A Study of the Successive Convexification Optimal Control Method for the Powered Descent Guidance Problem

by

Pádraig S. Lysandrou

B.S. ECE, Cornell University, 2018

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Masters of Science

Department of Aerospace Engineering Sciences

2019

This thesis entitled:
A Study of the Successive Convexification Optimal Control Method for the Powered Descent Guidance Problem
written by Pádraig S. Lysandrou
has been approved for the Department of Aerospace Engineering Sciences

_____
Prof. Robert D. Braun

_____
Dr. Jay McMahon

_____
Dr. Hanspeter Schaub

Date _____

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

Lysandrou, Pádraig S. (M.S., Aerospace Engineering)

A Study of the Successive Convexification Optimal Control Method for the Powered Descent Guidance Problem

Thesis directed by Prof. Robert D. Braun

add intro fluff kind of stff here

This thesis focuses on the implementation and development of a 6 degree-of-freedom (DoF) guidance algorithm that solves the powered descent guidance (PDG) problem. I present a modified version of the Szmuk-Acikmese 6-DoF algorithm presented in [9]. I present a similar algorithm utilizing Modified Rodrigues Parameters (MRPs) as the attitude formalism. This simplifies the formation to three parameters, from four, and does not require a norm constrain a every time step.

????

Dedication

In no particular order, to my loving family: Plato, Carolyn, Helena, Maria, and animals.

Acknowledgements

Foremost, I am thankful to Dr. Robert Braun for his support and advising through my work at Colorado Boulder. I would like to thank Professor Jay McMahon and Professor Hanspeter Schaub for serving on my committee and for their fulfilling coursework and advising. Additionally, I would like to thank Dr. Mason Peck for his four years of support at Cornell, which serve as my foundation in this field.

I am also grateful for the insightful and thrilling technical discussions I have had with Sven Niederberger, Benjamin Chung, Janis Maczijewski, Joe Barnard, Miki Szmuk, Skye Mceowen, Ian Garcia, Shez Virani, Daniel Aguilar-Marsillach, the EsDL research team, and many more.

Similarly, I would like to thank my SpaceX colleagues for grounding me in reality while letting me play with vehicles of the heavens. Go Falcon, Dragon, and Starlink!

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

1.1 Motivation

This thesis focuses on the implementation and development of a 6 degree-of-freedom (DoF) guidance algorithm that solves the powered descent guidance (PDG) problem. I present a modified version of the Szmuk-Açıkmeşe 6-DoF algorithm presented in [9].

1.2 Background

Pin-point landing has been of significant interest in the last 6 decades for a variety of applications. These include safely landing scientific payloads and humans on other planets or back on Earth. Having the ability to land near a site of scientific interest, a base, or refueling station will become a requirement as we build our space transportation infrastructure. Solving this problem has also been of considerable interest to those returning launch vehicle stages

The ability to soft-land a rocket is fundamentally disruptive to the launch industry and has already had a large impact in reducing the cost of getting to space. Additionally, similar methods, given navigational upgrades, will be conducive of the development of colonies on other planets. Pin-point landing and divert capability is a necessity in these situations.

Every pinpoint landing problem begins with an entry phase where the vehicle descends through an atmosphere to a point where powered descent must begin. Many Mars entry, descent, and landing schemes enter the atmosphere and slow down via parachute. This parachute is then cut away to allow powered descent. With atmospheric qualities being stochastic, the position

in which the descent phase must begin is uncertain. Therefore, we would like to look at algorithms which maximize the divert capability of the craft by minimizing the fuel consumption or final time from initial condition to terminal state. We define powered descent guidance as the generation of a fuel-optimal trajectory that takes the vehicle from some initial state condition to a prescribed final state in a uniform gravitational field with standard vehicle given thrust magnitude and direction constraints.

The convex optimization framework is exploited because it is conducive of real-time on-board implementation and has guaranteed convergence properties with deterministic criteria. The convex programming algorithm to solve powered descent guidance that I will present has non-convex controls constraints and will be posed as a finite-dimensional SOCP problem. SOCPs have low complexity and can be solved in polynomial time [5]. Interior-point numerical methods compute optimal solutions with deterministic stopping criteria and are, again conducive to on-board implementation.

1.3 Related Research

perhaps start with lawden functions linear in control?

"Lossless convexification can be used to convexify non-convex constraints without losing precision, unfortunately, only a few types of constraint can be convexified by lossless convexification. The nonlinear dynamics and constraints are the primary factors for non-convexity in P1 and they are not in the scope of lossless convexification. In this subsection, an improved successive convexification algorithm is proposed to circumvent the non-convexity from nonlinearity. "

COMPARE AND CONTRAST WITH SIMILAR ONLINE MTHODS LIKE LOSSLESS CONVEXIFICAION

COMPARE AND CONTRAST FRAMEWORKW ITH SQP

1.4 Statement of Scope

Chapter 2

The Powered Descent Guidance Problem

2.1 Problem Description

The goal of the presented algorithm is to generate optimal translational and attitude trajectory profiles that are dynamically feasible and amenable. This means that the modeled vehicle should abide by all state boundary conditions, actuator constraints, and the proposed dynamics. We shall initially define this problem as a continuous-time non-convex dynamical form and then explore converting this to a disciplined convex program. We shall discuss the dynamic, control, initial, and terminal constraints that must be met throughout the problem.

In order to maximize the divert capability of the vehicle, we propose the objective of minimizing the final time of the solution. Although not proven here, this can be considered a proxy to the minimum-fuel consumption problem, as the non-convex constraint of minimum-thrust and single-ignition requires that the engine be on for the duration of the landing phase. In this scenario, the fuel-consumption cost is strictly increasing monotonic, and the sooner the terminal conditions can be met, the fewer the total cost. A similar free-ignition-time modification can be made to further decrease this cost and additionally optimize the ignition time [8].

2.2 Definitions and Notation

Now we shall define the two frames of importance. The $\mathcal{F}_{\mathcal{N}} : \{\mathcal{O}_{\mathcal{N}}, \hat{\boldsymbol{n}}_1, \hat{\boldsymbol{n}}_2, \hat{\boldsymbol{n}}_3\}$ frame defines an inertially fixed Up-East-North reference frame where the origin $\mathcal{O}_{\mathcal{N}}$ located at the landing site. This can easily be changed to the local-vertical local-horizontal (LVLH) or other useful frame

definition. The $\mathcal{F}_\mathcal{B} : \{\mathcal{O}_\mathcal{B}, \hat{\boldsymbol{b}}_1, \hat{\boldsymbol{b}}_2, \hat{\boldsymbol{b}}_3\}$ frame is a body fixed frame where the x-axis is aligned vertically with the vehicle, or aligned with the thrust vector at zero thrust vector control (TVC) deflection angle. The Y-axis points out of the side of the cylindrical vehicle and the Z-axis completes the right handed triad.

