Closed-loop guidance for micro launchers

Improvement to robustness and failure tolerance

Benjamin Dutruel

TU Delft

Closed-loop guidance for micro launchers

Improvement to robustness and failure tolerance

by

Benjamin Dutruel

to obtain the degree of Master of Science
at the Delft University of Technology,

to be defended publicly on Wednesday, October 4th, 2023 at 10:00.

Student number:	4552229	
Project duration:	October 24th, 2023 – October 4th, 2023	
Version:	1.1	
Date:	September 18, 2023	
Thesis committee:	Ir. M. Naeije	TU Delft, Thesis Supervisor
	Dr.ir. Wouter van der Wal	TU Delft
	Dr. ir Erik-Jan van Kampen	TU Delft
	Dr. Christian Heise	Rocket Factory Augsburg

TU Delft

Preface

In this report I will present my master thesis which I have done at Rocket Factory Augsburg (RFA) where I have been working for over a year within the GNC team. Following a successful internship I was offered to stay longer and work on developing the guidance system. The main developments and results of this work are shown in this report, although I also had the opportunity to work on other domains. Moving to Germany was a challenging experience, yet I have greatly enjoyed working in such a young and dynamic company with colleagues of all nationalities.

I would like to thank all those at RFA who helped me throughout this period. In particular Francesc Betorz for supervising me in my first steps and taking time to explain the inner workings of the guidance system. I would also like to thank Nicolla Zappacosta and Christian Heise for taking the time to review the code and algorithms I produced and for all their recommendations. Furthermore I would like to thank Stefan Eisenknappl for giving me the opportunity to stay a RFA and for his suggestions on which analyses to perform. I would like to thank the GNC team for the amazing time I have had so far at RFA.

Of course I want to thank Marc Naeije from TU Delft who has been supervising me during all this time without showing any sign of impatience and always giving constructive feedback. So far our meetings have always been through Skype, so I am looking forward to finally seeing him again in person at the graduation.

Finally I would like to thank my family for their support throughout this whole journey. Their help with moving to Augsburg was much appreciated and helped lessen the stress with such large life change. I would also like to thank them for motivating me throughout some of the more difficult parts of this thesis. Likewise their assistance in proof-reading this report and all their suggestions were invaluable.

Going ahead I will continue to work at RFA to bring the guidance system to an operational state. I am looking forward to seeing it in action during the maiden flight of RFA ONE next year.

Benjamin Dutruel
Augsburg, September 2023

Summary

The launcher market is going through a major revolution with the emergence of many new startups trying to take a share of the ever growing space business. Rocket Factory Augsburg (RFA) is one of the emerging actors in this business and is facing numerous challenges in order to build an orbit capable launch vehicle, RFA ONE. The development of a closed-loop guidance system is one of those challenges and some of the design features of RFA ONE make this task particularly complicated. One of the primary issues is that the third stage which has been designed for in-orbit operations will be fully required for insertion during the first flight. This leads to very long burn-arcs which cause both convergence and optimality issues in traditional closed-loop guidance algorithms for orbit insertion. Due to a new approach to testing there is an increased uncertainty on various parameters that are critical for the guidance as well as a higher probability of component failures. The guidance algorithm needs to cope with this uncertain environment to ensure a successful flight. This leads to the following research objective:

The aim of this research is to improve the existing exo-atmospheric closed loop guidance system at RFA by increasing the probability that it will successfully guide the rocket to orbit insertion in all cases where reaching orbit is feasible.

This problem was broken down in three areas of improvement to the existing algorithms. The first was a complete redesign of the optimal control based closed-loop guidance. By using less assumptions and simplifications the optimality can be maintained for much longer thrust-arcs. Furthermore the use of a standard root-finding formulation allowed for the use of robust solvers. The second improvement was the addition of in-flight estimation of the vehicle performance parameters. By using the on-board measurement of acceleration and a linear Kalman Filter the parameter uncertainties can be largely mitigated. The last new component seeks to ensure that the rocket can divert to an alternate target orbit if the nominal one becomes unfeasible. It uses a ranked list of predefined targets and chooses the best feasible orbit. The feasibility criterion is the estimated performance margin computed using information from the closed-loop guidance and parameter estimation.

The new closed-loop guidance method has been proven to converge under a much wider range of conditions compared to traditional methods. It was also demonstrated that it yields functionally identical results to an industry-proven off-line trajectory optimizer for the mission studied in this thesis. While the parameter estimation initially showed some promising results, after testing in the high fidelity simulator it became apparent that it had difficulty dealing with all the perturbations. Nevertheless it was still shown to achieve better results compared to not performing any estimation of the vehicle parameters. Finally it was shown that the target selection algorithm was able to greatly increase the probability of reaching a viable orbit should the rocket encounter some technical issues during its ascent, such as an engine out.

List of Figures

List of Tables

List of abbreviations

AMT-HI High energy alternate MECO target. 59

AMT-LO Low energy alternate MECO target. 59

ATO Abort to orbit. 58

CoM Center of Mass. 9, 52, 55, 112

DOF Degrees of freedom, with 3-DOF referring to just translational dynamics and 6-DOF adding the rotational dynamics. 3, 19, 20, 31, 61, 82, 84, 98, 99, 105, 108, 112, 115

ECEF Earth Centered Earth Fixed frame. xiii, 15, 16

ECI Earth Centered Inertial. 16

ECI-J2000 Earth Centered Inertial frame defined by the mean equator and mean equinox on January 1, 2000, at 12:00 . 16

ECI-T0 Earth Centered Inertial frame defined by freezing the ECEF frame just before lift-off. 16

