Efficient Trajectory Optimization for Constrained Spacecraft Attitude Maneuvers using Momentum Exchange Devices

by

Thomas Lee Dearing

B.S., New Mexico Institute of Mining and Technology, 2018

M.S., University of Colorado Boulder, 2021

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

Department of Electrical, Computer, and Energy Engineering

2023

Committee Members:

Xudong Chen, Chair

Marco Nicotra

John Hauser

Christopher Petersen

Hanspeter Schaub

Dearing, Thomas Lee (Ph.D., Dynamics and Controls)

Efficient Trajectory Optimization

for Constrained Spacecraft Attitude Maneuvers

using Momentum Exchange Devices

Thesis directed by Profs. Xudong Chen

Existing solutions for spacecraft attitude control have historically accommodated the limited processing power of spacecraft CPU's through the use of simplified dynamical models and traditional feedback laws. However, while such approximations improve the computational complexity (and reliability) of the resulting control solutions (a necessary feature for a viable Attitude Determination and Control System (ADCS)), the maximum performance of control strategies developed under these simplified dynamics is limited compared to the potential capabilities of the physical system. In contrast, this thesis examines the optimal control behaviors and performance capabilities of *physical* spacecraft dynamical models by developing and solving trajectory optimization problems for rest-to-rest attitude maneuvers under standard operational constraints on input saturation, maximum angular rates, and solar exclusion cones.

In particular, this thesis examines three problems: (1) maneuvers planned using traditional abstract body torques and approximated dynamics (a model problem to validate the solver), (2) maneuvers planned using a comprehensive, momentum-conserving model for a spacecraft CMG array, and (3) the latter problem augmented with an accurate model for the array's energy usage. In each case, the optimal solutions guarantee constraint feasibility and present substantial performance improvements over the original guess (determined in most cases using accepted feedback solutions). Additionally, the proper energy penalty employed in the final case yields substantially improved performance and efficiency over the accepted Singularity-Robust (SR) feedback law, with this increased performance correlating with several notable control behaviors observed across the solution family: (a) a model torque profile for the CMG array analogous to the initial and final

burns used in orbital transitions and (b) variable CMG wheel speed regulation which is largely inactive during the maneuver's 'cruise' phase. Finally, potential adaptations of existing feedback control solutions which incorporate the above behaviors are proposed, achieving the corresponding performance improvements without the computational burden of strict optimality.

Acknowledgements

I cannot thank my advisors Xudong Chen and Marco Nicotra enough for the all training, guidance, and support they provided me during my graduate student career. This also extends to the other members of my committee, who each actively participated in developing my knowledge and skills as a researcher. I would also like to thank the National Science Foundation Graduate Research Fellowship Program for funding my graduate education and enabling me to pursue my own research interests.

I would also like to thank my fellow labmates Umar, Dave, Daniel, Felipe, Terry, and Jay for their companionship and solidarity during the many late nights and deadlines we braved together.

I also thank my parents, whose love, tireless efforts, and endless support gave me the opportunity and determination to reach this far. My greatest thanks however must go to my wife Carley, whose unconditional support, patience, and encouragement give me the courage to better myself every day.

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

A fundamental challenge when designing a spacecraft is achieving an appropriate balance between the performance capabilities necessary to complete the mission, the reliability (both in hardware *and* software) to survive multiple years in the extreme environment of space, and the required electrical power available from the onboard solar panels ([31]). As one might suspect, this balance plays a key role in the design of the spacecraft's Attitude Determination and Control System (ADCS), both in system's engineering (the efficacy, size, and number of attitude actuators used rotate the platform) and in control system's engineering (the required control effort and computational cost of the integrated control system). From this perspective, it is unsurprising that spacecraft attitude control (particularly *optimal* control and optimization) remains an extremely active research topic both in industry and in academia.

Unfortunately, the limited computational power available from existing spacecraft processors (which prefer older, bulkier electronics to resist the effects of radiation) has correspondingly restricted the scope of practical control solutions. In particular, the large majority of existing control solutions (both integrated and in the literature) employ simplified or *approximated* dynamical models for the spacecraft's attitude dynamics: a strategy which can provide valuable improvements in both the complexity (and reliability) of control solutions. In particular, consider the most prevalent actuation technique for spacecraft attitude control: momentum exchange. In essence, a Momentum Exchange Device (MED) generates attitude control torques by using electric motors to exchange angular momentum between itself and the main spacecraft body [33]. Thus, the 'true' dynamics of

an MED array (and its unique state manifold) are fundamentally governed by the conservation of the spacecraft's total inertial angular momentum: a complex constraint requiring a comprehensive model of *all* angular momentum exchanged within the array and spacecraft frame. Naturally, this complex model is challenging (both behaviorally and computationally) for direct application in spacecraft controller design, where simplicity and reliability are traditionally preferred over raw performance. As a result, existing ADCS generally simplify the internal momentum dynamics of their MED arrays, instead modelling them in the top level attitude controller as simplified, externally generated torques akin to those produced using thrusters. For the momentum array itself, a separate controller is then used to convert these abstracted 'command torques' to appropriate inputs for the MED's (or indeed any such actuator). This divide-and-conquer strategy greatly simplifies the design (and optimization of) the top level attitude controller, but sacrifices performance by treating the original actuator dynamics like a disturbance, thereby distorting the system's true state space.

Naturally, the above complications present an even greater challenge in the context of *optimal* control and trajectory optimization where the complexity of the resulting problem makes solutions challenging to obtain even using modern hardware. Correspondingly, the majority of approaches in the literature employ similar dynamical approximations to ensure the resulting control solutions remain relevant and compatible with those deployed in existing spacecraft. However, while it is certainly idealistic to develop a computational-tractable trajectory optimization problem (i.e. solvable in real time) under the true, physical dynamics of an MED-driven spacecraft, it *is* possible to examine the off-line results of such problems to identify the *strategies* and *behaviors* employed by the optimal controller which measurably improve ADCS performance and efficiency. These strategies can then potentially be incorporated into the more accepted, reliable, and computational control strategies to improve their performance without the computational burden of strict optimality. It is this perspective which motivates this thesis.

1.1 Research Overview

The overarching research objective of this thesis is to develop novel, robust, and computational tractable control strategies for spacecraft attitude maneuvers using techniques in trajectory optimization and optimal control while enforcing all relevant operational safety constraints. In pursuance of this goal, we formulate and solve several constrained trajectory optimization problems for constrained rest-to-rest attitude maneuvers for spacecraft driven by traditional abstract body torques and momentum-conserving Control Moment Gyroscope (CMG) arrays. In the process of solving these problems, we develop new strategies for the application of our chosen trajectory optimization solver (PRONTO) to problems evolving on constrained nonlinear manifolds, resulting in substantially improved performance over other techniques (both commercially and in the literature).

After validating this general approach, problem formulation, and solver on a model trajectory optimization problem in the literature (i.e. a maneuver planner using abstract body command torques), we apply this approach to solve the (previously *open*) constrained trajectory optimization problem for rest-to-rest attitude transfers using a momentum-conserving CMG array. In particular, our formulation employs a dynamical model which preserves the array's (conservative) momentum exchange dynamics, a power model directly tracking the usage of the individual CMG motors, and typical operational safety constraints on input saturation, angular velocity, and camera exclusion cones. The optimal control strategies produced under this comprehensive formulation present substantial improvements to mean maneuver performance and efficiency, and identify an acute shortcoming in cost functions which use the control input (rather than accurately modelled power usage) to penalize maneuver energy cost.

To address this shortcoming, we augment the above trajectory optimization problem with a specialized energy penalty in the cost function to accurately limit the electric power used by the array. Unlike existing approaches, this comprehensive formulation enables optimal solutions to display their full range of (potentially non-intuitive) behaviors while satisfying constraints and

reducing the true total electric power used by the array. Finding these solutions to be substantially more performant *and* efficient than existing feedback solutions, we primarily attribute this performance increase to two nonstandard control strategies observed across the solution family: (1) a bang-bang model torque profile for the CMG array analogous to the initial and final burns used in orbital transitions and (2) *variable* CMG wheel speed regulation which is largely inactive during the maneuver's 'cruise' phase. Finally, we suggest potential adaptations of existing feedback control solutions which incorporate the above behaviors, achieving the corresponding performance improvements without the computational burden of strict optimality. With this final step, we achieve our original research objective and demonstrate the immense value in off-line solutions of traditionally challenging trajectory optimization problems for high performance systems.

1.2 Outline of the Dissertation

Beginning in chapter 2, we review the definitions and relative strengths of popular attitude models, including Euler angles, quaternions, rotation matrices, and modified Rodriquez parameters. We next review the basic dynamical models, operational constraints, actuators, and feedback control strategies employed in classic spacecraft attitude control problems. In particular, we review in detail the popular Singularity Robust (SR) feedback law as a prime example of the prevalent inner-outer-loop controller archetype employed in existing spacecraft ADCS's.

In chapter 3, we review the fundamentals for formulating well-posed trajectory optimization problems as well as common numeric and analytic strategies for solving them. This includes a review of classic direct and indirect methods as well as newer hybrid strategies presented in the literature. In the second half of this chapter, we develop in detail the trajectory optimization algorithm used in this thesis: the PRONTO solver. In this discussion, we also review the required adaptations and (regulator) design strategies necessary for addressing constrained maneuver planning problems which evolve on nonlinear configuration manifolds.

In chapter 4, we examine a traditional trajectory optimization problem formulation for rest-to-rest attitude maneuvers under simplified body-torque driven dynamics and common operational

constraints on input saturation, angular rotation rate, and solar exclusion cones. This chapter largely reviews the implementation of the PRONTO algorithm presented in [12] and highlights the impressive capabilities and performance benefits offered by the PRONTO solver over popular commercial solvers, particularly in the case of systems whose dynamics evolve on nonlinear manifolds.

In chapter 5, we develop the fundamental momentum exchange physics and dynamics for Single-Gimble Control Moment Gyroscope arrays. In particular, this includes a nonlinear coordinate transformation of existing dynamical results to one appropriate for our problem formulation as well as a parameterization of the constraint manifold induced by the array's conservation of inertial angular momentum. Additionally, we develop an alternative model for the electrical power consumed by the CMG array's component motors, which later proves useful in formulating an energy-optimal trajectory optimization problem.

In chapter 6, we examine the more sophisticated trajectory optimization problem for rest-to-rest attitude maneuvers for spacecraft using CMG arrays. Specifically, the formulated maneuver planner evolves the constrained problem discussed in chapter 4 to the comprehensive (conservative) dynamics developed in chapter 5. Necessary adaptations to the PRONTO solver to accommodate the (nontrivial) momentum conservation constraints are discussed. Finally, two concurrent studies are performed (summarizing the results of [14] and [13] respectively) which highlight the potential performance improvements offered by optimal solutions and the critical importance of proper models for system dynamics and energy usage in trajectory optimization problems. Finally, this section concludes by presenting two potential control strategies observed in the optimal solutions which greatly improve maneuver efficiency. In particular the presented control strategies do not require strict optimality and may be be implemented using accepted feedback control strategies.

Finally, chapter 7 summarizes this thesis with concluding remarks.

Chapter 2

Classical Spacecraft Attitude Control

In this chapter, we will review the fundamental models used for spacecraft rotations, including common parameterizations of the attitude (orientation) space and the rotation dynamics of rigid bodies. In addition, we will review common feedback control strategies and the operational safety constraints they must satisfy to be employed in modern spacecraft Attitude Determination and Control Systems (ADCS's). Finally, we will review the Singularity-Robust (SR) feedback law as a prime example of the prevalent inner-outer-loop controller archetype employed in existing spacecraft ADCS's.

2.1 Spacecraft Attitude Modelling

Unlike with linear coordinates like position or velocity, there is no single 'correct' way to express the orientation or *attitude* of a rigid body in three dimensions. Indeed, the many challenges resulting from the symmetries and periodicity of the attitude space have inspired a large variety of coordinate systems, each with their own unique complications. Perhaps the most intuitive of these coordinate systems is the simple 3×3 rotation matrix $C \in \mathrm{SO}(3)$ whose columns directly list the x, y, and z unit basis vectors for the object's body frame $\mathcal{F}_b$ viewed in the fixed inertial frame $\mathcal{F}_i$ [24]. That is, a vector $\boldsymbol{v}^b \in \mathbb{R}^3$ written in $\mathcal{F}_b$ can be expressed (or re-coordinatized) in $\mathcal{F}_i$ using the matrix $C_b^i \in \mathrm{SO}(3)$ as follows

$$\boldsymbol{v}^i = C_b^i \, \boldsymbol{v}^b.$$

Unfortunately, while rotation matrices are extremely convenient and precise for expressing attitudes (directly listing the reference frame of the rotated body), the nine parameters they require and the nontrivial parent space they are constrained to, the special orthogonal group SO(3), make them computationally undesirable given the limited processing power available on most spacecraft.

With this limitation in mind, it is naturally tempting to instead seek out a *minimal* attitude representation requiring the fewest possible coordinates. For example, a rotation can alternatively be specified using a sequence of three rotations around the object's principle (body frame) axes. These yaw, roll, and pitch angles (commonly called Euler angles) are frequently used to express attitudes. While this representation is useful for static models however, it also possesses intrinsic coordinate singularities which make it unsuitable for modelling attitude dynamics. Specifically, the parent 3-torus T^3 for Euler angles is not homeomorphic to SO(3) at every point, meaning that certain motions in the true attitude of an object (i.e., on SO(3)) cannot be captured using Euler angles. Practically, this issue manifests in controller design as the well-known problem of gimbal lock [24]. These coordinate singularities (and the numerical ill-conditioning produced in their immediate vicinity) make minimal representations unsuitable in many applications, including the trajectory optimization problems developed in this thesis. Notably, certain specialized minimal representations like Modified Rodriguez Parameters (MRP's) can (and frequently are) used for attitude controller design and offer many practical advantages. However, additional considerations are required for controllers using MRP's (e.g. shadow sets) to ensure they safely and reliably avoid their singular points.

Thankfully, these coordinate singularities can be avoided by using a 4-parameter attitude representation. Specifically, the *quaternion* (or Euler parameters) is extremely prevalent in existing spacecraft guidance, navigation, and control algorithms due to its compactness and numerical efficiency. By definition, a quaternion $\mathfrak{q} \in \mathbb{H}$ is a hypercomplex number of the form

$$\mathfrak{q} \coloneqq q_s + \underbrace{q_x \boldsymbol{i} + q_y \boldsymbol{j} + q_z \boldsymbol{k}}_{\mathfrak{q}_v \coloneqq},$$

with real (scalar) part $q_s \coloneqq \mathrm{Re}(\mathfrak{q}) \in \mathbb{R}$ and imaginary (or vector) part $\mathfrak{q}_v \coloneqq \mathrm{Im}(\mathfrak{q})$ written using

the extended complex basis $\boldsymbol{i}, \boldsymbol{j}$, and $\boldsymbol{k}$ [11, 46]. Computationally, $\mathfrak{q}$ is commonly represented using the real vector $\boldsymbol{q} := [q_s; \boldsymbol{q}_v] \in \mathbb{R}^4$ with $\boldsymbol{q}_v := [q_x; q_y; q_z] \in \mathbb{R}^3$, where we use the notation $[\boldsymbol{a}; \boldsymbol{b}]$ to denote vertically concatenated vectors. The *rotation* action of a quaternion is achieved using the non-commutative quaternion product "$\circ$", which admits the following vector notation equivalent:

$$
\begin{aligned}
\mathfrak{h} &= \mathfrak{q} \circ \mathfrak{p}, \qquad\qquad \mathfrak{h}, \mathfrak{p}, \mathfrak{q} \in \mathbb{H}, \\
\begin{bmatrix} h_s \\ \boldsymbol{h}_v \end{bmatrix} &= \begin{bmatrix} q_s p_s - \boldsymbol{q}_v^\top \boldsymbol{p}_v \\ q_s \boldsymbol{p}_v + p_s \boldsymbol{q}_v + \boldsymbol{q}_v \times \boldsymbol{p}_v \end{bmatrix}, \\
&= \underbrace{\begin{bmatrix} q_s & -\boldsymbol{q}_v^\top \\ \boldsymbol{q}_v & q_s \mathbb{I}_3 + \widehat{\boldsymbol{q}}_v \end{bmatrix}}_{O_L(\boldsymbol{q}) :=} \begin{bmatrix} p_s \\ \boldsymbol{p}_v \end{bmatrix} = \underbrace{\begin{bmatrix} p_s & -\boldsymbol{p}_v^\top \\ \boldsymbol{p}_v & p_s \mathbb{I}_3 - \widehat{\boldsymbol{p}}_v \end{bmatrix}}_{O_R(\boldsymbol{p}) :=} \begin{bmatrix} q_s \\ \boldsymbol{q}_v \end{bmatrix},
\end{aligned}
\tag{2.1}
$$

where $\mathbb{I}_3$ denotes the 3×3 identity matrix and the 4×4 matrices $O_L(\boldsymbol{q}), O_R(\boldsymbol{p}) \in \mathrm{SO}(4)$ (the special orthogonal group) are the matrix representations of the quaternion product from the left (by $\mathfrak{q}$) and from the right (by $\mathfrak{p}$), respectively. This definition also employs the matrix representation of the cross product $\widehat{\boldsymbol{\omega}} \boldsymbol{v} = \boldsymbol{\omega} \times \boldsymbol{v}$ denoted using the hat operator

$$
\widehat{\boldsymbol{\omega}} := \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}.
$$

The *conjugate* (or adjoint) for a quaternion $\mathfrak{q}$ is defined to be $\mathfrak{q}^* = q_s - \mathfrak{q}_v$ and, noting that the vector part of $\mathfrak{q}^* \circ \mathfrak{q}$ is zero, its norm is given by $\|\mathfrak{q}\|^2 = \mathfrak{q}^* \circ \mathfrak{q} = \boldsymbol{q}^\top \boldsymbol{q}$ which agrees with the usual Euclidian norm on $\mathbb{R}^4$. The inverse of $\mathfrak{q}$ is then given by $\mathfrak{q}^{-1} = \mathfrak{q}^* / \|\mathfrak{q}\|^2$.

Like conventional rotation matrices, *unit* quaternions ($\mathfrak{q}$ with $\|\mathfrak{q}\| = 1$ or, equivalently, $\boldsymbol{q}$ on the unit sphere S^3) can be used to represent spacecraft attitudes and rotations. Returning to our previous example, the vector $\boldsymbol{v}^b \in \mathbb{R}^3$ written in $\mathcal{F}_b$ can re-coordinatized in $\mathcal{F}_i$ using the quaternion

$\mathfrak{q} \in \mathbb{H}$ or its corresponding rotation matrix $C(\boldsymbol{q}) \in \mathrm{SO}(3)$ as follows:

$$\widetilde{\boldsymbol{v}}^{\,i} = \mathfrak{q} \circ \widetilde{\boldsymbol{v}}^{\,b} \circ \mathfrak{q}^*, \tag{2.2a}$$

$$= O_L(\boldsymbol{q})\, O_R(\boldsymbol{q}^*)\, \widetilde{\boldsymbol{v}}^{\,b},$$

$$\boldsymbol{v}^i = C(\boldsymbol{q})\, \boldsymbol{v}^b. \tag{2.2b}$$

where $\widetilde{\boldsymbol{v}} \coloneqq [0; \boldsymbol{v}]$ is the so-called *pure* representation of the vector $\boldsymbol{v}$ and

$$C(\boldsymbol{q}) \coloneqq q_s^2\, \mathbb{I}_3 + 2q_s \widehat{\boldsymbol{q}}_v + \boldsymbol{q}_v \boldsymbol{q}_v^\top + \widehat{\boldsymbol{q}}_v^{\,2}.$$

The above definition for $C(\boldsymbol{q})$ reveals one of the primary challenges in the practical implementation of quaternions: the quaternion space S^3 is a double covering of the attitude space SO(3). That is, the quaternions $\pm\boldsymbol{q}$ (antipodal points on the sphere S^3) represent the *same* orientation and yield the same rotation matrix. Correspondingly, any formula for the reverse mapping $\bar{C} : \mathrm{SO}(3) \to \mathrm{S}^3$ (such as that given by [47]) cannot produce unique results. The equator between these dual representations shown in black in Figure 2.1 then (also redundantly) collects all possible 180° rotations from $\boldsymbol{q}_d$. As a result, the full space of physical attitudes is included in any *single* closed hemisphere of S^3: a feature which produces problematic results for some control laws. For example, consider travelling from $-\boldsymbol{q}_0$ to $\boldsymbol{q}_d$ along the magenta path shown in Figure 2.1. In this case, the observed rotation appears to go the 'wrong' way, travelling more than 180° to reach the target attitude. Extending this example, controllers which follow the longer geodesic from $-\boldsymbol{q_0}$ to $\boldsymbol{q_d}$ (fully around the opposite side S^3) will, in practice, appear to initially spin the correct direction. However, they will then appear to spin *past* the target attitude, completing an additional 360° rotation. This inefficient behavior is known as *unwinding* and is produced by control laws which stabilize almost all of S^3 to $\boldsymbol{q}_d$ only (as opposed to stabilizing each hemisphere to $\pm\boldsymbol{q}_d$ respectively). Modern feedback regulating controllers prevent unwinding via symmetric control laws (satisfying $\boldsymbol{u}(q) = \boldsymbol{u}(-q)$) which independently stabilize each half of S^3 to the appropriate $\pm\boldsymbol{q}_d$ (using Hybrid techniques to include the equator) [35]. We elaborate on these challenges (as well as potential solutions) in later chapters.

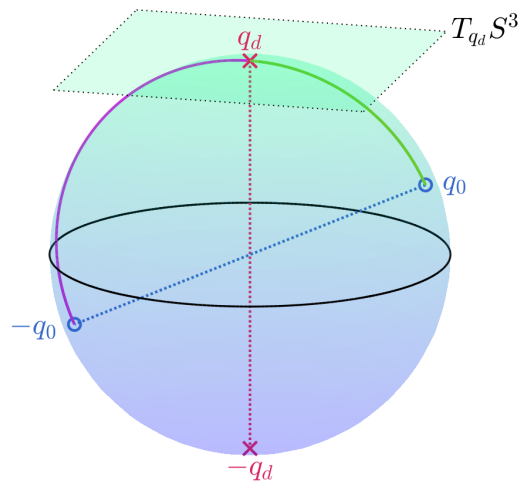

Figure 2.1: Illustration of quaternion attitude space S^3.

Another useful attitude representation that is frequently employed in spacecraft modelling are Modified Rodriguez Parameters (MRP's). Specifically, a quaternion $\boldsymbol{q}$ can be converted to its equivalent MRP vector in $\mathbb{R}^3$ as follows

$$\boldsymbol{e} \tan\left(\frac{\phi}{4}\right) = \frac{\boldsymbol{q}_v}{1 + q_s},$$

where the unit vector $\boldsymbol{e} \in \mathrm{S}^2$ defines the principal axis of the applied rotation and $\phi \in [0, 2\pi)$ yields the principal angle. Notably, this representation is ultimately a vector in $\mathbb{R}^3$ (with the principle angle encoded as the magnitude), and so is a minimal attitude representation. Correspondingly, the MRP encounters a coordinate singularity when describing a full rotation ($\phi = 360°$), though this can be avoided in practice by pro-actively switching between dual (shadow) MRP representations before the principle angle grows too large [10]. Although they are not used for modelling attitudes in this work (due to the dynamical challenges with coordinate singularities discussed above), this compact form is often useful for visualization and error tracking. In particular, the principle rotation angle ϕ serves as a useful performance metric for assessing the convergence rate and accuracy of attitude controllers, and is used for this purpose in the study in chapter 4.

2.2 Body-Torque Spacecraft Dynamics and Actuators

Having defined the spacecraft's attitude model, we may now discuss the most prevalent model for a spacecraft's attitude dynamics as well as models for many common attitude actuators. To define these dynamics, let $\mathcal{F}_b$ and $\mathcal{F}_i$ define the reference frames for the spacecraft body and inertial space as before, let the quaternion $\boldsymbol{q}$ represent the spacecraft's orientation, and let $\boldsymbol{\omega} \in \mathbb{R}^3$ define the angular rotation rate of the satellite frame (i.e. the rotation rate of $\mathcal{F}_b$ with respect to $\mathcal{F}_i$) written in $\mathcal{F}_b$. In addition, we require the following standard assumptions to simplify and standardize controller design under this model:

(1) The mass of the spacecraft remains constant,

(2) The principle Inertia $J \in \mathbb{R}^{3\times 3}$ of the platform remains constant,

(3) All actuators can be modelled using external body-torques.

In practice, assumption (1) requires that the propellant (fuel) used by the spacecraft is used sufficiently slowly so that the mass of the spacecraft is effectively constant over any single attitude maneuver (both reasonable and commonplace in most applications). Assumption (2) is arguably more restrictive, as it assumes that the spacecraft is entirely rigid and includes no moving parts with substantial mass (not always true for larger spacecraft such as the ISS). Finally, assumption (3) requires that every attitude actuator on the spacecraft can ultimately be modelled using torques acting *externally* on the spacecraft body (e.g. leveraged against the surrounding space or electromagnetic fields). While this proves to be a reasonable assumption for certain classes of actuators (e.g. conventional thrusters), it presents substantial challenges when applied to the Momentum Exchange Devices (MED's) used in a substantial portion of spacecraft.

Under these assumptions, the body-frame dynamics for a torque-driven satellite can be written as follows

$$\dot{\boldsymbol{q}} = {}^1\!/_2\, O_L(\boldsymbol{q})\, \widetilde{\boldsymbol{\omega}}\,, \tag{2.3a}$$

$$J\dot{\boldsymbol{\omega}} = -\widehat{\boldsymbol{\omega}}\, J\boldsymbol{\omega} + \boldsymbol{\tau}_c + \boldsymbol{\tau}_e\,. \tag{2.3b}$$

where J is the satellite's inertia tensor, $\boldsymbol{\tau}_c \in \mathbb{R}^3$ is the applied control torque, and $\boldsymbol{\tau}_e \in \mathbb{R}^3$ collects all measured external disturbance torques on the spacecraft (e.g. solar pressure, atmospheric drag, etc) [24]. Although included here for completeness, the external torques $\boldsymbol{\tau}_e$ are frequently ignored as negligible disturbances in feedback controller design. For comparison, the equivalent rotation matrix dynamics to (2.3a) on SO(3) is

$$\dot{C}_b^i = C_b^i \, \widehat{\boldsymbol{\omega}},$$

where, as before, the columns of $C_b^i = C(\boldsymbol{q})$ yield the axes of the body frame $\mathcal{F}_b$ viewed in the inertial frame $\mathcal{F}_i$.

In practice, the control torque $\boldsymbol{\tau}_c$ is purposefully left arbitrary, as it can be generated by a variety (and usually a combination) of different spacecraft actuators. These actuators commonly include thrusters, magnetic torquers, and MED's such as reaction wheels and control-moment-gyroscopes. We briefly review the relative strengths and challenges these common actuators below.

2.2.1 Thrusters

From the perspective of a control systems engineer, gas jets or *thrusters* provide the most convenient and direct actuator model for translating from requested (planned) maneuver command torques into actuator control inputs. Specifically, given the requested command torque $\boldsymbol{\tau}_c \in \mathbb{R}^3$, the actuator equation for an array of m thrusters with control inputs $\boldsymbol{u}_t \in \mathbb{R}^m$ can be written as follows:

$$\boldsymbol{\tau}_c = D_T \, \boldsymbol{u}_t, \tag{2.4}$$

Where the columns of the matrix $D_T \in \mathbb{R}^{3\times m}$ yield the appropriate torque axis and scaling coefficients for the moment produced by each individual thruster [28]. Naturally, this simple model readily meets all of the above design assumptions for the dynamical model (2.3), with the fuel requirements for standard rotations usually having a negligible impact on the satellite's mass and inertia. Additionally, thrusters provide a substantial amount of torque, with 'cold' gas jets (produce thrust via phase change) and 'hot' gas jets (produce thrust via chemical reaction) generating

nominal thrusts of 5 N and 9 kN respectively [31]. Additionally, thrusters can produce these control torques at any time, with the satellite's ultimate propellant capacity (minus requirements for orbital station keeping) being the only major limitation.

Naturally, thrusters do present certain engineering challenges in their implementation. First, thrusters are highly nonlinear in their production of thrust due to a wide range of mechanical challenges in their design. As a result, control commands sent to them are normally encoded using Pulse Width Modulation, thereby requiring a discrete time model when designing a stabilizing feedback law [28]. Secondly, thruster exhaust can contaminate and degrade surfaces on the spacecraft, potentially limiting mission performance and lifetime. Finally and most critically, thrusters require consumable propellants for their operation: a finite resource that is normally reserved for necessary operations (e.g. orbital station keeping). As a result, thrusters are *rarely* used during standard attitude maneuvers, and are reserved for emergencies or to regulate (or dump) the angular momentum accumulated from the external disturbance torques $\boldsymbol{\tau}_e$ [31].

2.2.2 Magnetic Torquers

As their name might suggest, magnetic torquers use magnetic coils or electromagnets to generate magnetic dipole moments against the Earth's natural magnetic field. The torque produced by this dipole moment is orthogonal (and proportional) to the Earth's magnetic field and so is less effective at higher altitudes [31]. However, these devices do not require propellants and produce torques leveraged on the magnetic field external to the satellite's frame, perfectly satisfying all of the assumptions for use in the dynamics (2.3). Given the time-varying magnetic field $\boldsymbol{B}_{EM}(t) \in \mathbb{R}^3$ along the satellite's orbit, the actuator equation for m magnetic torquers with magnetic moments $\boldsymbol{m}_i \in \mathbb{R}^3$ and current inputs $\boldsymbol{u}_m \in \mathbb{R}^m$ is given by [28] to be

$$\begin{aligned} \boldsymbol{\tau}_c &= \sum_i \left[\boldsymbol{m}_i \times \boldsymbol{B}_{EM}(t)\right] u_{m,i} \,, \\ &= D_B(t)\, \boldsymbol{u}_m \,. \end{aligned} \tag{2.5}$$

Apart from its natural time dependency, the actuator Jacobian $D_B(t) \in \mathbb{R}^{3\times m}$ bears a striking resemblance to D_T in (2.4), though it is necessarily low rank due to the single direction of the Earth's magnetic field. Additionally, magnetic torquers are *far* less powerful than thrusters, with normal torque ranges being measured in mN's. As a result, magnetic torquers are also used mainly for periodic removal of accumulated momentum from the disturbance torques $\boldsymbol{\tau}_e$.

2.2.3 Reaction Wheels

In much the same way that orbits are best described and planned referencing a spacecraft's linear momentum (and the momentum available from the onboard propellant), spacecraft attitude control can be understood as the management of the platform's *angular* momentum. From this perspective, the above attitude actuators (thrusters and magnetic torquers) generate torques *external* to the spacecraft body, exchanging the platform's inertial angular momentum with the surrounding space. However, the propellant (or time) required by such actuators make them problematically inefficient for use in routine attitude maneuvers. Instead, these actuators are primarily used to regulate the accumulated external disturbance torques $\boldsymbol{\tau}_e$ on the satellite frame. This strategy simultaneously minimizes propellant use and ensures that the inertial angular momentum of the platform remains a conserved quantity.

To avoid unnecessarily dumping angular momentum into the surrounding space, modern spacecraft ADCS's instead employ simple electric motors (and renewable electric power) to redistribute the satellite's *internal* angular momentum between its various components. The simplest such MED is called the Reaction Wheel (RW): an electric motor with a high-inertia rotor mounted to the satellite's frame [31]. When this motor applies torque to rotate the wheel, the resulting *reaction* torque of the wheel is used to rotate the satellite's frame. Provided that the wheels are not near saturation (their maximum stable spin rate), this torque ($\leq 1\,\mathrm{N\,m}$) is perfectly suitable for smaller spacecraft and (unlike magnetic torquers) can be generated at any time [33]. As such, an array of RW's mounted on the platform's principle inertia axes produces an effective and reliable attitude control system for smaller spacecraft.