Here forward, it should also be assumed that the vectorial derivative, shown by $\dot{\mathbf{r}}$, is an inertial time derivative. Derivatives in frames other than our inertial will be indicated otherwise as ${}^{\mathcal{X}}\frac{d\mathbf{r}}{dt}$. Any vector shown as ${}^{\mathcal{X}}\mathbf{r}$ is in the $\mathcal{X}$ frame, and similarly anything without this left superscript is frameless. Therefore the vector $\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}$ is the frameless angular velocity vector of the body frame with respect to the inertial frame. We also use the notation $\mathbb{R}$, $\mathbb{R}_+$, and $\mathbb{R}_{++}$ to denote the set of real values, non-negative real values, and positive real values respectively.

2.3 Vehicle Dynamics

Translational Dynamics Given that most powered descent maneuvers are done within kilometers of a site, and at speeds much less than orbital velocities, we can use a simplified gravitational acceleration assumption with a non-rotation sphere. Similarly, we assume aerodynamic forces to be negligible. However, as we will show in a later section, any nonlinear dynamics can be incorporated, as they will be represented as a linear time-varying (LTV) system in the successive convexification routine.

The algorithm as presented has the vehicle actuated by a single gimbaled thruster at the bottom of the vehicle. It should be stated that the algorithm can readily accommodate other actuator geometries and configurations. This engine has feasible thrust magnitude and efficiency, as well as standard gimbal range for agile landing or VTVL vehicles. To be inclusive of actuator dynamics, we assume that the engine has maximum and minimum thrust bounds. Most rocket engines have a minimum throttle percentage, below which the engine does not perform well or in a stable manner. Additionally, during powered descent routine, once ignition occurs the engine is not turned off or re-ignited until commanded at the terminal state. This minimum thrust constraint is

a source of non-convexity.

It is critical to capture the mass depletion dynamics, proportional to the magnitude of the thrust generated by the engine. For simplicity, we assume that the inertia matrix and the position of the center-of-mass is constant throughout the trajectory although these modifications can also be included in the dynamic formulation. We use the constant $\alpha_{\dot{m}}$, a function of the specific impulse, as the mass depletion parameter. This is the inverse of the mass flow rate, which is the total flow rate of the propellants to the engine. Additionally, we assume a constant specific impulse throughout the throttleable region, which may not always be true. Normally an engine operating at a lower thrust than nominal may may be less efficient and have a lower specific impulse. These effects are small enough to be ignored in this problem statement. Therefore we have that

$$\alpha_{\dot{m}} = \frac{1}{I_{sp} g_0} \tag{2.1}$$

$$\dot{m}(t) = -\alpha_{\dot{m}} \left\| \mathbf{F}_{thrust}(t) \right\|_2 \tag{2.2}$$

With our assumptions, we can trivially express the translational dynamics and forces acting on the vehicle in the inertially frame. They are as follows:

$$\dot{\mathbf{r}}(t) = \mathbf{v}(t) \tag{2.3}$$

$$\dot{\mathbf{v}}(t) = \frac{\mathbf{F}_{thrust}(t)}{m(t)} + \mathbf{g} \tag{2.4}$$

Moving forward, we will assume the translational dynamics to be in the inertial frame and all rotational dynamics to be in the body fixed frame.

HERE PERHAPS MAKE THE SRP MODIFICATIONS REQUIRED

2.3.1 Attitude Dynamics and Formalisms

In this formulation, the vehicle is treated as a rigid body, although the successive framework could readily support additional non-rigid nonlinear dynamics. From Euler's principal rotation theorem, any coordinate reference frame can be brought from an arbitrary initial condition to an arbitrary final orientation by a single rigid rotation through a principle angle ϕ about a principal

axis $\hat{\boldsymbol{e}}$. This axis is fixed in both the initial and final orientation. Therefore $\hat{\boldsymbol{e}} = [C]\hat{\boldsymbol{e}}$, where $[C] \in SO(3)$ is a rotation mapping in $\mathbb{R}^{3\times3}$ taking a vector from the initial orientation to the final, shows that $\hat{\boldsymbol{e}}$ is an unit eigenvector of the $[C]$ transform whose eigenvalue is $+1$. This rotation mapping matrix, a direction cosine matrix (DCM), has nine parameters and can be cumbersome although exact. Generally, we prefer to use mappings with fewer parameters. The right-handed $SO(3)$ group DCM has a determinant $+1$ with inverse/transpose both being the inverse mapping: $[\mathcal{BN}]^T = [\mathcal{NB}] = [\mathcal{N} \leftarrow \mathcal{B}]$. Similarly, they can be multiplied to encode compound rotations, say, from sensor frame to body frame to inertial.

Quaternions The attitude formalism initially used in [10] are Euler parameters, or quaternions. They to denote the attitude of the vehicle between the $\mathcal{F}_\mathcal{B}$ and $\mathcal{F}_\mathcal{N}$ frames, $q_{\mathcal{B}/\mathcal{N}}(t)$ on the unit sphere. This produces the DCM $[\mathcal{BN}]$. We use the angle-axis form of the quaternion, noted here:

$$\boldsymbol{q}_{\mathcal{B}/\mathcal{N}} \triangleq \begin{bmatrix} \cos(\frac{\phi}{2}) \\ \hat{\boldsymbol{e}}\sin(\frac{\phi}{2}) \end{bmatrix} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} \tag{2.5}$$

Note that the Euclidean norm $||\boldsymbol{q}_{\mathcal{B}/\mathcal{N}}||_2 = 1$ must be constrained to the unit sphere at all times. Lie group methods of integration are normally employed to ensure that the unit norm constraint is satisfied [2]. It soon becomes apparent that there can become a hemispherical ambiguity in the quaternion parameter as ϕ grows larger than π. However, this is resolved as the sign of the q_0 parameter can be checked to switch from long to short rotation quaternions. We similarly see that quaternion symmetries are such that $\boldsymbol{q}$ and $-\boldsymbol{q}$ produce the same rotation mapping. As it will become useful later, we must now introduce the quaternion-to-DCM mapping in equation 2.6 [7].

$$C_{\mathcal{B}/\mathcal{N}} = [\mathcal{BN}] = \begin{bmatrix} 1-2(q_2^2+q_3^2) & 2(q_1q_2-q_0q_3) & 2(q_0q_2+q_1q_3) \\ 2(q_1q_2+q_0q_3) & 1-2(q_1^2+q_3^2) & 2(q_2q_3-q_0q_1) \\ 2(q_1q_3-q_0q_2) & 2(q_0q_1+q_2q_3) & 1-2(q_1^2+q_2^2) \end{bmatrix} \tag{2.6}$$

