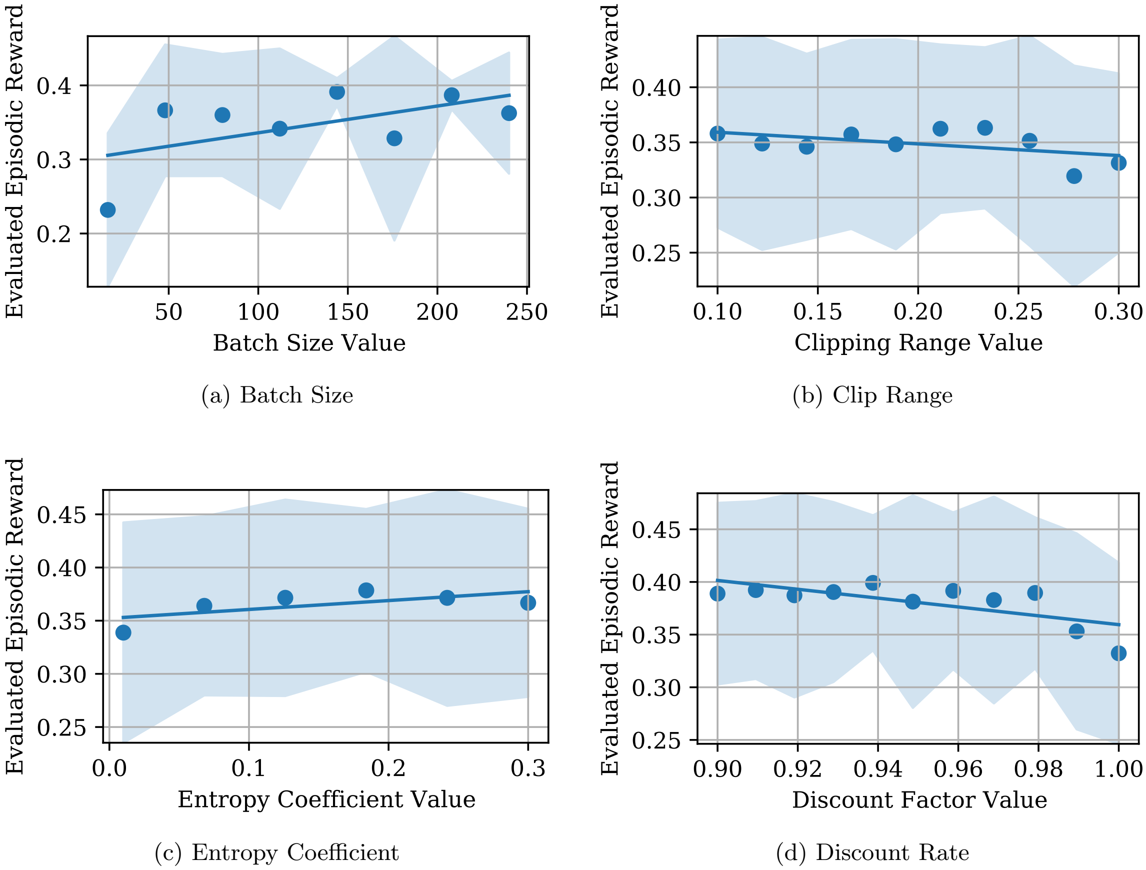
This section aims to identify the challenges associated with achieving acceptable tasking performance on the tasks described in Section 5.2.1.1 and the attributes of successful policies on said tasks. Specifically, this section seeks to address concerns relating to classic criticisms of DRL algorithms and their resulting policies; that results are heavily dependent on hyperparameter selection and random seed, that policies are overfit on environment conditions and fail to generalize, and that results are often comparable to other, simpler optimization strategies. In addition to addressing these challenges, the intrinsic dimension of the Mars station-keeping task and LEO attitude health management task are estimated and contextualized with other common benchmarks in the deep learning community.

5.3.1 Hyperparameter Sensitivity

Most deep learning approaches are dependent on proper selection of hyperparameters such as learning rate, network size, or reward discount factors for good performance; indeed, on many classic DRL tasks the selection of correct hyperparameters can be the distinction between successful agents and policies that are worse than random. It is expected that through the correct construction of spacecraft operations policy problems that this extreme sensitivity to hyperparameter selection can be avoided. To examine this sensitivity, DRL agents were trained over multiple random seeds at a grid set of hyperparameters listed in Table 5.2 in both the simple-science and LEO attitude management environments; after training, simple linear fits were performed on both sets of data to establish the sensitivity of agent returns to these hyperparameters.

Table 5.2: Parameters and Parameter Ranges used in hyperparameter search

Parameter	Baseline Value	Range
Discount Factor γ	0.99	(0.9,1)
Batch Size n_{steps}	64	(32,240)
Clip Range ϵ	0.3	(0.1,0.3)
Entropy Coefficient	0.1	(0.,0.3)
Network Shape	(64,64)	N/A



(a) Batch Size

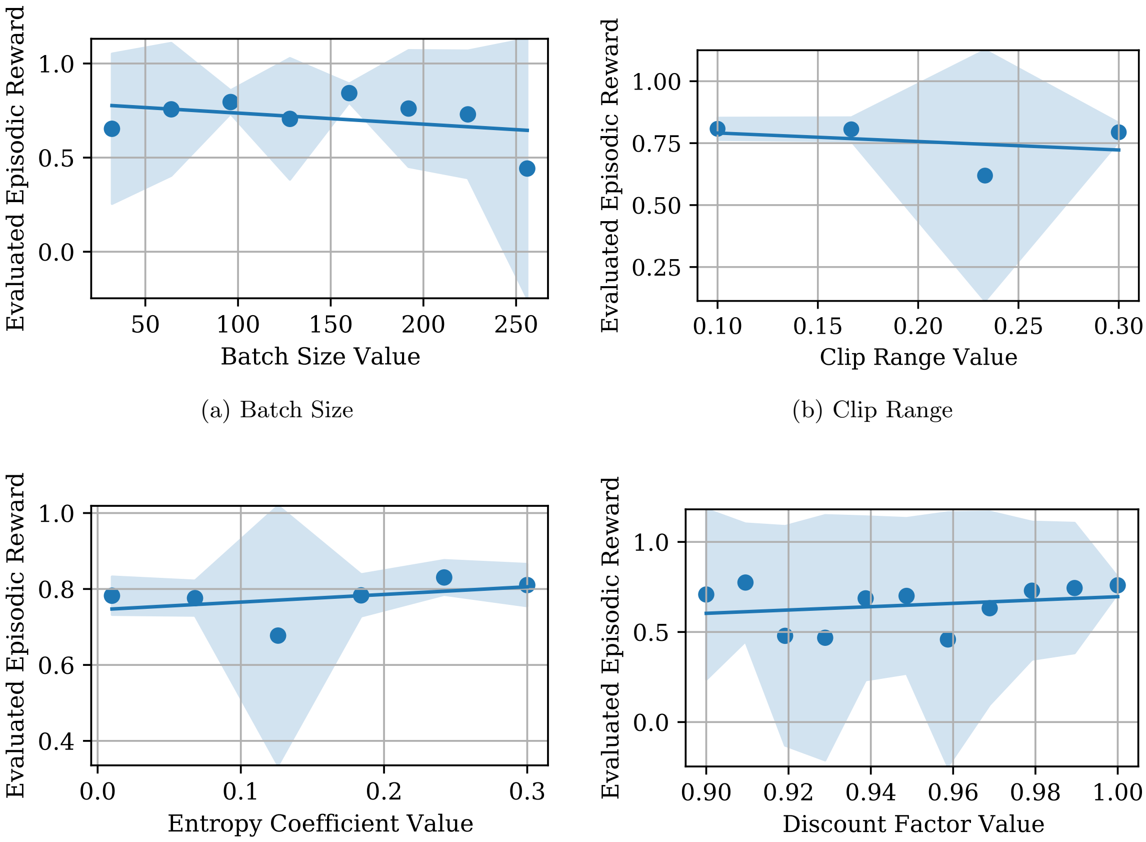
(b) Clip Range

(c) Entropy Coefficient

(d) Discount Rate

Figure 5.12: StationKeep performance across selected hyperparameters. Dots represent mean performance, shaded regions indicate 1-σ covariance bounds.

The results of this survey are shown in Figs. 5.12-5.13. In general, we find weak correlations



(a) Batch Size

(b) Clip Range

(c) Entropy Coefficient

(d) Discount Rate

Figure 5.13: leoPowerAttEnv performance across selected hyperparameters.

between both overall agent performance and specific hyperparameters, with the exception of the discount factor γ, which generally produces better performance at higher values in the LEO attitude mode selection problem. High discount factors reflect long periods of viability for rewards in a specific environment, meaning that rewards (and penalties) should propagate backwards farther during training. This result suggests that the LEO operations problem has a complex and long-lived dynamics which must be accounted for in the planning process.

5.3.2 Performance Comparison

This analysis aims to compare the performance of DRL-derived agents against other black box approaches and heuristic state-driven algorithms. To this end, a GA-based optimizer was used to search over ideal mode-based schedules for both of the problem environments; to mitigate performance losses associated with brittleness, a 'nominal' set of initial conditions was selected for both environments that is intended to represent realistic, feasible operational constraints a tasking agent may be faced with in flight; as a result, the GA-optimized timeline for that operational condition is treated as an upper bound on the optimal reward achievable in that condition. In addition to the GA-based approach, a heuristic agent based on the safety shield strategy was also devised, following the approach described in Alg. 2. This approach is used as a stand-in for simple state-machine driven autonomy approaches.

Algorithm 2: Heuristic Greedy-Safe Action Selection

```
Result: $a$
$q_k$ = ShieldDiscretizer($\boldsymbol{o}_k$);
if $q_k \in Q_{\textbf{nominal}}$ then
    $a$ = Reward Mode;
else
    $a$ = ShieldPolicy($q_k$)
end
return $a$
```

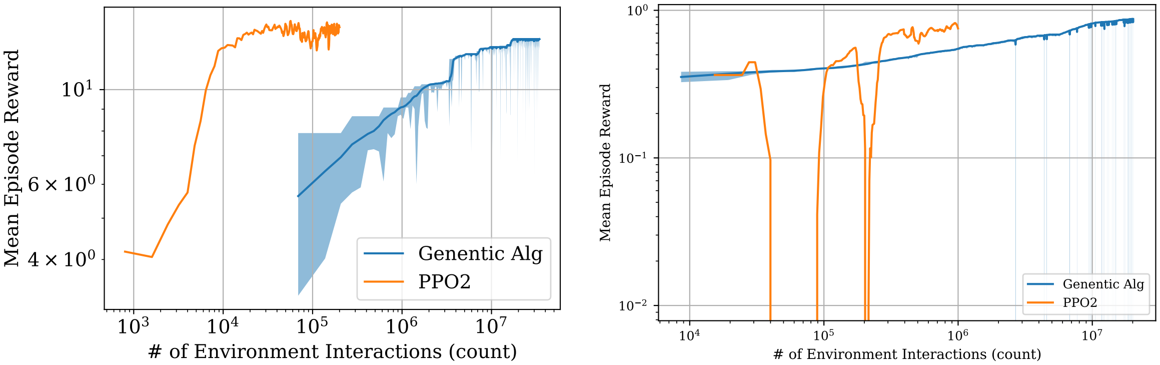
Given the high-level nature of the designed reference environments, a genetic-algorithm based scheduler was implemented to ground the results of the DRL-driven responsive tasking approach. The genetic algorithm, built on the DEAP evolutionary computing toolbox, encodes an action sequence for a given agent as a list of integers reflecting operational modes, using one mode per decision interval and the same decision interval and overall timeline as the DRL agents. The parameters and selection mechanism for this GA are listed in Table 5.3. A comparison of the final evaluated reward between the heuristic agent, GA-based scheduler and the DRL-based agents is shown in Table 5.4. While the GA-driven approaches generally perform 3-5% better on the reward metrics in the nominal environment, they struggle to find operational timelines that are well-suited to a variety of initial conditions. On the other hand, the heuristic agents utilizing the shield policy find better-than-random performance but generally do not match the performance of either the DRL or SDRL agents. This result suggests that the addition of DRL to the mode optimization problem provides benefits to mission performance beyond heuristic policies alone.