However, since MED's are fundamentally governed the conservation of angular momentum (e.g. do *not* use external body torques), they do *not* naturally satisfy the second and third assumptions for the dynamics (2.3). Nonetheless, the resulting non-physicality can be (and frequently is) treated as a disturbance in the resulting feedback law, which has a substantially improved capacity to regulate such disturbances given the improved availability of renewable electric power. Specifically, let each reaction wheel in an array of m wheels have the scalar momentum $h_{swr,i}$ along its (unit) spin axis $\boldsymbol{a}_{s,i} \in \mathbb{R}^3$ in the body frame $\mathcal{F}_b$. For compactness, let these spin axes be collected in the matrix

$$A_s \coloneqq [\boldsymbol{a}_{s,1}, \cdots, \boldsymbol{a}_{s,m}] \in \mathbb{R}^{3\times m},$$

and let the wheel momenta be collected in the vector $\boldsymbol{h}_{swr,i} \in \mathbb{R}^m$. The resulting actuator equation for an array of m RW's can then be written as follows:

$$\boldsymbol{\tau}_c = \widehat{\boldsymbol{\omega}} A_s \boldsymbol{h}_{swr} + A_s\, \boldsymbol{u}_{RW}\,, \tag{2.6}$$

where $\boldsymbol{u}_{RW} \in \mathbb{R}^m$ are the torque control inputs for the individual reaction wheels [28]. To accommodate the dynamics (2.3) via dynamic cancellation, the gyroscopic reaction term (and, potentially, the entire right side of (2.3b)) can be directly counteracted in the control design by simply redefining the target command torque as follows:

$$\underbrace{\boldsymbol{\tau}_c - \widehat{\boldsymbol{\omega}} A_s \boldsymbol{h}_{swr}}_{\boldsymbol{\tau}_{rw} \coloneqq} = A_s\, \boldsymbol{u}_{RW}\,. \tag{2.7a}$$

That is, given the desired command torque $\boldsymbol{\tau}_c$ for the dynamics (2.3), the required RW command torque $\boldsymbol{\tau}_{rw}$ can be computed following the left side of (2.7). This required torque can then be generated by the wheels according to the wheel allocation specified by the right side of (2.7). Note once again that this method ultimately produces an (adjusted) command torque $\boldsymbol{\tau}_{rw}$ and a static actuator Jacobian $D_{RW} = A_s$ following the thruster archetype in (2.4). The motivation for this very intentional symmetry will become readily apparent in the next section.

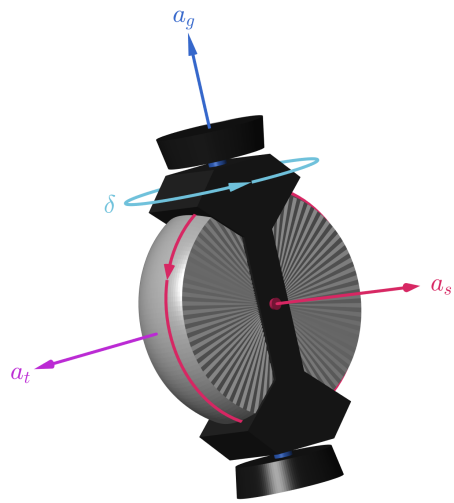

Figure 2.2: Control Moment Gyroscope coordinate frame.

2.2.4 Control Moment Gyroscopes

In practice, the limited momentum capacity of simple RW's make them inefficient for heavier spacecraft. Specifically, the mechanical shaft power $P_m = \omega_m \tau_m$ required for a motor to generate a torque τ_m increases linearly with the motor shaft speed ω_m. Thus, RW torque generation is inefficient at high wheel speeds, while complex friction effects also make RW's challenging to accurately model at low speeds. Thus, even for smaller spacecraft, RW control systems require active wheel speed regulation to avoid both effects [33].

Evolving from the design of the RW, the Control Moment Gyroscope (CMG) is a reaction wheel mounted on a rotating gimbal as shown in Figure 2.2, where the wheel (red) and gimbal (blue) motors act along the $\boldsymbol{a}_s$ and $\boldsymbol{a}_g$ axes respectively. Rather than using the direct motor reaction torques for attitude control, a CMG instead relies on the *gyroscopic* reaction torque

$$\boldsymbol{\tau}_r = \dot{\delta} h_{swr} \boldsymbol{a}_t,$$

produced along the transverse axis $\boldsymbol{a}_t \coloneqq \boldsymbol{a}_s \times \boldsymbol{a}_g$. This design is more mechanically complex, but offers substantial performance benefits. Specifically, the reaction torque $\boldsymbol{\tau}_r$ is proportional to the rotation rate $\dot{\delta} \in \mathbb{R}$ of the gimbal frame (*not* the gimbal motor torque $\boldsymbol{\tau}_g$) and is amplified by the

rotor momentum $h_{swr} \in \mathbb{R}$. This so-called *torque amplification* effect allows CMG's to efficiently generate much larger output torques than RW's (e.g. $\geq 1000\,\mathrm{N\,m}$) [33].

However, while a RW's torque axis remains fixed in the body frame $\mathcal{F}_b$, a CMG's output torque axis $\boldsymbol{a}_t$ rotates with the gimbal angle $\delta \in [0, 2\pi)$. As such, the available torque from an array of m CMG's varies with the array's *configuration* $\boldsymbol{\delta} := [\delta_1; \cdots; \delta_m]$, requiring more sophisticated maneuver planning strategies. Specifically, following the notation in [17], the available torque spaces for the CMG gimbal and wheel motors are spanned respectively by the column spaces of the matrices

$$\begin{aligned} A_s &:= [\boldsymbol{a}_{s,1}, \cdots, \boldsymbol{a}_{s,m}] \in \mathbb{R}^{3\times m}, \\ A_t &:= [\boldsymbol{a}_{t,1}, \cdots, \boldsymbol{a}_{t,m}] \in \mathbb{R}^{3\times m}, \end{aligned} \tag{2.8}$$

which vary with the array configuration $\boldsymbol{\delta}$ following

$$\begin{aligned} A_s(\boldsymbol{\delta}) &:= A_{s0}\,\mathrm{diag}(\cos(\boldsymbol{\delta})) - A_{t0}\,\mathrm{diag}(\sin(\boldsymbol{\delta})), \\ A_t(\boldsymbol{\delta}) &:= A_{t0}\,\mathrm{diag}(\cos(\boldsymbol{\delta})) + A_{s0}\,\mathrm{diag}(\sin(\boldsymbol{\delta})), \end{aligned} \tag{2.9}$$

where the functions $\sin(\cdot)$ and $\cos(\cdot)$ act entry-wise for vector inputs and the matrices $A_s(0) = A_{s0}$ and $A_t(0) = A_{t0}$ define the default configuration of the array geometry. For completeness, the gimbal axes (fixed in the satellite body frame) are collected in the constant matrix $A_g = [\boldsymbol{a}_{g,1}, \cdots, \boldsymbol{a}_{g,m}]$. Under this notation, the column space of A_t compactly summarizes the available gyroscopic reaction torques from the gimbal motors, while that of A_s and A_g describe the direct reaction torques available from the wheel and gimbal motors respectively. For example, consider the default configuration for the popular rooftop array geometry shown in Figure 2.3. In this configuration, the array can generate direct (red) reaction torques around the y-axis using the wheel motors and within the xz-plane using a combination of (purple) gyroscopic reaction torques from the gimbal motors.

Like with RWs, a feedback control law can be designed by employing dynamic cancellation and an appropriate actuator Jacobian $D_\omega \in \mathbb{R}^{3\times m}$ to yield the now familiar actuator equation form:

$$\boldsymbol{\tau}_{SR} = D_\omega\, \boldsymbol{u}_{SR,\dot{\delta}}\,, \tag{2.10}$$

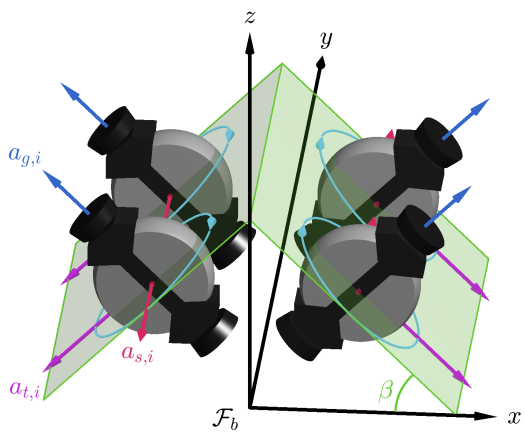

Figure 2.3: A 4-CMG array in rooftop configuration with inclination $\beta = 45°$.

where $\boldsymbol{u}_{SR,\dot{\delta}} \in \mathbb{R}^m$ collects the CMG gimbal rotation rates $\dot{\boldsymbol{\delta}} \in \mathbb{R}^m$ for the m CMG's in the array, which are themselves controlled by the gimbal motor torques $\boldsymbol{u}_g \in \mathbb{R}^m$. Notably, while the m CMG wheel motor inputs $\boldsymbol{u}_w \in \mathbb{R}^m$ can *also* be used to produce additional control torques, they are frequently omitted from the actuator equation (2.10) and are instead used solely for internal regulation of the CMG wheel speeds. The specifics, benefits, and challenges of this control strategy are developed in the next section.

2.3 Spacecraft Operational Constraints

Before investigating existing control solutions, it is first prudent to review the predominant operational (safety) restrictions for a spacecraft attitude control law. Broadly speaking, this includes (1) classic control input saturation constraints (to avoid damage or performance issues outside the actuator's operational limits), angular rate limits for the spacecraft's rotation (to avoid straining the platform frame), and exclusion cone constraints for the onboard optical sensors to avoid damage or signal loss when pointing them at bright stellar objects [19, 31]. These constraints

can each be written in the nonlinear inequality form $c_j(\boldsymbol{x}, \boldsymbol{u}) \leq 0$ as follows:

$$u_j^2 - u_{j+}^2 \leq 0, \quad j \in 1, \ldots, m, \tag{2.11a}$$

$$\omega_j^2 - \omega_{j+}^2 \leq 0, \quad j \in 1, 2, 3, \tag{2.11b}$$

$$(\boldsymbol{\psi}_s^{\,i})^\top C(\boldsymbol{q})\, \boldsymbol{\psi}_c^{\,b} - \cos(\phi_{ko}) \leq 0, \tag{2.11c}$$

where $u_{j+} \in \mathbb{R}_{>0}$ are the control saturation limits for the particular actuator, $\omega_{j+} \in \mathbb{R}_{>0}$ are the maximum rotation rates around each body frame axis, $\boldsymbol{\psi}_c^{\,b} \in \mathrm{S}^2$ is the body-fixed frame orientation of the onboard optical sensor, and $\boldsymbol{\psi}_s^{\,i} \in \mathrm{S}^2$ and $\phi_{ko} \in [0, \pi]$ are the inertial orientation and minimum avoidance angle for the light source. Notably, the exclusion cone constraint (2.11c) ensures that the vectors $\boldsymbol{\psi}_s$ and $\boldsymbol{\psi}_c$, when expressed in the same reference frame, remain at least ϕ_{ko} radians apart as the spacecraft rotates. As such, this constraint is non-convex and presents the greatest challenge for real-time optimization.

2.4 Prevalent Attitude Feedback Control Solutions

Armed now with basic models for spacecraft dynamics and actuators, we now review prevalent approaches for designing the feedback control law in a spacecraft's Attitude Determination and Control System (ADCS). In particular, we will discuss the popular *Singularity Robust* (SR) control law presented in [38]: a feedback law which uses an actively regularized pseudo-inverse to compute the CMG gimbal rates for a stabilizing body torque $\boldsymbol{\tau}_c$. This feedback law serves as an excellent example of the underlying design philosophy and the inherent challenges resulting from the required dynamical approximations. Additionally, this law will be used in later sections as a relevant comparison for the performance improvements presented by our optimal solutions.

2.4.1 Pseudo-Inverse Feedback, Singularities, and the SR-law

As discussed above, this controller design strategy begins with the assumptions motivating the approximated dynamics (2.3) which we repeat below for reference:

$$\dot{\boldsymbol{q}} = {}^1\!/_2\, O_L(\boldsymbol{q})\, \widetilde{\boldsymbol{\omega}}\,,$$

$$J\dot{\boldsymbol{\omega}} = -\widehat{\boldsymbol{\omega}}\, J\boldsymbol{\omega} + \boldsymbol{\tau}_c + \boldsymbol{\tau}_e\,.$$

Namely, these dynamics are considered 'approximated' as they only model the effects of external body-torques, ignoring the momentum exchange dynamics governing the function of momentum exchange devices like reaction wheels and CMG's. Indeed, the 'true' dynamics of an MED array (and its unique state manifold) are fundamentally governed by the conservation of the spacecraft's total angular momentum: a complex constraint requiring a comprehensive model of *all* angular momentum exchanged within the array and spacecraft frame. Naturally, this complete model is extremely challenging (both behaviorally and computationally) for direct use in spacecraft control design which, historically, has preferred simplicity and reliability over raw performance. As a result, the *intentionally* simplified dynamical model above presents compelling benefits in computational complexity (within a field well known for limited available processing power), simplicity, and reliability.

Under these approximated dynamics, the actuator dynamics are fully decoupled from the spacecraft's attitude dynamics. As a result,the problems of attitude stabilization and the control of the relevant actuators can be separated, simplifying the individual controller designs and improving reliability. Specifically, we can first design a top-level attitude feedback controller to stabilize the body-torque dynamics (2.3) using the generalized command torques $\boldsymbol{\tau}_c$. For example, the SR control law in [38] stabilizes both the attitude $\boldsymbol{q}$ and the angular rate $\boldsymbol{\omega}$ to the target values $\boldsymbol{q}_f, \boldsymbol{\omega}_f$ respectively using the following feedback law:

$$\boldsymbol{\tau}_{SR} = (-\widehat{\omega} J\boldsymbol{\omega} + \boldsymbol{\tau}_e) + k_q \operatorname{Im}\left(O_R(\boldsymbol{q}_f^*)\, \boldsymbol{q}\right) + K_\omega(\boldsymbol{\omega} - \boldsymbol{\omega}_f) - J\dot{\boldsymbol{\omega}}_f, \tag{2.12}$$

where the first term enforces the dynamic cancellation for (2.3b) (i.e. that $J\dot{\boldsymbol{\omega}} = \boldsymbol{\tau}_{SR}$) and the remaining three terms provide Proportional-Derivative feedback on errors in the attitude and an-

gular rates with the gains $k_q \in \mathbb{R}_{>0}$ and $K_\omega \in \mathbb{R}^{3\times 3}$ respectively (normally chosen diagonal and critically damped with diagonal elements $K_{\omega,i} = \sqrt{2k_q J_i}$).

With the stabilizing command torque $\boldsymbol{\tau}_{SR}$ in hand, we now develop a *secondary* controller to produce this torque from the available actuators. As developed above, the common spacecraft actuator equations can all be written in the form

$$\boldsymbol{\tau}_c = D\,\boldsymbol{u},$$

using the appropriate actuator Jacobian $D \in \mathbb{R}^{3\times m}$ and control inputs $\boldsymbol{u} \in \mathbb{R}^m$. For the CMG's considered in the SR control law, this is given by

$$\boldsymbol{\tau}_{SR} = D_\omega\,\boldsymbol{u}_{\boldsymbol{SR},\dot{\boldsymbol{\delta}}},$$

where the actuator Jacobian D_ω is state-dependent and takes the form

$$D_\omega \coloneqq {}^1\!/_2\Big[A_s \operatorname{diag}\Big(A_t^\top(\boldsymbol{\omega}+\boldsymbol{\omega}_f)\Big) + A_t \operatorname{diag}\Big(A_s^\top(\boldsymbol{\omega}+\boldsymbol{\omega}_f)\Big)\Big](\boldsymbol{J}_t - \boldsymbol{J}_s) - A_t \operatorname{diag}(\boldsymbol{h}_{swr}), \tag{2.13}$$

where the matrices $m \times m$ diagonal matrices $\boldsymbol{J}_s$, $\boldsymbol{J}_t$, and $\boldsymbol{J}_g$ are constructed using the principle inertias $J_{g,i}$, $J_{s,i}$, and $J_{t,i} \in \mathbb{R}$ of the individual CMG's along their gimbal, spin, and transverse axes respectively (e.g. $\boldsymbol{J}_s \coloneqq \operatorname{diag}([J_{s,1}, \cdots, J_{s,m}])$). Following a strategy often employed in the control of robotic manipulators, the command torques $\boldsymbol{\tau}_{SR}$ are then converted to the minimum norm CMG gimbal rates $\dot{\boldsymbol{\delta}} = D^\dagger \boldsymbol{\tau}_r$ using a Moore-Penrose pseudo-inverse

$$D^\dagger \coloneqq D^\top (DD^\top)^{-1},$$

of the actuator Jacobian (2.13). This pseudo-inverse can be similarly implemented for each of the actuators discussed above (as well as combinations of them) using the appropriate Jacobian to produce the individual actuator inputs.

As one might suspect, this pseudo-inverse strategy is not universally effective and, in fact, presents serious challenges for many of the actuators discussed above. Like with robotic manipulators, this feedback control strategy suffers from the kinematic singularities produced in any

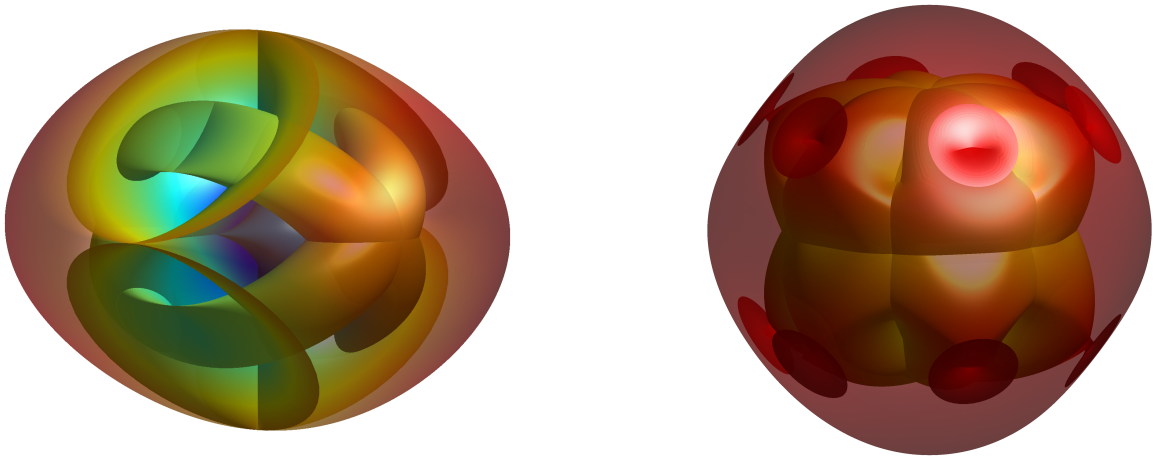

Figure 2.4: Singular configurations of a 6-CMG (a) rooftop and (b) pyramid array projected into the array's momentum workspace.

configuration where the actuator Jacobian D is low rank. In and around such *singular configurations* or *singularities*, the matrix $DD^\top$ is either ill-conditioned or not invertible, resulting in inefficient or impossible numerical actuator commands from the pseudo-inverse $D^\dagger$.

This issue is particularly prevalent in (single-speed) CMG's, as every array geometry possesses a nontrivial locus of array configurations $\boldsymbol{\delta}$ for which the matrix $A_t(\boldsymbol{\delta})$ (which shares rank with the Jacobian D_ω above) is low rank. As shown by the projections in Figure 2.4, these singular configurations are numerous in any array's configuration space and are highly dependent on the specifics of the array geometry (see [4] for the most prevalent classification system). In each of these configurations, the array is unable to return arbitrary torques that may be demanded by this psuedo-inverse feedback. For example, the default configuration (all $\delta_i = 0$) of the rooftop array geometry shown in Figure 2.3 cannot produce torques along the y axis because all the CMG torque axes are coplanar. Notably, this issue arises primarily for CMG designs which elect to use *only* the CMG gimbal motors to produce $\boldsymbol{\tau}_{SR}$ as in the SR law above, which assumes a fixed (internally regulated) wheel speed maintained using the individual CMG wheel motors. Feedback laws which employ the wheel motors for additional torque generation (i.e. as conventional reaction wheels) are separately classified as *Variable Speed* CMG's (VSCMG's) and present substantially

fewer kinematic singularities due to the additional control authority (i.e. only those produced by fully stopping or saturating the wheels). Notably, VSCMG's are necessarily less efficient in the vicinity of traditional CMG singularities (due to additional reliance on the less efficient reaction torque), though this is almost universally preferable to the alternative loss in control authority.

As one might suspect from its name, the Singularity Robust (SR) control law accommodates these singularities in an inefficient, but safe manner. Specifically, the law adds an adaptive regularization term to the pseudo-inverse which scales based on the singularity metric $\det(DD^\top)$. This robust pseudoinverse takes the form

$$D^\ddagger := D^\top \left[DD^\top + \alpha_0 e^{-\det(DD^\top)} \mathbb{I}_m \right]^{-1},$$

where $\alpha_0 \in \mathbb{R}_{>0}$ is a (small) weighting coefficient. Notably, this solution produces errant torques in the vicinity of singularities, but is robust in that an approximate solution is always available. Given the target attitude $\boldsymbol{q}_f$ and angular rate $\boldsymbol{\omega}_f$, the complete SR feedback law (including the internal wheel speed regulation) for the CMG array is then given by:

$$\boldsymbol{u}_{SR,\dot{\delta}} = D^\ddagger_\omega \boldsymbol{\tau}_{SR}, \tag{2.14a}$$

$$\boldsymbol{u}_{SR,\ddot{\delta}} = k_\delta \left[\boldsymbol{u}_{SR,\dot{\delta}} - \dot{\boldsymbol{\delta}} \right], \tag{2.14b}$$

$$\boldsymbol{u}_{SR,g} = \boldsymbol{J}_g \left[\boldsymbol{u}_{SR,\ddot{\delta}} + A_g^\top \boldsymbol{\omega} \right] - \dot{\boldsymbol{h}}_{ga}, \tag{2.14c}$$

$$\boldsymbol{u}_{SR,w} = k_w(\boldsymbol{h}_{swr} - h_w \mathbf{1}_m), \tag{2.14d}$$

where $\mathbf{1}_m \in \mathbb{R}^m$ is an m-vector of 1's, $k_\delta, k_w \in \mathbb{R}_{>0}$ are scalar feedback gains for state and tracking errors, $\boldsymbol{u}_{SR,g}, \boldsymbol{u}_{SR,w} \in \mathbb{R}^m$ are the motor control inputs for the CMG gimbal and wheel respectively, and $\boldsymbol{h}_{ga} \in \mathbb{R}^m$ collects the absolute CMG gimbal momenta (see chapter 5 for specifics).

Examining the complete feedback strategy in detail, (2.12) first computes a $(\boldsymbol{q}, \boldsymbol{\omega})$ stabilizing command torque for the satellite under the approximate attitude dynamics (2.3). Using the SR-law gimbal rates from (2.14a), the remaining internal array feedback equations (2.14b) and (2.14c) then produce the gimbal motor inputs $\boldsymbol{u}_{SR,g}$ which stabilize the gimbals to these target gimbal rates. Finally, (2.14d) computes the CMG wheel motor feedback $\boldsymbol{u}_{SR,g}$ to regulate the

CMG wheels to the target momentum h_w. The divide-and-conquer strategy used to construct this feedback strategy greatly simplifies the design (and, eventually, the optimization of) the top level attitude controller. Additionally (and of likely far greater significance for system's engineers), this formulation is naturally resilient to inevitable failure of individual actuators, as the actuator Jacobian can quickly be adjusted to exclude nonfunctional hardware. However, this strategy necessarily sacrifices performance by treating the original (conserved) momentum dynamics like a disturbance, thereby distorting the system's true state space.

2.4.2 Alternative Feedback Approaches

In addition to the SR law, many other feedback approaches have been developed in the literature to improve performance, reliability, and directly incorporate the operational safety constraints listed in section 2.3. Following the trends of industry and deployed systems, primary research in this area has generally adopted the accepted tiered controller design described above (i.e. attitude maneuver torque planning and actuator command torque tracking). As a result, the majority of efforts in feedback law design focus on each of these controllers individually and are summarized thus below.

Research dedicated solely to real-time feedback control solutions for the top level attitude maneuver planner is surprisingly sparse, as most results for this controller involve some manner of comprehensive maneuver planning (i.e. trajectory optimization, which will be reviewed and discussed later in chapter 3). As a direct alternative to the SR attitude feedback law (2.12) developed above, alternative PD and optimal Lyapunov feedback strategies for the attitude error were presented by [49]. Additionally, noting that (2.12) does not implicitly satisfy any of the constraints presented in section 2.3, both [26] and [53] incorporate the input saturation and exclusion cone constraints directly into the control feedback. In particular, [26] does this by discretizing the dynamics, convexifying the constraints, then solving for the corresponding quadratically constrained attitude feedback using semi-definite programming. Alternatively, [53] defines a finite set of constraint-admissable waypoints in $(C, \boldsymbol{\omega}) \in \mathrm{SO}(3) \times \mathbb{R}^3$, then develops a complementary outer-

loop supervisory switching strategy and inner-loop Lyapunov feedback law to track between feasible waypoints. Finally, the approach developed in [29] presents a unique feedback strategy which avoids CMG singularities by stabilizing to a finite set of pre-computed (offline) 'safe' attitude trajectories. Although somewhat limited in its capabilities and efficacy, this approach is unique in this category as it addresses an issue commonly left to the actuator controller directly in the maneuver planner.

In contrast, research on alternatives for or improvements upon the pseudo-inverse feedback solution for the actuator tracking controller are far more prevalent and can be divided into two fundamental approaches. The first approach focuses in on the specific choice or structure of the pseudo-inverse for the specific actuator, revising both to improve power consumption, provide tracking for specific power profiles, and avoid singularities using the null space for the specific actuator Jacobian. Regarding reaction wheels, relevant works include [44] and [8], which respectively employ a re-weighted Singular-Value Decomposition (SVD) for the pseudo-inverse and an optimization over an explicit parameterization of the Jacobian's null space to reduce power consumption in the generation of pre-planned command torques.

Naturally, research on control solutions for CMG's (and their many variants) which reduce the practical and numerical effects of kinematic singularities is extremely prevalent in the literature. Regarding classic CMG implementation's (which only use the gimbal motors for direct attitude control), a thorough review of classic steering law approaches can be found in [30]. Moving to VSCMG's (i.e. CMG's which also use reaction torque's from the the wheel motors for attitude control), [43, 45] follows a similar strategy to [8] discussed above and develops a parameterization of the Jacobian's null space, then employs that null space to avoid regions of lowered efficiency in the array's configuration space (i.e. the neighborhoods of singularities for classical CMG models). Interestingly, [43, 45] also employ a more comprehensive (momentum-conserving) model of the spacecraft's rotational dynamics to define an alternative feedback law for the top-level maneuver planner which actively uses both sets of actuators (the CMG gimbal and wheel motors) and their combined reaction and gyrosopic reaction torques (i.e. not using dynamic cancelation). Using this same model, [55] instead *extends* the Jacobian (and pseudo-inverse) to additionally minimize

deviation from a pre-planned power profile for the array. Finally, [42] extends the rotation physics and top level maneuver planner to Double-Gimble VSCMG's (the actuators employed on the ISS), and uses a re-weighted SVD for the corresponding psuedo-inverse to minimize array power usage.

The second approach used for improving the actuator tracking controller avoids the pseudo-inverse entirely, electing instead to use a different method for solving the actuator equation $\boldsymbol{\tau}_c = D\,\boldsymbol{u}$. Relevant works include [15] and [34] which, respectively, use lexographic convex optimization and the maximum principle to minimize the error $\|D\,\boldsymbol{u} - \boldsymbol{\tau}_c\|$ in the actuator equations for RW's and CMG's. Due to the complexity of the resulting problems, both solutions included iterative elements in their solvers, but additionally provide explicit feasibility guarantees for input saturation constraints.

Chapter 3

Trajectory Optimization Techniques

In this chapter, we will examine a review the broad subject of Trajectory Optimization (TO) and its applications in constrained spacecraft maneuver planning. Specifically, we will introduce the general structure of a constrained trajectory optimization problem as well as alternative interpretations and formulations. Next, we will examine a broad overview of existing numerical approaches for solving these problems and recent applicable research in the aerospace regime. Finally, we will specifically examine the PRojection-Operator based Newton method for Trajectory Optimization (PRONTO) used to solve the challenging problems considered in this thesis (see chapters 4 and 6) and its comparative benefits to other solvers.

3.1 A General Trajectory Optimization Problem

We begin by introducing the general structure for a finite-horizon continuous-time constrained trajectory optimization problem. Specifically, we will consider a nonlinear system with state $\boldsymbol{x}$ and control inputs $\boldsymbol{u}$ evolving within the constrained spaces X and U respectively (for example, the manifolds S^3 or SO(3) for quaternions and rotation matrices, and the respective regions in $\mathbb{R}^m$ defined by input saturation constraints). We further assume that this nonlinear system evolves under the C^2 dynamics $\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u})$ (twice continuously differentiable in both $\boldsymbol{x}$ and $\boldsymbol{u}$) over the finite horizon $t \in [0, T]$. Under this system, we desire a feasible solution maneuver satisfying the path constraints $c_j(\boldsymbol{x}, \boldsymbol{u})$ (for example, see section 2.3) and which minimizes the C^2 cost (or objective) functional h from the initial condition $\boldsymbol{x}(0) = \boldsymbol{x}_0$. This trajectory optimization and the

cost functional h can be written in the following form:

$$\begin{aligned} \min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)} \quad & h\left(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)\right) = \int_0^T \ell(\boldsymbol{x}(t),\boldsymbol{u}(t))\,dt \;+\; m(\boldsymbol{x}(T)), \\ \text{s.t.} \quad & \dot{\boldsymbol{x}} = f(\boldsymbol{x},\boldsymbol{u}), \quad \boldsymbol{x}(0) = \boldsymbol{x}_0, \\ & c(\boldsymbol{x},\boldsymbol{u}) \leq 0, \end{aligned} \tag{3.1}$$

where the notation $\boldsymbol{x}(\cdot)$ denotes the entire curve $\boldsymbol{x}(t), t \in [0,T]$ and the functions $\ell(\boldsymbol{x},\boldsymbol{u})$ and $m(\boldsymbol{x})$ denote general C^2 incremental (running) and terminal cost functionals respectively. This construction is extremely general, with the only noteworthy restrictions being the assumed structure for the cost function h (which can easily be exchanged if necessary) and the assumption that both the dynamics f and cost functional h both be C^2 (a necessity for establishing the local optimality of a solution).

While (3.1) accurately summarizes the problem and its complications however, there exists an alternative functional space representation that more intuitively expresses its greatest complication: the dynamical path constraint $\dot{\boldsymbol{x}} = f(\boldsymbol{x},\boldsymbol{u})$. Noting that we are discussing the subject of Trajectory Optimization, recall that a *trajectory* of the dynamics f is defined as any bounded curve $\boldsymbol{\xi}(t) := (\boldsymbol{x}(t),\boldsymbol{u}(t)), t \in [0,T]$ which satisfies an initial condition $\boldsymbol{x}(0) = \boldsymbol{x}_0$ and the dynamics $\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t),\boldsymbol{u}(t))$ for all $t \in [0,T]$. More intuitively, a trajectory ξ is any solution to the Ordinary Differential Equation (ODE)

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x},\boldsymbol{u}), \quad \boldsymbol{x}(0) = \boldsymbol{x}_0,$$

on the horizon $t \in [0,T]$ obtained via a control input $\boldsymbol{u}(t)$. Such trajectories are collected on a Banach manifold $\mathcal{T}$ called the *trajectory manifold* which is embedded in the ambient Banach space $\mathcal{X} \subset L^\infty([0,T],\mathbb{R}^{n+m})$ of bounded curves $\boldsymbol{\xi}(t), t \in [0,T]$ continuous in x and satisfying $\boldsymbol{x}(0) = \boldsymbol{x}_0$ [20]. This manifold $\mathcal{T}$ is precisely the search space for the trajectory optimization problem (3.1), which may be compactly rewritten as follows:

$$\begin{aligned} \min_{\boldsymbol{\xi}\in\mathcal{T}} \quad & h(\boldsymbol{\xi}) = \int_0^T \ell(\boldsymbol{x}(t),\boldsymbol{u}(t))\,dt \;+\; m(\boldsymbol{x}(T)), \\ \text{s.t.} \quad & c(\boldsymbol{x},\boldsymbol{u}) \leq \boldsymbol{0}. \end{aligned} \tag{3.2}$$

In addition to being more compact, this formulation is more representative of the optimal control problem we aim to solve. Specifically, by enforcing the system's dynamics implicitly in the search space, this formulation emphasizes that $\boldsymbol{x}(t)$ is a *dependent* variable determined from the control input $\boldsymbol{u}(t)$ via the integration of f. In contrast, (3.1), which encodes the dynamics as a constraint and searches over the ambient space $\mathcal{X}$, is written in a form that can be easily sampled, discretized, and transcribed as a Nonlinear Program (NLP). This subtle distinction illustrates a key practical (algorithmic) difference between the classic direct and indirect approaches to trajectory optimization discussed below.