In attitude dynamics, we use $\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}(t) \in \mathbb{R}^3$ to denote the angular velocity vector of the vehicle rigid body frame with respect to the inertial frame. This should recognized as the nominal

output of your gyroscope, the body frame angular rate. We also use the $\left[\mathbf{r}^{\times}\right]$ operator to denote the skew symmetric matrix form of the vector $\mathbf{r}$. The inertia tensor instantiated in the $\mathcal{F}_B$ frame, about the body center of mass, is written as ${}^{\mathcal{B}}[I_c] \in \mathbb{R}^{3\times3}$. The body frame inertia matrix is the solution to the ${}^{\mathcal{B}}[I_c] = \int_B -[\boldsymbol{r}^{\times}][\boldsymbol{r}^{\times}]dm$ where $\boldsymbol{r}$ are the vectors to each infinitesimal mass element from the center of mass. This matrix must be symmetric positive semi-definite and abide by the triangle inequality.

Quaternions are non-unique and non-singular with their kinematic differential equation being elegant and bilinear 2.7. These can be attractive for problems where the dynamics are linearized.

$$\dot{\boldsymbol{q}}_{\mathcal{B}/\mathcal{N}} = \frac{1}{2}B_w({}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}})\boldsymbol{q}_{\mathcal{B}/\mathcal{N}} = \frac{1}{2}B_q(\boldsymbol{q}_{\mathcal{B}/\mathcal{N}})\ {}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}} \tag{2.7}$$

where the B_w and B_q matrices are defined as

$$B_w(\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}) = \begin{bmatrix} 0 & -\omega_1 & -\omega_2 & -\omega_3 \\ \omega_1 & 0 & \omega_3 & -\omega_2 \\ \omega_2 & -\omega_3 & 0 & \omega_1 \\ \omega_3 & \omega_2 & -\omega_1 & 0 \end{bmatrix} \qquad B_q(\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}) = \begin{bmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & -q_3 & -q_2 \\ q_3 & q_0 & -q_1 \\ -q_2 & q_1 & q_0 \end{bmatrix} \tag{2.8}$$

Modified Rodrigues Parameters Now that we are intimately familiar with Euler parameters, we shall introduce Modified Rodrigues Parameters (MRPs), another useful and popular rotation formalism. These form a three parameter set without a norm constraint. We derive them from quaternions, as shown in 2.9

$$\sigma_i = \frac{q_i}{1+q_0} \quad i = 1,2,3 \tag{2.9}$$

$$\boldsymbol{\sigma}_{\mathcal{B}/\mathcal{N}} = [\sigma_1\ \sigma_2\ \sigma_3]^T = \tan\frac{\phi}{4}\hat{\boldsymbol{e}} \tag{2.10}$$

taking an argument of $\boldsymbol{q}_{\mathcal{B}/\mathcal{N}}$. The reverse mapping is simply $q_0 = \frac{1-\sigma^2}{1+\sigma^2}$ and $q_{1:3} = \frac{2\sigma_{1:3}}{1+\sigma^2}$ where σ^2 here forward will represent the norm squared of the MRP. We see that small angle approximations can be made for a wider range of displacements. It is clear that there exists a singularity at $\pm2\pi$. However, handling this singularity, in a computational sense, is very simple. We perform a switch to the MRP shadow set when the angular displacement is $\phi \geq \pi$ and define the shadow set as $\boldsymbol{\sigma}^{\mathcal{S}} = \frac{-\boldsymbol{\sigma}}{\sigma^2} = \tan\frac{\phi-2\pi}{4}\hat{\boldsymbol{e}}$. Simply:

Algorithm 1 MRP Switching

```
procedure MRP_SWITCH(σ_k)
    if ‖σ_k‖ > 1 then
        σ_k = −σ_k/σ_k^2
    end if
end procedure
```

However, if we constrain ourselves to never encounter a full rotation, we will not need such a structure. In optimization routines where large angular displacements must be made, it could be possible to encode switches as a state triggered constraint (STC) [8]. Now we shall recognize the MRP-to-DCM mapping that will be required 2.11:

$$C_{\mathcal{B}/\mathcal{N}} = [\mathcal{BN}] = \mathbb{I}_3 + \frac{8[\boldsymbol{\sigma}^\times] - 4(1-\sigma^2)[\boldsymbol{\sigma}^\times]}{(1+\sigma^2)^2} \tag{2.11}$$

taking an argument of $\boldsymbol{\sigma}_{\mathcal{B}/\mathcal{N}}$ and where $\mathbb{I}_3$ is the identity matrix. The kinematic differential equation is 2.12

$$\dot{\boldsymbol{\sigma}} = \frac{1}{4}[(1-\sigma^2)\mathbb{I}_3 + 2[\boldsymbol{\sigma}^\times] + 2\boldsymbol{\sigma}\boldsymbol{\sigma}^T]\ {}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}} = \frac{1}{4}B_\sigma(\boldsymbol{\sigma})\ {}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}} \tag{2.12}$$

as a function of the body frame vehicle angular rate.

Differentiating the angular momentum vector of system, and making the rigid body assumption, we have:

$$\dot{\boldsymbol{H}} = \mathbf{L}_c = [I_c]\dot{\boldsymbol{\omega}} + [\boldsymbol{\omega}^\times][I_c]\boldsymbol{\omega} \tag{2.13}$$

where the torque acting on the vehicle is written as $\mathbf{L}_c(t) \in \mathbb{R}^3$ in the body frame. Rearranged, we have the canonical Euler rotational equations of motion

$$\dot{\boldsymbol{\omega}} = [I_c]^{-1}(\mathbf{L}_c - [\boldsymbol{\omega}^\times][I_c]\boldsymbol{\omega}) \tag{2.14}$$

2.4 Boundary Conditions and State Constraints

The boundary conditions for the proposed guidance routine are simple. We must have hard constraints for the initial conditions, being the state vector of the vehicle when the routine is

initialized, and terminal state constraints. At the start of the guidance routine, we initialize the states with the current vehicle state vector:

$$m(0) = m_0, \quad {}^{\mathcal{N}}\mathbf{r}(0) = \mathbf{r}_0, \quad {}^{\mathcal{N}}\mathbf{v}(0) = \mathbf{v}_0, \quad \boldsymbol{\sigma}_{\mathcal{B}/\mathcal{N}}(0) = \boldsymbol{\sigma}_0, \quad {}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}(0) = \boldsymbol{\omega}_0 \tag{2.15}$$

with our given terminal state constraints:

$${}^{\mathcal{N}}\mathbf{r}(0) = \mathbf{0}, \quad {}^{\mathcal{N}}\mathbf{v}(0) = \mathbf{0}, \quad \boldsymbol{\sigma}_{\mathcal{B}/\mathcal{N}}(0) = \mathbf{0}, \quad {}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}(0) = \mathbf{0} \tag{2.16}$$

leaving the final mass unconstrained and assuming the landing site to be the origin. An upright attitude assuming the vehicle has landing hardware. We leave the terminal mass unconstrained. Of course these can be modified to fit arbitrary landing requirements. The problem proposed in [9] does not constrain the initial attitude of the vehicle, but we will in this formulation. TRY TO TELL WHY HERE!!