Table 5.3: Timeline-Optimization Genetic Algorithm Parameters.

Parameter	Value
Selection criteria	Tournament
Tournament Size	3
Crossover mechanism	None
Mutation Mechanism	Uniform w/ probability 0.01
Mutation Probability	0.75
Generation Size	24

Table 5.4: Summary of performance for PPO2, Shielded, Timeline, and Heuristic agents

Algorithm	Station Keep - Demo	Station Keep - Random	LEO Attitude - Demo	LEO Attitude - Random
Heuristic	0.0001779 ± 4.524 $\times 10^{-7}$	0.002402 ± 2.5299 $\times 10^{-7}$	0.8500 ±0.000	0.7372 ± 0.09812
GA	0.1513 ± 0.0001408	0.1445 ± 0.000557	0.8700 ±0.00	-0.5576 ± 0.03426
PPO2	0.2983 ± 7.4785$\times 10^{-5}$	0.3001 ± 0.000152	0.8024 ± 1.381$\times 10^{-4}$	0.7624 ± 0.05000
Shielded PPO2	0.2955 ± 9.489$\times 10^{-5}$	0.2919 ± 0.01	0.8406 ± 5.4521$\times 10^{-5}$	0.8038 ± 0.0001516



(a) StationKeep environment

(b) LEO Attitude Environment

Figure 5.14: Comparison of training curves versus environment interaction count for both PPO2 and a timeline-driven GA.

One notable area of difference between both approaches is the computational complexity induced by the timeline-driven approach. In this case, evaluating one gene requires a full run through the simulation environment; given that large populations are typically required for good convergence properties, this approach requires the dedication of substantial computational resources to simulating these action trajectories. While DRL is frequently described as data-intensive, Figure

5.14 shows that the DRL agent approaches similar mean episodic rewards to the maximum produced by the genetic algorithm approach while requiring 10-100 times fewer environment evaluations. This can be attributed to the fact that the GA-driven optimizer does not utilize information about the environment dynamics outside of the reward associated with a specific action sequence, whereas the DRL approaches by definition operate on and learn the relationships between observed states, actions, and rewards.

To demonstrate the relative merits of DRL-based policies for general operations, the genetic algorithm scheduler, heuristic policy agent, and best-performing PPO2 and SPPO2 approaches were evaluated on 100 initializations of both the demonstration and training environments for both scenarios; the resulting mean reward and 1-σ bounds are listed in Table 5.4. On the demonstration environment used to construct the point solution produced by the GA, DRL-based approaches produce comparable mean rewards, falling 2-5% short on the LEO attitude management environment and actually exceeding the performance of the GA on the Station Keeping environment. When considering environments with randomly sampled initial conditions, the DRL approaches out-perform the point solution produced by the GA in all circumstances. Importantly, both the default PPO2 implementation and the SPPO2 extension out-perform the greedy heuristic agent on both tasks, demonstrating both the increased performance possible with the adaptation of DRL versus hand-tuned policies and the benefit of combining correct-by-construction approaches with DRL techniques via shielding.

5.3.3 Sensitivity to Environment Parameters

The viability of this work hinges on the ability to train data-intensive DRL agents in simulation before applying them on-board. A natural shortcoming of this approach is the fact that simulations may not reflect the exact environment in which an agent might be deployed, resulting in degraded behavior; this phenomenon is described as the “simulation gap” in deep learning literature. Modeling errors are classified as either parametric errors (wherein dynamics are modeled correctly but constants that govern those dynamics are mis-matched) or systemic errors (wherein additional dynamics are not included in the training model). During nominal operations, mission analysts are typically able to account for systemic errors to an extremely high degree of precision. Prior work in astrodynamics problems has shown that perturbation methods, which assume small variations from a prescribed dominant dynamical regime, work very well for spacecraft trajectory design and navigation problems. For these reasons, it is desirable to evaluate the robustness of each agent architecture to parametric uncertainties, such as mis-modeled spacecraft inertias or environmental perturbation strengths.

Table 5.5: Parameters varied in Station Keeping and Attitude Management problems

Station Keeping Parameter	Range	Attitude Management Parameter	Range
J_2	$0.1 \times J_2 - 100 \times J_2$	m	200 – 400
$[Q]$	$0.1 \times [Q] - 100 \times [Q]$	P_{out}	3W – 7W

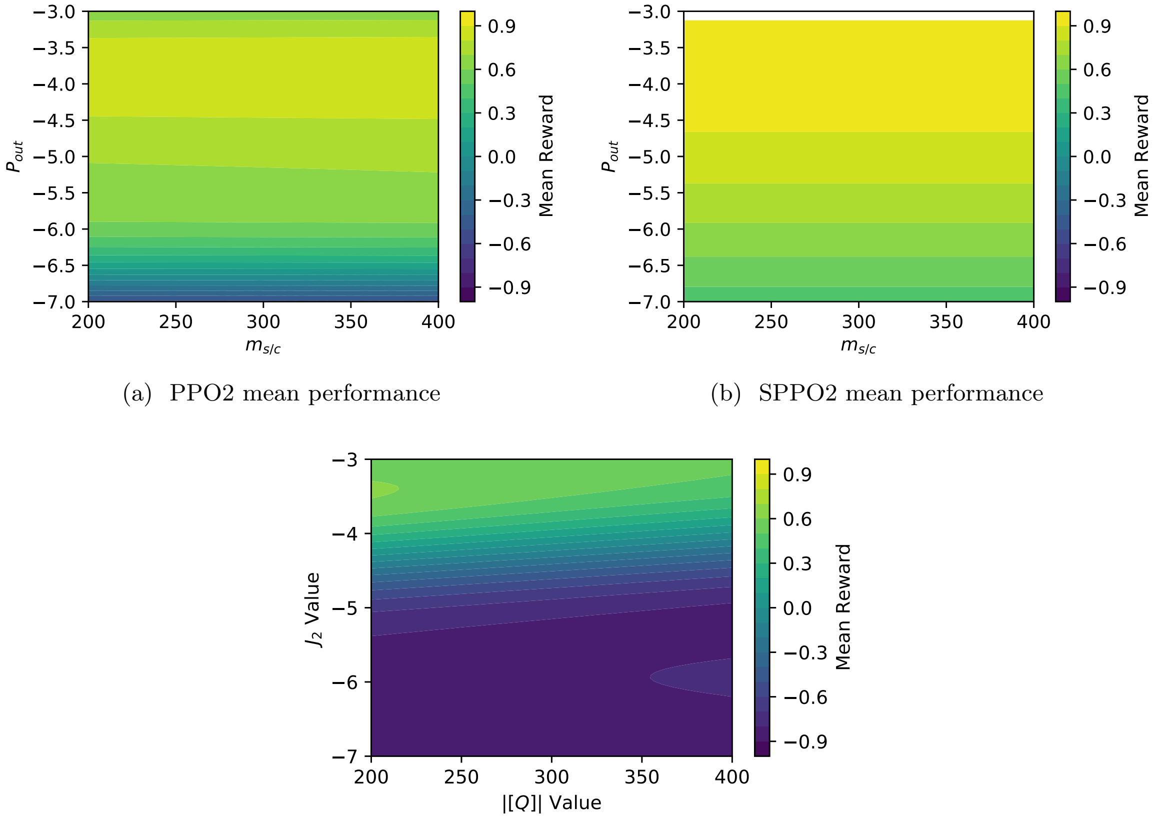
To demonstrate the empirical robustness of trained DRL algorithms in the reference scenarios described in Section 5.2, agents were evaluated against environments with parametric differences in their dynamics from the training environments; varied parameters and their ranges are listed in Table 5.5. Once again, the best-performing agents analyzed in Section 5.3.2 are used as benchmark agents for each approach. Each agent is run in the demonstration initial condition for each environment while taking 3 samples at each parameter combination; to evaluate mean performance, a Gaussian process regressor was fit to these samples to predict mean episodic reward of each algorithm, on each environment, at each set of varied parameters.

The resulting reward contours are shown in Figures 5.15-5.16. The LEO attitude management task shows a wide domain near the training condition in which the DRL-derived policies provide good performance, with degrading performance as power consumption increases and marginal differences in performance as the spacecraft mass and therefore inertia is varied. On the other hand, the GA-optimized timeline provides decent performance at the specific combination of mass and power consumption in the optimization environment, but degrades rapidly as either parameter is varied away from the reference condition. In the station keeping task, the performance of the DRL-based agent tends to improve as both the J_2 parameter and magnitude of optimization noise are shrunk, while the shielded PPO2 implementation shows broader areas of high performance and a higher overall floor on performance with parametric variation. In a reflection of the partially observable nature of this environment, the GA-optimized timeline does substantially well across a range of observation noise variables (because the GA optimizer does not take into account observations when identifying action sequences

In addition, these plots can also be considered as a reflection of the challenges presented by each environment; for example, small increases in power consumption in the LEO attitude management problem rapidly cause agents to fail. This is ultimately a result of the balanced power generation and consumption models used in the environment; the agent can generate, at most, 20 Watts of power in the sun-pointing mode. If an eclipse lasts 30% of an orbit–a common figure for the LEO environment during unfavorable beta angles–the agent can rapidly burn through its power reserve even before accounting for increases in power consumption; on the other hand, changes in mass and therefore inertia (and therefore the settling time and accuracy of pointing modes) has a more modest effect on overall system performance.