3.2 Conventional Approaches

Existing solutions for trajectory optimization problems usually fall into two primary categories: *Direct* and *Indirect* Methods. Not all approaches fit neatly into these categories (genetic algorithms and Dynamic Programming being notable exceptions) and many methods combine elements of both to improve computation times or provide additional desirable properties. In general however, these two approach categories provide an intuitive overview of the diverse approaches for this problem. For a concise survey of many of the trajectory optimization strategies below and their applications to aerospace problems, I recommend [6] and [57] respectively.

3.2.1 Direct Methods

Speaking broadly, *Direct* methods consider the first problem formulation (3.1), with the system's dynamics viewed as an additional constraint to be numerically managed by the solver. In general, direct methods operate by generating a sequence of curves $(\boldsymbol{x}_k(\cdot), \boldsymbol{u}_k(\cdot))$ which progressively descend the objective function, usually achieving feasibility in the path constraints $c(\boldsymbol{x}, \boldsymbol{u})$ by including them as an additive penalty (i.e. barrier function) in the objective. This descent is usually computed numerically by *transcribing* (3.2) into a finite-dimensional form by sampling the curves $\boldsymbol{x}_k(\cdot)$ and $\boldsymbol{u}_k(\cdot)$ in time and incorporating the dynamics as an additional constraint on and between those samples. These samples can be single points, time-segments of the curves

represented by some polynomial fit, or even time-segments obtained by expressly integrating the dynamics. In any case, these samples do not implicitly connect at the join times between them, creating *defects* (with respect to the system's dynamics) which are then actively minimized by the solver as additional constraints.

As a result of this approximation, the iterates of direct methods cannot remain *on* the trajectory manifold $\mathcal{T}$ as they descend the objective function, but instead remain near to it (with their proximity managed as a separate element of the optimization problem). This transcription of the problem presents appealing speed benefits as the algorithm can step further at each iterate in a locally unconstrained way. However, this finite-dimensional representation of $\boldsymbol{x}_k(\cdot), \boldsymbol{u}_k(\cdot)$ can only be guaranteed to satisfy the dynamics and constraints *at* the sample join times. As a result, these solutions must be sampled finely enough so that potential violations between samples remain appropriately small. Common direct methods applied in aerospace research include Model Predictive Control (MPC), Differential Evolution (DE), and Pseudospectral methods and are reviewed below.

Broadly speaking, Model Predictive Control is a solver archetype which prioritizes solver efficiency and constraint feasibility guarantees by using simplified (usually discrete time) dynamics over a shortened, receding horizon which shifts with the system as it tracks the current solution iterate in real time. Unlike many optimal maneuver planners, the reduced complexity of this shortened horizon enables the solver to adapt to disturbances and constraints in real-time, providing many of the same benefits and guarantees of a traditional feedback solution. To account for the shortened horizon, a complementary stabilizing control law around the target state is included in this algorithm to ensure the solution iterates remain recursively feasible (i.e. able to reach the target using this control law past the solution horizon). Recent works employing this approach include [36] and [50], which plan optimal attitude maneuvers using command torques on S^3 (quaternions) and SO(3) respectively, with [36] additionally including exclusion cone constraints. Additionally, [56] develops a constrained MPC solution for safe relative spacecraft orbital maneuvers in the Hill reference frame (a linear reference frame centered on the approximate orbit): a challenging but critical problem for modern spacecraft design to enable automated refueling and repair operations

in orbit.

Alternatively, Pseudospectral methods present an alternative method for transcribing the optimization problem into a numerically tractable Non-Linear Program (NLP) which can then be solved rapidly using established high performance solvers and techniques (e.g. IPOPT, SNOPT, etc.). Rather than representing the curves $\boldsymbol{x}_k(\cdot)$ and $\boldsymbol{u}_k(\cdot)$ using simple time-value samples (which themselves are connected by dynamically infeasible straight lines or splines), these curves are represented as sequences of variable-order polynomials of a problem appropriate basis function (e.g. sinusoids, Bessel functions, Lagrange polynomials, etc.). The length, number, order, and fit precision of each of these curve segments are then managed as parameters of the NLP, allowing the solver to more easily capture traditionally challenging solution properties (e.g. the rapid changes found in bang-bang control solutions). Notably, this solution method does not contain a natural feedback solution for the resulting optimizer, but accommodates a far broader scope of challenging dynamical models due to the flexibility provided by the curve approximation. Regarding research using this approach, [40] provides an introduction and survey of pseudospectral methods in aerospace problems, while [3] provides a practical example of a Body-torque maneuver planner on S^3 which includes input saturation constraints. One commercial solver in particular, GPOPS II, has seen widespread adoption in this research space, and employs a basis of variable-order Lagrange polynomials which are actively adapted in number and (time) length to accommodate rapid changes in the solution (a technique called mesh-refinement) [39]. The Gauss-Pseudospectral method is developed in [5], presented as a commercial solver in [39], and applied to the body-torque maneuver planning problem on S^3 in [9]. Due to its widespread adoption and accommodation of highly nonlinear dynamics, GPOPS II will serve as a comparative example for the algorithm developed and applied to the problems in chapters 4 and 6.

Finally, Differential Evolution (as its name might suggest) selectively combines and refines a randomized population of potential solutions to the optimization problem to descend the cost function. Like psuedospectral methods, this approach provides no intrinsic method for tracking the resulting solution and the overall speed of the algorithm is highly dependent on the accuracy of

the initial guess (population). This method is relatively new within the aerospace problem space, but has still seen successful application by [54] and [52] for constrained body-torque maneuver planning using quaternions and MRP's. Notably, these papers also provide direct comparisons to pseudospectral approaches and demonstrate comparable computational performance.

3.2.2 Indirect Methods

Unlike direct methods, *indirect* methods consider the second problem formulation (3.2) and (at least in direct implementations) avoid the inherent transcriptions used in direct methods. Rather than iteratively descending the cost functional using approximate solution trajectories, they function by determining *properties* of the optimizer obtained by analytically solving the first-order necessary condition $\nabla h = 0$. For example, approaches using the maximum principle (the most common approach in this context), this yields an optimal control policy $\boldsymbol{u}^*(\boldsymbol{x}, \boldsymbol{\lambda})$ and dynamics $\dot{\boldsymbol{\lambda}}$ for a costate $\boldsymbol{\lambda}$. This costate evolves backwards in time from a specified boundary condition $\boldsymbol{\lambda}(T)$ and, when combined with the policy $\boldsymbol{u}^*$ and dynamics f, determines the optimizer $\boldsymbol{\xi}^*$ as the solution to a Two Point Boundary Value Problem (TPBVP). This approach converts the problem from finding an infinite dimensional $\boldsymbol{\xi}^* \in \mathcal{T}$ to finding a *finite* dimensional initial condition $\boldsymbol{\lambda}(0)$ which evolves over $\boldsymbol{\xi}^*(t)$ to $\boldsymbol{\lambda}(T)$. Once this $\boldsymbol{\lambda}(0)$ is determined, the solution $\boldsymbol{\xi}^*$ is then generated directly *on* $\mathcal{T}$ with numerical errors originating *only* from the tolerance of the numerical integrator. This first approach is commonly referred to as single-shooting, as the state and costate both evolve continuously over the entire problem horizon $t \in [0, T]$. Alternatively, multiple shooting involves subdividing the problem into multiple subintervals (each of which are independently solved using the above approach), with some outer-loop collocation strategy (i.e. pseudospectral methods) used to reduce the errors between the boundary conditions of subsequent trajectory segments [25].

Unfortunately, indirect methods tend to be very slow for nonlinear constraints or dynamics because the effects of the iterate $\boldsymbol{u}_k(\cdot)$ are only measured *cumulatively* (i.e. via integration of $\dot{\boldsymbol{\lambda}}$). This means that any derivatives used to update the guess $\boldsymbol{\lambda}(0)$ must be computed numerically. This sensitivity gives these solvers a narrow region of convergence and tends to make results highly

numerically unstable with respect to variations in the initial condition [6, 25]. Most critically, the use of an open-loop integration step in computing the iterates makes such indirect methods impractical for highly unstable systems as the resulting solutions can diverge rapidly even on short time horizons. This last issue is also present in some integration-based direct approaches (e.g. single or multiple shooting) and serves as the original inspiration for the approach discussed below.

Due to the high nonlinearity and complexity of spacecraft dynamics, aerospace research using purely using indirect methods is quite limited. Approaches using traditional single shooting include [18] and [32], both of which employ a variational integrator on SO(3) and use the method to solve the MPC sub-problem. For [18], this involves planning maneuver body torques, while [32] also plans the momentum exchange for a RW array by additionally including the momentum exchange dynamics in the variational integrator. Finally, [27] explores a multiple shooting approach to solve a unique formulation of the attitude maneuver planner using a RW array while optimizing the power balance equation of the RW motors and the onboard solar panels.

3.3 The PRONTO Approach

We now develop the unique trajectory optimization solution used to solve the challenging problems developed in this thesis: the PRojection-Operator based Newton method for Trajectory Optimization (PRONTO). As discussed in [1, 23], PRONTO is a *direct* method that implements Newton descent on elements of the trajectory manifold $\mathcal{T}$ (if the reader is unfamiliar traditional Newton descent methods or desires a brief review, we recommend they review the summary of these methods presented in Appendix A). Unlike other direct methods, the core PRONTO algorithm is designed to operate *directly* on the infinite dimensional space of the trajectory manifold with the continuous elements of the algorithm (including the trajectory iterate $\boldsymbol{\xi}_i$, regulator $K_{\boldsymbol{\xi}}$, and descent direction $\boldsymbol{\zeta}_k$) being computed using numerical integration. This allows the representation and precision of these results to be controlled by the selected integrator and so implicitly supports the mesh refinement techniques developed in other conventional direct methods. In application, PRONTO itself requires only the selection of the regulator $K_{\boldsymbol{\xi}}(t)$ (discussed below) as well as C^2

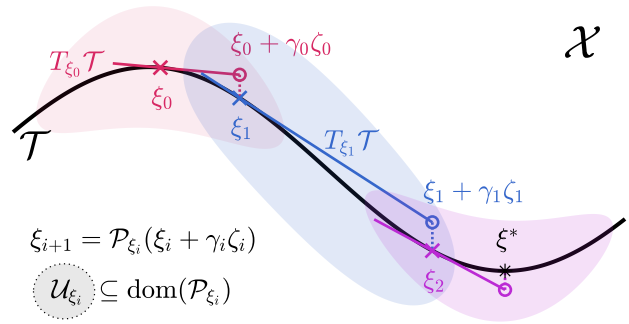

Figure 3.1: Outline of PRONTO algorithm showing the trajectory iterates $\boldsymbol{\xi}_i$, their tangent spaces $T_{\boldsymbol{\xi}_i}\mathcal{T}$, the computed steps $\gamma_i\boldsymbol{\zeta}_i$, and the application (and domain $\mathcal{U}_{\boldsymbol{\xi}_i}$) of the projection operator $\mathcal{P}_{\boldsymbol{\xi}_i}$ to generate $\boldsymbol{\xi}_{i+1}$.

differentiability of the dynamics f and cost functions ℓ, m [20, 22]. Naturally, convexity of the cost and the constraints is appreciated when available.

The centerpiece of the PRONTO algorithm is the nonlinear *Projection operator* $\mathcal{P}$: a specialized trajectory-tracking regulator which projects curve iterates in $\mathcal{X}$ onto $\mathcal{T}$ provided that they are sufficiently L_∞ close to $\mathcal{T}$. This operator allows PRONTO to take larger steps in descending the objective function than conventional direct methods without requiring extra iterations to remain near to $\mathcal{T}$. The second critical element is the geometric insight used to precisely (and efficiently) compute the descent direction $\boldsymbol{\zeta}_i$. Following the approach used in traditional Newton Descent methods, this descent direction is computed within the tangent space $T_{\boldsymbol{\xi}_i}\mathcal{T}$ of the current trajectory iterate $\boldsymbol{\xi}_i$ by minimizing a positive definite quadratic approximation of the objective function.

The central recursion of the PRONTO algorithm is summarized in steps (a-d) of the inner loop in algorithm 1. Following these steps along with the illustration in Figure 3.1, the algorithm first computes the additive descent step $\boldsymbol{\zeta}_i$ (scaled by an appropriate step-size γ_i), adds it to the current trajectory $\boldsymbol{\xi}_i$, then projects the resulting curves back onto the trajectory manifold $\mathcal{T}$. The remainder of algorithm 1 then adapts this approach to an Interior Point (IP) barrier functional method to incorporate the constraints and achieve feasibility at the minimizer. Once the inner loop has converged in the predicted descent step size (specified by the tolerance Δ_h^-), the outer loop then

Algorithm 1: Constrained PRONTO Method

```
Given Descent & Constraint tolerances Δ_h^-, ε_j^- > 0;
Initialize Trajectory ξ_0 ∈ 𝒯, and ε_j, δ_j = 1 ∀ j;
while any ε_j > ε_j^- do
    while h(ξ_{i+1}) − h(ξ_i) > Δ_h^- do
        if ξ_i satisfies all constraints then
            δ_j ← −1/2 · max_{t∈[0,T]} c_j(ξ_i(t));
        else
            δ_j ← δ_j/2;
        end
        a) Design Regulator K_{ξ_i}(·);
        b) Compute Descent Direction ζ_i;
        c) Compute step size γ_i;
        d) Project Update ξ_{i+1} = 𝒫(ξ_i + γ_i ζ_i);
    end
    For each ε_j > ε_j^-, reduce ε_j ← ε_j/10;
end
```

steepens the constraint barrier functions via the coefficients $\epsilon_j, \delta_j > 0$. This is repeated until the solution is sufficiently close to each constraint; a condition managed by the tolerance parameters $\epsilon_j^- > 0$. Each step a-d for the main PRONTO algorithm, as well as the IP constraint adaptation, is discussed in detail in the following sections.

3.4 The Projection Operator

The *projection operator* $\mathscr{P}$ is a generalization of a trajectory tracking regulator which projects a bounded, continuous curve $\boldsymbol{\eta} = (\boldsymbol{\alpha}(\cdot), \boldsymbol{\mu}(\cdot)) \in \mathscr{X}$ to a trajectory $\boldsymbol{\xi} = (\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot)) \in \mathscr{T}$ via the nonlinear feedback system

$$\begin{aligned} \dot{\boldsymbol{x}} &= f(\boldsymbol{x}, \boldsymbol{u}), \qquad \boldsymbol{x}(0) = \boldsymbol{x}_0, \\ \boldsymbol{u} &= \boldsymbol{\mu}(t) + K(t)\left[\boldsymbol{\alpha}(t) - \boldsymbol{x}(t)\right]. \end{aligned} \tag{3.3}$$

Under this simple definition, the choice of the regulator $K(\cdot)$ is not immediately obvious as the above construction will produce a projection operator for *any* $K(\cdot)$. That is, even $K \equiv 0$ satisfies $\boldsymbol{\xi} = \mathscr{P}(\boldsymbol{\xi})$ if and only if $\boldsymbol{\xi} \in \mathscr{T}$. In practice, the purpose of $K(\cdot)$ is best understood in the context of the infinite-horizon optimization problem ($T = \infty$) where it must provide local exponential stability in the error

$(\boldsymbol{\alpha} - \boldsymbol{x})$ for the closed loop system (3.3). This stabilizing property ensures that solutions of (3.3) remain finite and computationally tractable for systems with *unstable* dynamics (an extremely useful property which ultimately extends to the solver's iterates and descent directions as discussed below). More directly, the choice and weighting of the feedback gains in $K(\cdot)$ affects *which* $\boldsymbol{\xi}$ we obtain for a given $\boldsymbol{\eta}$, affecting the size and shape of $\mathrm{dom}(\mathscr{P})$ (i.e. the maximal effective step size of the algorithm).

However, while local exponential stability is also sufficient for the finite horizon case, it is unnecessarily strict. In practice, we can ensure that (3.3) remains computationally tractable on a finite horizon by selecting $K(\cdot)$ which provides a locally stabilizing *effect* around the trajectories of a given system. Since the family of trajectories is quite large even for simple nonlinear systems however, it is often simpler to select a *local* regulator $K_{\boldsymbol{\xi}'}(t)$ around a *specific* nearby trajectory $\boldsymbol{\xi}'$ (e.g. the previous algorithm iterate). In this case, $K_{\boldsymbol{\xi}'}(t)$ can be quickly designed around $\boldsymbol{\xi}'$ using classic time-varying linear-quadratic optimal control techniques: a process which guarantees the desired stability in $(\boldsymbol{\alpha} - \boldsymbol{x})$ for nearby $\boldsymbol{\eta} \in \mathscr{X}$. This regulator then defines a local projection operator $\mathscr{P}_{\boldsymbol{\xi}'}$, which includes in its domain an L_∞ neighborhood $\mathscr{U}_{\boldsymbol{\xi}'} \subset \mathrm{dom}(\mathscr{P}_{\boldsymbol{\xi}'}) \subset \mathscr{X}$ indicated by the colored regions in Figure 3.1. Provided that our step size remains within the (often quite large) set $\mathscr{U}_{\boldsymbol{\xi}'}$, the two optimization problems

$$\min_{\boldsymbol{\xi} \in \mathscr{T}} h(\boldsymbol{\xi}), \qquad \min_{\boldsymbol{\xi} \in \mathscr{U}_{\boldsymbol{\xi}'}} h(\mathscr{P}_{\boldsymbol{\xi}'}(\boldsymbol{\xi})), \tag{3.4}$$

are then equivalent in the following sense: if $\boldsymbol{\xi}^* \in \mathscr{T} \cap \mathscr{U}_{\boldsymbol{\xi}'}$ is a constrained local minimum of h, then it is an *unconstrained* local minimum of $g := h \circ \mathscr{P}_{\boldsymbol{\xi}'}$. Furthermore, if $\boldsymbol{\xi}^+ \in \mathscr{U}_{\boldsymbol{\xi}'}$ is an unconstrained local minimum of g, then $\boldsymbol{\xi}^* = \mathscr{P}_{\boldsymbol{\xi}'}(\boldsymbol{\xi}^+)$ is a constrained local minimum of $\mathscr{T}$ [20]. This observation motivates the core PRONTO loop structure shown in algorithm 1, where the computation of the update step $\boldsymbol{\zeta}_i$ for the iterate $\boldsymbol{\xi}_i$ can now be developed using an unconstrained approach. Moreover, provided that $\boldsymbol{\xi}_{i+1} = \mathscr{P}(\boldsymbol{\xi}_i + \gamma_i \boldsymbol{\zeta}_i)$ is projected back onto $\mathscr{T}$ at each iteration, this unconstrained method can also converge to a shared minimizer $\boldsymbol{\xi}^*$ satisfying the nonlinear dynamics.

3.5 Projection Regulator Design

Following the discussion above, we now consider the problem of designing a time-varying regulator around trajectories for the (potentially unstable) dynamics $\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u})$. For the purposes of this thesis, we refer to the well known instability observed in the angular rates of rigid-bodies along their intermediate principle inertia axis (corresponding to J_2 if $J_1 > J_2 > J_3$ are the diagonal elements of the inertia tensor J) [24]. To limit the effects of this instability in the iterations of our optimizer, we desire a time-varying regulator $K_{\boldsymbol{\xi}}(t)$ defined on an known trajectory ξ which attracts nearby curves by stabilizing the time-varying *linearized* dynamics

$$\dot{\boldsymbol{z}} = A_{\boldsymbol{\xi}}(t)\,\boldsymbol{z} + B_{\boldsymbol{\xi}}(t)\,\boldsymbol{v}, \quad t \in [0, T], \tag{3.5}$$

where $\boldsymbol{z} \in \mathbb{R}^n$, $\boldsymbol{v} \in \mathbb{R}^m$ represent linear perturbations in the state and control from the trajectory $\boldsymbol{\xi}$ around the linearized dynamics $A_{\boldsymbol{\xi}}(t) := {}^{\partial f(\boldsymbol{\xi}(t))}/_{\partial x}$ and $B_{\boldsymbol{\xi}}(t) := {}^{\partial f(\boldsymbol{\xi}(t))}/_{\partial u}$ and live in the ambient linear spaces of the state and control spaces X and U. In this case, an effective regulator can readily be obtained as the solution to the classic finite-horizon linear quadratic control problem

$$\min \quad \int_0^T \frac{1}{2}\|\boldsymbol{z}\|^2_{Q_{reg}} + \frac{1}{2}\|\boldsymbol{v}\|^2_{R_{reg}}\,dt \;+\; \frac{1}{2}\|\boldsymbol{z}(T)\|^2_{P_{reg}}. \tag{3.6}$$

where $\|\boldsymbol{v}\|^2_R$ is shorthand for the semi-norm $\boldsymbol{v}^T R\,\boldsymbol{v}$ (technically not a true norm unless R is positive definite) and the matrices $Q_{reg}, P_{reg} \in \mathbb{R}^{n\times n}$ and $R_{reg} \in \mathbb{R}^{m\times m}$ are suitable positive semi-definite and positive definite quadratic weights on the state and control inputs respectively (details for the selection of these matrices are developed below). This well-known problem admits the solution $K_{\boldsymbol{\xi}}(t) = R^{-1}_{reg}B(t)P(t)$ where $P(t)$ is obtained by integrating the Differential Riccati Equation (DRE)

$$\dot{P} = PB_{\boldsymbol{\xi}}(t)R^{-1}_{reg}B^{\top}_{\boldsymbol{\xi}}(t)P - PA_{\boldsymbol{\xi}}(t) - A^{\top}_{\boldsymbol{\xi}}(t)P - Q_{reg}, \tag{3.7}$$

backwards in time from the boundary condition $P(T) = P_{reg}$ [2]. Although other regulator designs are certainly possible (and indeed should be explored), this design approach for the projection regulator has been found to be effective for a variety of different nonlinear systems [1, 41].

While the state and control weight matrices Q_{reg} and R_{reg} can be chosen in an intuitive manner appropriate to the regulatory priorities of the specific problem and dynamics, the positive definite terminal cost matrix P_{reg} which serves as the boundary condition $P(T)$ for the DRE above is perhaps not so intuitive. Provided that this terminal cost is chosen non-negative definite (and, without loss of generality, symmetric), the choice of P_{reg} is relatively arbitrary [2], ultimately only affecting the capabilities of the resulting projection regulator $K_\xi(t)$. As described above, even the null projection regulator $K_\xi = 0$ resulting from $P_{reg} = 0$ will still possess the projection property (albeit with a substantially reduced basin of attraction around the base trajectory ξ). Below, we develop one particular strategy for determining the terminal cost and regulator which has proven particularly effective in the existing PRONTO literature [1, 20, 23].

3.6 Determination of the Terminal Cost on Constrained Manifolds

Following the above developments in Linear Quadratic control theory, a suitable terminal cost $P_{reg} = P(T)$ for the DRE (3.7) can be readily obtained from the solution for an exponentially stabilizing (infinite horizon) Linear Quadratic Regulator (LQR) designed at the target state $\boldsymbol{x}_d \in X$ using the same cost function (3.6) (i.e. solving for an equilibrium point of the DRE at the desired target state $\boldsymbol{x}_d$). Specifically, given symmetric positive definite matrices $Q_{reg} \in \mathbb{R}^{n\times n}$ and $R_{reg} \in \mathbb{R}^{m\times m}$, there will be a positive definite matrix $P_{reg} \in \mathbb{R}^{n\times n}$ satisfying the Algebraic Riccati Equation (ARE):

$$Q_{reg} = P_{reg} B_d R_r^{-1} B_d^\top P_{reg} - A_d^\top P_{reg} - P_{reg} A_d. \tag{3.8}$$

where, following the notation above, the matrices $A_d \coloneqq {}^{\partial f(\boldsymbol{x}_d,0)}/_{\partial x}$ and $B_d \coloneqq {}^{\partial f(\boldsymbol{x}_d,0)}/_{\partial u}$ specify the local linearized dynamics around the desired target (equilibrium) state $\boldsymbol{x}_d$.

Note that the above strategy requires both Q_{reg} and P_{reg} to be positive definite (unlike the DRE above which only requires them to be positive *semi*-definite). While this restriction is inconsequential in many applications, it poses a potential problem for systems with states that evolve on constrained submanifolds of their ambient linear space (e.g. quaternions $\boldsymbol{q}$ whose vector

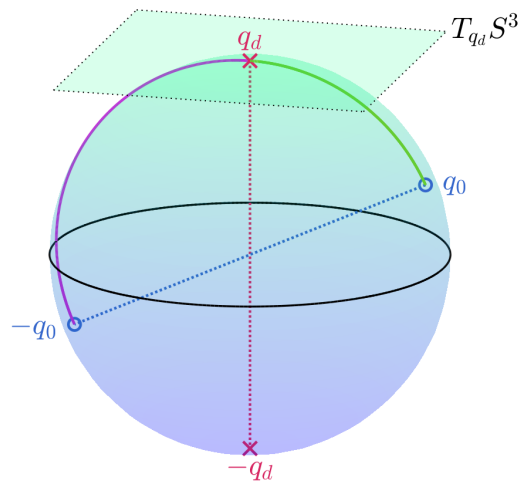

Figure 3.2: Illustration of quaternion attitude space S^3.

definition lives in $\mathbb{R}^4$ but are constrained to the sphere S^3). Due to the restricted configuration space of such systems, the linearized dynamics (A_d, B_d) of such systems *cannot* be locally linearly controllable in the ambient linear space. As such, no exponentially stabilizing regulator exists, and the ARE (3.8) has no solution.

Thankfully, this restriction can be circumvented in case where the tangent space of the state's constraint submanifold (i.e. the state's true configuration space) is known [12]. In this case, the regulator $K_\xi(T)$ and the terminal cost P_{reg} can instead be constructed on this (controllable) subspace then lifted into the ambient linear space. For example, recall the simplified body-torque dynamics developed in chapter 2:

$$\dot{\boldsymbol{q}} = {}^1\!/_2\, O_L(\boldsymbol{q})\, \widetilde{\boldsymbol{\omega}}\,, \tag{3.9a}$$

$$J\dot{\boldsymbol{\omega}} = -\widehat{\boldsymbol{\omega}}\, J\boldsymbol{\omega} + \boldsymbol{u}\,, \tag{3.9b}$$

with state $\boldsymbol{x} = [\boldsymbol{q}, \boldsymbol{\omega}] \in \mathrm{S}^3 \times \mathbb{R}^3$ and control input $\boldsymbol{u} \in \mathbb{R}^3$. As discussed in chapter 2, quaternions are naturally constrained the unit sphere S^3 embedded in $\mathbb{R}^4$. As such, the quaternion dynamics in (3.9) above can alternatively be written in the form

$$\begin{aligned} \dot{\boldsymbol{q}} &= {}^1\!/_2\, O_R(\widetilde{\boldsymbol{\omega}})\, \boldsymbol{q}\,, \\ &= {}^1\!/_2\, Z(\boldsymbol{q})\, \boldsymbol{\omega}\,, \end{aligned} \tag{3.10}$$

where the matrix $Z(\boldsymbol{q})$ collects the right three (orthogonal) columns of $O_L(\boldsymbol{q})$ so that $O_L(\boldsymbol{q}) = [\boldsymbol{q},\, Z(\boldsymbol{q})]$ and $\boldsymbol{q}^\top Z(\boldsymbol{q}) = \boldsymbol{0}$. Letting $(\boldsymbol{p}, \boldsymbol{w}, \boldsymbol{v})$ represent the linear perturbations about the target final state and control $(\boldsymbol{q}_d, \boldsymbol{\omega}_d, \boldsymbol{u}_d)$ (with $\boldsymbol{\omega}_d, \boldsymbol{u}_d = \boldsymbol{0}_{3\times 1}$ to ensure a resting equilibrium), the linearized dynamics (3.9) around $\boldsymbol{x}_d$ can then be written as follows:

$$\begin{aligned} \dot{\boldsymbol{p}} &= \frac{1}{2} Z(\boldsymbol{q}_d)\, \boldsymbol{w}\,, \\ \dot{\boldsymbol{w}} &= J^{-1} \boldsymbol{v}\,. \end{aligned} \tag{3.11}$$

As alluded above, because the attitude $\boldsymbol{q}(t)$ evolves on the sphere S^3, the system matrix pair (A_d, B_d) for the LTI system (3.11) is *not* linearly controllable in the ambient space $\mathbb{R}^4 \times \mathbb{R}^3$. Instead, the columns of $Z(\boldsymbol{q}_d)$ form a basis for the tangent space $T_{\boldsymbol{q}_d}\mathrm{S}^3$ (orthogonal to $\boldsymbol{q}_d$ as shown in Figure 3.2), making the reachable subspace for (3.11) $T_{\boldsymbol{q}_d}\mathrm{S}^3 \times \mathbb{R}^3$. However, noting that $Z(\boldsymbol{q}_d)^\top Z(\boldsymbol{q}_d) = \mathbb{I}_3$, we can project (3.11) to this controllable subspace by defining the projected coordinate $\boldsymbol{s} \in \mathbb{R}^3$ by $\boldsymbol{p} = Z(\boldsymbol{q}_d)\, \boldsymbol{s}$, so that

$$\begin{aligned} \dot{\boldsymbol{s}} &= \frac{1}{2}\, Z(\boldsymbol{q}_d)^\top Z(\boldsymbol{q}_d)\, \boldsymbol{w} = \frac{1}{2} \boldsymbol{w}\,, \\ \dot{\boldsymbol{w}} &= J^{-1} \boldsymbol{v}\,. \end{aligned} \tag{3.12}$$

Unlike (3.11), the projected dynamics are a three dimensional double integrator system that is clearly controllable (independent of $\boldsymbol{q}_d$) and therefore stabilizable using a standard linear quadratic regulator.

Summarizing and generalizing the above example, let us return to the general dynamics $\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u})$ whose state evolves a submanifold X of the ambient linear space $\mathbb{R}^n$. Around the target state $\boldsymbol{x}_d$, we can locally parameterize this submanifold by obtaining an orthonormal basis spanning the tangent space $T_{\boldsymbol{x}_d}X$. For example, a basis for the above example is given by the columns of the matrix $Z(\boldsymbol{q}_d)$. Using this basis, the local linearized dynamics around the target state

$$\dot{\boldsymbol{z}} = A_d\, \boldsymbol{z} + B_d\, \boldsymbol{v},$$

can then be projected onto the tangent space $T_{\boldsymbol{x}_d}X$ using the projected coordinate $\boldsymbol{r} \coloneqq M(\boldsymbol{x}_d)\boldsymbol{z} \in$

$\mathbb{R}^k$, yielding the reduced form

$$\dot{\boldsymbol{r}} = \underbrace{\left[M(\boldsymbol{x}_d)A_d M(\boldsymbol{x}_d)^\top\right]}_{A_r:=} \boldsymbol{r} + \underbrace{[M(\boldsymbol{x}_d)B_d]}_{B_r:=} \boldsymbol{v}, \tag{3.13}$$

where the rows of the projection matrix $M(\boldsymbol{x}_d) \in \mathbb{R}^{k\times n}$ are the orthonormal basis of $T_{\boldsymbol{x}_d}X$ determined above. For the above example, this projection from $\mathbb{R}^4 \times \mathbb{R}^3$ to $\mathbb{R}^3 \times \mathbb{R}^3$ is given by

$$M(\boldsymbol{q}_d) = \begin{bmatrix} Z(\boldsymbol{q}_d)^\top & 0 \\ 0 & \mathbb{I}_3 \end{bmatrix}, \tag{3.14}$$

In this projected subspace, the pair (A_r, B_r) should now ideally be controllable (though if not, the same technique above may be repeated, projecting onto the subspace which is locally linearly controllable at the target). Thus, an exponentially stabilizing LQR can now be designed directly on this controllable subspace by solving the (projected) ARE

$$Q_r = P_r B_r R_{reg}^{-1} B_r^\top P_r - A_r^\top P_r - P_r A_r, \tag{3.15}$$

where Q_r is a symmetric positive definite cost weight matrices defined on the projected subspace, which can be easily obtained by projecting the original Q_{reg} (or indeed any positive definite weight matrix) to this subspace following

$$Q_r := M(\boldsymbol{x}_d) Q_{reg} M(\boldsymbol{x}_d)^\top.$$

Having obtained the Riccati solution P_r and the corresponding regulator $K_r = R^{-1}B_r^\top P_r$ on the reduced subspace, we can now *lift* these results (as well as the projected state cost Q_r) into the ambient space as follows

$$\begin{aligned} Q_{reg}(\boldsymbol{x}_d) &= M(\boldsymbol{x}_d)^\top Q_r M(\boldsymbol{x}_d), \\ P_{reg}(\boldsymbol{x}_d) &= M(\boldsymbol{x}_d)^\top P_r M(\boldsymbol{x}_d), \\ K_\xi(T) &= K_r M(\boldsymbol{x}_d), \end{aligned} \tag{3.16}$$

where, by design, both $Q_{reg}(\boldsymbol{x}_d)$ and $P_{reg}(\boldsymbol{x}_d)$ are now appropriately positive definite on the tangent space $T_{\boldsymbol{x}_d}X$ and zero otherwise (i.e. their kernels are the orthogonal complement of $T_{\boldsymbol{x}_d}X$). As such, the lifted terminal cost $P_{reg}(\boldsymbol{x}_d)$ now serves as a suitable boundary condition $P(T)$ for solving the DRE (3.7) to obtain the desired projection regulator K_ξ.