Now, let us look at the state constraints that must be met. The vehicle propulsion system is limited in fuel which manifests itself as this inequality constraint:

$$m_{dry} - m(t) \leq 0 \tag{2.17}$$

We shall apply a glide-slope constraint such that the vehicle approaches the landing point from above, limiting large lateral diverts in the terminal phase. We form the convex constraint using the angle γ_{gs}. This becomes a simple geometrical argument that $\tan\gamma_{gs} \leq \frac{r_{Up}}{\left|\left|[r_{East} \;\; r_{North}]\right|\right|}$. We can form this with the following:

$$\tan\gamma_{gs}||[\hat{\boldsymbol{n}}_2 \quad \hat{\boldsymbol{n}}_3]^T\mathbf{r}(t)||_2 - \hat{\boldsymbol{n}}_1^T\mathbf{r}(t) \leq 0 \tag{2.18}$$

This creates an upward facing cone about the landing point that the vehicle must not lie outside of. This type of convex constraint can also be useful in avoiding rocky terrain and enforcing a landing from directly above an area, minimizing lateral movement close to the ground.

It is additionally helpful to restrict the attitude of the vehicle such that it does not tilt over a prescribed angular displacement. This could be to maintain visibility for terrain relative navigation sensors or to give human passengers visibility over the terrain or landing surface. Therefore we can

constrain the angle between the inertial frame Up unit vector and the bore-sight body vector of the vehicle. We start with the constraint derivation $\hat{\boldsymbol{b}}_1 \cdot \hat{\boldsymbol{n}}_1 \geq \cos\psi_{max}$:

$$\left([\mathcal{N}\mathcal{B}] \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}^{\mathcal{B}} \right)^T \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}^{\mathcal{N}} \geq \cos\psi_{max} \tag{2.19}$$

this selects the $(1,1)$ element of the MRP-to-DCM matrix in equation 2.11. Therefore the constraint becomes:

$$\frac{1}{(1+\sigma^2)^2}\left(4(\sigma_1^2 - \sigma_2^2 - \sigma_3^2) + (1-\sigma^2)^2\right) \geq \cos\psi_{max} \tag{2.20}$$

$$(1+\sigma^2)^2\cos\psi_{max} - \left(4(\sigma_1^2 - \sigma_2^2 - \sigma_3^2) + (1-\sigma^2)^2\right) \leq 0 \tag{2.21}$$

This quaternion version of this same constraint is $\cos(\psi_{max}) \leq 1 - 2\sqrt{q_2^2 + q_3^2}$. The quaternion unity identity must be used in this derivation. Similarly, we can also constrain the entire MRP such that the vehicle does not exceed π radians in total angular displacement.

$$\left\|\boldsymbol{\sigma}_{\mathcal{B}/\mathcal{N}}\right\|_2 \leq 1 \tag{2.22}$$

Nominally the vehicle should not have large angular rates throughout the descent and landing, so we can simply constrain this as well.

$$\left\|\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}(t)\right\|_2 \leq \omega_{max} \tag{2.23}$$

Finally we must constrain the commanded thrust magnitude. As stated before, engines have a minimum and maximum thrust region $[T_{min}, T_{max}] \in \mathbb{R}_{++}$ in which they operate. Also, recall that we have made the single-ignition assumption. The engine thrust vector control system there is some limit to the amount of gimbal angle we can perform δ_{max}.

$$0 < T_{min} \leq \left\|\boldsymbol{F}_{thrust}(t)\right\|_2 \leq T_{max} \tag{2.24}$$

$$\cos(\delta_{max})\left\|\boldsymbol{F}_{thrust}(t)\right\|_2 \leq \hat{\boldsymbol{b}}_1^T \boldsymbol{F}_{thrust}(t) \tag{2.25}$$

it is clear that the upper thrust bound is clearly convex but the lower bound creates a non-convex constraint. This shall be handled later with a slack variable.

2.5 Continuous Time Problem

Putting this all together, we can pose the continuous time optimization problem. In this form, it is non-convex and requires conditioning to work into the convex programming framework. As stated, our objective is to minimize the time-of-flight required to get to the terminal conditions subject to the aforementioned constraints, dynamics, and boundary conditions. The state vector, thrust commands, gimbal angles, and final time are optimization variables and are considered the solution to this problem. We pose this in problem 2.5.

Problem 1: Continuous Time Non-Convex Free-Final-Time

Cost Function:

$$\min_{\mathbf{x},\ \mathbf{F}_{th}} \ t_f$$

Boundary Conditions:

$$m(0) = m_0, \qquad {}^{\mathcal{N}}\mathbf{r}(0) = \mathbf{r}_0, \qquad {}^{\mathcal{N}}\mathbf{v}(0) = \mathbf{v}_0, \qquad \boldsymbol{\sigma}_{\mathcal{B}/\mathcal{N}}(0) = \boldsymbol{\sigma}_0, \qquad {}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}(0) = \boldsymbol{\omega}_0$$

$$ {}^{\mathcal{N}}\mathbf{r}(0) = \mathbf{0}, \qquad {}^{\mathcal{N}}\mathbf{v}(0) = \mathbf{0}, \qquad \boldsymbol{\sigma}_{\mathcal{B}/\mathcal{N}}(0) = \mathbf{0}, \qquad {}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}(0) = \mathbf{0}$$