5.3.4 Intrinsic Dimension Survey

Finally, it is desirable to understand where the spacecraft operations procedure problem is situated with other common deep learning problems, such as image classification or game-playing. The intrinsic dimension approach described in Section 5.1.4 was applied to both the trained station-keeping agent and the trained attitude manager to evaluate the intrinsic dimension of this solution approach. In addition, the Station Keep environment was evaluated with multiple observation models, ranging from full observations of the agent's estimated and reference state, to observations of the estimated error vector and covariance, to simply the norm of the error vector and covariance (referred to as 'full','semi',and 'simple' respectively), reflecting increasing application of the LDR hypothesis described in Section 5.1.1.1. The resulting plots of reward vs network dimensionality are shown in Figure 5.17. Table 5.6 presents the intrinsic dimension of several common reference



(a) PPO2 mean performance

(b) SPPO2 mean performance

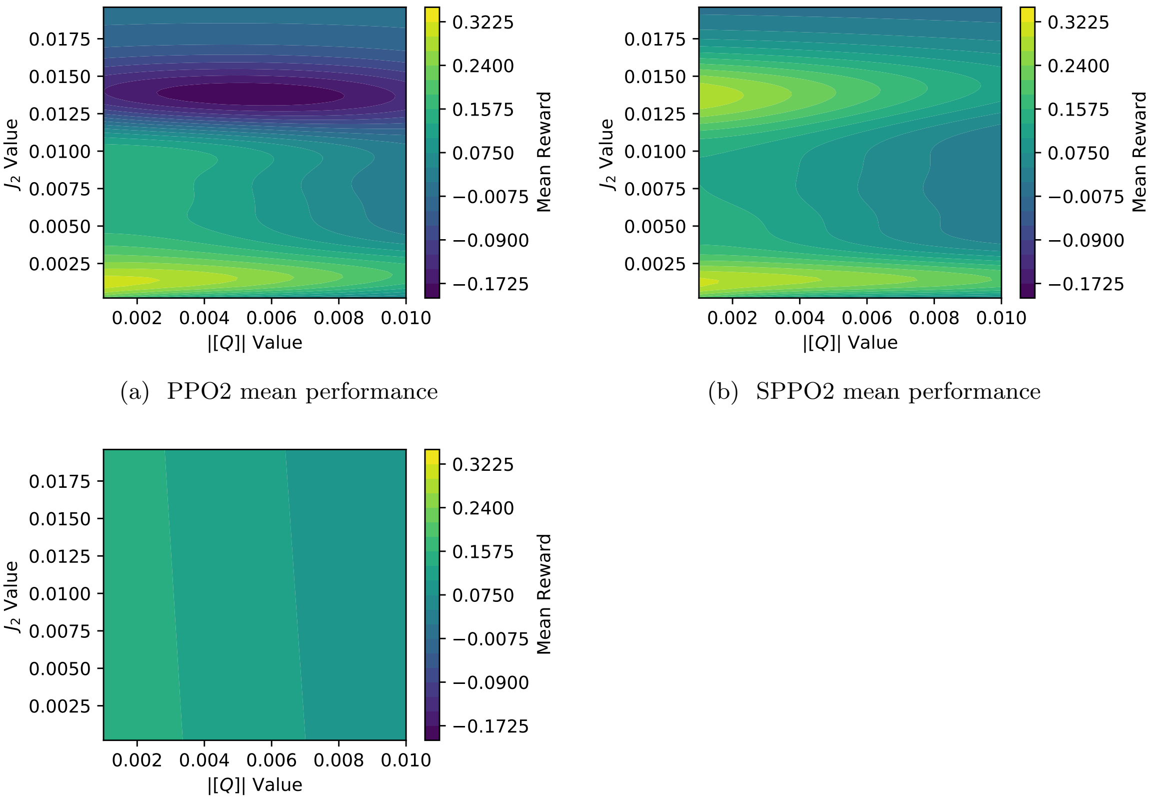
(c) GA-Optimized Timeline mean performance

Figure 5.15: Comparison of GA-optimized timeline and PPO2 on LEO attitude pointing task demonstration initial conditions with variation in spacecraft mass and power consumption.

problems and datasets in the broader machine learning community alongside these benchmark values.

Table 5.6: Intrinsic Dimension of PPO2 and SPPO2 Solutions for Station Keep and LEO Attitude Mode versus other problems in machine learning

Problem	d_{int90}
StationKeep - Simple Obs	2000
StationKeep - Semi Obs	¿10000
StationKeep - Full Obs	¿¿10000
LEO Attitude Management	≈ 1000-3500
CIFAR-10	2900-9000
Humanoid	700
Atari Pong	6000



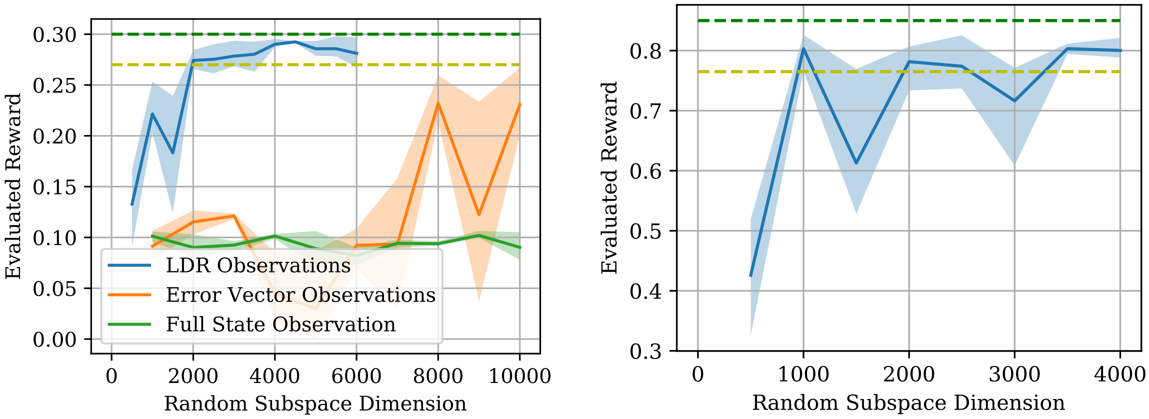
(a) PPO2 mean performance

(b) SPPO2 mean performance

(c) GA-Optimized Timeline mean performance

Figure 5.16: Comparison of GA-optimized timeline and PPO2 in StationKeep environment w/ demonstration initial conditions showing performance variation with J_2 and estimation noise changes in environment.

These results demonstrate several notable findings. First, it apparent that the reference problems presented herein are comparable in complexity to other classic deep learning and deep reinforcement learning benchmarks, falling between the image classification task CIFAR-10 and the deep reinforcement learning task Atari Pong in terms of intrinsic dimension. Second, these results suggest that acceptable performance can be obtained with substantially smaller neural networks than was originally used for training, a feature which is extremely important in the context of compute-constrained on-board decision-making. Finally, Fig. 5.17a demonstrates the relative advantage of the LDR hypothesis in terms of training complexity for systems which resemble switched hybrid systems in practice, demonstrating that solutions to the 'simplified' Station Keep environ-



(a) Station Keep

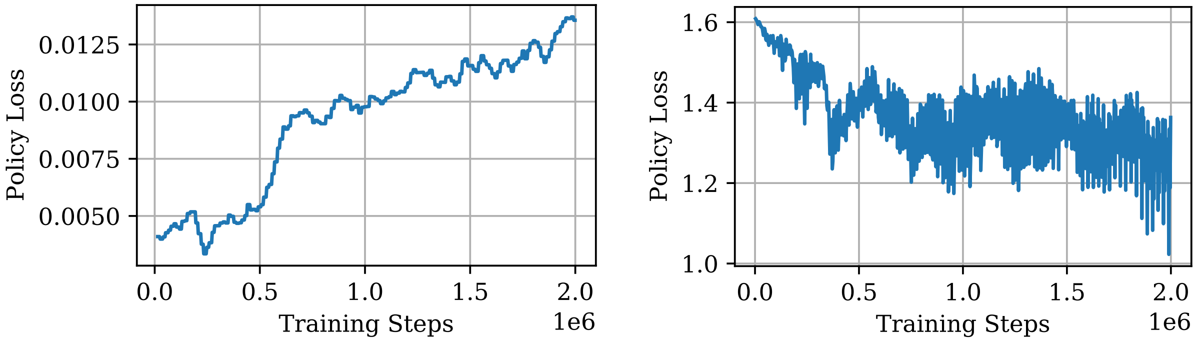
(b) LEO Attitude Guidance

Figure 5.17: Evaluated mean performance vs intrinsic dimension for PPO2 on LEO attitude guidance environment; shaded regions represent 1-σ performance bounds over 100 evaluations for 3 seeds, green dashed line represents benchmark performance, yellow dashed line represents 90% of benchmark performance.

ment can be obtained with smaller parameter sets than larger ones with no loss in performance. Taken together, these results suggest that the spacecraft operations problem can be solved with reasonably-sized neural networks that fall well within the current state of the art for the field of deep learning, especially when prior knowledge is leveraged in problem construction.

5.3.5 Application to Coordinated Satellite Tasking

Finally, the coordinated multi-satellite, multi-sensor, single-target problem defined in Section 5.2.5 is presented to demonstrate architectures for scalable, coordinated autonomy that leverages DRL. In this case, a single agent is trained on two orbit initial conditions in LEO and MEO, selected at random alongside various health initial conditions in the manner used for the station keeping and LEO attitude mode management tasks. Each orbit provides a handful of ground passes over the target region during a single episode. During training, substantial noise is apparent in both the mean episodic reward and policy loss as agents are thrown back and forth between the two orbit regimes. Because information about the target location and current desired image type is provided by the environment, an arbitrary number of actual satellites could be operated with only a single trained agent. Figure 5.18 demonstrates the impact of switched initial conditions on the training curves for the agent, with the mean episodic reward oscillating between extremes as the agent learns to maneuver in both orbits. This uncertainty is additionally reflected in the large policy entropy shown in Figure 5.18b After training, this agent was evaluated over 100 episodes and achieved an average reward of 0.014145 ± 0.0026, demonstrating a 50% improvement over an policy which ignored the state machine state and maneuvers ($R_{\text{max}} = 0.009259$). This performance demonstrates that for a single agent, SDRL is capable of learning both to meet the requested observation type and effectively maneuver to increase its observation time.



(a) Mean episodic reward in training

(b) Policy entropy during training

Figure 5.18: Mean performance and policy entropy during training on a single-satellite ground sensing task.

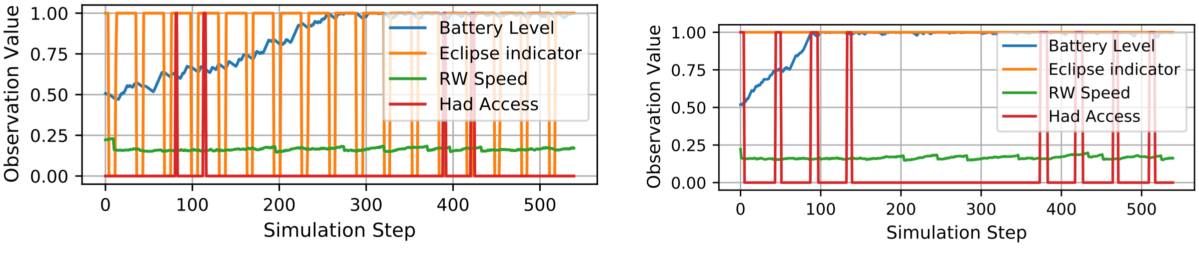
One key benefit of this training approach and architecture is the potential for generalization across a larger constellation. Other architectures, such as the treatment of the overall system as a single agent with concatenated state vectors and one-hot encoded action vectors, would scale poorly as the constellation size increases and require re-training at any time satellites are added or removed from the constellation. The single-agent/multi-satellite architecture, in which agents are trained to generalize across a range of initial conditions and operating conditions, would require training only one time and allow the addition of an arbitrary number of additional spacecraft.