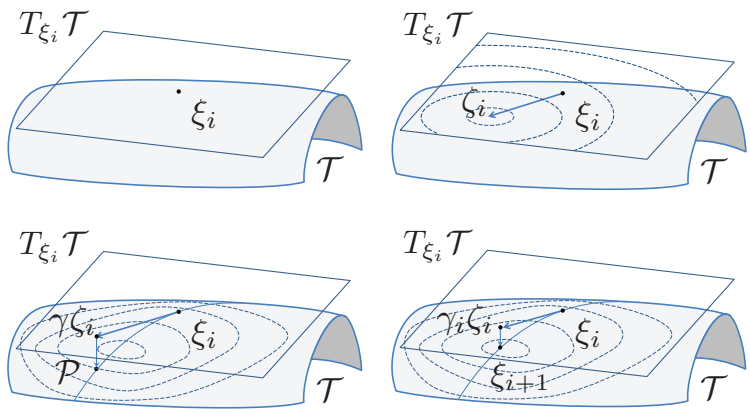

Figure 3.3: Illustration of PRONTO descent search process by Alessandro Saccon (reprinted from [1]), where the descent direction $\boldsymbol{\zeta}_i$ is determined by minimizing (3.17) over $T_{\boldsymbol{\xi}_i}\mathcal{T}$, while the step *size* γ_i is computed via line search on $\mathcal{T}$.

3.7 Computing the PRONTO Descent Direction

Next, we need to determine an appropriate descent direction by exploring a neighborhood of the trajectory iterate $\boldsymbol{\xi} \in \mathcal{T}$ as illustrated in Figure 3.3. To parametrize this space, let the curve $\boldsymbol{\eta} \in \mathcal{X} \cap \mathrm{dom}(\mathscr{P}_{\boldsymbol{\xi}})$ and let the trajectory $\boldsymbol{\xi}' \in \mathcal{T}$ be the projection $\boldsymbol{\xi}' = \mathscr{P}_{\boldsymbol{\xi}}(\boldsymbol{\eta})$ of this curve. Given $\boldsymbol{\zeta} \in T_{\boldsymbol{\xi}}\mathcal{T}$, we summarize the following Fréchet derivatives of the projection operator

$$
\begin{aligned}
\boldsymbol{\xi}' = (\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot)) \quad &= \mathscr{P}_{\boldsymbol{\xi}}(\boldsymbol{\eta}), && \boldsymbol{\eta} = (\boldsymbol{\alpha}(\cdot), \boldsymbol{\mu}(\cdot)),\\
\boldsymbol{\sigma} = (\boldsymbol{z}(\cdot), \boldsymbol{v}(\cdot)) \quad &= D\mathscr{P}_{\boldsymbol{\xi}}(\boldsymbol{\eta}) \cdot \boldsymbol{\zeta}, && \boldsymbol{\zeta} = (\boldsymbol{\beta}(\cdot), \boldsymbol{\nu}(\cdot)),\\
\boldsymbol{\theta} = (\boldsymbol{y}(\cdot), \boldsymbol{\upsilon}(\cdot)) \quad &= D^2\mathscr{P}_{\boldsymbol{\xi}}(\boldsymbol{\eta}) \cdot (\boldsymbol{\zeta}, \boldsymbol{\zeta}),
\end{aligned}
$$

which are computed in practice by solving the following ODE's

$$
\begin{aligned}
\boldsymbol{\xi}' :\quad & \dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}), && \boldsymbol{x}(0) = \boldsymbol{x}_0,\\
& \boldsymbol{u} = \boldsymbol{\mu}(t) + K_{\boldsymbol{\xi}}(t)[\boldsymbol{\alpha}(t) - \boldsymbol{x}(t)],\\
\boldsymbol{\sigma} :\quad & \dot{\boldsymbol{z}} = A_{\boldsymbol{\xi}'}(t)\boldsymbol{z} + B_{\boldsymbol{\xi}'}(t)\boldsymbol{v}, && \boldsymbol{z}(0) = \boldsymbol{0},\\
& \boldsymbol{v} = \boldsymbol{\nu}(t) + K_{\boldsymbol{\xi}}(t)[\boldsymbol{\beta}(t) - \boldsymbol{z}],\\
\boldsymbol{\theta} :\quad & \dot{\boldsymbol{y}} = A_{\boldsymbol{\xi}'}(t)\boldsymbol{y} + B_{\boldsymbol{\xi}'}(t)\boldsymbol{\upsilon} + D^2 f(\boldsymbol{\xi}'(t)) \cdot (\boldsymbol{\sigma}(t), \boldsymbol{\sigma}(t)),\\
& \boldsymbol{\upsilon} = -K_{\boldsymbol{\xi}}(t)\boldsymbol{y}, && \boldsymbol{y}(0) = \boldsymbol{0}.
\end{aligned}
$$

Note that the stabilizing effects of the projection regulator $K_{\boldsymbol{\xi}}$ are also imparted to the above derivatives, thus allowing these solutions to be computed numerically in the cases where f is unstable. Using these derivatives and following the mentality used to derive the Newton descent method, we note that trajectories in a neighborhood of $\boldsymbol{\xi}$ can be examined using the Taylor series expansion

$$\mathscr{P}_{\boldsymbol{\xi}}(\boldsymbol{\xi}+\boldsymbol{\zeta}) = \mathscr{P}_{\boldsymbol{\xi}}(\boldsymbol{\xi}) + D\mathscr{P}_{\boldsymbol{\xi}}(\boldsymbol{\xi}) \cdot \boldsymbol{\zeta} + \frac{1}{2} D^2 \mathscr{P}_{\boldsymbol{\xi}}(\boldsymbol{\xi}) \cdot (\boldsymbol{\zeta}, \boldsymbol{\zeta}) \cdots$$

When examining the above expansion, note that we need only consider descent directions $\boldsymbol{\zeta}$ in the tangent space of the trajectory manifold $\boldsymbol{\zeta} \in T_{\boldsymbol{\xi}}\mathscr{T}$ rather than all arbitrary curves $\boldsymbol{\eta} \in \mathscr{X}$. This reduction of the search space originates from [22], where it is shown that $\mathscr{T}$ is a Banach manifold. That is, a neighborhood of $\boldsymbol{\xi}$ is homeomorphic to the tangent space $T_{\boldsymbol{\xi}}\mathscr{T}$ so that every nearby trajectory $\boldsymbol{\xi}'$ to $\boldsymbol{\xi}$ can be *uniquely* represented by some $\boldsymbol{\zeta} \in T_{\boldsymbol{\xi}}\mathscr{T}$ via

$$\boldsymbol{\xi}' = \mathscr{P}_{\boldsymbol{\xi}}(\boldsymbol{\xi}+\boldsymbol{\zeta}),$$

Under this greatly simplified search space, a Newton descent direction $\boldsymbol{\zeta}^*$ can now be determined as the solution to

$$\min_{\boldsymbol{\zeta} \in T_{\boldsymbol{\xi}}\mathscr{T}} \; Dg(\boldsymbol{\xi}) \cdot \boldsymbol{\zeta} + \frac{1}{2} D^2 g(\boldsymbol{\xi}) \cdot (\boldsymbol{\zeta}, \boldsymbol{\zeta}), \tag{3.17}$$

where the composite cost functional $g \coloneqq h \circ \mathscr{P}_{\boldsymbol{\xi}}$ computes the cost of the step $\boldsymbol{\xi} + \boldsymbol{\zeta}$ when projected onto $\mathscr{T}$. To tackle this infinite dimensional optimization problem, we can now rewrite (3.17) to leverage techniques from linear quadratic optimal control. Noting that we have $\boldsymbol{\xi} \in \mathscr{T}$ and $\boldsymbol{\zeta} \in T_{\boldsymbol{\xi}}\mathscr{T}$ from the above results, the above unconstrained derivatives of the projected cost g can be simplified

to the following

$$\begin{aligned}
Dg(\boldsymbol{\xi}) \cdot \boldsymbol{\zeta} &= Dh(\mathscr{P}_{\boldsymbol{\xi}}(\boldsymbol{\xi})) \cdot D\mathscr{P}_{\boldsymbol{\xi}}(\boldsymbol{\xi}) \cdot \boldsymbol{\zeta}, \\
&= Dh(\boldsymbol{\xi}) \cdot \boldsymbol{\zeta}, \\
D^2 g(\boldsymbol{\xi}) \cdot (\boldsymbol{\zeta}, \boldsymbol{\zeta}) &= \quad D^2 h(\mathscr{P}_{\boldsymbol{\xi}}(\boldsymbol{\xi})) \cdot (D\mathscr{P}_{\boldsymbol{\xi}}(\boldsymbol{\xi}) \cdot \boldsymbol{\zeta}, D\mathscr{P}_{\boldsymbol{\xi}}(\boldsymbol{\xi}) \cdot \boldsymbol{\zeta}) \\
&\quad + Dh(\mathscr{P}_{\boldsymbol{\xi}}(\boldsymbol{\xi})) \cdot D^2 \mathscr{P}_{\boldsymbol{\xi}}(\boldsymbol{\xi}) \cdot (\boldsymbol{\zeta}, \boldsymbol{\zeta}), \\
&= \quad D^2 h(\boldsymbol{\xi}) \cdot (\boldsymbol{\zeta}, \boldsymbol{\zeta}) \\
&\quad + Dh(\boldsymbol{\xi}) \cdot D^2 \mathscr{P}_{\boldsymbol{\xi}}(\boldsymbol{\xi}) \cdot (\boldsymbol{\zeta}, \boldsymbol{\zeta}).
\end{aligned}$$

In practice, the final term $Dh(\boldsymbol{\xi}) \cdot D^2 \mathscr{P}(\boldsymbol{\xi}) \cdot (\boldsymbol{\zeta}, \boldsymbol{\zeta})$ can be computed by first rewriting in the form

$$\int_0^T \boldsymbol{\lambda}(t)^\top D^2 f(\boldsymbol{\xi}(t)) \cdot (\boldsymbol{\zeta}(t), \boldsymbol{\zeta}(t))\, dt,$$

where the adjoint vector $\boldsymbol{\lambda}(t)$, much like a Lagrange multiplier, evolves backwards in time from $\boldsymbol{\lambda}(T) = m_{\boldsymbol{x}}^\top(\boldsymbol{x}(T))$ according to

$$\dot{\boldsymbol{\lambda}} = -[A_{\boldsymbol{\xi}}(t) - B_{\boldsymbol{\xi}}(t) K_{\boldsymbol{\xi}}(t)]^\top \boldsymbol{\lambda} - \ell_{\boldsymbol{x}}^\top(t) + K_{\boldsymbol{\xi}}^\top(t)\, \ell_{\boldsymbol{u}}^\top(t)\,,$$

where we now adopt the common notation $f_x \coloneqq {}^{\partial f}/_{\partial x}$ used for partial derivatives. Notably, the above differential equation is *also* stabilized by the regulator $K_{\boldsymbol{\xi}}$. Under these simplifications, the quadratic term $D^2 g(\boldsymbol{\xi}) \cdot (\boldsymbol{\zeta}, \boldsymbol{\zeta})$ can be rewritten in the following quadratic form

$$\int_0^T \begin{bmatrix} \boldsymbol{z}(t) \\ \boldsymbol{v}(t) \end{bmatrix}^\top \underbrace{\begin{bmatrix} Q_{\boldsymbol{\xi}}(t) & S_{\boldsymbol{\xi}}(t) \\ S_{\boldsymbol{\xi}}(t)^\top & R_{\boldsymbol{\xi}}(t) \end{bmatrix}}_{W_{\boldsymbol{\xi}}(t)} \underbrace{\begin{bmatrix} \boldsymbol{z}(t) \\ \boldsymbol{v}(t) \end{bmatrix}}_{\boldsymbol{\zeta}(t)} dt + \boldsymbol{z}(T)^\top P_{\boldsymbol{\xi}} \boldsymbol{z}(T),$$

where $P_{\boldsymbol{\xi}} = m_{\boldsymbol{xx}}(\boldsymbol{x}(T))$ and the matrix $W_{\boldsymbol{\xi}}(t)$ has entries given by

$$w_{ij}(t) = \frac{\partial^2 \ell}{\partial \boldsymbol{\xi}_i \partial \boldsymbol{\xi}_j}(t, \boldsymbol{\xi}(t)) + \sum_{k=1}^n \boldsymbol{\lambda}_k(t) \frac{\partial^2 f_k}{\partial \boldsymbol{\xi}_i \partial \boldsymbol{\xi}_j}(\boldsymbol{\xi}(t)), \tag{3.18}$$

Finally, letting $a_{\boldsymbol{\xi}}(t) = \ell_{\boldsymbol{x}}(\boldsymbol{\xi}(t))$, $b_{\boldsymbol{\xi}}(t) = \ell_{\boldsymbol{u}}(\boldsymbol{\xi}(t))$, and $r_{\boldsymbol{\xi}} = m_{\boldsymbol{x}}^\top(\boldsymbol{x}(t))$, we can now rewrite the minimization problem (3.17) as a conventional constrained linear quadratic optimal control problem

$$\begin{aligned} \min \int_0^T \begin{bmatrix} \boldsymbol{a}_{\boldsymbol{\xi}}(t) \\ \boldsymbol{b}_{\boldsymbol{\xi}}(t) \end{bmatrix}^\top \boldsymbol{\zeta}(t) + \frac{1}{2}\boldsymbol{\zeta}(t)^\top W_{\boldsymbol{\xi}}(t)\boldsymbol{\zeta}(t)\, dt \\ + \quad \boldsymbol{r}_{\boldsymbol{\xi}}^\top \boldsymbol{z}(T) + \frac{1}{2}\boldsymbol{z}(T)^\top P_{\boldsymbol{\xi}}\boldsymbol{z}(T), \\ \text{s.t.} \quad \dot{\boldsymbol{z}} = A_{\boldsymbol{\xi}}(t)\boldsymbol{z} + B_{\boldsymbol{\xi}}(t)\boldsymbol{v}, \quad \boldsymbol{z}(0) = \boldsymbol{0}, \end{aligned} \tag{3.19}$$

The optimal descent direction $\boldsymbol{\zeta}^*$ can then be obtained by solving the above using the DRE. Following established methods, this is done by first solving the following ODE's backwards in time for the adjoint states, optimal regulator K_o, and optimal control input v_o:

$$\begin{aligned} -\dot{P} &= A_{\boldsymbol{\xi}}^\top P + P^\top A_{\boldsymbol{\xi}} - K_o^\top R_{\boldsymbol{\xi}} K_o + Q_{\boldsymbol{\xi}}, \quad P(T) = P(\boldsymbol{q}_d), \\ -\dot{\boldsymbol{r}} &= (A_{\boldsymbol{\xi}} - B_{\boldsymbol{\xi}} K_o)^\top \boldsymbol{r} + (\boldsymbol{a}_{\boldsymbol{\xi}} - K_o^\top \boldsymbol{b}_{\boldsymbol{\xi}}), \quad \boldsymbol{r}(T) = \boldsymbol{r}_{\boldsymbol{\xi}}, \\ -\dot{\boldsymbol{\lambda}} &= (A_{\boldsymbol{\xi}} - B_{\boldsymbol{\xi}} K_{\boldsymbol{\xi}})^\top \boldsymbol{\lambda} + (\boldsymbol{a}_{\boldsymbol{\xi}} - K_{\boldsymbol{\xi}}^\top \boldsymbol{b}_{\boldsymbol{\xi}}), \quad \boldsymbol{\lambda}(T) = \boldsymbol{r}_{\boldsymbol{\xi}}, \\ K_o &= \quad R_{\boldsymbol{\xi}}^{-1}(S_{\boldsymbol{\xi}}^\top + B_{\boldsymbol{\xi}}^\top P), \\ v_o &= -R_{\boldsymbol{\xi}}^{-1}(B_{\boldsymbol{\xi}}^\top \boldsymbol{r} + \boldsymbol{b}_{\boldsymbol{\xi}}). \end{aligned} \tag{3.20}$$

Following intuitively from the restrictions of the classical Newton Method, the Ricatti solution $P(\cdot)$ above will diverge *before* reaching $t = 0$ if the quadratic functional $D^2 g(\boldsymbol{\xi})$ is not strongly positive definite on $T_{\boldsymbol{\xi}}\mathcal{T}$ (indicating that no unique $\boldsymbol{\zeta}^*$ solving (3.17) exists) [21]. Since the incremental cost ℓ and terminal cost m are both positive definite by design, this can only occur when $W_{\boldsymbol{\xi}}$ is perturbed in a negative direction by the product of the costate $\boldsymbol{\lambda}$ and the 2nd-order derivatives of the dynamics f in (3.18) (or by the 2nd-order constraint derivatives in the augmented cost $\bar{\ell}$ developed below). In either case, we can temporarily shift to an infinite dimensional analog of the Quasi-Newton method by *approximating* $D^2 g(\boldsymbol{\xi})$ with the quadratic cost $D^2 h(\boldsymbol{\xi})$ obtained by omitting these additional terms. Specifically, we approximate the entries of $W_{\boldsymbol{\xi}}(t)$ given in (3.18) by

$$\bar{w}_{ij}(t) = \frac{\partial^2 \ell}{\partial \boldsymbol{\xi}_i \partial \boldsymbol{\xi}_j}(t, \boldsymbol{\xi}(t)).$$

This approximation is often required when far away from a minimizer and (typically) restricts the solver to linear (or superlinear) descent rate (i.e. accelerated gradient descent). However, because $D^2 g$ is necessarily positive definite in a neighborhood of a feasible minimizer, the algorithm must eventually achieve a quadratic descent rate. Thankfully, potentially divergent behavior in the costate solution $P(t)$ is simple to detect numerically and can thus be used as an reliable switching condition in the algorithm for applying this approximation.

Having now obtained a bounded solution to (3.20), the descent direction $\boldsymbol{\zeta}^*$ is finally obtained by solving

$$\dot{\boldsymbol{z}} = A_{\boldsymbol{\xi}}(t)\,\boldsymbol{z} + B_{\boldsymbol{\xi}}(t)\,\boldsymbol{v}, \quad \boldsymbol{z}(0) = \boldsymbol{0},$$

$$\boldsymbol{v} = -K_o(t)\,\boldsymbol{z} + \boldsymbol{v}_o,$$

forward in time. With a descent direction now determined, standard discrete line-search techniques (e.g. Armijo backstepping) are then used to select an appropriate step size $\gamma \in \mathbb{R}_{>0}$ by evaluating the cost $g(\boldsymbol{\xi} + \gamma\boldsymbol{\zeta}^*)$ and feasibility of the projected curve on $\mathcal{T}$ as shown in Figure 3.3. This step guarantees sufficient descent of the objective, ensures a quadratic convergence rate near the optimizer, and preserves feasibility once a feasible trajectory has been obtained.

3.8 Inclusion of Constraints

To incorporate general nonlinear constraints of the form $c(\boldsymbol{x}, \boldsymbol{u}) \leq \boldsymbol{0}$ into the PRONTO solver, we employ a modified Interior-Point (IP) barrier functional approach which is commonly effective in convex optimization problems [23]. Specifically, we modify the original problem (3.2) to the following form

$$\min_{\boldsymbol{\xi}\in\mathcal{T}} \int_0^T \overbrace{\ell(\boldsymbol{x}(t), \boldsymbol{u}(t)) + \sum_j \epsilon_j \beta_{\delta_j}(-c_j(\boldsymbol{x}(t), \boldsymbol{u}(t)))}^{\bar{\ell}(\boldsymbol{x}(t),\boldsymbol{u}(t))}\, dt + m(\boldsymbol{x}(t)), \tag{3.21}$$

where the augmented incremental cost $\bar{\ell}$ now includes the barrier functions $\beta_\delta : \mathbb{R} \to \mathbb{R}$ which add a rapidly increasing cost penalty for approaching or violating each constraint. The weights $\epsilon_j > 0$ are used to control the individual scaling of these barrier functions and are progressively lowered in later stages of the algorithm to allow the solution iterates progressively closer to the constraint

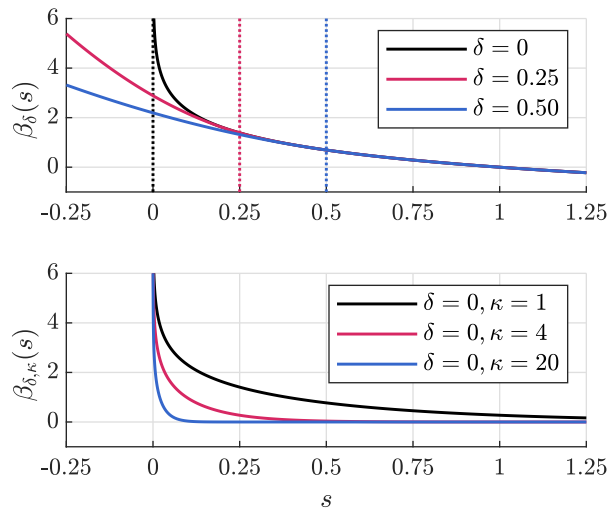

Figure 3.4: Illustrations of (a) the barrier function $\beta_\delta(s)$ and (b) the composition $\beta_{\delta,\kappa}(s)$ with the input-scaling hockeystick function $\sigma_\kappa(s)$.

boundaries. The augmented cost $\bar{\ell}$ of this (soft) constraint implementation then approaches the original $\ell(\boldsymbol{x}, \boldsymbol{u})$ as $\epsilon_j \to 0^+$.

The selected barrier functions are C^2 modifications of the classic log() function given by

$$\beta_\delta(s) = \begin{cases} -\log(s), & s > \delta, \\ \frac{1}{2}\left[\left(\frac{s-2\delta}{\delta}\right)^2 - 1\right] - \log(\delta), & s \leq \delta. \end{cases} \tag{3.22}$$

As shown in Figure 3.4, the parameters $\delta_j \in (0, 1]$ control the shape of this barrier function by specifying where it changes from a classic log() function to a C^2 extension that is finite on the entire real line. While it might initially appear extraneous, this extension is necessary for PRONTO to operate on constrained problems, as it ensures that infeasible trajectories possess a *finite* cost and a well-defined descent direction. Once a feasible trajectory is obtained, the shaping parameter δ_j can then be lowered, effectively modeling the constraint barrier as a true log() barrier function on the interior of the feasible set. This modification easily blends with the original IP approach as, for fixed $\epsilon_j > 0$, the problems (3.2) and (3.21) are both locally convex and share the *same* solution $\boldsymbol{\xi}^*_\epsilon = (\boldsymbol{x}^*_\epsilon(t), \boldsymbol{u}^*_\epsilon(t))$ if

$$\delta_j < \min_{t \in [0,T]} -c_j(\boldsymbol{x}^*_\epsilon(t), \boldsymbol{u}^*_\epsilon(t))$$

is satisfied for each j [23]. As shown in algorithm 1, this requirement is satisfied over two stages. First, we allow the barrier functions to push the solution iterates into the feasible set: a process that can be accelerated by decreasing δ_j at each iteration. Once feasibility is obtained, δ_j can be actively set below this threshold at each iteration.

However, while the default log-like barrier construction in (3.21) is sufficient to ensure optimizer feasibility, we can further improve its performance by ensuring that the constraints $c_j(\boldsymbol{x}, \boldsymbol{u})$ are well-conditioned. By default, when a constraint is able to satisfy $c_j(\boldsymbol{x}, \boldsymbol{u}) < -1$, the default cost penalty $\beta_\delta(-c_j(\boldsymbol{x}, \boldsymbol{u}))$ can erroneously become a negative cost *incentive* as shown in Figure 3.4. Specifically, this incentive can cause issues with non-convex obstacle avoidance or exclusion cone constraints like (2.11c), as it promotes behaviors which over-avoid the exclusion region. For convex constraints like (2.11a) and (2.11b), this issue is easily addressed by scaling the constraints (e.g. dividing (2.11a) and (2.11b) by $u_j{+}^2$ and ω_+^2 respectively) to ensure that this incentive region cannot be reached.

However, this solution is insufficient for non-convex exclusion type constraints (such as (2.11c)) as the surrounding space cannot (generally) be bounded, allowing the exclusion barrier functional to erroneously affect the entire parent space rather than the immediate neighborhood of the constraint boundary. To minimize the effect of this perturbation (and ensure that the constraint penalty is always positive), we can reshape the original barrier function (3.22) using the 'hockeystick' function

$$\sigma_\kappa(s) = \begin{cases} \tanh(\kappa s), & s \geq 0, \\ \kappa s \ , & s < 0, \end{cases}$$

via the composition

$$\beta_{\delta,\kappa}(s) \coloneqq \beta_\delta(\sigma_\kappa(s)).$$

As shown in Figure 3.4, the asymptote in the tanh() function causes $\beta_{\delta,\kappa}(s)$ to quickly flatten to zero on the interior of the feasible set at a rate determined by the coefficient $\kappa > 0$. For example, a value of $\kappa = 125$ is sufficient to ensure perturbations $< 0.2°$ by the exclusion cone constraint (2.11c) for camera trajectories ending at least $5°$ from the edge of the exclusion cone.

Chapter 4

Maneuver Planning via Tracked Command Torques

In this chapter, we will employ the Trajectory Optimization solvers PRONTO and GPOPS II developed in chapter 3 to solve a family of standard rest-to-rest constrained attitude maneuvers for simple body-torque (thruster) driven spacecraft subject to input saturation, angular velocity, and non-convex exclusion cone constraints. In addition to providing a demonstrative example of PRONTO's capabilities for problems with dynamics evolving on constrained nonlinear manifolds, this broad family of sample problems will also highlight its comparative computational efficiency with a state-of-the-art commercial solver. These results will present a strongly recommendation for PRONTO as a suitable real-time optimal maneuver planner for the more complex (momentum exchange) spacecraft attitude control problems developed in chapter 6. The following results reiterate the results presented in [12].

4.1 System Dynamics and Constraints

In this chapter, we wish to determine an optimal rest-to-rest attitude transfer from the initial attitude $\boldsymbol{q}_0 \in \mathrm{S}^3$ to the target attitude $\boldsymbol{q}_d \in \mathrm{S}^3$ with $\boldsymbol{\omega}_0 = \boldsymbol{\omega}_d = 0$) using generic body torques. That is, we wish to plan the optimal rotation from $\boldsymbol{x}_0 = (\boldsymbol{q_0}, 0)$ to $\boldsymbol{x}_d = (\boldsymbol{q_d}, 0)$ under the standard body-torque dynamics

$$\dot{\boldsymbol{q}} = {}^1\!/_2\, O_L(\boldsymbol{q})\, \widetilde{\boldsymbol{\omega}}\,, \tag{4.1a}$$

$$J\dot{\boldsymbol{\omega}} = -\widehat{\boldsymbol{\omega}}\, J\boldsymbol{\omega} + \boldsymbol{u}\,. \tag{4.1b}$$

where, for simplicity, the generalized control torques $\boldsymbol{u} \in \mathbb{R}^3$ assumes that all external disturbance body torques $\boldsymbol{\tau}_e(t) \equiv 0$.

Notably, the restriction to rest-to-rest transfers in our problem formulation is somewhat restrictive when considering the full maneuver space considered in practical spacecraft operations (e.g. de-tumbling maneuvers and sensor tracking problems frequently require $\omega_0, \omega_d \neq 0$). However, it should be noted that this restricted scope was intentionally chosen to simplify problem generation, analysis, and presentation rather than enforced by restrictions in our approach or solver. Indeed, arbitrary ω_0, ω_d and arbitrary state tracking problems are certainly possible using the PRONTO solver (though dynamic equilibria certainly simplify the design of the PRONTO projection regulator). For the purposes of this thesis however, the family of rest-to-rest maneuvers is sufficiently rich to capture the primary performance improvements and unique control behaviors for the majority of standard ADCS operations.

In addition to the above dynamics and boundary conditions, the desired maneuver must be feasible under the following standard operational safety constraints on maximum control torque (input saturation), maximum angular slew rate, and optical sensor exclusion cones

$$u_j^2 - u_+^2 \leq 0, \quad j \in 1, 2, 3, \tag{4.2a}$$

$$\omega_j^2 - \omega_+^2 \leq 0, \quad j \in 1, 2, 3, \tag{4.2b}$$

$$(\boldsymbol{\psi}_s^{\,i})^\top C(\boldsymbol{q})\, \boldsymbol{\psi}_c^{\,b} - \cos(\phi_{ko}) \leq 0, \tag{4.2c}$$

where, to review, $u_+, \omega_+ \in \mathbb{R}_{>0}$ are the maximum control thrusts and angular rates, respectively, $\boldsymbol{\psi}_c^{\,b} \in \mathrm{S}^2$ is the body-fixed frame orientation of the onboard sensor, and $\boldsymbol{\psi}_s^{\,i} \in \mathrm{S}^2$ and $\phi_{ko} \in [0, \pi]$ are the inertial orientation and minimum avoidance angle for the light source. Intuitively, the non-convex exclusion cone constraint (4.2c) ensures that the vectors $\boldsymbol{\psi}_s$ and $\boldsymbol{\psi}_c$, when expressed in the same reference frame, remain at least ϕ_{ko} radians apart as the spacecraft rotates and presents the greatest challenge for real-time optimization. In some cases, this is further complicated by an additional convex *inclusion* cone constraint needed for sensors that must remain pointed in certain directions to function (e.g. star or horizon trackers, communications equipment, etc.). We consider

only (4.2c) here to simplify presentation.

In addition to the *operational* constraints (4.2), note that the dynamics (4.1) above are also *dynamically* constrained to the manifold $\boldsymbol{x} \in X = \mathrm{S}^3 \times \mathbb{R}^3$. While this constraint is implicitly satisfied by iterates of the PRONTO solver (with complications addressed in the design of the cost function below), it should be noted that solvers which do not obtain the attitude path $\boldsymbol{q}(t)$ via precise integration of (4.1a) (e.g. transcription based solvers like GPOPS II) must also explicitly enforce the constraint $\|\boldsymbol{q}(t)\| = 1$ as a member of (4.2) to ensure $\boldsymbol{x}(t)$ evolves on X.

4.2 Problem Formulation

With the goals, dynamics, and constraints now specified, we may now specify the complete constrained trajectory optimization problem on the finite horizon $t \in [0, T]$ as follows

$$\begin{aligned} \min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)} \quad & \int_0^T \ell(\boldsymbol{x}(t), \boldsymbol{u}(t))\, dt \;+\; m(\boldsymbol{x}(T)), \\ \text{s.t.} \quad & \dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}),\; \boldsymbol{x}(0) = \boldsymbol{x}_0, \\ & c(\boldsymbol{x}, \boldsymbol{u}) \leq \boldsymbol{0}, \end{aligned} \tag{4.3}$$

where $c(\boldsymbol{x}, \boldsymbol{u})$ collects the constraints in (4.2), the functions $\ell(\boldsymbol{x}, \boldsymbol{u})$ and $m(\boldsymbol{x})$ specify the running and terminal costs, and the notation $\boldsymbol{x}(\cdot)$ denotes the entire curve $\boldsymbol{x}(t), t \in [0, T]$. Alternatively, the problem (4.3) can be written more compactly on the trajectory manifold as follows:

$$\begin{aligned} \min_{\boldsymbol{\xi} \in \mathcal{T}} \quad & h(\boldsymbol{\xi}) = \int_0^T \ell(\boldsymbol{x}(t), \boldsymbol{u}(t))\, dt \;+\; m(\boldsymbol{x}(T)), \\ \text{s.t.} \quad & c(\boldsymbol{x}, \boldsymbol{u}) \leq \boldsymbol{0}. \end{aligned} \tag{4.4}$$

Although there are certainly many suitable potential options for the cost functionals ℓ and m, a classically effective choice are the quadratic forms

$$\ell(\boldsymbol{x}, \boldsymbol{u}) \coloneqq \frac{1}{2}\|\boldsymbol{x}\|^2_{Q(\boldsymbol{x}_d)} + \frac{1}{2}\|\boldsymbol{u}\|^2_R, \tag{4.5a}$$

$$m(\boldsymbol{x}) \coloneqq \frac{1}{2}\|\boldsymbol{x}\|^2_{P(\boldsymbol{x}_d)}, \tag{4.5b}$$

where $\|\boldsymbol{v}\|^2_R$ is shorthand for the semi-norm $\boldsymbol{v}^T R\, \boldsymbol{v}$ and the matrix weights $Q(\boldsymbol{x}_d), P(\boldsymbol{x}_d) \in \mathbb{R}^{7\times 7}$ for the state and $R \in \mathbb{R}^{3\times 3}$ for the control are chosen positive semi-definite and positive definite

respectively.