Dynamics:

$$\begin{aligned}
\dot{m}(t) &= -\alpha_{\dot{m}} \left\| \mathbf{F}_{th}(t) \right\|_2 \\
{}^{\mathcal{N}}\dot{\mathbf{r}}(t) &= \mathbf{v}(t) \\
{}^{\mathcal{N}}\dot{\mathbf{v}}(t) &= \frac{[\mathcal{N}\mathcal{B}(\boldsymbol{\sigma})]\ {}^{\mathcal{B}}\mathbf{F}_{th}(t)}{m(t)} + {}^{\mathcal{N}}\mathbf{g} \\
\dot{\boldsymbol{\sigma}}_{\mathcal{B}/\mathcal{N}} &= \frac{1}{4}\left[(1-\sigma^2)\mathbb{I}_3 + 2[\boldsymbol{\sigma}^{\times}] + 2\boldsymbol{\sigma}\boldsymbol{\sigma}^T\right]\ {}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}} \\
{}^{\mathcal{B}}\dot{\boldsymbol{\omega}}_{\mathcal{B}/\mathcal{N}} &= [I_c]^{-1}([\boldsymbol{r}^{\times}_{\mathrm{COM}}]\ {}^{\mathcal{B}}\mathbf{F}_{th}(t) - [\boldsymbol{\omega}^{\times}][I_c]\boldsymbol{\omega})
\end{aligned}$$

State Constraints:

$$\begin{aligned}
m_{dry} - m(t) &\leq 0 \\
||[\hat{\boldsymbol{n}}_2 \ \ \hat{\boldsymbol{n}}_3]^T \mathbf{r}(t)||_2 \ \tan\gamma_{gs} - \hat{\boldsymbol{n}}_1^T \mathbf{r}(t) &\leq 0 \\
(1+\sigma^2)^2 \cos\psi_{max} - \left(4(\sigma_1^2 - \sigma_2^2 - \sigma_3^2) + (1-\sigma^2)^2\right) &\leq 0 \\
\left\| \boldsymbol{\sigma}_{\mathcal{B}/\mathcal{N}}(t) \right\|_2 &\leq 1 \\
\left\| \boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}(t) \right\|_2 &\leq \omega_{max}
\end{aligned}$$

Control Constraints:

$$\begin{aligned}
0 < T_{min} \leq \left\| \boldsymbol{F}_{th}(t) \right\|_2 &\leq T_{max} \\
\cos(\delta_{max}) \left\| \boldsymbol{F}_{th}(t) \right\|_2 &\leq \hat{\boldsymbol{b}}_1^T \boldsymbol{F}_{th}(t)
\end{aligned}$$

Chapter 3

Convex Formulation

Now we shall derive the convex form of Problem 1. We will convert the non-convex continuous free-final-time problem to a convex fixed-final-time problem. This will be a second order cone sub-problem. We will solve this sub-problem repeatedly to convergence or "successively." This successive process turns each subproblem into a larger free-final-time algorithm.

3.1 Linearization

Let us define the state vector $\mathbf{x}(t) \in \mathbb{R}^{14\times 1}$ and our control vector $\mathbf{u}(t) \in \mathbb{R}^{3\times 1}$:

$$\mathbf{x}(t) \triangleq \begin{bmatrix} m(t) & \mathbf{r}_I^T(t) & \mathbf{v}_I^T(t) & q_{B/I}^T(t) & \boldsymbol{\omega}_B^T(t) \end{bmatrix}^T \tag{3.1}$$

$$\mathbf{u}(t) \triangleq \mathbf{T}_B(t) \tag{3.2}$$

Therefore we can express the nonlinear dynamics in the following form

$$\frac{d}{dt}\mathbf{x}(t) = f(\mathbf{x}(t), \mathbf{u}(t)) = \begin{bmatrix} \dot{m}(t) & \dot{\mathbf{r}}_I^T(t) & \dot{\mathbf{v}}_I^T(t) & \dot{q}_{B/I}^T(t) & \dot{\boldsymbol{\omega}}_B^T(t) \end{bmatrix}^T \tag{3.3}$$

In order to make the problem free-final-time, we must manipulate the how time works. We are going to replace t with a normalized trajectory time $\tau \in [0, 1]$. We apply the chain run:

$$\frac{d}{dt}\mathbf{x}(t) = \frac{d\tau}{dt}\frac{d}{d\tau}\mathbf{t}(t) \tag{3.4}$$

And we can translate between the two by using the dilation coefficient σ which is defined

$$\sigma \triangleq \left(\frac{d\tau}{dt}\right)^{-1} \tag{3.5}$$

This σ will become a variable in the convex subproblem that acts as the non-dimensionalized final time. It is a scaling factor that translates between real work differential time and the normalized version used for our algorithm. We can now write the nonlinear dynamics to take advantage of this normalized time:

$$\mathbf{x}'(\tau) \triangleq \frac{d}{d\tau}\mathbf{x}(\tau) = \sigma f(\mathbf{x}(\tau), \mathbf{u}(\tau)) \tag{3.6}$$

We can fit the nonlinear dynamics from problem 1 into a linear framework by performing a first-order Taylor approximation about a reference state, input, and dilation coefficient $(\hat{\mathbf{x}}, \hat{\mathbf{u}}, \hat{\sigma})$. Therefore we can represent the system using normalized time:

$$\mathbf{x}'(\tau) = \hat{\sigma}\frac{\partial}{\partial x}f(x,u)\bigg|_{\hat{x},\hat{u}}(x(\tau)-\hat{x}(\tau)) + \hat{\sigma}\frac{\partial}{\partial u}f(x,u)\bigg|_{\hat{x},\hat{u}}(u(\tau)-\hat{u}(\tau)) + \sigma f(\hat{x}(\tau),\hat{u}(\tau)) \tag{3.7}$$

$$\mathbf{x}'(\tau) = A(\tau)\mathbf{x}(\tau) + B(\tau)\mathbf{u}(\tau) + \Sigma(\tau)\sigma + \mathbf{z}(\tau) \tag{3.8}$$

We can break the Taylor expansion into matrices to make things simpler later on:

$$A(\tau) \triangleq \hat{\sigma}\frac{\partial}{\partial x}f(x,u)\bigg|_{\hat{x},\hat{u}} \tag{3.9}$$

$$B(\tau) \triangleq \hat{\sigma}\frac{\partial}{\partial u}f(x,u)\bigg|_{\hat{x},\hat{u}} \tag{3.10}$$

$$\Sigma(\tau) \triangleq f(\hat{x}(\tau),\hat{u}(\tau)) \tag{3.11}$$

$$\mathbf{z}(\tau) \triangleq -A(\tau)\hat{(x)}(\tau) - B(\tau)\hat{u}(\tau) \tag{3.12}$$

With this done, we tackle the only source of non-convexity: the non-zero lower bound to our actuator thrust. With a Taylor series approximation we can say that $T_{min} \leq B_g(\tau)(u)(\tau)$ for which $B_g(\tau) \triangleq \frac{\hat{\mathbf{u}}^T(\tau)}{\|\hat{\mathbf{u}}\|_2}$.