To evaluate this generalizability, the trained agent was deployed across two satellites to form a coordinated imaging task in which both agents affect and must satisfy a changing image type

Table 5.7: Additional parameters for multi-satellite coordination run with $R = 0.06$ and $R_{\max} = 0.13$.

Parameter	Satellite 1	Satellite 2	Overall
Imaging Fraction	1.0	0.8793	N/A
Max Possible Reward	0.02	0.1074	0.1296
Achieved Reward	N/A	N/A	0.0600

request. Observation and action trajectories for both agents over a representative run for the task are displayed in Figure 5.19. These trajectories demonstrate that the pair of individual agents effectively generalize their understanding of safety states and dynamics to both spacecraft in spite of radically different orbital regimes, echoing the successful generalization of the LEO attitude health management agent. However, this same generalizability is only partly demonstrated in the performance results listed in Table 5.7. While both spacecraft effectively use their respective ground pass opportunities for sensing modes, they fail to utilize the correct imaging mode when requested, leading to an overall system reward that is half of the theoretical max. One explanation for this poor performance is the fact that, in training, the requested imaging mode only updates when the agent takes a particular action, leading to a lack of training information on how image type requests and reward is related. However, the promising generalization of image opportunity utilization demonstrates the merits of this approach.



(a) Satellite 1 states

(b) Satellite 2 states

Figure 5.19: Observation trajectories for Satellite 1 and 2 for the coordinated imaging task.

5.4 Conclusion

This work has established the viability of Deep Reinforcement Learning for generating operations procedures for next-generation space mission autonomy. Mode-based operations design, wherein specific software and hardware status combinations are represented by higher-level 'modes' provides both a common entry point for modeling day-to-day spacecraft operations problems as a mode-selection task that can be readily transformed into Markov Decision Processes. In addition, challenges inherent to the use of reinforcement learning for operations policy creation – such as safety and the use of existing information – have been addressed through the application of Shielded Reinforcement Learning. In comparison with both heuristic and timeline-optimization approaches, DRL-driven procedures provide comparable or improved performance with respect to mission objective satisfaction while generalizing to a wider range of initial conditions and parametric uncertainties, providing additional robustness.

Summary of Results

- **Role for Deep Reinforcement Learning:** This dissertation has identified potential roles for deep reinforcement learning techniques to play in spacecraft operations and command and control.
- **Problem Representation:** Techniques and best practices for representing spacecraft decision-making problems, such as Lyapunov dimensionality reduction, have been identified and presented.
- **Safety and Verification:** This work has demonstrated avenues for ensuring the safety of learning-based decision agents using shielded learning, including the efficacy of shielded learning versus reward engineering. Agents have additionally been verified against random initial conditions, demonstrating generalizability, and parametric variations in problem dynamics.

- **Scalability:** Estimates for the intrinsic dimensionality of reference problems in spacecraft operations have been identified and fall well-within the range of benchmark problems in deep learning. In addition, results from a demonstrative coordinated sensing task show that individual operations agents trained with DRL can be scaled and coordinated.

- **Simulation Capability and Reference Problems:** High-performance, high-fidelity simulation tools have been created by using and extending the Basilisk simulation framework. These tools have been used to create reference problems which are publicly available to the community.

Chapter 6

Autonomous, Health-Aware Mission Management for Aero-Assisted Missions

Building on the fundamental components of linear differential-drag controllers for low-impact orbit station-keeping and machine-learning driven autonomous mission operations procedure, this chapter aims to combine both to the management of a set of spacecraft in LEO that use differential drag to manage their relative orbital state.

6.1 Introduction

This scenario considers a pair of spacecraft in LEO that are maintaining a specified phasing offset from one another for mission-relevant purposes. Inspired by multi-satellite Earth observation missions with phasing constraints, such as the A-Train as described in Section 2.1, this scenario considers a set of spacecraft maintaining their respective positions within a one-plane constellation using differential drag as the primary means of actuation. Differential drag control is achieved by using the differential drag attitude guidance module derived in Chapter 3, tied into the LEO attitude health management environment and flight software stack as indicated in Figure 6.1. As a result, agents in this environment must learn to manage both system health states (battery charge, reaction wheel saturation) *and* maintain their phasing while maximizing time spent in their science modes. To further reflect the A-Train as a motivating mission, the simulated spacecraft is scaled up to small satellite mass (albeit with a large solar panel to match the low ballistic coefficient used for simulations in Chapter 4.) Table 6.1 describes the initial conditions and spacecraft parameters used in the simulation environment. To allow the results of Chapter 4 to be used as a point of comparison, the exponential atmospheric model is once again used as the truth atmosphere.

Table 6.1: Major simulation parameters, initial conditions, and distributions for the differential drag station-keeping task.

Parameter	Reference Value	Distribution
h	300 km	(290 km, 330 km)
Δy_0	700 m	(500m, 1000m)
ρ_0	2.2×10^{-11} $\frac{\text{kg}}{\text{m}^3}$	N/A
$J_{\max}$	1100 W-Hr	1100 W-Hr
A_i	3.0 m^2	
m_i	300 kg	
$C_{d,i}$	2.2	

6.2 MDP Design

6.2.1 Objective Reward Design

This scenario considers a pair of spacecraft in LEO that are maintaining a specified phasing offset from one another for mission-relevant purposes. Inspired by multi-satellite Earth observation missions with phasing constraints, such as the A-Train as described in Section 2.1. The aim of this MDP is to allocate as much time as possible to performing science observations while also remaining as close as possible to a designated orbit slot while subject to power and reaction wheel momentum constraints. Following the single-agent, multiple-spacecraft framework described in Section 5, this environment primarily considers the problem of an individual spacecraft station-keeping about its slot in a larger constellation, with the understanding that an agent trained to perform this task could be readily applied to spacecraft station keeping in separate slots. As a result, the attitude and mode-based reward function used in the LEO attitude mode management environment is modified to penalize the agent for relative position errors, especially errors beyond a desired radius ($r_{\max}$) :

$$r_o = \frac{r_{\text{mult}}}{(\boldsymbol{r}^T\boldsymbol{r})/r_{\max} + 1} \tag{6.1}$$

where $\boldsymbol{r}$ is the relative error state $\boldsymbol{r}_{sc} - \boldsymbol{r}_{ref}$ and $r_{\max}$ is the desired pointing accuracy. This reward structure provides an incentive for the agent to provide marginal improvements to its station-keeping accuracy within the reference distance, while additionally providing a lower shaped reward to intermediate positions outside $r_{\max}$.

6.2.2 Action and State Design

To achieve this relative station-keeping objective, the tuned attitude-driven differential drag control law defined in Chapter 4 has been added to the spacecraft simulation stack, thereby extending the action space available to the agent; this is reflected in the addition of the differential drag attitude guidance stack shown in Figure 6.1. Here, several of the operational benefits for the attitude-driven differential drag controller derived in Chapter 3 are made apparent; while within the domain of validity for the controller's linearization, the drag-driven attitude guidance law causes the relative position and velocity to decay towards 0. The attitude of the reference spacecraft is chosen to coincide closely with the attitude used for the science mode, being offset by 45 degrees in yaw alone and ensuring that only small maneuvers are required to station-keep. To ensure that the Markov condition holds for this new attitude objective, heterogeneous time-steps are taken depending on the action type, in accordance with the state-action selection paradigm described in section 5.1.1.1; steps taken in the differential drag mode are executed for five times longer than a typical step (15 minutes versus 3 minutes in other modes). In the same vein, the state observation for the relative position is reduced to observations of the relative distance and velocity rather than the full state vectors. These observations are additionally normalized by $r_{\max}$.

Two atmospheric density models are considered; the simple exponential model used in the differential drag sections, fit about a point evaluation for NRLMSISE-00, and NRLMSISE-00 itself with space weather parameters drawn from the NOAA database for August 2016. In a similar manner, this work considers three observation models for atmospheric density:

(1) No observation: Under this model, atmospheric density and related factors are treated as pure environmental dynamics and are not provided to the decision-making agent for consideration.

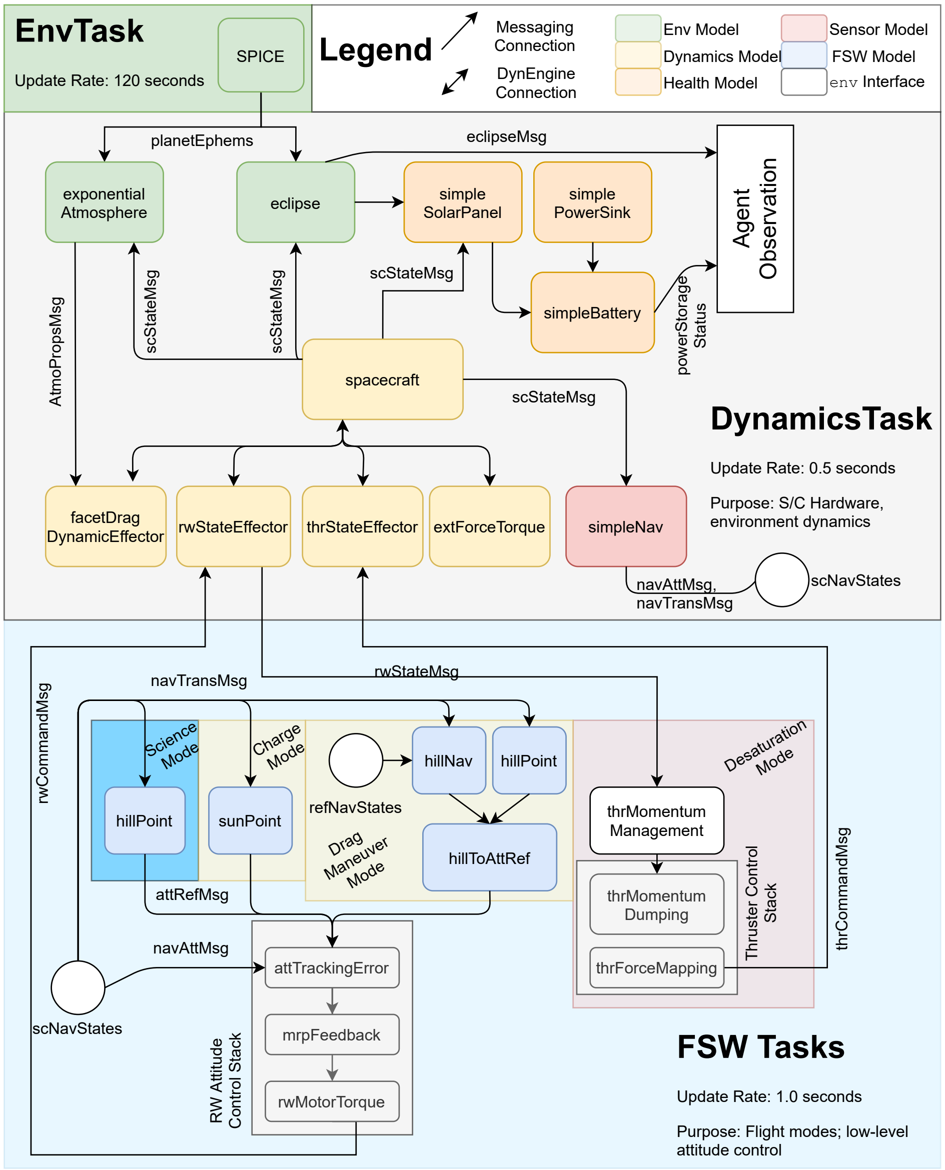


Figure 6.1: BSK Simulation block diagram for the differential drag station keeping task.