Recalling that the above cost function must be appropriately defined on the nonlinear manifold X (rather than the parent linear space $\mathbb{R}^7$), the use of a classic quadratic cost in the above problem allows us to employ the same projection strategy used to define the PRONTO projection regulator to choose suitable weights $Q(\boldsymbol{x}_d), P(\boldsymbol{x}_d)$. This strategy, (and indeed, this specific example) is developed in detail in section 3.5. To briefly re-iterate, we first define the projection matrix $M(\boldsymbol{x}_d) \in \mathbb{R}^{6\times 7}$

$$M(\boldsymbol{q}_d) = \begin{bmatrix} Z(\boldsymbol{q}_d)^\top & 0 \\ 0 & \mathbb{I}_3 \end{bmatrix}, \tag{4.6}$$

used to project the local linear coordinates $\boldsymbol{z} \in R^7$ (of $\boldsymbol{x}$) to the tangent space $T_{\boldsymbol{x}_d}X$ around the target state $\boldsymbol{x}_d$ via the coordinate transformation $\boldsymbol{r} := M(\boldsymbol{x}_d)\boldsymbol{z}$. Applying this projection to the linearization of the dynamics (4.1) around $\boldsymbol{x}_d$ (given by the pair (A_d, B_d)), we obtain the simplified dynamics

$$\dot{\boldsymbol{r}} = \underbrace{\left[M(\boldsymbol{x}_d)A_d M(\boldsymbol{x}_d)^\top\right]}_{A_r :=} \boldsymbol{r} + \underbrace{[M(\boldsymbol{x}_d)B_d]}_{B_r :=} \boldsymbol{v}, \tag{4.7}$$

where, in this projected subspace, the pair (A_r, B_r) is now completely linearly controllable (see Appendix B for a complete proof). Thus, an exponentially stabilizing LQR can now be designed directly on this controllable subspace by solving the (projected) ARE

$$Q_r = P_r B_r R_{reg}^{-1} B_r^\top P_r - A_r^\top P_r - P_r A_r, \tag{4.8}$$

where Q_r is a symmetric positive definite cost weight matrices defined on the projected subspace appropriate to either the state cost for the original trajectory optimization problem (4.4) or the LQ feedback problem (3.6) defining the PRONTO projection regulator. For each of these problems, Q_r can be readily obtained by projecting a standard diagonal weight matrix $Q_c \in \mathbb{R}^{7\times 7}$ (with diagonal entries scaled intuitively based on the desired individual state errors or regulation) to this subspace following

$$Q_r := M(\boldsymbol{x}_d) Q_c M(\boldsymbol{x}_d)^\top.$$

Having obtained the Riccati solutions P_r on the reduced subspace for the regulator and cost problems respectively, we can now *lift* these results into the ambient space as follows

$$\begin{aligned} Q(\boldsymbol{x}_d) &= M(\boldsymbol{x}_d)^\top Q_r M(\boldsymbol{x}_d), \\ P(\boldsymbol{x}_d) &= M(\boldsymbol{x}_d)^\top P_r M(\boldsymbol{x}_d), \\ K(T) &= K_r M(\boldsymbol{x}_d), \end{aligned} \tag{4.9}$$

where, by design, both $Q(\boldsymbol{x}_d)$ and $P(\boldsymbol{x}_d)$ are now appropriately positive definite on the tangent space $T_{\boldsymbol{x}_d}X$ and zero otherwise (i.e. their kernels are the orthogonal complement of $T_{\boldsymbol{x}_d}X$).

For the purpose of defining the cost functionals ℓ and m, the above procedure yields cost functionals which are symmetrically defined between the two hemispheres of S^3, conveniently ensuring that the traditional 'unwinding' problem encountered under the quaternion attitude model is avoided for rest-to-rest transfers [12]. For the purposes of designing an effecitive PRONTO regulator, the lifted terminal cost $P(\boldsymbol{x}_d)$ now serves as a suitable boundary condition $P(T)$ for solving the DRE (3.7) to obtain the desired projection regulator K_ξ, and is suitably defined to be locally exponentially stabilizing around the terminal state $\boldsymbol{x}_d$.

4.3 Trajectory Initialization

As direct methods approaching problem (4.4), both PRONTO and GPOPS II require an initial guess for the curves $\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot)$. For performance reasons, this guess should (at least approximately satisfy the dynamics $f(\boldsymbol{x}, \boldsymbol{u})$ and boundary conditions $\boldsymbol{x}(0) = \boldsymbol{x}_0$ and $\boldsymbol{x}(T) = \boldsymbol{x}_d$. For a rest-to-rest attitude transfer using simple body-torque dynamics, a reasonable initial guess for the attitude path $\boldsymbol{q}(t)$ can be determined analytically from a geodesic on S^3 from $\boldsymbol{q}_0$ to $\bar{\boldsymbol{q}}_d$. To select the shortest rotation, the variant $\bar{\boldsymbol{q}}_d$ for the final attitude is chosen from $\pm\boldsymbol{q}_d$ to ensure that $\boldsymbol{q}_0$ and $\bar{\boldsymbol{q}}_d$ lie in the same hemisphere (e.g. $\boldsymbol{q}_0^\top \bar{\boldsymbol{q}}_d \geq 0$). Note that, if $\boldsymbol{q}_0^\top \bar{\boldsymbol{q}}_d = 0$, this choice is arbitrary.

On SO(3), a geodesic for the net rotation $\tilde{\boldsymbol{q}} = \boldsymbol{q}_0^* \circ \bar{\boldsymbol{q}}_d$ corresponds to a *single* principal rotation of $\phi_d \in [0, \pi]$ around a unit axis $\boldsymbol{e}_d \in \mathrm{S}^2$. This angle-axis representation is obtained directly from

the MRP

$$\boldsymbol{e}_d \tan\left(\frac{\phi_d}{4}\right) = \frac{\tilde{\boldsymbol{q}}_v}{1+\tilde{q}_s}.$$

Using Spherical Linear Interpolation [48], $\boldsymbol{q}(t)$ is then given by

$$\boldsymbol{q}(t) = \left(\frac{\sin\left((1-\tau)\phi_d/2\right)}{\sin\left(\phi_d/2\right)}\right)\boldsymbol{q}_0 + \left(\frac{\sin\left(\tau\phi_d/2\right)}{\sin\left(\phi_d/2\right)}\right)\bar{\boldsymbol{q}}_d,$$

where $\tau(t) \in [0,1]$ shapes the velocity profile along this geodesic. Since we know $\boldsymbol{q}(t)$ rotates around $\boldsymbol{e}_d$, all that remains is to select a rest-to-rest rotation rate $\|\boldsymbol{\omega}(t)\|$. A suitable choice is given by

$$\tau(t) = {}^1\!/_2\left(1-\cos\left({}^{\pi t}\!/_{T}\right)\right),$$

$$\boldsymbol{\omega}(t) = \dot{\tau}(t)\cdot\phi_d\boldsymbol{e}_d.$$

The torque control $\boldsymbol{u}(t)$ is then obtained by inverting the angular rate dynamics (4.1b) as follows

$$\boldsymbol{u}(t) = \left(\ddot{\tau}(t)\,J\boldsymbol{\omega}(t) + \widehat{\boldsymbol{\omega}}(t)\,J\boldsymbol{\omega}(t)\right),$$

This procedure generates an initial guess $\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot)$ which satisfies our dynamic model and boundary conditions, but leaves optimality and constraint satisfaction to the solver.

4.4 PRONTO Solver Demonstration

Before moving on to the experimental design forming the major results of this chapter, it is worthwhile to first briefly examine some example results (intermediate and final) obtained using the PRONTO algorithm. As a practical example of solutions to problem (4.4), consider now a model half rotation (180°) about the spacecraft's z-axis. Examining the (operationally) unconstrained problem first, a simple geodesic guess solution to this problem is given by the dashed lines in Figure 4.1(a), while the dotted lines show how this guess trajectory is deformed along $\mathcal{T}$ by the central PRONTO loop in algorithm 1. Even on the highly nonlinear attitude dynamics for the quaternion sphere S^3, the PRONTO approach is able to reach the optimizer in only 4 iterations.

Next, let us modify the problem to include the operational constraints given by (4.2). With the constraints enabled, the completed PRONTO algorithm (see algorithm 1) now solves for a

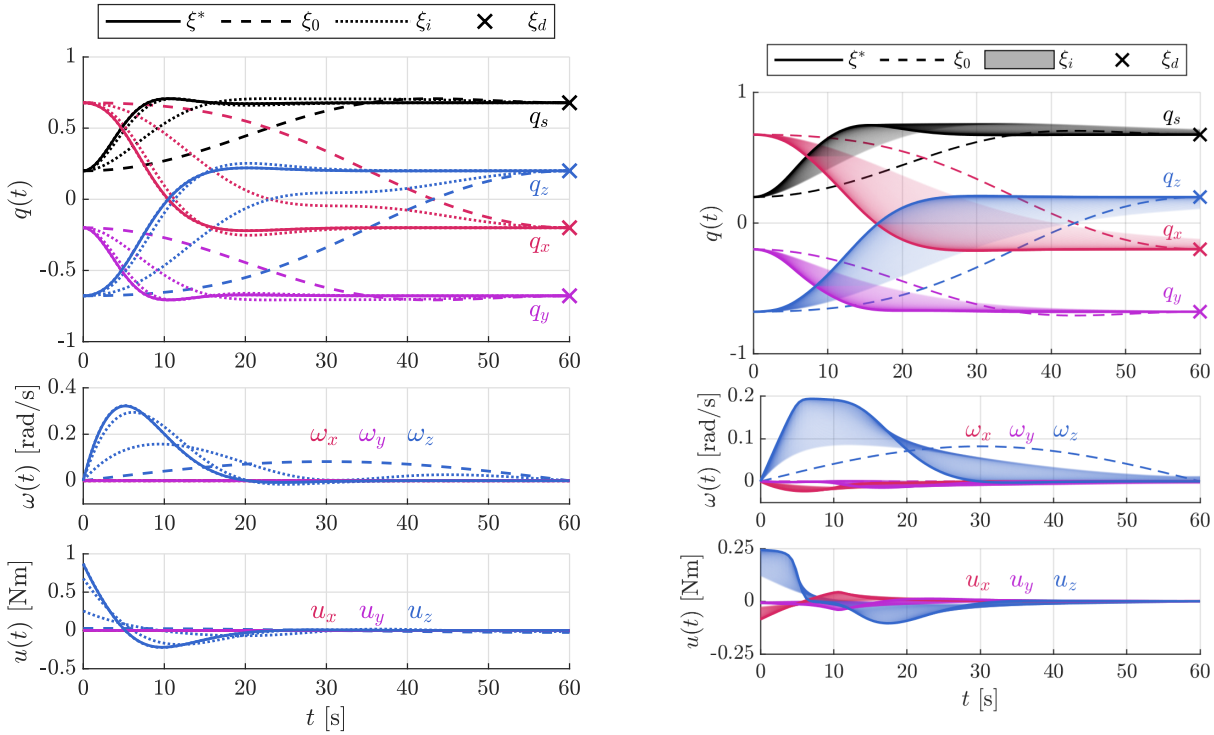

Figure 4.1: Example PRONTO guess, iterates, and solution for a 180° rotation about the z-axis of the spacecraft. Panel (a) (left) shows the unconstrained problem and the required 4 iterations to reach the solution, while Panel(b) shows the fully constrained problem and the (IP) central path for the solver.

feasible, optimal trajectory $\boldsymbol{\xi}$ for a particular set of constraint weights ϵ_j. Once this intermediate result is obtained, the solver proceeds down the so-called 'central path' by lowering ϵ_j by a fixed ratio and then re-solving for a new optimizer that is closer to the constraint boundary $c_j(\boldsymbol{x}, \boldsymbol{u}) = 0$. This process is repeated until ϵ_j reaches a predetermined threshold. A very fine sampling of the iterates of this central path for our 180° rotation example are shown in Figure 4.1(b) where the constraints for control saturation and angular velocity are visibly satisfied. Additionally, note the appearance of nonzero rotations along the x and y-axes to swerve around the exclusion cone constraint. An animation of these results is also available on our lab's website. Finally note that, with the exception of the initial guess, each iterate displayed in Figure 4.1(b) is an optimal, feasible result for (4.4) under a particular weight choice. As such, these intermediate solutions can be safely used in circumstances where rapid decisions are required (e.g. docking maneuvers or

collision avoidance).

4.5 Experimental Design

In this section, we evaluate the computational and operational performance of the PRONTO algorithm when applied to the constrained satellite attitude problem (4.4). To give these results context amongst the huge family of modern commercial solvers, we provide a comparison against the solutions obtained from the trajectory optimization package GPOPS II which has become a respected benchmark for evaluating new solvers [9, 51]. This comparison against an established benchmark serves to highlight the unique benefits of the PRONTO approach for real-time maneuver planning as well as to illustrate several potential improvements to the algorithm. All solutions obtained for both solvers were computed in Matlab on an Intel Core I5 6600K CPU.

In this study, both solvers were given 500 randomized problems generated via the attitudes $\boldsymbol{q}_0, \boldsymbol{q}_d$ sampled randomly on S^3. The exclusion cone vector $\boldsymbol{\psi}_s$ was defined in each case to *nearly* intersect the path of the fixed camera vector $\boldsymbol{\psi}_c$ on the initial guess trajectory with an exclusion angle of $\phi_{ko} = 10°$. An intentional offset of $1°$ was applied to the cone's center $\boldsymbol{\psi}_s$ to encourage both solvers to take the same direction around the exclusion region (though this is not necessary for either method to converge). For the remaining constraints, we selected $u_+ = 0.25\,\mathrm{N}$ and $\omega_+ = 0.2\,\mathrm{rad\,s^{-1}}$ to ensure that each constraint was actively considered in each problem. The spacecraft was modelled as a 100 kg CubeSat with an inertia tensor $J = \mathrm{diag}([10.6, 10.6, 6.2])\mathrm{kg\,m^2}$. Finally, the quadratic cost functionals ℓ and m and the projection regulator used by the PRONTO solver were each constructed following the procedure outlined above (i.e. solving (3.8) on the controllable subspace and lifting via (3.16)), where the weight matrix $Q_r := M(\boldsymbol{x}_d)Q_cM(\boldsymbol{x}_d)^\top$ on the controllable subspace was generated using $Q_c := \mathbb{I}_7$ and $R := \mathbb{I}_3$ for both the regulator and cost functional.

To further ensure that both PRONTO and GPOPS II were compared in a consistent and controlled way, both solvers were configured with

(1) Identical problems and initial guesses

(2) Matching cost, constraint, and solution tolerances.

(3) Matching Matlab MEX executables to compute the dynamics, cost, and constraints.

This last point ensures that both programs are solving *identical* optimization problems and that evaluations of the objective function (frequently the most computationally expensive step) are computed identically. The remaining configuration options for each solver were optimized to minimize computation time over a group of 20 test cases. Unfortunately, some minor intrinsic differences in the solvers could not be controlled for:

(1) PRONTO uses a 4x finer time-sampling of $\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot)$.

(2) GPOPS uses an *exterior* point approach for constraints.

(3) GPOPS computes its derivatives numerically.

This first discrepancy originates from PRONTO not natively including a mesh refinement strategy to exactly model sharp changes in the solution curves $\boldsymbol{\xi}_i$ and, correspondingly, in the descent direction $\boldsymbol{\zeta}_i$. However, because both $\boldsymbol{\xi}_i$ and $\boldsymbol{\zeta}_i$ are generated via continuous time integration, the default sampling employed by the integrator *can* be used as a suitable form of mesh refinement. For this study however, PRONTO was instead configured with a finer, fixed resolution to simplify presentation. The second point is a simple difference in how the constraints were implemented in each solver. PRONTO solutions approach the constraints with feasible curves, while GPOPS II solutions approach with infeasible curves. This distinction makes little difference in practice as the constraint definitions can be modified to ensure both algorithms return feasible results. This difference is visible in the final costs of each solution however, as exterior point methods are naturally able to get *slightly* closer to the constraints. Finally, simplifications in the commercial release of GPOPS II prevent it from directly using PRONTO's functions for computation of the analytic derivatives of the cost, dynamics, and constraint functions. However, GPOPS II's use of sparsity

to reduce complexity when computing numerical derivatives heavily reduces the impact of this difference on actual computation time.

4.6 Solution Features and Accuracy

To begin our comparison of the two solvers, we first compare the respective solutions produced by each solver for the problem shown in Figure 4.2. First, a brief inspection will show that both solutions are feasible in all constraints (with the yellow cone and green line indicating the solar exclusion cone and camera direction respectively) and successfully reach the target in approximately half the time of the original (geodesic) guess. However, while these two solutions appear to be nearly identical, the exterior point approach and dynamically-sampled trajectory representation strategy used by GPOPS II enables it to push its solution slightly closer to each of the constraints. This difference is visible in the camera path $\boldsymbol{\psi}_c(t)$ (which passes slightly closer to the exclusion cone) as well as in the control trajectory (which shows a sharper jump from max y-axis torque at $t \approx 7\,\mathrm{s}$).

Indeed, this observation is further supported when comparing the relative cost improvement of the GPOPS II solution over PRONTO as shown in Figure 4.3. Specifically, for the (449) cases in which both solvers chose the same direction around the exclusion cone constraint (and so approached the same optimizer), GPOPS II was able to descend the cost functional 0.2% further on average than PRONTO from the same initial guess. Critically, we note that this difference is *not* reflected in the final attitude errors shown in Figure 4.4(a). In fact, both solvers return nearly identical decay profiles and residual distributions in the MRP attitude error (i.e. the principal angle to rotate $\boldsymbol{q}^*(t)$ to $\boldsymbol{q}_d$). In practice, these distinctions are all minor and can be reduced by either lowering the solution constraint tolerances ϵ_j^- or increasing the constraint dropoff rate κ_j.

4.7 Algorithm Performance Comparison

We now reach the critical issues for the deployment of any real-time spacecraft maneuver-planner: algorithm computation time and feasibility. As shown in Figure 4.4(b), PRONTO takes 25 s on average to compute a feasible optimal result on a relatively modern CPU. In contrast,

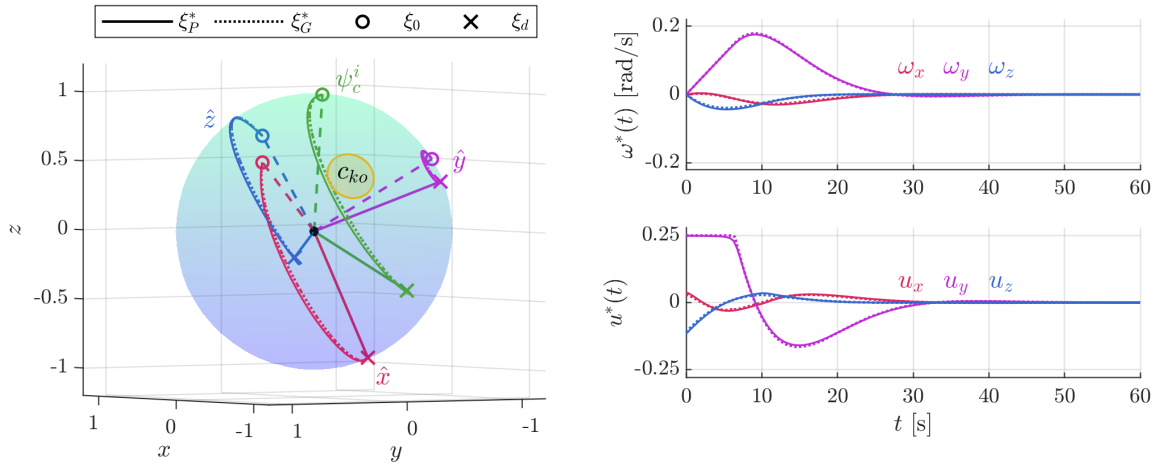

Figure 4.2: Optimal trajectories $\boldsymbol{\xi}_P^*$ and $\boldsymbol{\xi}_G^*$ and camera paths $\boldsymbol{\psi}_c^i(t)$ for PRONTO and GPOPS II algorithms respectively.

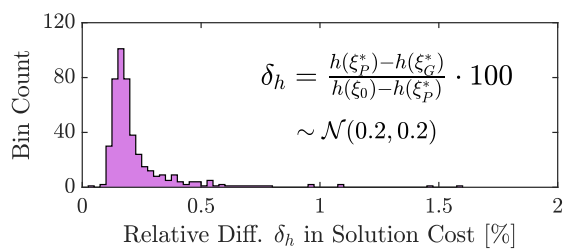

Figure 4.3: Distribution and Gaussian fit $N(\mu, \sigma)$ for the percentage descent improvement δ_h of the GPOPS II optimizer cost over that from PRONTO from the same initial guess.

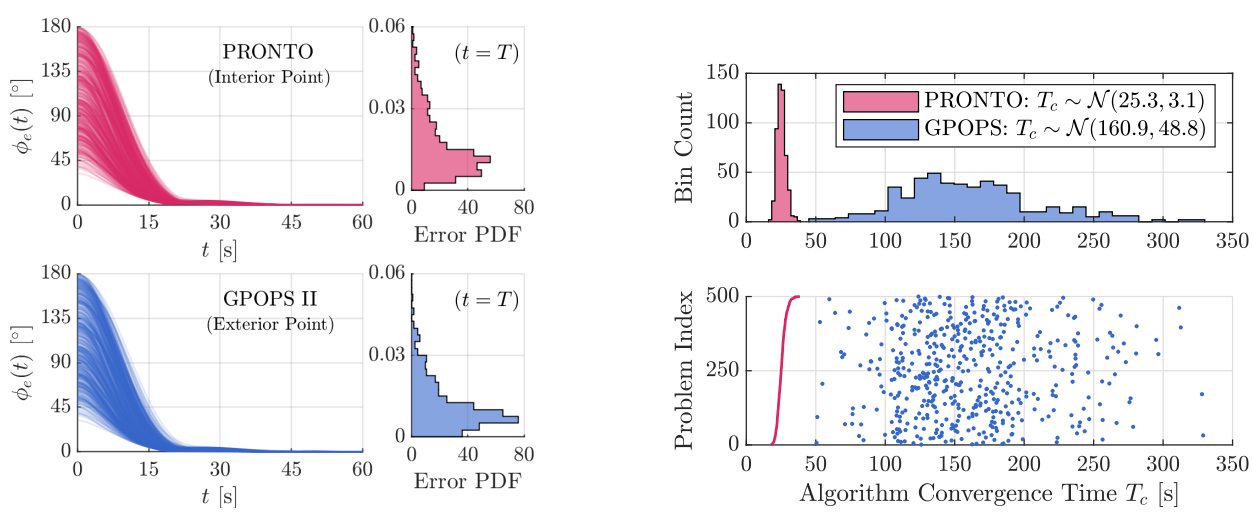

Figure 4.4: (a) Trajectories and final distributions for the principal rotation angle error $\phi_e(t)$ from the target attitude $\boldsymbol{q}_d$ and (b) Measured convergence times and Gaussian fits $T_c \sim N(\mu, \sigma)$ for PRONTO and GPOPS II algorithms.

GPOPS II takes approximately six times longer with a much wider distribution for its most difficult cases. While this difference can be partially attributed to GPOPS II's numeric computation of derivatives, the result clearly emphasizes the immediate computational benefits of optimization on the trajectory manifold. This significant improvement in computation time is further emphasized by the near-equivalence of the computed solutions displayed above.

However, there is an even greater advantage to this approach when discussing real-time implementation on spacecraft: the recursive feasibility (in both dynamics *and* constraints) of *intermediate* solutions. Specifically, recall that virtue of the projection operator, each PRONTO iterate is a valid trajectory of the system. Secondly, the feasibility requirement in the descent step-size calculation ensures that, once a feasible trajectory iterate is obtained, each successive iteration is also feasible. For the computed cases, this feasible iterate was *always* obtained within 6 iterations (or approximately 7 s) and, on average was obtained within 3 iterations (or 3 s). As a result, a spacecraft using PRONTO can reliably use sub-optimal intermediate solutions in cases where quick decisions are required for safe operation (e.g. docking or collision avoidance maneuvers). This capability is not available to classical direct collocation methods like GPOPS II, as both the dynamics and constraints are only loosely satisfied until the final solution is reached.

4.8 Conclusions

In this chapter, we specialized an existing optimal trajectory planning algorithm to the problem of rest-to-rest spacecraft attitude transfers subject to input saturation, angular rate, and non-convex state constraints. Specializing PRONTO to this uniquely nonlinear and non-convex problem required the design of custom quadratic cost functions, a custom regulator design, and an Interior Point modification to the algorithm to include constraints. After an extensive numerical comparison against the modern commercial trajectory optimization package GPOPS II, the completed PRONTO solver was found to return nearly identical solutions while taking only 16% of the computation time. It was also discussed that this performance gap could be further widened (perhaps significantly) by modifying PRONTO to use its native integrator time-sampling for the

curve iterates rather than a fixed representation. Finally it was noted that, in the realm of real-time trajectory optimization, this construction for PRONTO uniquely provides feasible intermediate solutions in circumstances where fast control action is required for the safety of the spacecraft. From this analysis, it is reasonable to conclude that PRONTO is a suitable optimal maneuver planning solution for spacecraft attitude maneuver planning. Following this final line of reasoning, the two subsequent chapters of this thesis will now develop the (substantially more complicated) spacecraft maneuver planning problem using a standard CMG momentum array.

Chapter 5

Spacecraft Attitude Control via Momentum Exchange

In this chapter, we will develop the fundamental (conservative) momentum exchange physics of a traditional Single-Gimbal Control Moment Gyroscope array. Using this foundation, we will then develop a comprehensive, conservative dynamical model for CMG-driven spacecraft with the intent of developing a conservative, computationally tractable trajectory optimization problem which utilizes the full capabilities of the momentum array. From the literature review presented in chapter 2, it should be noted that such a comprehensive approach (including momentum conserving dynamics, a *single* centralized control policy for the entire spacecraft, and common safety constraints) is *entirely novel* (likely owing to the bewildering complexity of the resulting problem). The dynamical model developed in this chapter was first presented in [13, 14], with the second work still under review.

5.1 Momentum and Inertia of a CMG array

Recall from chapter 2 that a Control Moment Gyroscope is a reaction wheel mounted on a rotating gimbal as shown in Figure 5.1, where the wheel (red) and gimbal (blue) motors act along the $\boldsymbol{a}_s$ and $\boldsymbol{a}_g$ axes respectively and the gimbal's orientation is measured by the angle $\delta \in [0, 2\pi)$. In addition to the standard (direct) reaction torques produced by the gimbal and wheel motors, a CMG additionally employs the *gyroscopic* reaction torque

$$\boldsymbol{\tau}_r = \dot{\delta} h_w \boldsymbol{a}_t,$$

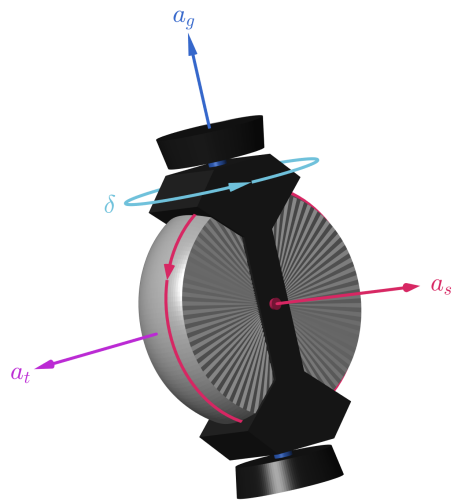

Figure 5.1: Control Moment Gyroscope coordinate frame.

produced along the transverse axis $\boldsymbol{a}_t \coloneqq \boldsymbol{a}_s \times \boldsymbol{a}_g$ for attitude control. This reaction torque $\boldsymbol{\tau}_r$ is proportional to the rotation rate $\dot{\delta} \in \mathbb{R}$ of the gimbal frame and is amplified by the rotor momentum $h_w \in \mathbb{R}$. Unlike with RW's however, the available torque from an array of m CMG's varies with the array's configuration $\boldsymbol{\delta} \coloneqq [\delta_1; \cdots; \delta_m]$, requiring more sophisticated maneuver planning strategies. The available torque spaces for the CMG gimbal and wheel motors are spanned respectively by the column spaces of the matrices

$$\begin{aligned} A_s &\coloneqq [\boldsymbol{a}_{s,1}, \cdots, \boldsymbol{a}_{s,m}] \in \mathbb{R}^{3\times m}, \\ A_t &\coloneqq [\boldsymbol{a}_{t,1}, \cdots, \boldsymbol{a}_{t,m}] \in \mathbb{R}^{3\times m}, \end{aligned} \tag{5.1}$$

which vary with the array configuration $\boldsymbol{\delta}$ following

$$\begin{aligned} A_s(\boldsymbol{\delta}) &\coloneqq A_{s0}\,\mathrm{diag}(\cos(\boldsymbol{\delta})) - A_{t0}\,\mathrm{diag}(\sin(\boldsymbol{\delta})), \\ A_t(\boldsymbol{\delta}) &\coloneqq A_{t0}\,\mathrm{diag}(\cos(\boldsymbol{\delta})) + A_{s0}\,\mathrm{diag}(\sin(\boldsymbol{\delta})), \end{aligned} \tag{5.2}$$

where the functions $\sin(\cdot)$ and $\cos(\cdot)$ act entry-wise for vector inputs and the matrices $A_s(0) = A_{s0}$ and $A_t(0) = A_{t0}$ define the default configuration of the array geometry. For completeness, the gimbal axes (fixed in the satellite body frame) are collected in the constant matrix $A_g = [\boldsymbol{a}_{g,1}, \cdots, \boldsymbol{a}_{g,m}]$.

With this notation established, we now examine the momentum exchange physics of a CMG

array to determine the satellite's variable Moment of Inertia (MoI) and body-frame angular momentum. First let $J_B \in \mathbb{R}^{3\times 3}$ be the constant diagonal inertia of the satellite (omitting the CMG array) in the body frame $\mathcal{F}_b$. Let the array have m CMG's with relative positions $\boldsymbol{b}_i \in \mathbb{R}^3$ and orientations in $\mathcal{F}_b$ given by the matrices A_g, A_s, and A_t as defined above. Let each CMG have mass m_i and principle inertia $J_{g,i}$, $J_{s,i}$, and $J_{t,i} \in \mathbb{R}$ along their gimbal, spin, and transverse axes respectively. Collecting these inertias into the $m \times m$ diagonal matrices $\boldsymbol{J}_s$, $\boldsymbol{J}_t$, and $\boldsymbol{J}_g$ (e.g. $\boldsymbol{J}_s \coloneqq \mathrm{diag}([J_{s,1}, \cdots, J_{s,m}])$) and applying the parallel axis theorem, the satellite's total MoI is assembled as follows:

$$J_{stg}(\boldsymbol{\delta}) \coloneqq J + A_g \boldsymbol{J}_g A_g^\top + A_s \boldsymbol{J}_s A_s^\top + A_t \boldsymbol{J}_t A_t^\top, \tag{5.3a}$$

$$J = J_B + \sum_{i=1}^{m} m_i \left(\mathbb{I}_3 \|\boldsymbol{b}_i\|^2 - \boldsymbol{b}_i \boldsymbol{b}_i^\top \right), \tag{5.3b}$$

where $\mathbb{I}_3$ is the 3×3 identity matrix. Note that the first two terms of (5.3a) are constant as the CMG gimbal axes and centers of mass are fixed in the body frame.