3.2 Discretization Scheme

This is where things get hairy. We need to "cast" the problem into a finite dimensional optimization problem where the trajectory is discretized into K evenly separated points with respect

to the normalized trajectory time τ. Let us define the set $\mathcal{K} \triangleq \{0, 1, \cdots, K-1\}$ with which many of the parameter arrays will have the same dimensions.

Given that the trajectory time is normalized on the interval $\tau \in [0, 1]$, we must define the discrete time step at point k as such

$$\tau_k \triangleq \frac{k}{K-1}, \quad \forall k \in \mathcal{K} \tag{3.13}$$

To help with numerical feasibility issues, we employ a first-order-hold linear scaling on the applied controls for each time step. Over the interval $\tau \in [\tau_k, \tau_{k+1}]$, we must express $\mathbf{u}(\tau)$ in terms of the $\mathbf{u}_k$ and $\mathbf{u}_{k+1}$:

$$\mathbf{u}(\tau) \triangleq \alpha_k(\tau)\mathbf{u}_k + \beta(\tau)\mathbf{u}_{k+1} \tag{3.14}$$

$$\alpha_k(\tau) = \frac{d\tau - \tau}{d\tau} \tag{3.15}$$

$$\beta_k(\tau) = \frac{\tau}{d\tau} \tag{3.16}$$

$$d\tau = \frac{1}{K-1} \tag{3.17}$$

I have shown these differently than the author for simplicity.

We can then use a state transition matrix $\Phi(\tau_{k+1}, \tau_k)$ to translate the system from k to $k+1$ with no input. This matrix follows these dynamics:

$$\frac{d}{d\tau}\Phi(\tau, \tau_k) = A(\tau)\Phi(\tau, \tau_k) \quad \forall k \in \mathcal{K} \tag{3.18}$$

Using our previous expressions, we can now discretize the dynamics such that each matrix,

at each time step requires a short integration:

$$\mathbf{x}_{k+1} = \bar{A}_k \mathbf{x_k} + \bar{B}_k \mathbf{u_k} + \bar{C} \mathbf{u_{k+1}} + \bar{\Sigma}_k \sigma + \bar{\mathbf{z}}_k \tag{3.19}$$

$$\bar{A}_k \triangleq \Phi(\tau_{k+1}, \tau_k) \tag{3.20}$$

$$\bar{B}_k \triangleq \int_{\tau_k}^{\tau_{k+1}} \alpha_k(\xi) \Phi(\tau_{k+1}, \xi) B(\xi) d\xi \tag{3.21}$$

$$\bar{C}_k \triangleq \int_{\tau_k}^{\tau_{k+1}} \beta_k(\xi) \Phi(\tau_{k+1}, \xi) B(\xi) d\xi \tag{3.22}$$

$$\bar{\Sigma}_k \triangleq \int_{\tau_k}^{\tau_{k+1}} \Phi(\tau_{k+1}, \xi) \Sigma(\xi) d\xi \tag{3.23}$$

$$\bar{\mathbf{z}}_k \triangleq \int_{\tau_k}^{\tau_{k+1}} \Phi(\tau_{k+1}, \xi) \mathbf{z}(\xi) d\xi \tag{3.24}$$

$$\tag{3.25}$$

The other boundary, state, and control constraints are already convex and just need to be instantiated for all time or at initial/terminal τ.

3.3 Successive Form, Trust Regions and Relaxations

In order to solve a non-convex problem, we iteratively solve a sequence of related convex subproblems. However, before we can reach a concluding framework, we must consider trust regions and dynamic relaxations. In order to make sure that this successive framework stays bounded and feasible through this convergence process, we must bound the divergence of state and inputs from one iteration to another. Unbounded problems can arise from constraints that admit an unbounded cost. To mitigate this issue, we augment the cost function with soft trust regions about the previous iterate's information. Let us define these deviations at iteration i as such:

$$\delta \mathbf{x}_k^i \triangleq \mathbf{x}_k^i - \mathbf{x}_k^{i-1} \tag{3.26}$$

$$\delta \mathbf{u}_k^i \triangleq \mathbf{u}_k^i - \mathbf{u}_k^{i-1}, \quad \forall k \in \mathcal{K} \tag{3.27}$$

$$\delta \sigma^i \triangleq \sigma^i - \sigma^{i-1} \tag{3.28}$$

We can then fabricate the following constraints with $\bar{\mathbf{\Delta}}^i \in \mathcal{R}^K$ and $\Delta_\sigma^i \in \mathcal{R}$

$$\delta \mathbf{x}_k^i \cdot \delta \mathbf{x}_k^i + \delta \mathbf{u}_k^i \cdot \delta \mathbf{u}_k^i \leq \mathbf{e}_k \cdot \bar{\mathbf{\Delta}}^i \tag{3.29}$$

$$\delta \sigma^i \cdot \delta \sigma^i \leq \Delta_\sigma^i \tag{3.30}$$

We can now append $w_\Delta^i \left\| \bar{\Delta}^i \right\| + w_{\Delta_\sigma} \left\| \Delta_\sigma^i \right\|$ to the cost function to attempt to minimize input, state, and final time deviations and keeping their deviation bounded via constraint, where w_Δ^i and w_{Δ_σ} are weighting scalars. Here the author makes an important note. Given that the trust regions are centered about previous points $(\mathbf{x}_k^{i-1}, \mathbf{u}_k^{i-1}, \sigma^{i-1})$, we must evaluate the Jacobian about the nonlinear trajectory beginning at $\mathbf{x}_k^{i-1}$. We can then use the zero-order-hold input vector $\mathbf{u}(\tau)$. Doing this $\forall k \in \mathcal{K}$ defines the linearization path $(\hat{\mathbf{x}}, \hat{\mathbf{u}}, \hat{\sigma})$.