(2) Direct observation: In this model, the local neutral atmospheric density is provided directly to the agent.

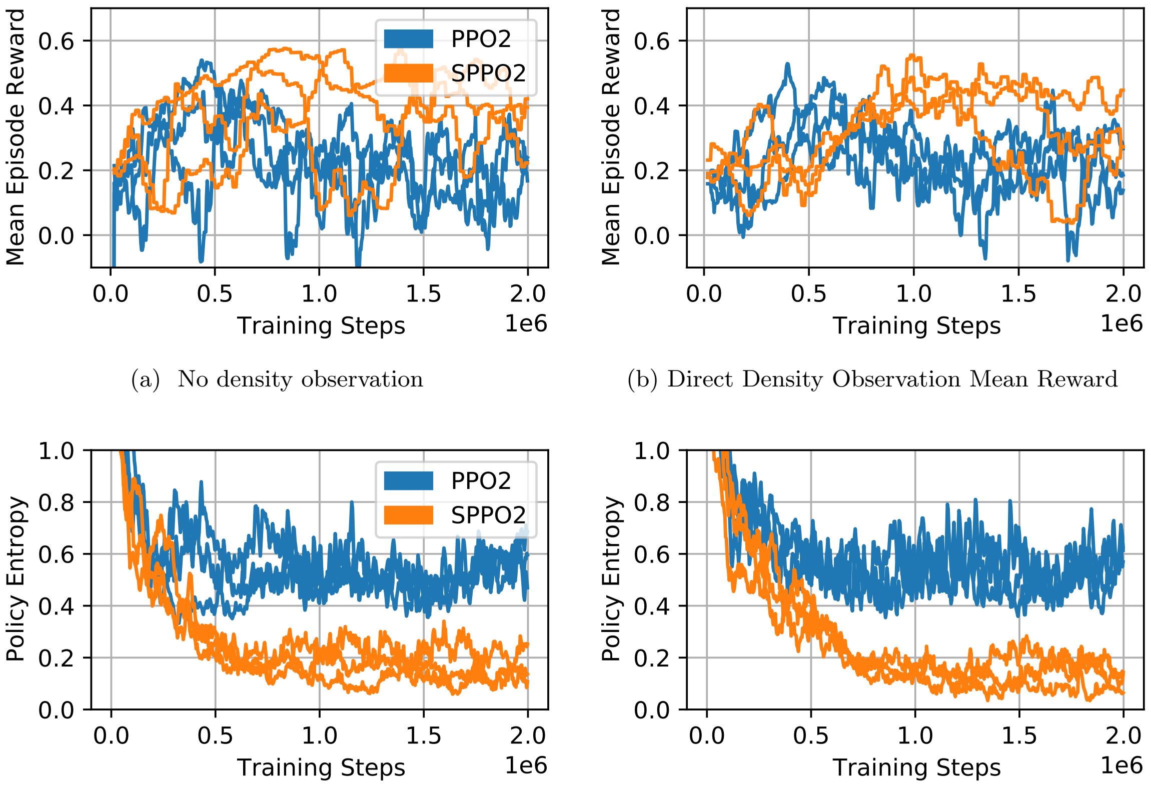
For practical purposes, the direct observation model is obviously unfavorable, as atmospheric neutral density is difficult to sense in-situ; however, parameters that are believed to affect local neutral density are readily available in near real-time. A null model, wherein atmospheric density is not observed, is also included to demonstrate the relative benefit (or lack thereof) to including information about atmospheric density to the drag-control agent. To ensure numerical conditioning, the 'direct observation' mode actually provides the agent with a measurement of the ratio of the current local neutral density to the design reference neutral density used in the controller gain calculation.

6.3 Training Results

To evaulate the fitness of SDRL techniques for drag-augmented spacecraft formation flight, drag-augmented LEO health management problem was trained using identical system parameters and algorithm hyperparameters to the best-performing SDRL agent for the LEO Attitude management task. Agents were trained using both PPO2 and SPPO2 on both environments over 3 random seeds. Because the problem formulation includes identical power and reaction wheel dynamics to the LEO attitude health management task described in Chapter 5, the same shield is re-applied to the SPPO2 agent here. The mean performance and variance for each of these agents over 100 random episodes on the drag environment is listed in Table 6.2; the mean performance of a uniform random policy over 100 random initial conditions is also included as a point of comparison.

Table 6.2: Summary of best performance for PPO2 and SPPO2 on direct density and null density observation environments.

Algorithm	Drag Env - No Density Obs	Drag Env - Direct Density Obs
Random Actions	0.15 ± 0.15	0.15 ± 0.15
PPO2	0.4622 ± 0.1137	0.4016 ± 0.07854
SPPO2	0.5579± 0.07709	0.4553 ± 0.06141



(a) No density observation

(b) Direct Density Observation Mean Reward

(c) No density observation policy entropy

(d) Direct Density Observation policy entropy

Figure 6.2: Mean episodic reward and policy entropy for each agent, environment over 3 random seeds and 2M timesteps.

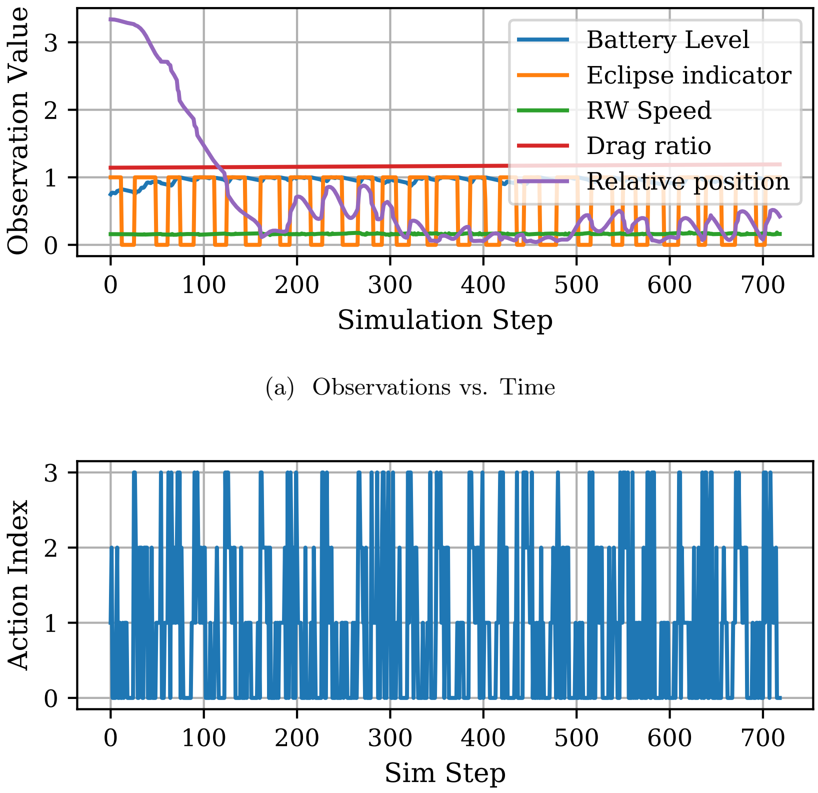
While both algorithms have similar maximum performance, several runs using pure PPO2 failed to converge to better-than-random performance on the environment (shown in Figure 6.2, indicating training instability. In addition, PPO2-trained agents tend to converge to higher levels of entropy in the final policy, indicating that agents learn to prioritize taking random actions more than the shielded agents. While SPPO2-trained policies have relatively high variance compared to the agents trained for the LEO attitude health management task, the same catastrophic lack of convergence is not observed in their results or policy entropies, indicating a clear benefit applying the shield in training.

Smaller differences are apparent in comparing agents trained on with and without density observations. While agents in both environments meet *can* exhibit similar peak performance, agents trained without density observations tended to perform better in evaluation. One way to interpret this is that both agents are trained with an equivalent amount of data (2M timesteps), but agents in the direct density observation problem must consider an additional parameter and its impact on the overall system. Drawing analogies to information dilution in filtering, these results suggest that additional observations may require additional training time to converge to similar performance, even on identical environment dynamics, a result that echos the concept of Lyapunov Dimensionality Reduction in Chapter 5.

6.4 Performance Analysis

During the application of trained DRL agents, it is desirable to understand not just how agents perform with respect to the objective function but also the characteristics of objective-maximizing behavior identified by the agents. To accomplish this, the best-performing shield PPO2 agent was run several times on the direct density observation environment to examine specific trajectories and action distributions.

Trained agents not only replicate the health-keeping behavior demonstrated by successful agents on the LEO attitude management task, but also switch to the differential drag mode periodically to reduce orbit error. Interestingly, agents do not chose to remain in drag modes for very long as demonstrated by both Fig. 6.3b. Instead, trained agents appear to use the drag-based station keeping mode periodically to 'kick' themselves onto trajectories that eventually move within the reference distance of the reference spacecraft, as shown by the Hill-frame trajectory in Figure 6.4. For comparison, the results of Chapters 3 and 4 assume that the attitude-guidance controller is activated continuously; these results indicate that even with small attitude variations triggered periodically, spacecraft could maintain their phase using a small fraction of the time needed for phasing maneuvers described in those chapters. Additionally, this indicates that the agent has learned to accounts for the passive drag impacts of other modes. This result suggests that differential drag station keeping can be accomplished with only occasional and minor attitude maneuvers, while using intermediate time to accomplish other mission needs.



(a) Observations vs. Time

(b) Actions vs. Time

Figure 6.3: Observation trajectories and actions over a representative simulation. ‘0’ actions indicate science mode, ‘1’ actions indicate sun-pointing, ‘2’ actions indicate wheel desaturation modes, and ‘3’ actions indicate drag-based station-keeping modes.