To define the individual momenta and inertias of the gimbal frame and wheel of each CMG, we now introduce the comprehensive notation system employed by [16]. Specifically, each scalar CMG momenta term $h_{(\cdots)} \in \mathbb{R}$ and single axis inertia term $J_{(\cdots)}$ will be identified using a specific sequence of subscripts. The first two subscripts directly identify a) the relevant CMG rotation axis (s for spin, g for gimbal, or t for transverse) and b) the rotating mass within the CMG (w for the CMG's wheel or g for its gimbal frame). For each momentum term, a third additional subscript is used to specify the specific Center of Mass (CoM) about which the angular momentum is modeled, with r (relative) indicating rotation around the CMG CoM, and a (absolute) indicating rotation around the satellite's CoM (i.e. including the extra orbital angular momentum of the CMG around $\mathcal{F}_b$). For example, the ith CMG's wheel momentum and inertia about its own spin axis is given by $h_{swr,i} \in \mathbb{R}$ and $J_{sw,i} \in \mathbb{R}_{\geq 0}$ respectively, and the vector $\boldsymbol{h}_{swr} \in \mathbb{R}^m$ lists this angular momentum for each CMG in the array (with inertias following the above diagonal structure for $\boldsymbol{J}_s$). For compactness, the second subscript may be omitted to indicate the total momentum in the CMG (e.g. $h_{sa} \coloneqq h_{swa} + h_{sga}$). It should be noted that some momentum terms in this notation are

zero by definition due to mechanical restrictions in the CMG's motion (e.g. $h_{sgr} \equiv 0$ as the CMG gimbal cannot rotate around the spin axis in $\mathcal{F}_b$).

To determine the satellite's angular momentum, let the body frame $\mathcal{F}_b$ have an angular rotation rate $\boldsymbol{\omega} \in \mathbb{R}^3$ (measured in $\mathcal{F}_b$) with respect to the inertial frame $\mathcal{F}_i$. The satellite's total angular momentum is then a combination of the momentum $J_{stg}\boldsymbol{\omega}$ from the rotation of $\mathcal{F}_b$ and the CMG's angular momentum relative to $\mathcal{F}_b$. This can be compactly written in $\mathcal{F}_b$ as

$$\boldsymbol{h} = J_{stg}\boldsymbol{\omega} + A_s\boldsymbol{h}_{sr} + A_g\boldsymbol{h}_{gr} + A_t\boldsymbol{h}_{tr}, \tag{5.4}$$

where, mechanically, $\boldsymbol{h}_{tr} \equiv 0$ by definition (i.e. the CMG frame is mechanically unable to rotate about the transverse axis). For the purposes of our optimization problem however, it will prove preferable to exchange some of the relative CMG momenta coordinates in (5.4) for their absolute variations given by

$$\boldsymbol{h}_{ga} = \boldsymbol{h}_{gr} + \boldsymbol{J}_g A_g^\top \boldsymbol{\omega}, \tag{5.5a}$$

$$\boldsymbol{h}_{swa} = \boldsymbol{h}_{swr} + \boldsymbol{J}_{sw} A_s^\top \boldsymbol{\omega}, \tag{5.5b}$$

$$\boldsymbol{h}_{sga} = \boldsymbol{h}_{sgr} + \boldsymbol{J}_{sg} A_s^\top \boldsymbol{\omega}, \tag{5.5c}$$

which incorporate the orbital component of their angular momentum (normally embedded in $J_{stg}\boldsymbol{\omega}$). Noting again that $\boldsymbol{h}_{sgr} \equiv 0$ or, notationally, $\boldsymbol{h}_{swr} = \boldsymbol{h}_{sr}$, (5.3) and (5.4) can be compactly rewritten as

$$J_{st}(\boldsymbol{\delta}) \coloneqq J + A_s\boldsymbol{J}_s A_s^\top + A_t\boldsymbol{J}_t A_t^\top, \tag{5.6a}$$

$$\boldsymbol{h} \coloneqq J_{st}\boldsymbol{\omega} + A_s\boldsymbol{h}_{swr} + A_g\boldsymbol{h}_{ga}. \tag{5.6b}$$

For compactness, we will often use (5.6b) to convert between $\boldsymbol{h}$ and $\boldsymbol{\omega}$ via the following transformations:

$$\bar{\boldsymbol{h}}(\boldsymbol{\omega}, \boldsymbol{\delta}, \boldsymbol{h}_{swr}, \boldsymbol{h}_{ga}) \coloneqq J_{st}\boldsymbol{\omega} + A_s\boldsymbol{h}_{swr} + A_g\boldsymbol{h}_{ga}, \tag{5.7a}$$

$$\bar{\boldsymbol{\omega}}(\boldsymbol{h}, \boldsymbol{\delta}, \boldsymbol{h}_{swr}, \boldsymbol{h}_{ga}) \coloneqq J_{st}^{-1}\left(\boldsymbol{h} - A_s\boldsymbol{h}_{swr} - A_g\boldsymbol{h}_{ga}\right), \tag{5.7b}$$

and will often write $\bar{\boldsymbol{h}}$ or $\bar{\boldsymbol{h}}(\boldsymbol{x})$ for (5.7a), and $\bar{\boldsymbol{\omega}}$ or $\bar{\boldsymbol{\omega}}(\boldsymbol{x})$ for (5.7b) respectively. Notably, (5.7a) can be used to determine the array's $3 \times m$ actuator Jacobian

$$\begin{aligned} D(\boldsymbol{\omega}, \boldsymbol{\delta}, \boldsymbol{h}_{swr}) &\coloneqq \frac{\partial \bar{\boldsymbol{h}}}{\partial \boldsymbol{\delta}} = \frac{\partial J_{st}}{\partial \boldsymbol{\delta}} \boldsymbol{\omega} + \frac{\partial A_s}{\partial \boldsymbol{\delta}} \boldsymbol{h}_{swr} \\ &= \left[A_s \operatorname{diag}(A_t^\top \boldsymbol{\omega}) + A_t \operatorname{diag}(A_s^\top \boldsymbol{\omega}) \right] (\boldsymbol{J}_t - \boldsymbol{J}_s) - A_t \operatorname{diag}(\boldsymbol{h}_{swr}). \end{aligned} \tag{5.8}$$

Note that this particular variation of the actuator Jacobian, while still relating the gimbal rates $\dot{\boldsymbol{\delta}}(t)$ to the array's output torque $\boldsymbol{\tau}_r$, differs slightly from that employed by the classic SR feedback law (see equation (2.13)). In particular, the variation employed in the SR law includes additional terms relating to the dynamic cancellation employed by that feedback law (a design strategy we are intentionally avoiding here).

5.2 CMG Energy Modeling

In addition to the comprehensive momentum exchange physics developed above, it will also prove useful to develop an accurate model of the true electric power consumed by the CMG motors. As introduced in chapter 2, the electrical power consumed by a motor is defined by the relation $P_m = \omega_m \tau_m$, where ω_m specifies the current shaft speed of the motor and τ_m specifies the requested torque. Extending this definition to the motors of a Single-Gimbal CMG, the individual power consumed by a CMG's gimbal and wheel motor respectively can be written as

$$P_{g,i} = \dot{\delta}_i \, u_{g,i}, \quad P_{w,i} \coloneqq J_{sw,i}^{-1} \, h_{swr,i} \, u_{w,i}, \tag{5.9}$$

where $u_{g,i}, u_{w,i} \in \mathbb{R}$ are the commanded (control input) torques for each gimbal and wheel motor respectively. Following the notation developed above, these powers can be grouped by motor type, yielding the following compact representations for total gimbal and wheel motor power consumed by the array

$$\boldsymbol{P}_g \coloneqq \operatorname{diag}(\boldsymbol{f}_{\boldsymbol{\delta}}) \, \boldsymbol{u}_g, \quad \boldsymbol{P}_w \coloneqq \boldsymbol{J}_{sw}^{-1} \operatorname{diag}(\boldsymbol{h}_{swr}) \, \boldsymbol{u}_w, \tag{5.10}$$

where the function $\boldsymbol{f}_{\boldsymbol{\delta}}(\boldsymbol{x}, \boldsymbol{u}) = \dot{\boldsymbol{\delta}}$ (developed below) captures the dynamics of the array gimbal angles. As with the single motor, these power equations are bilinear in the motor shaft speeds

and control torque, ensuring that the same torque requested at a higher shaft (or wheel) speeds will require a proportionally increase in power draw. With this in mind, the powerful efficiency improvement offered by CMG's over conventional RW's is apparent when comparing the coefficients for both of the above motor torques. In particular, the gimbal rate $\dot{\delta}$ of a CMG is nominally zero and (virtue of torque amplification) is normally much smaller than the speed of a traditional RW, requiring a much lower power to produce the same output torque. As a result, CMG's promote greater efficiency and agility for larger spacecraft like the ISS, which employs four double-gimbal CMGs in its design.

However, this engineering feature also critically impacts the formulation of our optimal control problem, as the absolute control torque $\|\tau_m\|$ (or 'control effort') alone is no longer sufficient to capture the true (state-dependent) energy usage of the array. As such, the standard penalty on $\|\boldsymbol{u}\|$ employed by standard incremental cost formulations is not suitable for this system. Instead, penalties of the form $\|\boldsymbol{P}_g\|$ and $\|\boldsymbol{P}_w\|$ are needed to specifically target the total electric power usage of the array. Notably, this subtlety cannot be considered by maneuver planners using approximated dynamics that omit the internal momentum states of the CMG.

5.3 Momentum Exchange Dynamics

With the satellite's momentum exchange physics fully modeled, we may now develop the complete momentum-conserving dynamics for a CMG driven spacecraft. Specifically, this derivation takes the form of a complex nonlinear coordinate transformation (see Appendix C for the complete derivation) from their original direct coordinate choice in [16] to a set of coordinates more appropriate to the relevant constraints and error metrics of our trajectory optimization problem. In particular, the relative CMG wheel momenta $\boldsymbol{h}_{swr}$ are preferable to their (marginally more obscure) absolute representation $\boldsymbol{h}_{swa}$, as the former are directly proportional to the individual CMG wheel speeds. As developed above, these wheel speeds (under any control law) require a minimum amount of regulation in order to prevent the unnecessary wear or modelling error encountered at very low or high speeds. By choosing the coordinates $\boldsymbol{h}_{swr}$, this performance goal can be directly incorporated

into the cost functional via a (nominal) state tracking penalty (or alternatively, included as a pair of convex box constraints). Similarly, we prefer the spacecraft's angular rate $\boldsymbol{\omega}$ to the body-frame angular momentum $\boldsymbol{\omega}$ as this simplifies the incorporation of standard angular rate slew constraints. Finally, the original CMG motor torques were a natural choice for the control inputs, as they give the optimizer maximal control over the array's momentum exchange dynamics and the resulting reaction control torques. With these considerations, the clear choice of coordinates for the state and control inputs are then

$$\begin{aligned} \boldsymbol{x} &\coloneqq [\,\boldsymbol{q}\,;\boldsymbol{h}_{swr}\,;\boldsymbol{\omega}\,;\boldsymbol{\delta}\,;\boldsymbol{h}_{ga}\,] \in \mathbb{R}^{3m+7}, \\ \boldsymbol{u} &\coloneqq [\,\boldsymbol{u}_g\,;\boldsymbol{u}_w\,] \in \mathbb{R}^{2m}, \end{aligned} \tag{5.11}$$

where $\boldsymbol{u}_w, \boldsymbol{u}_g \in \mathbb{R}^m$ collect the CMG motor torque inputs for the wheel and gimbal respectively. Using the notation $\dot{\boldsymbol{\nu}} \coloneqq \boldsymbol{f}_\nu(\boldsymbol{x}, \boldsymbol{u})$ for $\boldsymbol{\nu} \in \{\boldsymbol{h}, \boldsymbol{h}_{swr}, \boldsymbol{\omega}, \boldsymbol{\delta}, \boldsymbol{h}_{ga}\}$, the body-frame dynamics for a momentum conserving CMG array are

$$\dot{\boldsymbol{q}} = {}^1\!/_2\, O_L(\boldsymbol{q})\, \widetilde{\boldsymbol{\omega}}, \tag{5.12a}$$

$$\dot{\boldsymbol{h}}_{swr} = \boldsymbol{J}_{sw}\left[\operatorname{diag}(A_t^\top \boldsymbol{\omega})\boldsymbol{f}_\delta - A_s^\top \boldsymbol{f}_\omega\right] + \boldsymbol{u}_w, \tag{5.12b}$$

$$\dot{\boldsymbol{\omega}} = J_{st,a}^{-1}\left[\boldsymbol{f}_h - D_a \boldsymbol{f}_\delta - A_g \boldsymbol{f}_{hga} - A_s \boldsymbol{u}_w\right], \tag{5.12c}$$

$$\dot{\boldsymbol{\delta}} = \boldsymbol{J}_g^{-1}\boldsymbol{h}_{ga} - A_g^\top \boldsymbol{\omega}, \tag{5.12d}$$

$$\dot{\boldsymbol{h}}_{ga} = \operatorname{diag}\left(A_t^\top \boldsymbol{\omega}\right)\left[(\boldsymbol{J}_t - \boldsymbol{J}_s)A_s^\top \boldsymbol{\omega} - \boldsymbol{h}_{swr}\right] + \boldsymbol{u}_g, \tag{5.12e}$$

where the dynamics for the satellite's body frame angular momentum are given by

$$\boldsymbol{f}_h(\boldsymbol{x}, \boldsymbol{\tau}_e) \coloneqq \widehat{\bar{\boldsymbol{h}}}\boldsymbol{\omega} + \boldsymbol{\tau}_e, \tag{5.13}$$

and $\boldsymbol{\tau}_e \in \mathbb{R}^3$ collects all known external disturbance torques on the satellite body (atmospheric drag, solar pressure, etc.). To compactly accommodate the new coordinate choice, (5.12) employs minor variations on the functions for the satellite's MoI (5.6a) given by

$$J_{st,a}(\boldsymbol{\delta}) \coloneqq J + A_s \boldsymbol{J}_{sg} A_s^\top + A_t \boldsymbol{J}_t A_t^\top, \tag{5.14}$$

as well as for the actuator Jacobian (5.8) given by

$$D_a(\boldsymbol{\omega}, \boldsymbol{\delta}, \boldsymbol{h}_{swa}) \coloneqq \frac{\partial J_{st,a}}{\partial \boldsymbol{\delta}} \boldsymbol{\omega} + \frac{\partial A_s}{\partial \boldsymbol{\delta}} \boldsymbol{h}_{swa} \tag{5.15}$$
$$= \left[A_s \operatorname{diag}(A_t^\top \boldsymbol{\omega}) + A_t \operatorname{diag}(A_s^\top \boldsymbol{\omega}) \right] (\boldsymbol{J}_t - \boldsymbol{J}_{sg}) - A_t \operatorname{diag}(\boldsymbol{h}_{swa}).$$

Notably, the individual state dynamics in (5.12) are ordered by computational dependency (*not* integrator order) with each dynamical block requiring results only from lower blocks (e.g. $\boldsymbol{f}_\omega$ depends on $\boldsymbol{f}_\delta$ but not on $\boldsymbol{f}_{hswr}$).

5.4 Implicit Dynamical Constraints

As with the body-torque attitude dynamics considered in the previous chapter, the dynamics (5.12) naturally evolve along a nonlinear constraint manifold X induced (in part) by the quaternion constraint $\boldsymbol{q} \in \mathrm{S}^3$. Unlike these dynamics however, the state manifold X for the CMG driven system is also induced by a *physical* constraint: the conservation of the spacecraft's total inertial angular momentum. To examine this, we first write these two constraints in the following form

$$1 = \|\boldsymbol{q}\|, \tag{5.16a}$$

$$\boldsymbol{h}_0 = C(\boldsymbol{q}) \bar{\boldsymbol{h}}(\boldsymbol{\omega}, \boldsymbol{\delta}, \boldsymbol{h}_{ga}, \boldsymbol{h}_{swr}), \tag{5.16b}$$

where the satellite's inertial-frame angular momentum $\boldsymbol{h}_0 \in \mathbb{R}^3$ is conserved in the absence of external forces (and, indeed, nominally regulated by other attitude actuators aboard the spacecraft). Together, the constraints (5.16) implicitly constrain $\boldsymbol{x}$ to a $(3m+3)$-submanifold $X(\boldsymbol{h}_0)$ of the ambient linear space $\mathbb{R}^{3m+7}$. Notably, common dynamical approximations which compromise (5.16b) (e.g. modeling the MoI $J_{st}(\boldsymbol{\delta})$ as a constant) are not constrained to $X(\boldsymbol{h}_0)$ (ensuring that resulting trajectories are almost certainly not physical).

As with the body-torque attitude dynamics, the state manifold $X(\boldsymbol{h}_0)$ also implicitly constrains the local linear controllability of the linearized dynamics $(A(\boldsymbol{x}), B(\boldsymbol{x}))$ of the CMG array to the tangent space $T_{\boldsymbol{x}} X(\boldsymbol{h}_0)$. To develop a parameterization of this controllable subspace (which, as above, can be used to define a locally stabilizing LQ regulator and quadratic terminal cost

functional), we can write the 1st-order Taylor expansion of the constraints (5.16) around $\boldsymbol{x}$ in the direction $\boldsymbol{z}$ as follows

$$\begin{bmatrix} 1 \\ \bar{\boldsymbol{h}}(\boldsymbol{x}) \end{bmatrix} \approx \begin{bmatrix} 1 \\ \bar{\boldsymbol{h}}(\boldsymbol{x}) \end{bmatrix} + \underbrace{\begin{bmatrix} \boldsymbol{q}^\top/\|\boldsymbol{q}\| & 0 & 0 & 0 & 0 \\ 2\left[\bar{\boldsymbol{h}}, -\widehat{\bar{\boldsymbol{h}}}\right] O_L(\boldsymbol{q}^*) & A_s & J_{st} & D & A_g \end{bmatrix}}_{Z(\boldsymbol{x}):=} \boldsymbol{z}, \tag{5.17}$$

where we have simplified the expansion by applying the (full rank) transformation $C(\boldsymbol{q})^\top$ to the (normally constant) lower block (i.e. $C(\boldsymbol{q})^\top \boldsymbol{h}_0 = \bar{\boldsymbol{h}}(\boldsymbol{x})$). Noting the nonzero element $\boldsymbol{q}^\top/\|\boldsymbol{q}\|$ and strongly positive definite matrix J_{st}, it is clear that, for any $\boldsymbol{x} \in X(\boldsymbol{h}_0)$, the rows of the matrix $Z(\boldsymbol{x})$ are linearly independent and (by construction) span the null space of $A(\boldsymbol{x})$ (e.g. $A(\boldsymbol{x})Z(\boldsymbol{x})^\top = 0$). Using this parametrization of the orthogonal complement to the tangent space $T_{\boldsymbol{x}}X$, the desired projection matrix $M(\boldsymbol{x}) \in \mathbb{R}^{3m+3\times 3m+7}$ to $T_{\boldsymbol{x}}X$ can then be defined row-wise by any orthonormal basis spanning the kernel of $Z(\boldsymbol{x})$ (obtainable via $\texttt{null}(Z(\boldsymbol{x}))^\top$ in Matlab).

Having obtained the desired projection matrix $M(\boldsymbol{x})$ to the tangent space of the configuration manifold $X(\boldsymbol{h}_0)$, the only remaining requirement for obtaining a local LQ regulator stabilizing the dynamics (5.12) about a general point $\boldsymbol{x} \in X(\boldsymbol{h}_0)$ is that the local dynamics (A_r, B_r) in the reduced subspace are locally linearly controllable. While this is certainly true around the static equilibria $\boldsymbol{x}_d$ considered for the trajectory optimization problems in the next chapter, a general proof for arbitrary array geometries and target states $\boldsymbol{x} \in X(\boldsymbol{h}_0)$ is entirely nontrivial and absent from existing literature. However, [7] have shown a (comparatively underactuated) variation of the dynamics (5.12) to be linearly controllable around non-singular equilibrium points. While this result (and our own extensive numerical testing) indicate that this restriction is unlikely to have any significant practical impact, the general case remains an open problem.

Chapter 6

Trajectory Optimization with Momentum Arrays

In this chapter, we will develop an optimal spacecraft maneuver planner for rest-to-rest attitude transfers using a Control Moment Gyroscope array. In contrast to conventional formulations built using approximated dynamical models, our formulation will examine the optimal performance and unique control strategies available to a CMG array under comprehensive physical models for its dynamics and power consumption. In particular, our formulation employs the dynamical model (5.12) which preserves the array's (conservative) momentum exchange dynamics, typical operational safety constraints on input saturation, angular velocity, and camera exclusion cones, and in certain cases, a sophisticated power model (5.10) directly tracking the usage of the individual CMG motors. The optimal control strategies produced under this comprehensive formulation display substantial improvements to mean maneuver performance and efficiency. Additionally, several specific control behaviors are identified to correlate with these improvements across the solution family. Finally, potential approaches are suggested to reproduce these behaviors using existing feedback control methods without requiring strict optimality. This chapter summarizes the results presented in [13, 14].

6.1 System Dynamics and Constraints

Evolving from the problem developed in chapter 4, we examine a family of optimal attitude transfers from $\boldsymbol{x}_0$ to $\boldsymbol{x}_d \in X(\boldsymbol{h}_0)$ while satisfying the (momentum conserving) dynamics $\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u})$ in (5.12). For simplicity, we only consider rest-to-rest transfers between (non-singular) equilibrium

points (e.g. $\boldsymbol{x}_0, \boldsymbol{x}_d \in \{x \in X(\boldsymbol{h}_0) : f(\boldsymbol{x}, 0) = 0,\ \boldsymbol{\omega} = 0\}$). Additionally, feasible solutions must satisfy the following operational safety constraints on control input saturation, angular rate limits, and an exclusion cone constraint to avoid damaging onboard cameras or sensors by pointing them at bright stellar objects [31]. These constraints are given in the nonlinear inequality form $c_j(\boldsymbol{x}, \boldsymbol{u}) \leq 0$ as follows:

$$u_{g,j}^2 - u_{g+}^2 \leq 0, \quad j \in 1, \ldots, m, \tag{6.1a}$$

$$u_{w,j}^2 - u_{w+}^2 \leq 0, \quad j \in 1, \ldots, m, \tag{6.1b}$$

$$\omega_j^2 - \omega_{j+}^2 \leq 0, \quad j \in 1, 2, 3, \tag{6.1c}$$

$$(\boldsymbol{\psi}_s^{\,i})^\top C(\boldsymbol{q})\, \boldsymbol{\psi}_c^{\,b} - \cos(\phi_{ko}) \leq 0, \tag{6.1d}$$

where $u_{g+}, u_{w+} \in \mathbb{R}_{>0}$ are the maximum control torques for the gimbal and wheel motors, $\omega_{j+} \in \mathbb{R}_{>0}$ are the maximum rotation rates around each body frame axis, $\boldsymbol{\psi}_c^{\,b} \in \mathrm{S}^2$ is the body-fixed frame orientation of the onboard sensor, and $\boldsymbol{\psi}_s^{\,i} \in \mathrm{S}^2$ and $\phi_{ko} \in [0, \pi]$ are the inertial orientation and minimum avoidance angle for the light source. Notably, the exclusion cone constraint (6.1d) is non-convex and may prove challenging for traditional solvers.

6.2 Problem Formulations

Having specified the goals, dynamics, constraints, and boundary conditions for this problem, the general constrained trajectory optimization problem considered in this chapter can be written in the following standard form on the finite horizon $t \in [0, T]$

$$\begin{aligned} \min_{\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot)} \quad & \int_0^T \ell(\boldsymbol{x}(t), \boldsymbol{u}(t))\, dt \ + \ m(\boldsymbol{x}(T)), \\ \text{s.t.} \quad & \dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}),\ \boldsymbol{x}(0) = \boldsymbol{x}_0, \\ & c(\boldsymbol{x}, \boldsymbol{u}) \leq \boldsymbol{0}, \end{aligned} \tag{6.2}$$

where $c(\boldsymbol{x}, \boldsymbol{u})$ collects the constraints in (6.1), the functions $\ell(\boldsymbol{x}, \boldsymbol{u})$ and $m(\boldsymbol{x})$ specify the running and terminal costs, and the notation $\boldsymbol{x}(\cdot)$ denotes the entire curve $\boldsymbol{x}(t), t \in [0, T]$. Alternatively,

the general problem (6.2) can be written more compactly on the trajectory manifold as follows:

$$\begin{aligned} \min_{\boldsymbol{\xi}\in\mathcal{T}} \quad & h(\boldsymbol{\xi}) = \int_0^T \ell(\boldsymbol{x}(t), \boldsymbol{u}(t))\, dt \;+\; m(\boldsymbol{x}(T)), \\ \text{s.t.} \quad & c(\boldsymbol{x}, \boldsymbol{u}) \leq \mathbf{0}, \end{aligned} \tag{6.3}$$

where we again select the following familiar quadratic forms for the cost functionals ℓ and m

$$\ell_0(\boldsymbol{x}, \boldsymbol{u}) \coloneqq \frac{1}{2}\|\boldsymbol{x}\|^2_{Q(\boldsymbol{x}_d)} + \frac{1}{2}\|\boldsymbol{u}\|^2_R, \tag{6.4a}$$

$$m(\boldsymbol{x}) \coloneqq \frac{1}{2}\|\boldsymbol{x}\|^2_{P(\boldsymbol{x}_d)}, \tag{6.4b}$$

where $\|\boldsymbol{v}\|^2_R$ is shorthand for the semi-norm $\boldsymbol{v}^T R\, \boldsymbol{v}$ and the matrix weights $Q(\boldsymbol{x}_d), P(\boldsymbol{x}_d) \in \mathbb{R}^{7\times 7}$ for the state and $R \in \mathbb{R}^{3\times 3}$ for the control are chosen positive semi-definite and positive definite respectively. Also following the design procedure employed for the problem in chapter 4, the positive-semidefinite state weights $Q(\boldsymbol{x}_d), P(\boldsymbol{x}_d) \in \mathbb{R}^{3m+7\times 3m+7}$ are designed on the linear subspace at the target $\boldsymbol{x}_d$ (i.e. the tangent space $T_{\boldsymbol{x}_d}X(\boldsymbol{h}_0)$). The projection matrix $M(\boldsymbol{x}_d)$ defined immediately below equation (5.17) (designed for this system's specific configuration manifold) is used to affect the required linear transformation.

Additionally, note that the problem (6.3) considered in the first study of this chapter employs a classical penalty on the norm of the motor control inputs $\|\boldsymbol{u}\|$. As discussed in chapter 5, this standard approach does not adequately capture the true electrical energy use of the array. Referencing the CMG motor power equations (5.10), an appropriate additive quadratic penalty for the aggregate power used by the array is given by

$$\ell_e(\boldsymbol{x}, \boldsymbol{u}) \coloneqq \frac{1}{2}\|\boldsymbol{P}_g\|^2_{R_{eg}} + \frac{1}{2}\|\boldsymbol{P}_w\|^2_{R_{ew}}, \tag{6.5}$$

where $R_{eg}, R_{ew} \in \mathbb{R}^{m\times m}$ are positive definite diagonal weight matrices. In particular, we choose the weights $R_{eg}, R_{ew} \coloneqq \rho_e \mathbb{I}_m$ to ensure that energy usage is uniformly penalized on all devices in the array. The resulting additive penalty can then be incorporated directly into the incremental cost in (6.3) (i.e. letting $\ell = \ell_0 + \ell_e$).

Finally, while (6.2) specifies the *general* problem under consideration, it should be noted that the specific problems considered in sections 6.5 and 6.6 include minor variations of this base

problem. In order to introduce a simpler variation of the problem, the initial study in section 6.5 considers the default cost functional ($\ell = \ell_0$) and omits the standard operational constraints $c(\boldsymbol{x}, \boldsymbol{u})$ for simplicity. Building on these results, the more advanced study in section 6.6 does enforce these constraints and compares the optimal solutions from omitting ($\ell = \ell_0$) and including ($\ell = \ell_0 + \ell_e$) the additional cost penalty.

6.3 Trajectory Initialization

As with the body-torque maneuver planner developed in chapter 4, PRONTO requires a suitable initial guess ξ_0 for the optimization problem (6.3) which (at least approximately) satisfies the dynamics $f(\boldsymbol{x}, \boldsymbol{u})$ and the boundary conditions $\boldsymbol{x}(0) = \boldsymbol{x}_0$ and $\boldsymbol{x}(T) = \boldsymbol{x}_d$. While the dynamics (5.12) do not directly yield a simple analytic solution like the geodesics in chapter 4, existing control laws provide an effective and simple way of obtaining such guess solutions when given a sufficiently long time horizon. For this problem in particular, we return to the accepted *Singularity Robust* (SR) control law presented in [38] and specified in section 2.4. To briefly review, this feedback solution stabilizes the classic torque dynamics (2.3) by converting the desired (planned) command torques $\boldsymbol{\tau}_r$ to the minimum norm CMG gimbal rates $\dot{\boldsymbol{\delta}} = D^{\ddagger}\boldsymbol{\tau}_r$ using the following singularity-perturbed variant of the Moore-Penrose Pseudoinverse

$$D^{\ddagger} := D^{\top}\left[DD^{\top} + \alpha_0 e^{-\det(DD^{\top})}\mathbb{I}_m\right]^{-1},$$

where $\alpha_0 \in \mathbb{R}_{>0}$ is a (small) weighting coefficient. Given the target attitude $\boldsymbol{q}_f$, angular rate $\boldsymbol{\omega}_f$, and corresponding angular torque $\dot{\boldsymbol{\omega}}_f = \boldsymbol{f}_{\omega}(\boldsymbol{x_f})$ from the target $\boldsymbol{x}_f \in X(\boldsymbol{h}_0)$ (which we assume zero for rest-to-rest mnaneuvers), the complete SR feedback law for the satellite attitude and CMG

array is then given by:

$$\boldsymbol{\tau}_{SR} = k_q \operatorname{Im}\left(O_R(\boldsymbol{q}_f^*)\,\boldsymbol{q}\right) + K_\omega(\boldsymbol{\omega} - \boldsymbol{\omega}_f) + \boldsymbol{f}_h(\boldsymbol{x}) - J\dot{\boldsymbol{\omega}}_f, \tag{6.6a}$$

$$\boldsymbol{u}_{SR,\dot{\delta}} = D_\omega^\ddagger\,\boldsymbol{\tau}_{SR}, \tag{6.6b}$$

$$\boldsymbol{u}_{SR,\ddot{\delta}} = k_\delta\left[\boldsymbol{u}_{SR,\dot{\delta}} - \boldsymbol{f}_\delta(\boldsymbol{x})\right], \tag{6.6c}$$

$$\boldsymbol{u}_{SR,g} = \boldsymbol{J}_g\left[\boldsymbol{u}_{SR,\ddot{\delta}} + A_g^\top\boldsymbol{\omega}\right] - \boldsymbol{f}_{hga}(\boldsymbol{x}), \tag{6.6d}$$

$$\boldsymbol{u}_{SR,w} = k_w(\boldsymbol{h}_{swr} - h_w\mathbf{1}_m), \tag{6.6e}$$

where $\mathbf{1}_m \in \mathbb{R}^m$ is an m-vector of 1's, $k_q, k_\delta, k_w \in \mathbb{R}_{>0}$ are scalar feedback gains for state and tracking errors, and $K_\omega \in \mathbb{R}^{3\times 3}$ is a diagonal feedback gain on the angular rate error (critically damped with diagonal elements $K_{\omega,i} = \sqrt{2k_qJ_i}$). The specific actuator Jacobian used to convert torques to CMG gimbal rates for (6.6) while also stabilizing angular rates is given by

$$D_\omega \coloneqq {}^1\!/_2\Big[A_s \operatorname{diag}\left(A_t^\top(\boldsymbol{\omega}+\boldsymbol{\omega}_f)\right) + A_t \operatorname{diag}\left(A_s^\top(\boldsymbol{\omega}+\boldsymbol{\omega}_f)\right)\Big](\boldsymbol{J}_t - \boldsymbol{J}_s) - A_t \operatorname{diag}(\boldsymbol{h}_{swr}), \tag{6.7}$$

One further consideration is necessary when applying the above feedback to generate initial solutions for the optimization problem (6.3). While the feedback law (6.6) asymptotically stabilizes $\boldsymbol{q} \to \pm\boldsymbol{q}_f$, $\boldsymbol{\omega} \to \boldsymbol{\omega}_f$, and $\boldsymbol{h}_{swr} \to h_w\mathbf{1}_m$, the array configuration $\boldsymbol{\delta}$ and momentum $\boldsymbol{h}_{ga}$ may vary freely along $X(\boldsymbol{h}_0)$. As a result, flows under (6.6) will not generally satisfy the complete boundary condition $\boldsymbol{x}(T) \approx \boldsymbol{x}_f$, producing a large initial penalty in the terminal cost function. For the purposes of rest-to-rest maneuvers however, achieving a *specific* $\boldsymbol{\delta}_f$ and $\boldsymbol{h}_{ga,f}$ is generally unnecessary as most mission objectives require only a resting, non-singular array configuration at the target attitude. Given this extra freedom, we may generate our guess solution curve from $\boldsymbol{x}_0$ to $\boldsymbol{x}_f$, then mildly adjust $\boldsymbol{\delta}_f$ and $\boldsymbol{h}_{ga,f}$ to prevent unnecessary efforts from the solver. Specifically, we generate initial guesses using the following procedure:

(1) Flow $\boldsymbol{x}(t)$ from $\boldsymbol{x}_0$ using (5.12) with the feedback (6.6),

(2) Determine $T > 0$ such that $\boldsymbol{q}(T) \approx \boldsymbol{q}_f$ and $\boldsymbol{\omega}(T) \approx \boldsymbol{\omega}_f$,

(3) Adjust $\boldsymbol{x}_f \in X(\boldsymbol{h}_0)$ such that $\boldsymbol{x}(T) \approx \boldsymbol{x}_f$.