Finally we must speak to dynamic relaxation. "Artificial Infeasibility" is encountered during the convergence process when the linearization becomes infeasible. For example, if the dynamics are linearized about unrealistic conditions, it because dynamically inconsistent. It will not produce a feasible solution. This is encountered during the first couple iterations of a successive convexification due to poor initial trajectory or time-of-flight estimation. To get rid of this issue, we employ dynamic relaxation, where a slack variable is added to the dynamics in order to "make room" for the iteration to proceed. However, you can guess that this will inevitably be something we try to minimize in the cost function in order to make sure that our final trajectories are as dynamically consistent as possible. Therefore, we must now write our dynamics as follows:

$$\mathbf{x}_{k+1}^i = \bar{A}_k^i \mathbf{x_k}^i + \bar{B}_k^i \mathbf{u_k}^i + \bar{C}^i \mathbf{u_{k+1}}^i + \bar{\Sigma}_k^i \sigma^i + \bar{\mathbf{z}}_k^i + \boldsymbol{\nu}_k^i \tag{3.31}$$

Because we are going to use this in our cost function as a norm, we shall concatenate the time history of the slack variable as

$$\bar{\boldsymbol{\nu}}^i = \begin{bmatrix} \boldsymbol{\nu}_0^{iT} & \cdots & \boldsymbol{\nu}_{K-2}^{iT} \end{bmatrix}^T \tag{3.32}$$

And of course we augment the cost function again with $w_v \left\| \bar{\boldsymbol{\nu}}^i \right\|_1$. As the iteration continues, this value is minimized as the solution becomes more dynamically feasible. Therefore, the magnitude of this norm is indicative of a final solution in the successive iteration process.

3.4 Convex Sub-Problem

We can now put all of the components together and form the convex version of the continuous problem, but with fixed final time.

Problem 2: Discretized Convex Fixed-Final-Time Sub-Problem (i^{th} iteration)

Cost Function:

$$\min_{\sigma^i, \mathbf{u}_k^i} \quad \sigma^i + w_\Delta^i \left\| \bar{\Delta}^i \right\|_2 + w_{\Delta_\sigma} \left\| \Delta_\sigma^i \right\|_1 + w_v \left\| \bar{\boldsymbol{\nu}}^i \right\|_1$$

Boundary Conditions:

$$m_0 = m_{wet} \quad \mathbf{r}_{I_0} = \mathbf{r}_i \quad \mathbf{v}_{I_0} = \mathbf{v}_i \qquad q_{B/I_0} = q_{B/I_i} \quad \boldsymbol{\omega}_B(0) = \boldsymbol{\omega}_{B_i}$$

$$\mathbf{r}_{I_K} = \mathbf{0} \qquad \mathbf{v}_{I_K} = \mathbf{0} \quad q_{B/I_K} = q_{B/I_f} \quad \boldsymbol{\omega}_{B_K} = \mathbf{0}$$

$$\mathbf{e}_2 \cdot \mathbf{T}_{B_K} = \mathbf{e}_3 \cdot \mathbf{T}_{B_K} = 0$$

Dynamics:

$$\mathbf{x}_{k+1}^i = \bar{A}_k^i \mathbf{x_k}^i + \bar{B}_k^i \mathbf{u_k}^i + \bar{C}^i \mathbf{u_{k+1}}^i + \bar{\Sigma}_k^i \sigma^i + \bar{\mathbf{z}}_k^i + \boldsymbol{\nu}_k^i$$

State Constraints:

$$m_k \geq m_{dry}$$

$$\mathbf{e}_1 \cdot \mathbf{r}_{I_k} \geq \tan(\gamma_{gs}) \left\| \begin{bmatrix} \mathbf{e}_2 & \mathbf{e}_3 \end{bmatrix}^T \mathbf{r}_{I_k} \right\|_2$$

$$\cos(\theta_{max}) \leq 1 - 2(q_{2_k}^2 + q_{3_k}^2)$$

$$\left\| \boldsymbol{\omega}_{B_k} \right\|_2 \leq \omega_{max}$$

Control Constraints:

$$T_{min} \leq B_g(\tau_k) \mathbf{u}_k^i$$

$$\left\| \mathbf{u}_k^i \right\|_2 \leq T_{max}$$

$$\cos(\delta_{max}) \left\| \mathbf{u}_k^i \right\|_2 \leq \boldsymbol{e}_1 \cdot \mathbf{u}_k$$

Trust Regions:

$$\delta \mathbf{x}_k^i \cdot \delta \mathbf{x}_k^i + \delta \mathbf{u}_k^i \cdot \delta \mathbf{u}_k^i \leq \mathbf{e}_k \cdot \bar{\boldsymbol{\Delta}}^i$$

$$\delta \sigma^i \cdot \delta \sigma^i \leq \Delta_\sigma^i$$

3.5 Algorithm

The goal is for the successive convexification algorithm to continue iterating until Δ_{tol} and ν_{tol} are met. These are defined by the magnitude of each vector.

Chapter 4

Results

Algorithm 2 Successive Convexification

procedure SCVX(x_{ref}, σ_{ref})
    Generate initial trajectory by linearly spanning from initial condition to terminal conditions on each state
    while $||\Delta|| \geq \Delta_{tol}$ && $||\nu|| \geq \nu_{tol}$ **do**
        Compute $\bar{A}_k^{i-1}, \bar{B}_k^{i-1}, \bar{C}_k^{i-1}, \bar{\Sigma}_k^{i-1}, \bar{z}_k^{i-1}$ from $\mathbf{x}_k^{i-1}, \mathbf{u}_k^{i-1}, \sigma^{i-1}$
        Solve Problem 2 using $\mathbf{x}_k^{i-1}, \mathbf{u}_k^{i-1}, \sigma^{i-1}, \bar{A}_k^{i-1}, \bar{B}_k^{i-1}, \bar{C}_k^{i-1}, \bar{\Sigma}_k^{i-1}, \bar{z}_k^{i-1}$
        Store the newly found $\mathbf{x}_k^i, \mathbf{u}_k^i, \sigma^i$
        $i++$;
    end while
    return $\mathbf{x}, \mathbf{u}$
end procedure

It turns out that this tool is incredibly powerful. One can modify the dynamics quite a bit, and depending on the robustness of the linearization scheme, this could be used as a generalized tool. The only glaring obstacle is state and actuator constraints. These types of nonconvexities need to be managed carefully.

4.1 Dimensionalized Simulation Results

The initial conditions to this problem must be small for numerical stability. Therefore, you must non-dimensionalize then on entry. Once the problem is solved, you can redimensionalize the solution, as shown in **??**. I implemented this algorithm in Python, utilizing the CVXPY library with ECOS solver. Below in 4.1 are some parameters I tested for an in-plane maneuver. I used the same constants and weightings as in [**?**]. For this scenario, to redimensionalize, $U_M = 10$kg,

$U_L = 100$meters, and $U_T = (10)^{1/2}$seconds.