6.5 Robustness Analysis

Finally, a critical question to ask of mission management agents for station-keeping is whether said agents are robust to density variation. To accomplish this, a parametric sensitivity analysis was conducted using the same methodology as the sensitivity analysis performed in Chapter 5, wherein agents are evaluated in an environment with parametric differences from the environment used in training. For agents in the drag environment, the base density used in simulation was varied between two orders of magnitude difference from the reference value; as a control, power consumption was also varied in a comparable range to that used in the LEO attitude health management environment. Each agent was evaluated for three random seeds at 100 grid points in the

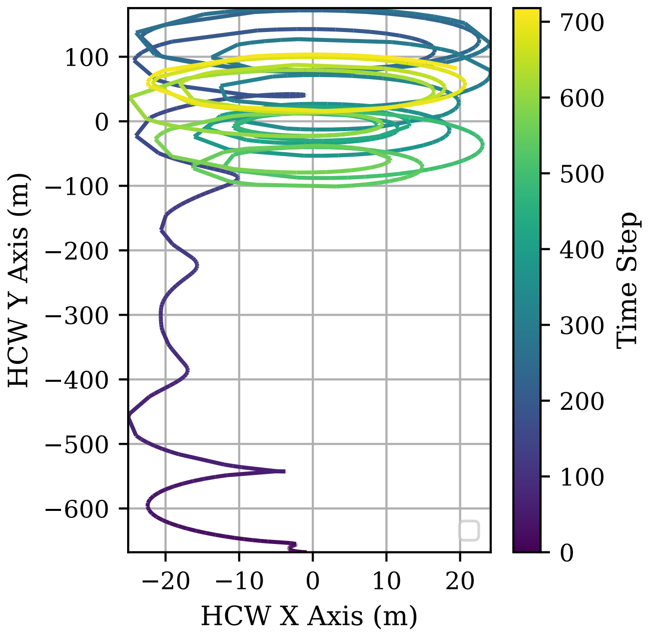


Figure 6.4: In-plane trajectory with respect to the reference position in the reference Hill frame.

parameter space summarized in Table 6.3, providing a set of points which were fit by Gaussian process regression.

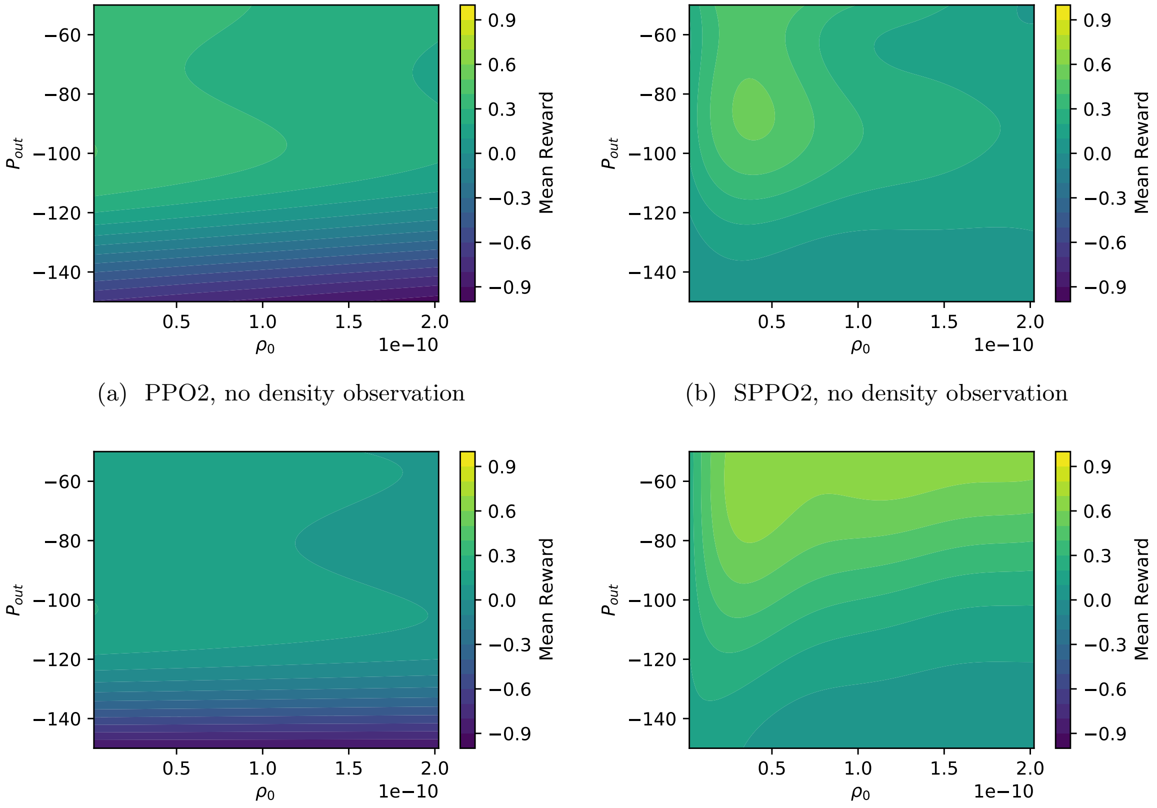
The results from the sensitivity analysis done in Chapter 4 show that the differential drag control law fails to converge when density varies below the design value as a result of reduced control authority. From the results shown in Figure 6.5, it is clear that the variation in episodic reward due to density variation are smaller than those resulting from changes in power-consumption, a challenge which in this case is well-resolved through the application of the LEO safety shield to the operations management. In both cases, shielded agents do not experience failures in the evaluation environment under any combination of power consumption and local density (though performance is degraded towards 0, indicating that less time can be spent in the mission or station-keeping modes.)

Notably, all agents see degraded performance as density increases and improved performance as density decreases. This is attributed to the impact of increased density and therefore drift rates in non-station-keeping modes: while the differential drag controller converges faster as density

Table 6.3: Parameters and ranges varied for drag agent sensitivity study.

Parameter	Training Value	Range
Ref Density (h=300km)	$2.022\times10^{-11}\,\frac{\mathrm{kg}}{\mathrm{m}^3}$	2.022×10^{-10}-2.022×10^{-12}
Power Consumption	100W	50-150

increases, it must be triggered more frequently, thereby negating the benefit of faster convergence; in addition, faster drift rates imply that the reward earned according to Eqn. 6.1 will be decreased. Notably, this degraded performance is dramatically reduced in the shielded direct-observation case, which implies that the agent was successfully able to learn to manage the drift-performance tradeoff.



(a) PPO2, no density observation

(b) SPPO2, no density observation

(c) PPO2, direct density observation

(d) SPPO2, direct density observation

Figure 6.5: Comparison of mean episode reward vs parametric variation for PPO2 and SPPO2 agents trained with and without direct density observations.

6.6 Conclusions

This chapter has demonstrated the benefits of combining the attitude-driven differential drag rendezvous controller from Chapters 3 and 4 and the shielded deep reinforcement-based autonomy strategy described in Chapter 5. Simulations presented in Section 6.3 demonstrate that DRL and SDRL agents improve on random policies for a representative extension of the LEO attitude and health management task that incorporates drag-based station-keeping. Agents that observed atmospheric densities in addition to other system states initially under-performed 'naïve' agents trained without knowledge of atmospheric density, but demonstrated improved robustness to density variations as studied in Section 6.5. Finally, the results of Section 6.4 show that trained agents discover novel uses for the continuous differential drag attitude-guidance law derived in Chapter 3, selecting it sparingly and periodically to re-adjust the relative trajectory to the reference position while maximizing time spent for other mission modes. This learned behavior is shown to manage the relative position to within a defined reference distance, demonstrating that acceptable station-keeping performance may be possible with only brief, periodic differential drag control modes as opposed to the long-duration continuous trajectories shown in prior work.

Chapter 7

Conclusion and Future Work

This work has contributed to the state of the art in techniques for both differential drag control and the adaptation of machine learning techniques for spacecraft operations. In this chapter, specific contributions will be summarized and contextualized. Additional future work and research questions posed by the results of this dissertation are also discussed for future research.

7.1 Major Highlights

Current operational examples of differential drag control identified in Chapter 1 rely on the use of either continuously variable deployable panels or the selection of discrete maximum/minimum drag configurations for formation flight to achieve relative accelerations. The approach described in Chapter 3 outlines a novel linearization strategy that compactly and elegantly encodes the geometry-attitude coupling to in-plane accelerations, allowing for the continuous differential drag control problem to be solved directly using small attitude motions defined by a linear control law.

In addition, the sensitivity analysis for differential drag outlined in Chapter 4 is to the author's knowledge the first explicit analysis of the impact of density variation on linear differential drag systems alongside the first adaptation of desensitized control to the differential drag problem. In addition to providing analytical insight into gain tuning for conventional controllers, control approaches that explicitly minimize sensitivity are shown to minimize control performance variation as density varies (albeit at the cost of control performance in nominal conditions). The results from this Chapter provide a rigorous framework for understanding which types of errors are likely to arise from density variation and provide insight into gain selection for future differential drag control laws to minimize (but not eliminate) the impact of density variation.

A major overall contribution of this work is the first adaptation of deep reinforcement learning and shielded learning to address high-level spacecraft operations tasking. This work examined not only performance of DRL techniques on benchmark tasks, but also how future challenges in spacecraft operations should be modeled and presented to be quickly and efficiently addressed with DRL techniques. The relative merits of DRL versus heuristic or black-box timeline optimization algorithms for performance, sample efficiency, and robustness to parametric modeling errors are also presented using empirical studies of trained agent performance. In comparison to other strategies, DRL-based approaches provide comparable performance to black-box timeline optimization while providing far greater robustness against parametric uncertainty and generalization across a range of operating conditions. The application of shielded learning, which allows for the enforcement of safety properties on DRL-driven decision making systems, is additionally shown to improve both performance and generalizability across multiple tasks while remaining simple to implement.

Finally, Chapter 6 demonstrates the merits of combining the small-attitude differential drag controller defined in Chapter 3 and analyzed in Chapter 4 with the SDRL tasking strategy and LEO health management tasks described in Chapter 5, thereby providing an elegant demonstration of this dissertation's technical solution to the challenge of formation- and constellation-scale flight with differential drag. DRL-based agents trained on a phase-keeping and health management task discover novel approaches to using the small-attitude differential drag control law, identifying that brief periods of control are sufficient to both fix large phasing errors and maintain a relative position without over-utilizing mission time. Finally, it is shown that knowledge of local atmospheric density both complicates the learning problem (leading to reduced reward under nominal conditions) and improves generalization of such agents with respect to moderate or large density variations.

7.2 Broader Impacts

During the writing and research efforts behind this dissertation, a keen eye has been turned towards ensuring that the results of this work have a broad impact upon the research and professional satellite communities.

Basilisk Contributions: This work has led to the development of a number of contributions to the open-source Basilisk astrodynamics framework, ranging from class hierarchies and implementations for atmospheric models to system-level models for power consumption Finally, to a formation flight control stack based on the work of Chapter 3. This work has resulted in a conference publication detailing the atmospheric and environmental models. Industry users of Basilisk have remarked upon the analytical utility provided by these models for early-stage mission design.