For the satellite platform considered in this work, we found $T \geq 180\,\mathrm{s}$ to be sufficient to accommodate any rotation $< 180°$. For the final step, the resting condition $\boldsymbol{f}(\boldsymbol{x}_f, 0) = 0$ and $\boldsymbol{\omega}_f = 0$ necessarily fix $\boldsymbol{h}_{ga,f} = 0$. Thus, we need only determine a compatible (non-singular) $\boldsymbol{\delta}_f \approx \boldsymbol{\delta}(T)$ on $X(\boldsymbol{h}_0)$. For this purpose, the recursion $\boldsymbol{\delta}_f \leftarrow \boldsymbol{\delta}_f + d(\boldsymbol{x}_f, \boldsymbol{x}(T))$ where

$$\begin{aligned} d(\boldsymbol{x}_f, \boldsymbol{x}) &= \Delta^{\dagger} \left(\bar{\boldsymbol{h}}(\boldsymbol{x}) - \bar{\boldsymbol{h}}(\boldsymbol{x}_f) \right), \\ \Delta &= -A_t(\boldsymbol{\delta}_f)\, \mathrm{diag}[\boldsymbol{h}_{swr} + \boldsymbol{J}_{sw} A_s^{\top}(\boldsymbol{\delta})\boldsymbol{\omega}], \end{aligned} \tag{6.8}$$

was found to reliably produce compatible, non-singular boundary conditions $\boldsymbol{\delta}_f$ near the horizon $\boldsymbol{\delta}(T)$ produced by (6.6). Notably, this recursion only defines a local contraction around $\boldsymbol{\delta}(T)$ (iterated until the step size $d(\boldsymbol{x}_f, \boldsymbol{x})$ is sufficiently small) and may fail if $\boldsymbol{\delta}_f$ approaches a singularity. However, this solution was found to be reliable for the purposes of generating initial guess solutions for our optimization problem.

Notably, as a classic CMG control law, the standard SR law formulation uses *only* the CMG gimbal motor inputs (defined by (6.6d)) to generate attitude control torques, with the wheel motors dedicated entirely to wheel speed regulation in (6.6e) (a standard assumption in classic CMG implementations). Given that the proposed trajectory optimizer directly leverages the wheel motors for additional control torques (i.e. operation more akin to VSCMG's), some component of measured performance improvements in the optimizer will likely result from this additional control authority (and so might also be found in existing VSCMG control laws). As such, the relative performance improvements discussed in the remainder of this section are qualified against the SR law *specifically*, which was chosen for prevalence, rather than peak performance.

6.4 Experimental Design

The following two sections summarize the results of two separate studies for PRONTO solutions to (variations of) the optimization problem (6.3). In each study, we consider 10 random rest-to-rest attitude transfers, each with random initial and final attitudes $\boldsymbol{q}_0, \boldsymbol{q}_d \in \mathrm{S}^3$, resting initial and final angular rates $\boldsymbol{\omega}_0, \boldsymbol{\omega}_d = 0$, and with CMG array boundary conditions selected to

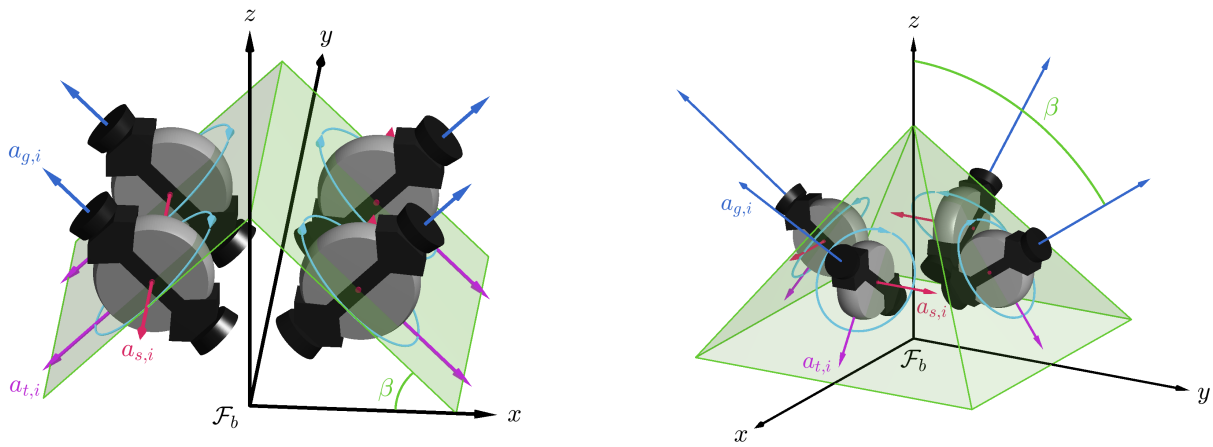

Figure 6.1: A 4-CMG array in the (a) Rooftop and (b) Pyramid configurations with inclinations of $\beta = 45°$ and $54.74°$ respectively.

be non-singular zero-momentum ($\boldsymbol{h}_0 = 0$) configurations in the feasible momentum workspace (e.g. satisfying $\boldsymbol{x}_0, \boldsymbol{x}_d \in X(\boldsymbol{h}_0)$). In addition to these random problems, we examine a typical 180° rotation about the satellite's z-axis to more closely examine specific features of the solution trajectories. The initial guess trajectory ξ_0 for each maneuver was generated using an extension of the classic SR feedback law presented above. Using aggressive feedback gains for this law, a maneuver time horizon of 180 s was found to allow sufficient convergence to $\boldsymbol{x}_d$ for all tested maneuvers. All solutions discussed in this study were computed in Matlab on an AMD Ryzen™5800x CPU using 32 MHz memory.

For our spacecraft and CMG-array models, we examine the popular rooftop and pyramid array geometries shown in Figure 6.1 with the platform inertias (in kgm^2)

$$\operatorname{diag}(J) = \begin{bmatrix} 1500 & 1500 & 2000 \end{bmatrix}, \quad J_g = 0.115,$$

$$J_{sw} = 0.075,\ J_{sg} = 0.015, \qquad J_t = 0.001,$$

and a target (and initial) CMG wheel momentum of $h_w = 25\,\mathrm{kgm^2s^{-1}}$. In each case, the base LQR cost functional (and the projection regulator used by the PRONTO solver) were constructed following (3.8) and (4.9), where the positive definite weight matrix $Q_r := M(\boldsymbol{x}_d) Q_c M(\boldsymbol{x}_d)^\top$ on the

controllable subspace was generated using the structure

$$Q_c \coloneqq \mathrm{diag}([\rho_q \mathbf{1}_4\,;\, \rho_{hswr}\mathbf{1}_m\,;\, \rho_\omega \mathbf{1}_3\,;\, \rho_\delta \mathbf{1}_m\,;\, \rho_{hga}\mathbf{1}_m]),$$

where individual state error weights are specified via the scalar weights ρ_i and $\mathbf{1}_m \in \mathbb{R}^m$ denotes a vector of 1's. For the cost functional and regulator respectively, these weights were chosen to be

$$[\rho_q, \rho_{hswr}, \rho_\omega, \rho_\delta, \rho_{hga}]_{\text{cost}} = [5, 10, 0.1, 0.01, 50],$$

$$[\rho_q, \rho_{hswr}, \rho_\omega, \rho_\delta, \rho_{hga}]_{\text{reg}} = [3 \cdot 10^4, 3, 200, 0.3, 3] \cdot 10^{-4}.$$

The control weights for $R \coloneqq \mathrm{diag}([\rho_{ug}\mathbf{1}_m; \rho_{uw}\mathbf{1}_m])$ for the cost function and regulator were likewise chosen as

$$[\rho_{ug}, \rho_{uw}]_{\text{cost}} = [1, 1], \qquad [\rho_{ug}, \rho_{uw}]_{\text{reg}} = [1, 3] \cdot 10^{-5}.$$

Finally, the control energy weight matrices for the energy penalty (6.5) were selected identically as $R_{eg} = R_{ew} \coloneqq 2\,\mathbb{I}_m$ to sufficiently and equivalently penalize energy usage for both motor types.

Regarding the operational constraints considered in study B (6.6), the exclusion cone direction $\boldsymbol{\psi}_s$ was oriented to *nearly* intersect the path of the camera vector $\boldsymbol{\psi}_c$ on the initial guess trajectory with a solar exclusion angle of $\phi_{ko} = 10°$. For the remaining operational safety constraints (2.11), we limit the maximum motor input torques to $u_{g+} = 5$ and $u_{w+} = 0.01\,\mathrm{N\,m}$ for the gimbal and wheel respectively. Finally, we limit the principle axes of the spacecraft to maximum slew rates of $\boldsymbol{\omega}_+ = [4.98, 9.95, 3.72] \cdot 10^2\,\mathrm{rad\,s^{-1}}$ respectively (roughly approximating the CMG array's spheroidal momentum envelope).

6.5 Study A: Variations with Array Geometry

In this initial study, we first examine the features and mean performance statistics for PRONTO solutions to the simplified optimization problem (6.3) obtained by *omitting* the operational constraints $c(\boldsymbol{x}, \boldsymbol{u})$ and using the (default) cost functional $\ell = \ell_0$. Specifically, we compare the optimal solutions of 11 (10 randomized and 1 model) sample problems to those of the default (SR) feedback law under for the rooftop and pyramid array geometries shown in Figure 6.1. The

Table 6.1: Mean Optimal Trajectory Statistics (w/o constraints under ℓ_0)

	Rooftop		Pyramid	
Metric	Guess	Opt.	Guess	Opt.
Computation Time [min]	NA	15.45	NA	36.47
Maneuver Cost	83.56	39.90	91.53	34.07
Maneuver Time [s]	95.70	47.93	100.35	37.39
Final Attitude Error [°]	0.83	0.06	1.68	0.10
Control Effort [N m s]	109.77	25.40	140.21	26.05
Max $\boldsymbol{u}_g$ [N m]	0.47	1.19	1.5E-4	3.9E-3
Max $\boldsymbol{u}_w$ [N m]	0.50	1.08	1.2E-4	4.0E-3
Maneuver Energy [J]	5.07	15.15	4.82	19.95

mean performance statistics for each solution method and geometry (averaged across problems) are presented in Table 6.1 below (*note that these results should only be considered for general impressions, as the variance between individual problems is not controlled in this analysis).

Examining Table 6.1, we first note that solutions to this challenging problem are not obtained easily. For the rooftop and pyramid geometries, PRONTO takes an average of 15 and 35 minutes respectively to reduce the objective cost to 47% and 42% of the original feedback solution. Notably, Matlab's `ode45` function limits the algorithm to a single CPU thread (a limitation shared by the majority of spacecraft CPU's). Interestingly, solutions for the pyramid geometry were far more computationally expensive than those of the rooftop array, indicating a higher intrinsic complexity in the effective operation of that geometry.

While the maneuver convergence time and terminal attitude error show similar reductions to that of the objective function, a more interesting effect is observed in the mean maneuver efficiency. In particular, while the total control effort (integration of $\sum_i |u_i|$) shows a reduction of 77% and 81% respectively, the optimal maneuver uses *far* more electric power (3-4x) than the initial guess. While this increased energy usage partially originates from an aggressive cost function weighting, it primarily highlights the fundamental modelling limitation of the standard model for control energy employed in the cost functional ℓ_0 chosen for this study (namely that the standard penalty on $\|\boldsymbol{u}\|$ does not capture the true electrical energy used by the array). This key observation presents the original motivation for the design of the additive penalty ℓ_e considered in the next section.

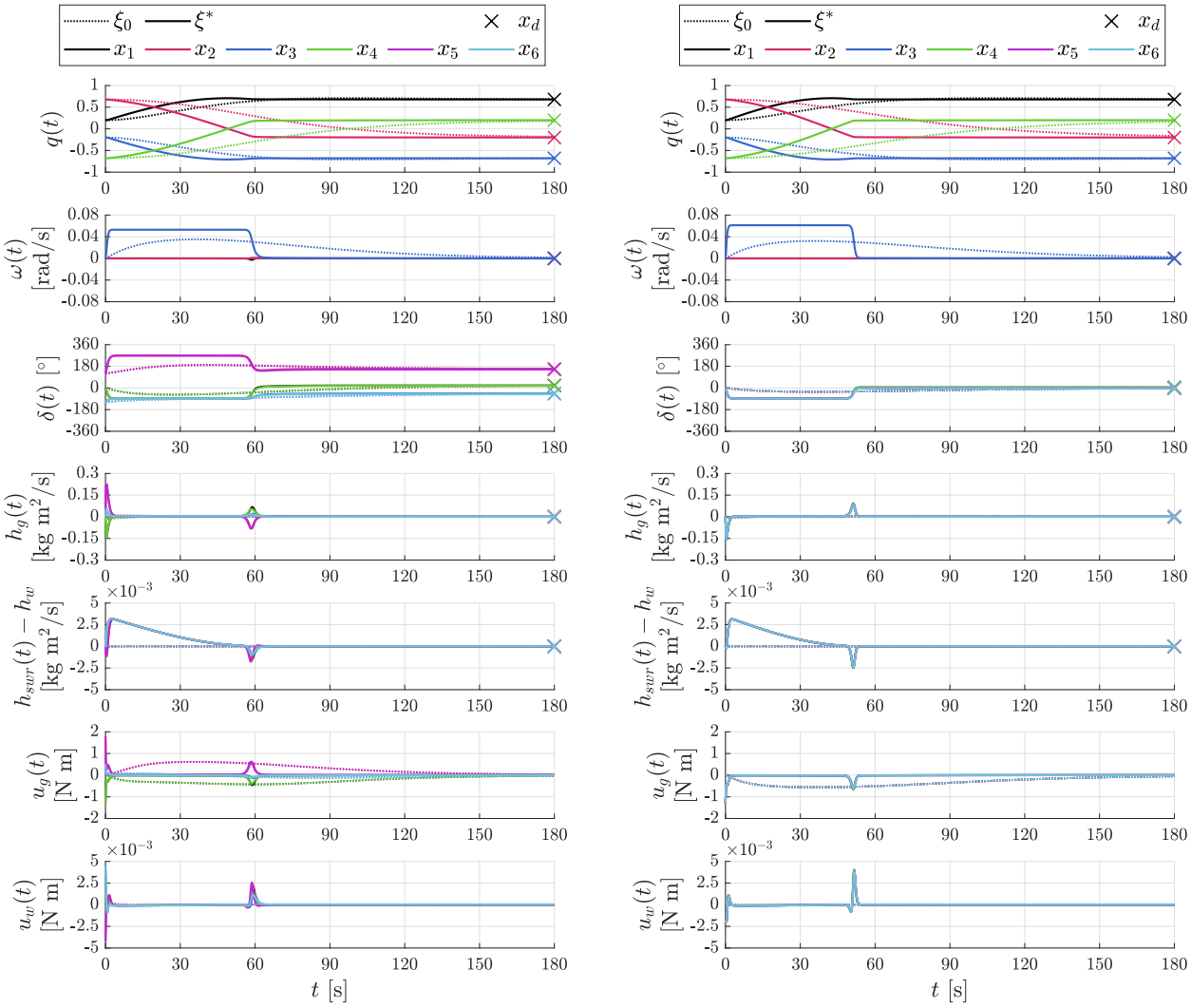

Figure 6.2: Guess (ξ_0) and unconstrained optimal (ξ^*) Trajectories for a 180° z-axis rotation of a spacecraft under the Rooftop (a) and Pyramid (b) CMG array geometries respectively.

Examining these solutions in more detail, the left and right panels of Figure 6.2 show the optimal and guess trajectories for a 180° rotation about the z-axis for the rooftop and pyramid geometries respectively. These optimal trajectories display several interesting features

(1) $\boldsymbol{\omega}(t)$ saturates in both maneuvers and geometries.

(2) Both $\boldsymbol{u}_w$ and $\boldsymbol{u}_g$ are impulsive in nature.

(3) The CMG angles δ_i display unusual coordination.

Regarding observation 1, we remind the reader that slew rate constraints are *not* considered in this simplified problem. Instead, the apparent maximum rotation rate results from the finite momentum capacity of a CMG array (the envelope of the momentum workspace of the array like that shown in Figure 2.4), which naturally enforces a maximum rotation rate along any axis. This natural property of MED's also informs upon observation 2, with each control input acting impulsively to rapidly achieve the array configuration for this maximum rotation rate. Regarding observation 3, the coordination of the pyramid array is intuitive as its symmetry with the requested rotation axis clearly promote symmetry in the actuators. However, the coordination for the rooftop geometry is far more interesting. While we might expect the CMG's to coordinate in groups with shared gimbal axes (as sides of the rooftop), they instead operate in pairs *across* the rooftop. This intriguing behavior was observed for multiple maneuvers with different rotation axes and warrants further investigation.

6.6 Study B: Solution Variations with Energy Model

Having established the potential performance gains offered by solutions to the simplified, unconstrained version of problem (6.3) in the previous section, we now consider the fully constrained problem. As before, we examine the features and performance improvements offered by solutions to (6.3) (both including and excluding the new energy penalty ℓ_e) over those produced by the accepted SR feedback. In order to isolate the performance effect produced by the CMG energy

Table 6.2: Mean Solution Performance

Solution	Guess	ξ^*	ξ_e^*
Feasible	No	Yes	Yes
Comp. Time [min]	0	58.38	50.70
Maneuver Time [s]	95.7	51.8	53.4
Final Attitude Error [°]	0.829	0.0728	0.0723
Control Effort [N m s]	109.77	33.63	34.50
Max $\boldsymbol{u}_g$ [N m]	0.47	1.23	1.22
Max $\boldsymbol{u}_w$ [N m]	1.47E-4	1.01E-3	3.29E-6
Maneuver Energy [J]	5.07	14.7	3.35

penalty, we compare three solution trajectories for each maneuver: (1) the unconstrained initial guess ξ_0 produced by an established feedback law (discussed below), (2) the constrained solution ξ^* to (6.3) using classic Linear Quadratic state and control weights in the incremental cost (e.g. $\ell = \ell_0$), and (3) the constrained solution ξ_e^* to (6.3) including the penalty (6.5) on the true CMG energy usage (e.g. $\ell = \ell_0 + \ell_e$).

Following the protocol developed in section 6.5, optimal solutions were computed using the PRONTO solver over the 11 sample maneuvers for the Rooftop array geometry. By design, this family of sample problems spans an enormous variety of different constrained maneuvers with different lengths, axes of rotation, and array configurations. To control for these differences in the performance analysis, we examine the relative performance improvement of each maneuver's optimal solutions over its initial guess (the SR feedback law). The average of these relative performance gains and their standard errors are collected in Table 6.3, with negative results indicating a performance loss. The average performance statistics and computational cost over each solution family (including the high variance between the maneuvers) are also included in Table 6.2 to provide real-world estimates for these gains.

Examining these results in detail, we first note that (as before) the aggressive cost weighting chosen for ℓ_0 improves the speed and accuracy of each optimal solution, with each reaching a neighborhood of the target in just over half the time achieved by the original feedback law (and getting nearly 10x closer to the target attitude over the full 180 s maneuver period). Additionally,

Table 6.3: Mean Sol. Performance Gain over SR-Law

Solution	ξ^*	ξ_e^*
Maneuver Time	46.1 ± 2.9%	44.3 ± 2.7%
Final Attitude Error	87.7 ± 2.1%	87.7 ± 2.2%
Control Effort	67.4 ± 3.1%	66.4 ± 3.2%
Max $\boldsymbol{u}_g$	−169 ± 18%	−170 ± 17%
Max $\boldsymbol{u}_w$	−596 ± 115%	97.7 ± 0.3%
Maneuver Energy	−207 ± 33%	35.0 ± 6.5%

the classic quadratic control penalty $\|\boldsymbol{u}\|_R^2$ ensures that both optimal solutions reduce the total motor torque applied in each maneuver. From the greatly increased motor torque maximums applied in the optimal maneuver, we can infer that the optimal strategy employs shorter, higher intensity torques to achieve the same rotation with reduced classical control effort. These results present a compelling (if somewhat unsurprising) argument of the powerful benefits of standard trajectory optimization techniques in complex high-performance systems.

At this point however, we reach a critical distinction between the two optimal solution families. While the classic LQ cost function ℓ_0 does improve the total applied motor *torque* over the maneuver, it requires nearly triple the total CMG motor power used by the standard feedback law (mirroring the result we obtained above for the unconstrained case). While one might assume that this loss is a result of the cost function weighting (trading efficiency for speed), the energy-penalized solution ξ_e^* shows differently. By properly modeling and penalizing the electric power employed by the array, this optimal strategy executes the same maneuver using 35% *less* energy than the original feedback law while *retaining* the improved performance in every other metric. Intriguingly, while the updated penalty ξ_e^* largely improves tracking of the gimbal motor's energy usage over the traditional LQ cost ℓ_0, the total energy used by the gimbal motors in the optimizers ξ^* and ξ_e^* is nearly identical. Instead, the observed efficiency improvement appears to strongly correlate with substantially lowered usage of the *wheel* motor: a feature we will examine in greater detail below. Again, it should be noted that a (potentially substantial) portion of the measured efficiency improvement of the optimizer over the SR law originates from the additional control authority

provided by the variable speed operation of the wheels. As such, one should not interpret this improvement as relative to the 'best' available VSCMG control solutions (many of which are likely not publicly available). Instead, the above performance comparison demonstrates the *limitations* imposed by the approximations leveraged in existing CMG control strategies (specifically the SR law) as well as an approximation of the remaining performance available to the fully modelled system. With this motivation in mind, our efforts are now directed to identifying the solution features which generate this performance difference.

To identify the origin of the performance improvements discussed above, we next examine the specific features of the optimal solutions ξ^* and ξ_e^*. In particular, Figure 6.3(a) compares the optimal and guess trajectories for a 180° rotation about the satellite's z-axis: a standard maneuver whose simplicity effectively highlights the differences between each strategy. Additionally, Figure 6.3(b) provides a helpful visualization of this rotation, showing the paths of the satellite's body-frame, camera orientation, and solar exclusion cone.

Comparing these solutions to the original guess, we first observe several similarities: Both solutions are feasible in all constraints, follow similar attitude paths, and employ short, rapid accelerations around the primary axis of rotation to arrive at the target more quickly than the gentler feedback law. In particular, we note that both optimal solutions immediately saturate the CMG array along the z-axis, maximizing ω_z on the array's momentum envelope and creating a 'cruise' phase for the rotation analogous to those seen in standard orbital maneuvers. Though each optimal solution differs slightly in the specific array configurations and particular transition maneuvers to and from this cruise phase, this general strategy is universal for both optimal solutions across the test cases.

As before, the key difference between ξ^* and ξ_e^* appears in the control and regulation of the CMG *wheels*. Specifically, while ξ^* follows a more conventional (if aggresive) regulatory approach during the cruise phase of the rotation (immediately regulating the wheel speeds to their standard operating values), the energy-penalized solution ξ_e^* instead allows the wheel momentum to deviate during this period, regulating only after the final deceleration. As is often the case in optimization,

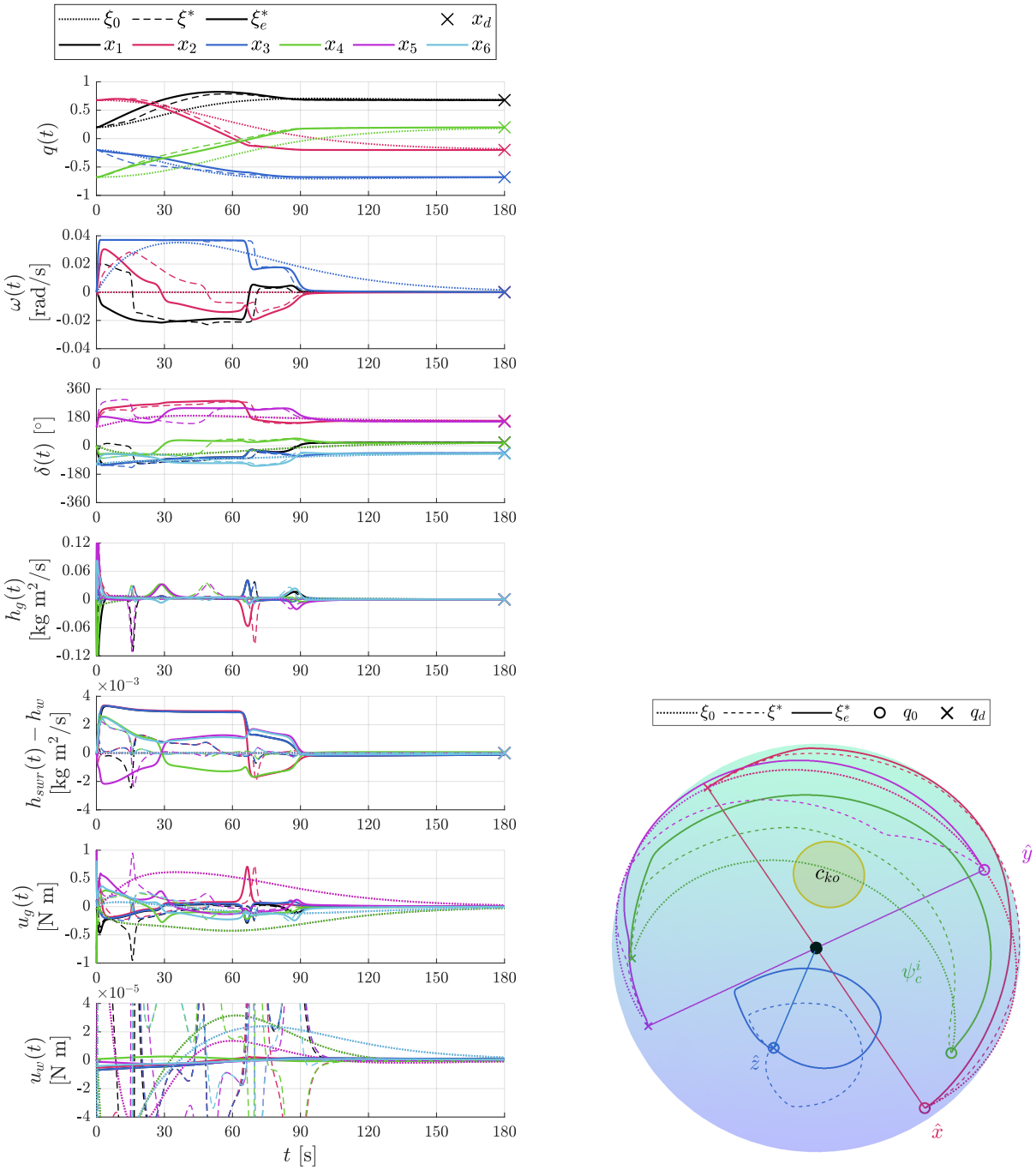

Figure 6.3: (a) Guess (ξ_0), constrained optimal (ξ^*), and constrained energy optimal (ξ_e^*) solution trajectories for a 180° z-axis rotation of the $n = 6$ Rooftop CMG geometry and (b) their projection to SO(3). The blue, red, and magenta lines track the satellite's body frame, while the green line and yellow circle indicate the satellite's camera path and solar exclusion cone.

the motivation behind this strategy is quite apparent in retrospect. Recall that the power usage $P = \tau_w \omega_w$ of a classic RW is proportional to its wheel speed ω_w. Because ω_w is regulated to be quite high in a CMG, the minor wheel speed regulation torques are substantially more expensive than the primary gimbal torques. As such, reducing regulation via the wheel motor control inputs $\boldsymbol{u}_w$ substantially lowers the array's energy usage *without* meaningfully impacting the performance of the array. As above, this behavior was observed in each of the other test problems, indicating this intermediate regulation to be a potentially widespread efficiency loss for existing feedback solutions.

The above strategy is particularly valuable for practical spacecraft design because it does *not* require optimality (an expensive property to achieve in existing spacecraft hardware). Instead, this strategy could be achieved using a traditional wheel speed regulator with variable (stabilizing) gains designed to reduce regulation during the maneuver's cruise phase. While more complex CMG array behaviors may be required to maximize the potential of this strategy (e.g. produce a distinct cruise phase and employ the deceleration action of the array to regulate the wheels), this strategy by itself could largely improve efficiency by eliminating unnecessary regulation. This powerful result demonstrates the often understated value of off-line trajectory optimization techniques, as well as the critical importance of developing accurate models for a system's dynamics and energy usage.

6.7 Conclusions

In this chapter, we formulated an optimal maneuver planner for rest-to-rest attitude transfers with CMG-driven spacecraft subject to input saturation, angular velocity, and non-convex exclusion cone constraints. To examine the true optimal performance of the CMG array and the strategies employed to achieve it, we employed comprehensive physical models for the CMG array's momentum exchange dynamics and electric power usage, avoiding the implicit behavioral restrictions introduced by prevalent approximations of CMG momentum dynamics. To formulate and solve this constrained trajectory optimization problem, we designed a locally stabilizing LQR on the system's configuration manifold, then lifted it into the ambient state space to produce suitable terminal and running LQ cost functionals. A family of random maneuver problems were then ex-

amined, with the optimal solutions presenting substantially improved performance and efficiency under the comprehensive dynamical models. In the first study, a subtle but critical shortcoming was identified in the energy penalty cost functional. This issue was then remedied using a specialized additive energy penalty term in the incremental cost functional which tracks the array's true power usage.

Upon examining the solutions under this constrained and properly energy-optimized system, two control behaviors were observed in correlation with the observed performance benefits. First, the CMG array produced rapid accelerations at the beginning and end of the maneuver, creating an intermediate and inactive 'cruise' period in the spacecraft's rotation analogous to that found in typical orbital transitions. Second, the wheel speeds of the CMG's were *not* regulated during this cruise interval, waiting instead until after the deceleration action of the gimbal motors to regulate the wheel momenta to their nominal values. In particular, this second behavior directly challenges the strategy employed by typical CMG feedback laws, achieving a 35% reduction in electric power for the same maneuver. Notably, these specific control strategies *do not require optimality*, and may potentially be implemented using minor modifications of typical CMG array feedback control laws.

Chapter 7

Summary and Conclusions

In this thesis, we formulated and solved several constrained trajectory optimization problems for constrained rest-to-rest attitude maneuvers for spacecraft driven by traditional abstract body torques and momentum-conserving Control Moment Gyroscope arrays. In the process of solving these problems, we developed new strategies for the application of our chosen trajectory optimization solver (PRONTO) to problems evolving on constrained nonlinear manifolds, resulting in substantially improved performance over other techniques (both commercially and in the literature).

After validating this general problem formulation and solver on a model trajectory optimization problem in the literature (i.e. a maneuver planner using abstract body command torques), we applied this approach to solve the (previously *open*) constrained trajectory optimization problem for rest-to-rest attitude transfers using a momentum-conserving CMG array. In particular, our formulation employed a dynamical model which preserves the array's (conservative) momentum exchange dynamics, a power model directly tracking the usage of the individual CMG motors, and typical operational safety constraints on input saturation, angular velocity, and camera exclusion cones. The optimal control strategies produced under this comprehensive formulation presented substantial improvements to mean maneuver performance and efficiency, and identified an acute shortcoming in cost functions which use the control input (rather than accurately modelled power usage) to penalize maneuver energy cost.