Table 4.1: Parameters Used

Param	Units	Value
$\mathbf{g}_I$	U_L/U_T^2	$-0.981\mathbf{e}_1$
$\alpha_{\dot{m}}$	-	0.004
m_{wet}	U_M	14
m_{dry}	U_M	3
r_{I_0}	U_L	$(4, 4, 0)$
v_{I_0}	U_L/U_T	$(-0.1, 0, 0)$
q_{BI_0}	-	$(1, 0, 0, 0)$
ω_{B_0}	rad/U_T	$(0, 0, 0)$

4.2 Attitude Trajectory Tracking with Cold Gas Thrusters

This portion of the project was developed for ASEN6010 Advanced Attitude Control. I will now show how control can be done with thrusters at the top of the vehicle. I am using the attitude trajectory generated from the convex optimal problem. I have converted those quaternions into MRPs and switched from the UEN frame to a more common ENU frame. This is to say that +x is in the direction of thruster 2 on figure **??**; +y goes along thruster 1, and thruster 6/8 generate a positive torque about the z-axis axially through the top.

Please note that I did not implement standard Schmitt trigger "bang bang" control here. This probably would have been a more realistic implementation.

The distribution matrix D for my configuration is the following:

$$[D] = \begin{bmatrix} \vec{r}_1 \times \vec{g}_{t_1} & \cdots & \vec{r}_N \times \vec{g}_{t_N} \end{bmatrix} =$$

$$\begin{bmatrix} -1.50 & 0 & 1.50 & 0 & 1.50 & -1.50 & -1.50 & 1.50 \\ 0 & 1.50 & 0 & -1.50 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.61 & -0.61 & 0.61 & -0.61 \end{bmatrix}$$

We shall no use the MRP version of the Lyapunov control function is the following [7]:

$$\mathbf{L}_r = -K\epsilon - [P]\,\delta\omega + [I]\,(\dot{\omega}_r - [\omega]^{\times}\,\dot{\omega}_r) + [\omega_{\mathbf{r}}]^{\times}\,[I]\,\omega - \mathbf{L} \tag{4.1}$$

The value K is a scalar gain on the attitude relative MRP (error) ϵ. The value P is a gain on the angular rate error $\delta\omega$. All values labeled with ω_r are the reference angular velocities that we get from the successive conovex problem. The value $\mathbf{L}$ is a measure of the exogenous torque. In our case, this is zero.

The torque imparted on the spacecraft from an active thruster is the following:

$$\tau_i = \vec{r}_i \times F_i \vec{g}_{t_i} \tag{4.2}$$

$$\tau_j = \sum_{i=1}^{N} (\vec{r}_i \times \vec{g}_{t_i}) \cdot \hat{c}_j F_i \tag{4.3}$$

Where $\hat{c}_j$ is the body vector we expect to torque about. We can also write this expression as such:

$$\tau_j = [d_1 \cdots d_N] \begin{bmatrix} F_1 \\ \vdots \\ F_N \end{bmatrix} = [D]_j \, \mathbf{F} \tag{4.4}$$

We see that $[D] \in \mathbb{R}^{1\times N}$ for our N thrusters. This matrix maps the thruster forces to the spacecraft torque τ_j along our control axis of interest $\hat{c}_j$. Now we want to find the thrusters that provide a positive force for the desired torque on the system $\mathbf{L}_r$. We can perform the minimum norm inverse to find the force matrix as such:

$$\mathbf{F}_j = [D]_j^T \left([D]_j \, [D]_j^T \right)^{-1} \mathbf{L}_r \cdot \hat{c}_j \tag{4.5}$$

We then pick which thrusters have positive force values, log their identity, and create a reduced matrix $\bar{\mathbf{F}}_j \in \mathcal{R}^{M\times 1}$. We can do the same with $[\bar{D}]$. And therefore $[\bar{D}]\bar{\mathbf{F}}_j$ is the torque on a single axis. Doing this for all control axes, we apply the final torque

$$\mathbf{L_{cg}} = [\bar{D}]_1 \bar{\mathbf{F}}_1 + [\bar{D}]_2 \bar{\mathbf{F}}_2 + [\bar{D}]_3 \bar{\mathbf{F}}_3 \tag{4.6}$$

4.2.1 Simulation Framework

I created a simulation where I numericall integrated the equations of motion for the vehicle. On each cycle I calculated the required torque and fed it back to the system dynamics.

4.2.2 Starting At Nominal Attitude

Let's look at the system response if our initial attitude is the same as the one performed in the convex problem. This would be the case if you performed control immediately after guidance finished, or if it converged quickly. See figure **??**.

4.2.3 Starting At Off-Nominal Attitude and Angular Rate

In this scenario, I have drifter farther from the convex problem initial condition to roughly 32 degrees off axis and with an angular rate of $\omega_0 = [.3.5.5]^T$ rad/s. This is a large error should be hard to converge to in final time. See figure **??**. We can also see the output force of each of the 8 thrusters with figure **??**. Notice that the magnitudes don't go over 40N, as that is a hard limit that I set.

Chapter 5

Analysis

Chapter 6

Conclusions

6.1 Conclusions and Future Work

It should be known that while I do not consider this work novel, I do think it may lend a useful insight to those interested in using or implementing such strategies of motion.

Bibliography

[1] Behcet Acikmese and Scott R Ploen. Convex programming approach to powered descent guidance for mars landing. Journal of Guidance, Control, and Dynamics, 30(5):1353–1366, 2007.

[2] Michael S Andrle and John L Crassidis. Geometric integration of quaternions. Journal of Guidance, Control, and Dynamics, 36(6):1762–1767, 2013.

[3] Lars Blackmore. Autonomous precision landing of space rockets. Winter Bridge on Frontiers of Engineerin, 4(46):15–29, 2016.

[4] Lars Blackmore, Behcet Acikmese, and Daniel P Scharf. Minimum-landing-error powered-descent guidance for mars landing using convex optimization. Journal of guidance, control, and dynamics, 33(4):1161–1171, 2010.

[5] Stephen Boyd and Lieven Vandenberghe. Convex optimization. Cambridge university press, 2004.

[6] F Landis Markley and John L Crassidis. Fundamentals of Spacecraft Attitude Determination and Control, volume 33. Springer, 2014.

[7] Hanspeter Schaub and John L. Junkins. Analytical Mechanics of Space Systems. American Institute of Aeronautics and Astronautics, Inc, Reston, Virginia, fourth edition, 2018.

[8] Michael Szmuk. Successive Convexification & High Performance Feedback Control for Agile Flight. PhD thesis, 2019.

[9] Michael Szmuk and Behcet Acikmese. Successive convexification for 6-dof mars rocket powered landing with free-final-time. In 2018 AIAA Guidance, Navigation, and Control Conference, page 0617, 2018.

[10] Michael Szmuk, Utku Eren, and Behcet Acikmese. Successive convexification for mars 6-dof powered descent landing guidance. In AIAA Guidance, Navigation, and Control Conference, page 1500, 2017.

Appendix A

Planetary Landing Testbed Development

About appendices: Each appendix follow the same page-numbering rules