Spacecraft Ops Reference Tasks: An understated but considerable contribution to the community is the creation of reference problems for future work in autonomous mission management. These tasks, contained in the open-source library `basilisk-env`, will provide future researchers in autonomous decision-making the ability to compare their results against the benchmarks presented in Chapter 5. These environments have already been utilized and extended for work within the AVS laboratory [89], and are actively being maintained to encode both the best practices for environment construction described in Chapter 5 and relevant, realistic dynamics for spacecraft behavior.

7.3 Future Work

7.3.1 Differential Drag Control

This work has been primarily concerned with short-baseline maneuvers that can take advantage of linearized relative motion, which is shown to be surprisingly robust despite relatively large separation distances. However, most constellation-scale formation flight missions will require the ability to conduct control on baselines longer than tens of kilometers. It is likely desirable to bring the attitude-to-relative acceleration mapping described in Chapter 3 and combine it with a relative orbital elements formulation for long-baseline relative motion.

In addition, this work has largely neglected the use of atmospheric interactions to achieve out-of-plane control. Early work attempted to utilize linearized relative orbital elements, combined with optimistic accommodation coefficients, to generate control laws for formation flight using Lyapunov's direct method; however, these results demonstrated instability due to coupling between drag and lift forces, resulting in poor performance. Other works have considered multi-phase lift and drag maneuvers to avoid this coupling. In addition, the impact of atmospheric winds on formation control is also largely neglected in this work but could be explored for out-of-plane maneuverability.

Finally, additional work should be done to analyze the robustness and stability of differential drag control systems in the face of atmospheric uncertainty. This work considers only constant density errors when considering robustness against density variation; however, the actual behavior of neutral density in the thermosphere involves complex dynamics which can vary substantially over the course of one orbit. Future work should examine the sensitivity of differential drag trajectories under realistic environmental conditions using the high-fidelity models of neutral density and drag described in Section 2.

7.3.2 Machine Learning for Space Applications

There is a rich future for researchers interested in adapting machine learning or reinforcement learning techniques to address problems in spacecraft operations. Over the course of constructing this dissertation, several promising future directions have been identified and explored to varying degrees.

Sim2Real Transfer and On-Line Learning: While this work has extensively studied the use of software agents trained on high-fidelity simulators, the question of generalizability remains for transferring knowledge learned in simulation to flight. A related question is whether agents can be successfully and safely trained on-board during flight. Such a capability would allow for missions to respond to changing hardware and software conditions on-the-fly rather than relying on pre-flight assumptions about performance. Both of these challenges could be explored in a low-cost

LEO technical demonstration flight.

Hierarchical RL for action mode selection: This work only considers space missions that operate using sequenced, pre-designed operational modes which encapsulate relevant software and hardware behaviors. However, for complex missions, these modes may themselves be non-trivial to construct. Hierarchical RL attempts to address this problem by learning low-level and high-level behaviors separately but simultaneously, and presents one RL-centric approach towards resolving this issue. Other work in astrodynamics has explicitly considered the problem of constructing motion primitives from sampled trajectories, but to the author's knowledge no relevant example exists for attitude dynamics, especially when system-scale impacts on power or sensor availability are considered.

Learning of World Models: Before settling on PPO, a model-free reinforcement learning algorithm, some effort was spent through the Discovery Learning Assistant program to explore the space of model-based alternative to conventional DRL algorithms. Model-based RL learns a transition model – i.e., a generative model of the dynamics of an MDP – during training alongside a policy to map from states to actions. Learning on transition models instead of in the policy space has a number of practical advantages for future space adaptations of DRL, including sample-efficient learning for on-board applications, replacement of computationally expensive models with approximated neural network models, and the construction of numerically-derived safety MDPs for shield construction in place of expert-derived ones.

Multi-Agent Coordination: This dissertation has touched on the challenges facing future satellite missions which will require the orchestration of heterogeneous spacecraft in a decentralized manner; however, it has largely resolved those challenges by examining missions with homogeneous capabilities, allowing for one trained agent to be deployed across multiple spacecraft.

Bibliography

[1] Ulrich Walter. Orbit Perturbations, pages 555–660. Springer International Publishing, Cham, 2018.

[2] National Aeronautics and Space Administration. NASA Technology Taxonomy 2020. Technical Report July 2015, 2020.

[3] Defense Advanced Research Projects Agency. DARPA 2019 Strategic Framework. Technical report, 2019.

[4] 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), 2018.

[5] David A Spencer and Robert Tolson. Aerobraking Cost and Risk Decisions. Journal of Spacecraft and Rockets, 44(6):1285–1293, 2007.

[6] David Carrelli, Daniel O'Shaughnessya, Thomas Strikwerda, James Kaidy, Jill Prince, Richard Powell, Daniel O'Shaughnessy, Thomas Strikwerda, James Kaidy, Jill Prince, and Richard Powell. Autonomous aerobraking for low-cost interplanetary missions. Acta Astronautica, 93:467–474, jan.

[7] Hanspeter Schaub and John L Junkins. Analytical Mechanics of Space Systems. American Institute of Aeronautics and Astronautics, 3rd edition, 2014.

[8] Frank Marcos, Bruce Bowman, and Robert Sheehan. Accuracy of Earth's Thermospheric Neutral Density Models. AIAA/AAS Astrodynamics Specialist Conference and Exhibit, pages 1–20, 2006.

[9] David Pérez and Riccardo Bevilacqua. Differential drag spacecraft rendezvous using an Adaptive Lyapunov Control strategy. In Advances in the Astronautical Sciences, volume 145, pages 973–991, 2012.

[10] 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, 2011.

[11] Joseph W. Gangestad, Brian S. Hardy, and David A. Hinkley. Operations, Orbit Determination, and Formation Control of the AeroCube-4 CubeSats. Proceedings of the AIAA/USU Conference on Small Satellites, SSC13:SSC13–X–4, 2013.

[12] M Horsley, S Nikolaev, and A Pertica. Small Satellite Rendezvous Using Differential Lift and Drag. Journal of Guidance, Control, and Dynamics, 36(2):445–453, 2013.

[13] Cyrus Foster, Henry Hallam, and James Mason. Orbit determination and differential-drag control of Planet Labs cubesat constellations. Advances in the Astronautical Sciences, 156:645–657, 2016.

[14] J. T. Emmert. Thermospheric mass density: A review, 2015.

[15] Shaylah Mutschler, Penina Axelrad, and Tomoko Matsuo. Harnessing Orbital Debris to Sense the Space Environment. In Proceedings of the Advanced Maui Optical and Space Surveillance (AMOS) Technologies Conference, Maui, HI, 2017.

[16] Ohad Ben-Yaacov and Pini Gurfil. Long-Term Cluster Flight of Multiple Satellites Using Differential Drag. Journal of Guidance, Control, and Dynamics, 36(6):1731–1740, 2013.

[17] L Dellelce and G Kerschen. Optimal propellantless rendez-vous using differential drag. Acta Astronautica, 109:112–123, 2015.

[18] D. Spiller, Ko Basu, Fabio Curti, and Christian Circi. On the optimal passive formation reconfiguration by using attitude control. Acta Astronautica, 153, 2018.

[19] Andrew Harris and Hanspeter Schaub. Towards reinforcement learning techniques for spacecraft autonomy. In Advances in the Astronautical Sciences, volume 164, pages 467–476, 2018.

[20] Andrew Harris, Thibaud Teil, and Hanspeter Schaub. Spacecraft Decision-Making Autonomy Using Deep Reinforcement Learning. 29th AAS/AIAA Space Flight Mechanics Meeting, Hawaii, (AAS 19-447):1–19, 2019.

[21] Christoph Steiger, Massimo Romanazzo, Pier P. Emanuelli, Rune Floberghagen, and Michael Fehringer. The deorbiting of ESA's gravity mission GOCE - Spacecraft operations in extreme drag conditions. 13th International Conference on Space Operations, SpaceOps 2014, (May):1–12, 2014.

[22] Christopher R. Boshuizen, James Mason, Pete Klupar, and Shannon Spanhake. Results from the Planet Labs Flock Constellation. 28th Annual AIAA/USU Conference on Small Satellites, pages SSC14–I–1, 2014.

[23] C R Frost. Challenges and Opportunities for Autonomous Systems in Space. National Academy of Engineering's U.S. Frontiers of Engineering Symposium, 2010.

[24] Teck H Choo and Joseph P Skura. SciBox: A software library for rapid development of science operation simulation, planning, and command tools. Johns Hopkins APL Technical Digest (Applied Physics Laboratory), 25(2):154–161, 2004.

[25] Steve Chien, Rob Sherwood, Daniel Tran, Rebecca Castano, Benjamin Cichy, Ashley Davies, Gregg Rabideau, Nghia Tang, Michael Burl, Dan Mandl, Stuart Frye, Jerry Hengemihle, Jeff D Agostino, Robert Bote, Bruce Trout, Seth Shulman, Stephen Ungar, Jim Van Gaasbeck, Darrell Boyer, Control Systems, Michael Griffin, and Hsiao-hua Burke Mit. Autonomous Science on the EO-1 Mission. Proceedings of International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS), (May), 2003.

[26] Steve Chien, Rob Sherwood, Daniel Tran, Benjamin Cichy, Gregg Rabideau, Rebecca Castano, Ashley Davis, Dan Mandl, Stuart Frye, Bruce Trout, and Seth Shulman. Using Autonomy Flight Software to Improve Science Return on Earth Observing One. 2(April):196–216, 2005.

[27] Steve A Chien, Daniel Tran, Gregg Rabideau, Steve R Schaffer, Dan Mandl, and Stuart Frye. Timeline-Based Space Operations Scheduling with External Constraints. Proceedings of the 20th International Conference on Automated Planning and Scheduling (ICAPS), (Icaps):34–41, 2010.

[28] Oriol Vinyals. Deep Learning in neural networks: An overview. Neural Networks, 61:85–117, 2015.

[29] Kyle D Julian and Mykel J Kochenderfer. Autonomous Distributed Wildfire Surveillance using Deep Reinforcement Learning. (January):1–16, 2018.

[30] Richard S. Sutton and Andrew G. Barto. Reinforcement learning. Learning, 3(9):322, 2012.

[31] Daniel G Kubitschek. Impactor Spacecraft Encounter Sequence Design for the Deep Impact Mission. Jet Propulsion, pages 1–14, 2005.

[32] Andrew Harris. Towards Reinforcement Learning Techniques For Spacecraft Autonomy. AAS Guidance, Navigation and Control Meeting, pages 1–10, 2018.

[33] Alicia D Cianciolo, Robert W Maddock, Jill L Prince, Angela Bowes, Richard W Powell, Joseph P White, Robert Tolson, O Shaughnessy, and David Carrelli. Autonomous Aerobraking Development Software : Phase 2 Summary. pages 1–16, 2013.

[34] Brian Gaudet, Roberto Furfaro, Markov Decision Process, Reinforcement Learning, Linear Quadratic Regulator, Tucson Arizona, and Tucson Arizona. Robust Spacecraft Hovering Near Small Bodies in. test, (August):1–20, 2012.