To address this shortcoming, we then augmented the above trajectory optimization problem

with a specialized energy penalty in the cost function to accurately limit the electric power used by the array. Unlike existing approaches, this formulation enabled optimal solutions to display their full range of (potentially non-intuitive) behaviors while reducing the true total electric power used over the maneuver. Finding these solutions to be substantially more performant *and* efficient than existing feedback solutions, we primarily attributed this performance increase to two nonstandard control strategies observed across the solution family: (1) a model torque profile for the CMG array analogous to the initial and final burns used in orbital transitions and (2) variable CMG wheel speed regulation which is largely inactive during the maneuver's 'cruise' phase. Finally, we suggested potential adaptations of existing feedback control solutions which incorporate the above behaviors, achieving the corresponding performance improvements without the computational burden of strict optimality. With this final step, we achieved our original research objective and demonstrate the immense value in off-line solutions of traditionally challenging trajectory optimization problems for high performance systems.

Bibliography

[1] A. Pedro Aguiar, Florian A. Bayer, John Hauser, Andreas J. Häusler, Giuseppe Notarstefano, Antonio M. Pascoal, Alessandro Rucco, and Alessandro Saccon. Constrained optimal motion planning for autonomous vehicles using PRONTO. In Thor I. Fossen, Kristin Y. Pettersen, and Henk Nijmeijer, editors, Lecture Notes in Control and Information Sciences, volume 474 of Lecture Notes in Control and Information Sciences, pages 207–226. Springer International Publishing, Cham, 2017.

[2] Brian D O Anderson and Moore. Optimal Control: Linear Quadratic Methods. Prentice-Hall International, 1989.

[3] Arunava Banerjee, Syed Muhammad Amrr, and M. Nabi. A pseudospectral method based robust-optimal attitude control strategy for spacecraft. Advances in Space Research, 64(9):1688–1700, 2019.

[4] Nazareth S. Bedrossian, Joseph Paradise, Edward V. Bergmann, and Derek Rowell. Redundant single gimbal control moment gyroscope singularity analysis. Journal of Guidance, Control, and Dynamics, 13(6):1096–1101, 1990.

[5] David A. Benson, Geoffrey T. Huntington, Tom P. Thorvaldsen, and Anil V. Rao. Direct trajectory optimization and costate estimation via an orthogonal collocation method. Journal of Guidance, Control, and Dynamics, 29(6):1435–1440, 2006.

[6] John T. Betts. Survey of numerical methods for trajectory optimization. Journal of Guidance, Control, and Dynamics, 21(2):193–207, 1998.

[7] Sanjay P Bhat, Tata Consultancy, and Services Limited. Small-time local controllability and stabilizability of spacecraft attitude dynamics under CMG actuation. SIAM Journal of Control and Optimization, 52(2):797–820, 2015.

[8] Robin Blenden and Hanspeter Schaub. Regenerative power-optimal reaction wheel attitude control. Journal of Guidance, Control, and Dynamics, 35(4):1208–1217, 2012.

[9] George A. Boyarko, Marcello Romano, and Oleg A. Yakimenko. Time-optimal reorientation of a spacecraft using an inverse dynamics optimization method. Journal of Guidance, Control, and Dynamics, 34(4):1197–1208, 2011.

[10] R Calaon and H Schaub. Constrained Attitude Maneuvering Via Modified Rodrigues Parameters - Based Motion Planning Algorithms. AAS/AIAA Astrodynamics Specialist Conference, pages 1–20, 2021.

[11] Anton H.J. De Ruiter, Christopher J. Damaren, and James R. Forbes. Spacecraft dynamics and control: an introduction. Wiley, 2013.

[12] Thomas L. Dearing, John Hauser, Xudong Chen, Marco M. Nicotra, and Christopher Petersen. Efficient trajectory optimization for constrained spacecraft attitude maneuvers. Journal of Guidance, Control, and Dynamics, 45(4):1–13, 2021.

[13] Thomas L. Dearing, John Hauser, Xudong Chen, Christopher D. Petersen, and Marco M. Nicotra. (Under Review) Energy-Optimal Attitude Control Strategies with Control Moment Gyroscopes. Journal of Guidance, Control, and Dynamics, 2023.

[14] Thomas L Dearing, John Hauser, Christopher Petersen, Marco M Nicotra, and Xudong Chen. Attitude Trajectory Optimization and Momentum Conservation with Control Moment Gyroscopes. In to appear in: IFAC World Congress, 2023.

[15] Dueri Dueri, Frederick Leve, and Behcet Aclkmese. Minimum Error Dissipative Power Reduction Control Allocation via Lexicographic Convex Optimization for Momentum Control Systems. IEEE Transactions on Control Systems Technology, 24(2):678–686, 2016.

[16] Kevin A. Ford. Reorientations of Flexible Spacecraft Using Momentum Exchange Devices. PhD thesis, Air Force Institute of Technology, 1997.

[17] Kevin A. Ford and Christopher D. Hall. Singular direction avoidance steering for control-moment gyros. Journal of Guidance, Control, and Dynamics, 23(4):648–656, 2000.

[18] Rohit Gupta, Uroš V. Kalabić, Stefano Di Cairano, Anthony M. Block, and Ilya V. Kolmanovsky. Constrained spacecraft attitude control on SO(3) using fast nonlinear model predictive control. In Proceedings of the American Control Conference, pages 2980–2986. IEEE, 2015.

[19] Hari B. Hablani. Attitude commands avoiding bright objects and maintaining communication with ground station. Journal of Guidance, Control, and Dynamics, 22(6):759–767, 1999.

[20] John Hauser. A projection operator approach to the optimization of trajectory functionals. IFAC Proceedings Volumes, 35(1):377–382, 2002.

[21] John Hauser. On the Computation of Optimal State Transfers with application to the Control of Quantum Spin Systems. Proceedings of the American Control Conference, 3:2169–2174, 2003.

[22] John Hauser and D.G. Meyer. The trajectory manifold of a nonlinear control system. In Proceedings of the IEEE Conference on Decision and Control, volume 1, pages 1034–1039. IEEE, 1998.

[23] John Hauser and Alessandro Saccon. A barrier function method for the optimization of trajectory functionals with constraints. In Proceedings of the IEEE Conference on Decision and Control, pages 864–869. IEEE, 2006.

[24] Peter C Hughes. Spacecraft attitude dynamics. Courier Corporation, 2012.

[25] Matthew P. Kelly. Transcription methods for trajectory optimization: A beginners tutorial. arXiv, pages 1–14, 2017.

[26] Yoonsoo Kim and Mehran Mesbahi. Quadratically constrained attitude control via semidefinite programming. IEEE Transactions on Automatic Control, 49(5):731–735, 2004.

[27] Bjørn Andreas Kristiansen, Jan Tommy Gravdahl, and Tor Arne Johansen. Energy optimal attitude control for a solar-powered spacecraft. European Journal of Control, 62:192–197, 2021.

[28] Raymond Kristiansen and David Hagen. Modelling of actuator dynamics for spacecraft attitude control. Journal of Guidance, Control, and Dynamics, 32(3):1022–1025, 2009.

[29] Mark D Kuhns and Armando A Rodriguez. Preferred trajectory tracking steering law for spacecraft with redundant CMGs. In Proceedings of the American Control Conference, volume 5, pages 3111–3115, 1995.

[30] Haruhisa Kurokawa. Survey of theory and steering laws of single-gimbal control moment gyros. Journal of Guidance, Control, and Dynamics, 30(5):1331–1340, 2007.

[31] Wiley J. Larson and James R. Wertz. Space mission analysis and design. Microcosm Press, Portland OR, 1999.

[32] Dae Young Lee, Rohit Gupta, Uroš V. Kalabić, Stefano Di Cairano, Anthony M. Bloch, James W. Cutler, and Ilya V. Kolmanovsky. Geometric mechanics based nonlinear model predictive spacecraft attitude control with reaction wheels. Journal of Guidance, Control, and Dynamics, 40(2):309–319, 2017.

[33] Frederick A. Leve, Brian J. Hamilton, and Mason A. Peck. Spacecraft momentum control systems. Springer, 2015.

[34] Daniel P. Lubey and Hanspeter Schaub. Instantaneous quadratic power-optimal attitude tracking with N control moment gyroscopes. Journal of Guidance, Control, and Dynamics, 40(3):701–708, 2017.

[35] Christopher G. Mayhew, Ricardo G. Sanfelice, and Andrew R. Teel. On quaternion-based attitude control and the unwinding phenomenon. In Proceedings of the American Control Conference, pages 299–304, 2011.

[36] Marco M. Nicotra, Dominic Liao-Mcpherson, Laurent Burlion, and Ilya V. Kolmanovsky. Spacecraft attitude control with nonconvex constraints: an explicit reference governor approach. IEEE Transactions on Automatic Control, 65(8):3677–3684, 2020.

[37] Jorge Nocedal and Stephen J Wright. Numerical optimization. Springer, second edition, 2006.

[38] H. S. Oh and S. R. Vadali. Feedback control and steering laws for spacecraft using single gimbal control moment gyros. Journal of the Astronautical Sciences, 39(2):183–203, 1991.

[39] Michael A. Patterson and Anil V. Rao. GPOPS-II: A MATLAB software for solving multiple-phase optimal control problems using hp–adaptive Gaussian quadrature collocation methods and sparse nonlinear programming. ACM Transactions on Mathematical Software, 41(1):1–37, oct 2014.

[40] I. Michael Ross and Mark Karpenko. A review of pseudospectral optimal control: From theory to flight. Annual Reviews in Control, 36(2):182–197, 2012.

[41] Alessandro Saccon, John Hauser, and A. Pedro Aguiar. Optimal control on Lie groups: The projection operator approach. IEEE Transactions on Automatic Control, 58(9):2230–2245, 2013.

[42] Takahiro Sasaki, John Alcorn, Hanspeter Schaub, and Takashi Shimomura. Convex optimization for power tracking of double-gimbal variable-speed control moment gyroscopes. Journal of Spacecraft and Rockets, 55(3):541–551, 2018.

[43] Hanspeter Schaub and John L. Junkins. Singularity avoidance using null motion and variable-speed control moment gyros. Journal of Guidance, Control, and Dynamics, 23(1):11–16, 2000.

[44] Hanspeter Schaub and Vaios J. Lappas. Redundant reaction wheel torque distribution yielding instantaneous L 2 power-optimal attitude control. Journal of Guidance, Control, and Dynamics, 32(4):1269–1276, 2009.

[45] Hanspeter Schaub, Srinivas R. Vadali, and John L. Junkins. Feedback control law for variable speed control moment gyros. Advances in the Astronautical Sciences, 99(1):581–600, 1998.

[46] Arend L Schwab. Quaternions, finite rotation and Euler parameters. Technical report, Delft University of Technology, Laboratory for Engineering Mechanics, 2002.

[47] Stanley W. Shepperd. Quaternion from rotation matrix. Journal of Guidance, Control, and Dynamics, 1(3):223–224, may 1978.

[48] Ken Shoemake. Animating Rotation with Quaternion Curves. In Proceedings of the 12th annual conference on computer graphics and interactive techniques, pages 245–254, 1985.

[49] Panagiotis Tsiotras. Stabilization and optimality results for the attitude control problem. Journal of Guidance, Control, and Dynamics, 19(4):772–779, 1996.

[50] Yulin Wang, Haichao Hong, and Shengjing Tang. Geometric control with model predictive static programming on SO(3). Acta Astronautica, 159(August 2018):471–479, 2019.

[51] Zhenbo Wang and Michael J. Grant. Constrained trajectory optimization for planetary entry via sequential convex programming. Journal of Guidance, Control, and Dynamics, 40(10):2603–2615, 2017.

[52] Zhuo Wang, Rui Xu, Shengying Zhu, Huiping Jiang, Zhaoyu Li, Zixuan Liang, and Da Luo. Integration planning of gimbal angle and attitude motion for zero propellant maneuver under attitude and control moment gyroscope constraints. Acta Astronautica, 172:123–133, 2020.

[53] Avishai Weiss, Frederick Leve, Morgan Baldwin, James R. Forbes, and Ilya Kolmanovsky. Spacecraft constrained attitude control using positively invariant constraint admissible sets on SO(3) $\times$ R3. Proceedings of the American Control Conference, pages 4955–4960, 2014.

[54] Changqing Wu, Rui Xu, Shengying Zhu, and Pingyuan Cui. Time-optimal spacecraft attitude maneuver path planning under boundary and pointing constraints. Acta Astronautica, 137(April):128–137, 2017.

[55] Hyungjoo Yoon and Panagiotis Tsiotras. Spacecraft adaptive attitude and power tracking with variable speed control moment gyroscopes. Journal of Guidance, Control, and Dynamics, 25(6):1081–1090, 2002.

[56] Costantinos Zagaris, Hyeongjun Park, Josep Virgili-Llop, Richard Zappulla, Marcello Romano, and Ilya Kolmanovsky. Model predictive control of spacecraft relative motion with convexified keep-out-zone constraints. Journal of Guidance, Control, and Dynamics, 41(9):2051–2059, 2018.

[57] Jiamin Zhu, Emmanuel Trélat, and Max Cerf. Geometric optimal control and applications to aerospace. Pacific Journal of Mathematics for Industry, 9(8):1–41, 2017.

Appendix A

Traditional Newton Descent Methods

The trajectory optimization technique discussed in this work is fundamentally an adaptation of the conventional Newton descent method developed for unconstrained optimization problems. For example, consider the following optimization problem on $\mathbb{R}^n$

$$\min_{\boldsymbol{\xi} \in \mathbb{R}^n} h(\boldsymbol{\xi}), \tag{A.1}$$

in which we desire the optimizer $\boldsymbol{\xi}^*$ which minimizes the objective functional $h(\boldsymbol{\xi}) \in C^2(\mathbb{R}^n, \mathbb{R})$. The classic Newton method is a direct method which approaches local minimizers of (A.1) from an initial guess $\boldsymbol{\xi}_0$ via the recursion:

$$\boldsymbol{\xi}_{k+1} = \boldsymbol{\xi}_k + \boldsymbol{\zeta}_k, \tag{A.2}$$

where $\boldsymbol{\zeta}_k \in \mathbb{R}^n$ is the optimal *descent direction* for h at $\boldsymbol{\xi}_k$ determined by examining the 2nd-order Taylor expansion of h around $\boldsymbol{\xi}_k$:

$$h(\boldsymbol{\xi}_k + \boldsymbol{\zeta}) \approx h(\boldsymbol{\xi}_k) + \nabla h(\boldsymbol{\xi}_k)^\top \boldsymbol{\zeta} + \frac{1}{2} \boldsymbol{\zeta}^\top \nabla^2 h(\boldsymbol{\xi}_k) \boldsymbol{\zeta}.$$

Observing this Taylor expansion, it is clear that the best improvement obtained by stepping in the direction ζ occurs when the difference $(h(\boldsymbol{\xi}_k + \boldsymbol{\zeta}) - h(\boldsymbol{\xi}_k))$ is minimized. As such, an optimal (to 2nd-order) descent direction along h is the solution to the problem

$$\boldsymbol{\zeta}_k = \operatorname*{argmin}_{\boldsymbol{\zeta} \in \mathbb{R}^n} \nabla h(\boldsymbol{\xi}_k)^\top \boldsymbol{\zeta} + \frac{1}{2} \boldsymbol{\zeta}^\top \nabla^2 h(\boldsymbol{\xi}_k) \boldsymbol{\zeta}, \tag{A.3}$$

which can be solved for the unique minimizer

$$\boldsymbol{\zeta}_k = -\nabla^2 h(\boldsymbol{\xi}_k)^{-1} \nabla h(\boldsymbol{\xi}_k),$$

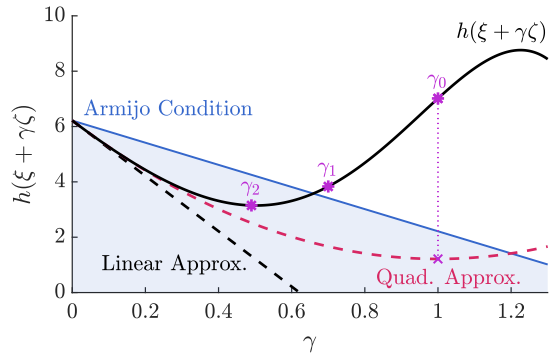

Figure A.1: Illustration of Armijo backtracking line search.

if and only if the hessian $\nabla^2 h(\boldsymbol{\xi}_k) > 0$. In the case that a local minimizer ξ^* of the C^2 function $h(\cdot)$ has a positive definite hessian, that minimizer will be an isolated local minimizer and the sequence $\|\xi_k - \xi^*\|$ produced by (A.2) converges to zero quadratically for initial points ξ_0 sufficiently close to ξ^*. Below, we introduce two compatible adjustments to the classic Newton method which are implemented in PRONTO.

A.1 Dampened Newton Method

One limitation of the classic Newton method arises from the fixed step size $\boldsymbol{\zeta}_k$ indicated by γ_0 in Figure A.1. In this example, the default step magnitude is large enough that the higher order nonlinear terms of h dominate the quadratic approximation, causing the computed step to overshoot and actually *increase* the cost. This issue can be addressed by using a variable step size $\gamma_k \in (0, 1]$ in the recursion (A.2)

$$\boldsymbol{\xi}_{k+1} = \boldsymbol{\xi}_k + \boldsymbol{\zeta}_k \gamma_k,$$

by enforcing an explicit descent *condition* on the iterates such as the linear descent enforced by the Armijo rule:

$$h(\boldsymbol{\xi}_k + \boldsymbol{\zeta}_k \gamma_k) \leq h(\boldsymbol{\xi}_k) + \alpha \nabla h(\boldsymbol{\xi}_k)\, \boldsymbol{\zeta}_k, \tag{A.4}$$

where $\alpha \in (0, 1/2)$ ensures linear descent (or better) at each iterate. A step size γ_k satisfying (A.4) can be determined rapidly using a backtracking line search via the recursion

$$\gamma_{k_{j+1}} = \beta \gamma_{k_j},$$

where $\gamma_{k_0} = 1$ and $\beta \in (0, 1)$. This recursion is illustrated in Figure A.1 (using the typical values $\alpha = 0.4$ and $\beta = 0.7$) and is repeated until (A.4) is satisfied.

A.2 Quasi-Newton Method

Another limitation of the classic Newton method arises when the local approximation of h is not positive definite ($\nabla^2 h(\boldsymbol{\xi}_k) \not> 0$), causing the solution of (A.3) for the descent direction to be unbounded. The Quasi-Newton method addresses this issue by approximating (A.3) with

$$\boldsymbol{\zeta}_k = \underset{\boldsymbol{\zeta} \in \mathbb{R}^n}{\operatorname{argmin}} \nabla h(\boldsymbol{\xi}_k)\, \boldsymbol{\zeta} + \frac{1}{2} \boldsymbol{\zeta}^\top H_k\, \boldsymbol{\zeta}, \tag{A.5}$$

where $H_k > 0$ is some positive definite matrix. Since the choice $H_k = \mathbb{I}$ leads to the gradient descent solution $\boldsymbol{\zeta}_k = -\nabla h(\boldsymbol{\xi}_k)$ (which has a linear convergence rate), we note that this method may achieve a superlinear convergence rate when $H_k > 0$ is a suitable approximation of $\nabla^2 h(\boldsymbol{\xi}_k)$. For a more thorough discussion of the Newton method and its variations in finite dimensions, we refer the reader to [37].

Appendix B

Lyapunov Stability of Body-Torque Dynamics

In this appendix, we briefly establish the local stability of the body-torque attitude dynamics (2.3) using Lyapunov methods and employing notations and definitions from 2 and Section 3.5. To study the stability of the satellite under linear feedback, we will make use of a number of inequalities regarding distances in the state manifold $\mathrm{S}^3 \times \mathbb{R}^3$. First, referring to Figure B.1, it is clear from geometry that

$$\frac{1}{\sqrt{2}}\|\boldsymbol{q} - \boldsymbol{q}_d\| \leq \|Z(\boldsymbol{q}_d)^\top(\boldsymbol{q} - \boldsymbol{q}_d)\| \leq \|\boldsymbol{q} - \boldsymbol{q}_d\|,$$

for all $\boldsymbol{q} \in \mathrm{S}^3$ with $\|\boldsymbol{q} - \boldsymbol{q}_d\| < \sqrt{2}$ (i.e., on the open hemisphere with $\boldsymbol{q}_d^T\boldsymbol{q} > 0$). Recalling that $\boldsymbol{\omega}_d = 0$ for rest to rest problems and that, by definition,

$$\|M(\boldsymbol{q}_d)(\boldsymbol{x} - \boldsymbol{x}_d)\|^2 = \|Z(\boldsymbol{q}_d)^\top(\boldsymbol{q} - \boldsymbol{q}_d)\|^2 + \|\boldsymbol{\omega}\|^2,$$

we have the bounds

$$\frac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}_d\|^2 \leq \|M(\boldsymbol{q}_d)(\boldsymbol{x} - \boldsymbol{x}_d)\|^2 \leq \|\boldsymbol{x} - \boldsymbol{x}_d\|^2,$$

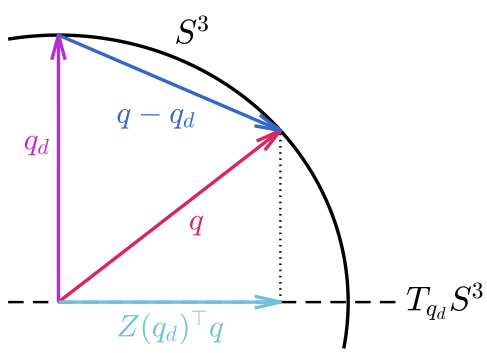

Figure B.1: Planar slice of S^3 containing q_d and q together with $Z(q_d)^\top q$.

for $\|\boldsymbol{x}-\boldsymbol{x}_d\|^2 < \sqrt{2}$. Thus, for any symmetric matrix $S > 0$, it is clear that

$$\frac{1}{2}\lambda_{\min}(S)\,\|\boldsymbol{x}-\boldsymbol{x}_d\|^2 \le \|M(\boldsymbol{q}_d)(\boldsymbol{x}-\boldsymbol{x}_d)\|_S^2 \le \lambda_{\max}(S)\|\boldsymbol{x}-\boldsymbol{x}_d\|^2,$$

holds on the same set.

Referring to the default dynamics (2.3) and the alternate representation (3.10) and noting that the operator $q \mapsto Z(q)$ is linear, we see that the dynamics of the satellite satisfies the following

$$\begin{bmatrix} \dot{\boldsymbol{q}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{2}Z(\boldsymbol{q}_d) \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{q}-\boldsymbol{q}_d \\ \boldsymbol{\omega} \end{bmatrix} + \begin{bmatrix} 0 \\ J^{-1} \end{bmatrix} \boldsymbol{u} + \begin{bmatrix} \frac{1}{2}Z(\boldsymbol{q}-\boldsymbol{q}_d)\,\boldsymbol{\omega} \\ -J^{-1}\widehat{\boldsymbol{\omega}}\,J\,\boldsymbol{\omega} \end{bmatrix} \tag{B.1}$$

or, more compactly

$$\dot{\boldsymbol{x}} = A(\boldsymbol{q}_d)\,(\boldsymbol{x}-\boldsymbol{x}_d) + B\,\boldsymbol{u} + f_2(\boldsymbol{x}-\boldsymbol{x}_d),$$

where the nonlinear term $f_2(\cdot)$ is quadratic with $\|f_2(\boldsymbol{x}-\boldsymbol{x}_d)\| \le c\,\|\boldsymbol{x}-\boldsymbol{x}_d\|^2$ for some $c > 0$ and all $\boldsymbol{x}, \boldsymbol{x}_d \in \mathrm{S}^3 \times \mathbb{R}^3$. Using the LQR feedback developed in section 3.5, we have

$$\dot{\boldsymbol{x}} = (A(\boldsymbol{q}_d) - B\,K(\boldsymbol{q}_d))\,(\boldsymbol{x}-\boldsymbol{x}_d) + f_2(\boldsymbol{x}-\boldsymbol{x}_d)\,. \tag{B.2}$$

Combining these results, we can now prove local exponential stability via our Lyapunov function. Specifically, the quadratic Lyapunov function $V(\boldsymbol{x}-\boldsymbol{x}_d) = \frac{1}{2}\|M(\boldsymbol{q}_d)(\boldsymbol{x}-\boldsymbol{x}_d)\|_{P_s}^2$ is positive definite (and decrescent)

$$\frac{1}{4}\lambda_{\min}(P_s)\,\|\boldsymbol{x}-\boldsymbol{x}_d\|^2 \le V(\boldsymbol{x}-\boldsymbol{x}_d) \le \frac{1}{2}\lambda_{\max}(P_s)\|\boldsymbol{x}-\boldsymbol{x}_d\|^2,$$

on $\|\boldsymbol{x}-\boldsymbol{x}_d\| < \sqrt{2}$. Finally, a straightforward calculation shows that

$$\begin{aligned}
\dot{V}(\boldsymbol{x}-\boldsymbol{x}_d) &= -\frac{1}{2}\|M(\boldsymbol{q}_d)(\boldsymbol{x}-\boldsymbol{x}_d)\|_{Q_c}^2 + (\boldsymbol{x}-\boldsymbol{x}_d)^\top P(\boldsymbol{q}_d)\,f_2(\boldsymbol{x}-\boldsymbol{x}_d), \\
&\le -\frac{1}{4}\lambda_{\min}(Q_c)\,\|\boldsymbol{x}-\boldsymbol{x}_d\|^2 + c\,\lambda_{\max}(P_s)\,\|\boldsymbol{x}-\boldsymbol{x}_d\|^3, \\
&\le -\frac{1}{8}\lambda_{\min}(Q_c)\,\|\boldsymbol{x}-\boldsymbol{x}_d\|^2, \\
&\le -\frac{1}{4}\frac{\lambda_{\min}(Q_c)}{\lambda_{\max}(P_s)}\,V(\boldsymbol{x}-\boldsymbol{x}_d),
\end{aligned}$$

for $\|\boldsymbol{x}-\boldsymbol{x}_d\| < \epsilon$ where $\epsilon > 0$ is the smaller of $\sqrt{2}$ and $\lambda_{\min}(Q_c)/(8\,c\,\lambda_{\max}(P_s))$. It follows that the equilibrium point $\boldsymbol{x} = \boldsymbol{x}_d$ is locally exponentially stable for the closed loop dynamics (B.2).

Appendix C

Derivation of CMG Momentum Dynamics

In this appendix, we examine the specific coordinate transformation used to obtain the dynamics (5.12) from established results. Using both Newtonian and Lagrangian methods, conservative body-frame momentum exchange dynamics for a rigid CMG driven satellite were originally derived by [16] to be

$$\dot{\boldsymbol{q}} = {}^1\!/_2\, O_L(\boldsymbol{q})\, \widetilde{\boldsymbol{\omega}}, \tag{C.1a}$$

$$\dot{\boldsymbol{h}} = \widehat{\boldsymbol{h}}\bar{\boldsymbol{\omega}} + \boldsymbol{\tau}_e, \tag{C.1b}$$

$$\dot{\boldsymbol{\delta}} = \boldsymbol{J}_g^{-1}\boldsymbol{h}_{ga} - A_g^\top \bar{\boldsymbol{\omega}}, \tag{C.1c}$$

$$\dot{\boldsymbol{h}}_{ga} = \operatorname{diag}\left(A_t^\top \bar{\boldsymbol{\omega}}\right)\left[(\boldsymbol{J}_t - \boldsymbol{J}_{sg})A_s^\top \bar{\boldsymbol{\omega}} - \boldsymbol{h}_{swa}\right] + \boldsymbol{u}_g, \tag{C.1d}$$

$$\dot{\boldsymbol{h}}_{swa} = \boldsymbol{u}_w, \tag{C.1e}$$

where $\boldsymbol{\tau}_e$ specifies external torques on the satellite body and $\boldsymbol{u}_w$, $\boldsymbol{u}_g \in \mathbb{R}^m$ collect the control inputs for the CMG wheel and gimbal motors respectively. However, while these dynamics are physical, the default coordinate choice used in their derivation is not ideal for the purposes of our optimization problem. In particular, the body-frame angular rate $\boldsymbol{\omega}$ of the satellite and the (classically regulated) relative CMG wheel momentum $\boldsymbol{h}_{swr}$ are preferable choices to $\boldsymbol{h}$ and $\boldsymbol{h}_{swa}$ respectively when specifying desirable boundary conditions and path constraints. To accomplish this change of coordinates, we first differentiate the definition (5.5b) and use (C.1e) to determine

the dynamics of $\boldsymbol{h}_{swr}$ as follows:

$$\begin{aligned}\dot{\boldsymbol{h}}_{swr} &= \boldsymbol{J}_{sw}\left[\mathrm{diag}(\dot{\boldsymbol{\delta}})A_t^\top\boldsymbol{\omega} - A_s^\top\dot{\boldsymbol{\omega}}\right] + \dot{\boldsymbol{h}}_{swa} \\ &= \boldsymbol{J}_{sw}\left[\mathrm{diag}(A_t^\top\boldsymbol{\omega})\dot{\boldsymbol{\delta}} - A_s^\top\dot{\boldsymbol{\omega}}\right] + \boldsymbol{u}_w.\end{aligned} \tag{C.2}$$

Following a similar approach to determine the dynamics of $\boldsymbol{\omega}$, we first differentiate the definition (5.7b) to yield

$$J_{st}\dot{\boldsymbol{\omega}} = \dot{\boldsymbol{h}} - D\dot{\boldsymbol{\delta}} - A_s\dot{\boldsymbol{h}}_{swr} - A_g\dot{\boldsymbol{h}}_{ga}. \tag{C.3}$$

Unfortunately, using the above expression above in conjunction with (C.2) produces an implicitly defined function. Thankfully, we can address this issue by substituting (C.2) into (C.3) and then simplifying as follows

$$\begin{aligned}J_{st}\dot{\boldsymbol{\omega}} &= \dot{\boldsymbol{h}} - D\dot{\boldsymbol{\delta}} - A_s\dot{\boldsymbol{h}}_{swr} - A_g\dot{\boldsymbol{h}}_{ga}, \\ J_{st}\dot{\boldsymbol{\omega}} &= \dot{\boldsymbol{h}} - D\dot{\boldsymbol{\delta}} - A_s\left(\boldsymbol{J}_{sw}\left[\mathrm{diag}(A_t^\top\boldsymbol{\omega})\dot{\boldsymbol{\delta}} - A_s^\top\dot{\boldsymbol{\omega}}\right] + \boldsymbol{u}_w\right) - A_g\dot{\boldsymbol{h}}_{ga}, \\ \left(J_{st} - A_s\boldsymbol{J}_{sw}A_s^\top\right)\dot{\boldsymbol{\omega}} &= \dot{\boldsymbol{h}} - \left(D + \boldsymbol{J}_{sw}\mathrm{diag}(A_t^\top\boldsymbol{\omega})\right)\dot{\boldsymbol{\delta}} - A_g\dot{\boldsymbol{h}}_{ga} - A_s\boldsymbol{u}_w, \\ J_{st,a}\dot{\boldsymbol{\omega}} &= \dot{\boldsymbol{h}} - D_a\dot{\boldsymbol{\delta}} - A_s\boldsymbol{u}_w - A_g\dot{\boldsymbol{h}}_{ga},\end{aligned} \tag{C.4}$$

which motivates the following definitions in the reduced expression

$$\begin{aligned}J_{st,a} &= J_{st} - A_s\boldsymbol{J}_{sw}A_s^\top, \\ D_a &= D + \boldsymbol{J}_{sw}\mathrm{diag}(A_t^\top\boldsymbol{\omega}).\end{aligned} \tag{C.5}$$

We then complete the coordinate transformation by substituting (5.5b) into (C.1d) as follows

$$\begin{aligned}\dot{\boldsymbol{h}}_{ga} &= \mathrm{diag}\left(A_t^\top\bar{\boldsymbol{\omega}}\right)\left[(\boldsymbol{J}_t - \boldsymbol{J}_{sg})A_s^\top\bar{\boldsymbol{\omega}} - \boldsymbol{h}_{swa}\right] + \boldsymbol{u}_g \\ &= \mathrm{diag}\left(A_t^\top\bar{\boldsymbol{\omega}}\right)\left[(\boldsymbol{J}_t - \boldsymbol{J}_s)A_s^\top\bar{\boldsymbol{\omega}} - \boldsymbol{h}_{swr}\right] + \boldsymbol{u}_g.\end{aligned} \tag{C.6}$$

Combining (C.1a), (C.1c), (C.2), (C.4), and (C.6) and ordering by computational dependence completes the dynamics (5.12).