[35] Ilaria Bloise Roberto Furfaro. Deep Learning for Autonomous Lunar Landing. Proceedings of the 2018 AAS/AIAA Astrodynamics Specialist Conference, Snowbird UT, 2018.

[36] Timothy E. Rumford. Demonstration of Autonomous Rendezvous Technology (DART). Space Systems Technology and Operations, 5088(August 2003):10–19, 2003.

[37] Duncan Eddy and Mykel Kochenderfer. Markov Decision Processes For Multi-Objective Satellite Task Planning. In 2020 IEEE Aerospace Conference, pages 1–12. IEEE, 2020.

[38] Sara Spangelo, James Cutler, Kyle Gilson, and Amy Cohn. Optimization-based scheduling for the single-satellite, multi-ground station communication problem. Computers and Operations Research, 57:1–16, 2015.

[39] Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John Agapiou, Julian Schrittwieser, John Quan, Stephen Gaffney, Stig Petersen, Karen Simonyan, Tom Schaul, Hado van Hasselt, David Silver, Timothy Lillicrap, Kevin Calderone, Paul Keet, Anthony Brunasso, David Lawrence, Anders Ekermo, Jacob Repp, and Rodney Tsing. StarCraft II: A New Challenge for Reinforcement Learning. 2017.

[40] Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander S. Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom L. Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. Nature, 575(7782):350–354, 2019.

[41] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Psyho Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique Pondé De Oliveira Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. Dota 2 with large scale deep reinforcement learning, 2019.

[42] Subramanya Nageshrao, Eric Tseng, and Dimitar Filev. Autonomous highway driving using deep reinforcement learning, 2019.

[43] B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A.Al Sallab, Senthil Yogamani, and Patrick Perez. Deep Reinforcement Learning for Autonomous Driving: A Survey. IEEE Transactions on Intelligent Transportation Systems, pages 1–18, 2021.

[44] Richard (DeepMind) Evans and Jim (DeepMind) Gao. DeepMind AI Reduces Google Data Centre Cooling Bill by 40

[45] Ashley D. Biria and Belinda G. Marchand. Constellation design for space-based space situational awareness applications: An analytical approach. Journal of Spacecraft and Rockets, 51(2):545–562, 2014.

[46] Adam Snow, Angela den Boer, Luke Alexander, and Marcus J. Holzinger. Design and Optimization of a Disaggregated Constellation for Space Situational Awareness. Proceedings of the AIAA/USU Conference on Small Satellites, pages SSC15—-VIII—-3, 2015.

[47] Kelly Cole, David Voss, Amanda Pietruszewski, Lyon B King, Philip Hohnstadt, Kelly Feirstine, J Crassidis, Michael D'Angelo, and Richard Linares. Space surveillance tech area benefits from university partnerships, 2011.

[48] E K Sutton. Effects of Solar Disturbances on the Thermosphere Densities and Winds from CHAMP and GRACE Satellite Accelerometer Data. page 149, 2008.

[49] G A Bird. Molecular Gas Dynamics and the Direct Simulation of Gas Flows. Number v. 1 in Molecular Gas Dynamics and the Direct Simulation of Gas Flows. Clarendon Press, 1994.

[50] Frank A. Marcos, Shu T. Lai, Cheryl Y. Huang, Chin S. Lin, John M. Retterer, Susan H. Delay, and Eric Sutton. Towards next level satellite drag modeling. AIAA Atmospheric and Space Environments Conference 2010, (August), 2010.

[51] Liying Qian and Stanley C. Solomon. Thermospheric density: An overview of temporal and spatial variations. Space Science Reviews, 168(1-4):147–173, 2012.

[52] David A. Vallado. Fundamentals of Astrodynamcs and Applications. Space Technology Library, 4th edition, 2013.

[53] Bruce R. Bowman, W. Kent Tobiska, Frank A. Marcos, Cheryl Y. Huang, Chin S. Lin, and William J. Burke. A new empirical thermospheric density model JB2008 using new solar and geomagnetic indices. AIAA/AAS Astrodynamics Specialist Conference and Exhibit, (August), 2008.

[54] J M Picone, A E Hedin, D P Drob, and A C Aikin. NRLMSISE-00 empirical model of the atmosphere: Statistical comparisons and scientific issues. Journal of Geophysical Research: Space Physics, 107(A12):SIA 15–1–SIA 15–16, 2002.

[55] Liying Qian, A Burns, Barbara Emery, B Foster, Gang Lu, Astrid Maute, A Richmond, R G Roble, Stanley Solomon, and Wei Wang. The NCAR TIE-GCM: A community model of the coupled thermosphere/ionosphere system. Geophysical Monograph Series, 201:73–83, 2013.

[56] D G KING-HELE and DIANA W SCOTT. Rotational Speed of the Upper Atmosphere, from the Orbits of Satellites 1966-51 A, B and C. Nature, 213(5081):1110–1111, 1967.

[57] Douglas P. Drob, John T. Emmert, John W. Meriwether, Jonathan J. Makela, Eelco Doornbos, Mark Conde, Gonzalo Hernandez, John Noto, Katherine A. Zawdie, Sarah E. McDonald, Joe D. Huba, and Jeff H. Klenzing. An update to the Horizontal Wind Model (HWM): The quiet time thermosphere. Earth and Space Science, 2(7), 2015.

[58] David A. Vallado and David Finkleman. A critical assessment of satellite drag and atmospheric density modeling. Acta Astronautica, 95(1):141–165, 2014.

[59] M D Pilinski. Dynamic Gas-Surface Interaction Modeling for Satellite Aerodynamic Computations. 2011.

[60] Nuno Filipe and Panagiotis Tsiotras. Adaptive position and attitude tracking controller for satellite proximity operations using dual quaternions. Advances in the Astronautical Sciences, 150:2313–2332, 2014.

[61] Mingying Huo, Jun Zhao, Shaobiao Xie, and Naiming Qi. Coupled attitude-orbit dynamics and control for an electric sail in a heliocentric transfer mission. PLoS ONE, 10(5):1–14, 2015.

[62] Junshan Mu, Shengping Gong, and Junfeng Li. Coupled Control of Reflectivity Modulated Solar Sail for GeoSail Formation Flying. Journal of Guidance, Control, and Dynamics, 38(4):740–751, 2015.

[63] Elvis D Silva. A Formulation of the Clohessy-Wiltshire Equations to Include Dynamic Atmospheric Drag. AIAA/AAS Astrodynamics Specialist Conference, (August), 2008.

[64] William L. Brogan. Modern Control Theory. Prentice-Hall Inc, Englewood Cliffs, New Jersey, 3rd edition, 1991.

[65] A. Harris, C.D. Petersen, and H. Schaub. Linear coupled attitude-orbit control through aerodynamic forces. In Space Flight Mechanics Meeting, 2018, number 210009, 2018.

[66] Liying Qian, Stanley C. Solomon, and Timothy J. Kane. Seasonal variation of thermospheric density and composition. Journal of Geophysical Research: Space Physics, 114(1):1–15, 2009.

[67] Stephen J Kahne. Low-Sensitivity Design of Optimal Linear Control Systems. IEEE Transactions on Aerospace and Electronic Systems, AES-4(3):374–379, 1968.

[68] Haijun Shen, Hans Seywald, and Richard W Powell. Desensitizing the Minimum-Fuel Powered Descent For Mars Pinpoint Landing. Journal of Guidance, Control, and Dynamics, 33(1):108–115, 2010.

[69] Marco Muenchhof and Tarunraj Singh. Desensitized Jerk Limited-Time Optimal Control of Multi-Input Systems. Journal of Guidance, Control, and Dynamics, 25(3):474–481, 2008.

[70] Kevin Seywald and Hans Seywald. Desensitized Optimal Control. (November), 1996.

[71] Venkata Ramana Makkapati, Mehregan Dor, and Panagiotis Tsiotras. Trajectory Desensitization in Optimal Control Problems. Proceedings of the IEEE Conference on Decision and Control, 2018-Decem:2478–2483, 2019.

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

[73] Thomas Uhlig, Florian Sellmaier, and Michael Schmidhuber. Spacecraft operations. 2015.

[74] Gabriel Dulac-Arnold, Daniel Mankowitz, and Todd Hester. Challenges of Real-World Reinforcement Learning. 2019.

[75] Eric Sample, Nisar Ahmed, and Mark Campbell. An Experimental Evaluation of Bayesian Soft Human Sensor Fusion in Robotic Systems. (August):1–19, 2012.

[76] Michael S Branicky. Multiple Lyapunov functions and other analysis tools for switched and hybrid systems. IEEE Transactions on Automatic Control, 43(4):475–482, 1998.

[77] Steve Chien, Russell Knight, Steve Schaffer, David Ray Thompson, Brian Bue, and Martina Troesch. An onboard autonomous response prototype for an earth observing spacecraft. In International Joint Conference on Artificial Intelligence Workshop on Artificial Intelligence in Space (AI Space, IJCAI 2015), Buenos Aires, Argentina, July 2015.

[78] Forest Agostinelli, Guillaume Hocquest, Sameer Singh, and Pierre Baldi. From Reinforcement Learning to Deep Reinforcement Learning: An Overview. In Lev Rozonoer, Ilya Muchnik, and Boris Mirkin, editors, Braverman Readings in Machine Learning, pages 298–328. Springer Nature Switzerland, 2017.

[79] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep Reinforcement Learning that Matters. 2017.

[80] Chen Debao. Degree of approximation by superpositions of a sigmoidal function. Approximation Theory and its Applications, 9(3):17–28, 1993.

[81] Yann Lecun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. Nature, 521(7553):436–444, 2015.

[82] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, and Others. Human-level control through deep reinforcement learning. Nature, 518(7540):529, 2015.

[83] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. Arxiv, pages 1–12, 2017.

[84] John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. 4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings, pages 1–14, 2016.

[85] Xiangyu Li, Rakesh R. Warier, Amit K. Sanyal, and Dong Qiao. Trajectory Tracking Near Small Bodies Using Only Attitude Control. Journal of Guidance, Control, and Dynamics, 42(1):1–14, 2018.

[86] Mohammed Alshiekh, Roderick Bloem, Ruediger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe Reinforcement Learning via Shielding. pages 1–23, 2017.

[87] Taolue Chen, Vojtěch Forejt, Marta Kwiatkowska, David Parker, and Aistis Simaitis. PRISM-games: A model checker for stochastic multi-player games. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 7795 LNCS:185–191, 2013.

[88] John Alcorn, Hanspeter Schaub, Scott Piggott, and Daniel Kubitschek. Simulating Attitude Actuation Options Using the Basilisk Astrodynamics Software Architecture. 67 th International Astronautical Congress, 2016.

[89] Adam Herrmann and Hanspeter Schaub. Autonomous spacecraft tasking using monte carlo tree search methods. In AAS/AIAA Space Flight Mechanics Meeting, Charlotte, NC, Jan. 31 – Feb. 4 2021. Paper No. AAS-21-228.