EO Engine out. 59

FES Functional Engineering Simulator; The combination of the GNC system with a plant model that allows for the simulation of the entire flight. viii, 2, 82–92, 95–97, 99, 100, 103, 105, 107, 108, 112, 115, 121

GNC Guidance, Navigation, and Control. vii, 2, 3, 31, 54, 61, 62, 82, 84, 93, 99, 103, 105, 108

GNSS Global Navigation Satellite System. 2, 108–110

GTO Geostationary Transfer Orbit, a highly elliptical orbit used for inserting satellites into a Geostationary Orbit. 33

HIL Hardware In the Loop. 2, 103

IMU Inertial Measurement Unit. 2, 9, 17, 51, 52, 55, 108, 112, 120, 122

LAN Longitude of the Ascending Node. 21, 32, 83, 84

LEO Low Earth Orbit. 40, 59

MECO Main engine cutoff. xiii, xiv, 59, 84

O/F Oxidiser to fuel ratio. 19, 75, 90, 107

PEG Powered Explicit Guidance, a closed loop guidance algorithm developed for the Space Shuttle. vii, 1, 6–11, 20, 32, 33, 37–44, 46–48, 50, 51, 61–67, 83, 84, 86, 96, 119, 120

PTM Push to MECO, proceed to nominal target. 59

QPEG Quadratic Powered Explicit Guidance. vii, viii, 39, 40, 42, 48–51, 60–72, 74, 75, 79, 83–87, 90, 93, 96, 98–101, 103, 108–110, 115, 116, 119–123

RCS Reaction Control System. 2, 3, 52, 105, 106, 121

RFA Rocket Factory Augsburg. v, 1, 4, 5, 19, 31, 32, 58–60, 97, 103, 112, 119–121

SECO Second-stage engine cutoff. 70, 80

SLS Space Launch System, a new American super heavy-lift launcher designed for deep space exploration. 25, 39, 51–53, 55, 58–60, 101, 119, 121

TVC Thrust Vector Control. 2, 3, 52, 112

TWR Thrust to Weight Ratio. 7, 12, 46, 48, 56, 97, 99, 113, 119

List of Symbols

a	Semi-major axis	m
a_T	Acceleration due to thrust	$\mathrm{m\,s^{-2}}$
e	Orbital eccentricity	–
$\mathbf{g}$	Gravity acceleration vector	$\mathrm{m\,s^{-2}}$
g_0	Standard gravity	$\mathrm{m\,s^{-2}}$
$\mathbf{H}$	Measurement matrix	–
$\mathbf{h}$	Angular momentum of the orbit	$\mathrm{m^2\,s^{-1}}$
H	Hamiltonian	–
i	Orbital inclination	rad
I_{sp}	Specific impulse	s
$\dot{m}$	Mass flow rate	$\mathrm{kg\,s^{-1}}$
m	Current vehicle mass	kg
$\hat{\mathbf{P}}$	Thrust direction vector	–
$\mathbf{P}$	State covariance matrix	–
$\mathbf{p}_r$	Position co-state, analogous to $\dot{\boldsymbol{\lambda}}$	–
$\mathbf{p}_v$	Velocity co-state, analogous to $\hat{\boldsymbol{\lambda}}$	–
$\mathbf{R}$	Measurement covariance matrix	–
$\mathbf{r}$	Position vector	m
$\mathbf{R}_d$	Desired cut-off position vector	m
$\mathbf{R}_p$	Predicted cut-off position vector	m
$\mathbf{R}_{go}$	Simplified position integral due to thrust	m
$\mathbf{R}_{thrust}$	Position integral due to thrust	m
T	Thrust force	N
t_b	Burn-time	s
t_{go}	Time to orbit insertion	s
t_λ	Time of average thrust	s
$\mathbf{u}$	Control vector	–

$\mathbf{V}$	Velocity vector	$\mathrm{m\,s^{-1}}$
$\mathbf{V}_d$	Desired cut-off velocity vector	$\mathrm{m\,s^{-1}}$
$\mathbf{V}_p$	Predicted cut-off velocity vector	$\mathrm{m\,s^{-1}}$
$\mathbf{V}_{go}$	Simplified velocity integral due to thrust	$\mathrm{m\,s^{-1}}$
$\mathbf{V}_{thrust}$	Velocity integral due to thrust	$\mathrm{m\,s^{-1}}$
v_{ex}	Exhaust velocity	$\mathrm{m\,s^{-1}}$
$\mathbf{x}$	State vector	–
γ	Flight path angle	rad
$\dot{\boldsymbol{\lambda}}$	Rate-vector of the thrust direction	$\mathrm{s^{-1}}$
$\hat{\boldsymbol{\lambda}}$	Unit-vector in the average thrust direction	–
$\boldsymbol{\lambda}$	Co-state vector	–
μ	Standard gravitational parameter	$\mathrm{m^3\,s^{-2}}$
$\boldsymbol{\Phi}$	State transition matrix	–
τ	Total mass to mass-flow rate ratio	s
ω	Local circular angular rotation rate	$\mathrm{rad\,s^{-1}}$

Contents

1 Project introduction and definition

Any launch vehicle needs to be guided through its ascent trajectory. While some of the trajectory optimization can be done before flight, the vehicle will always encounter perturbances throughout its flight which will lead to deviations from the nominal pre-defined flight path. It is therefore essential to include an on-board system which can correct for these deviations and ensure an optimal orbit injection. This topic has been researched since the very beginning of space rocketry. One notable early example includes the IGM (iterative guidance mode) [1] for the Saturn V which was based on a linear angle guidance law. This concept evolved into the PEG (Powered Explicit Guidance) algorithm used in the Space Shuttle [2], of which a variant is still used in the recent SLS rocket [3]. In the beginning these algorithms were limited in their complexity due to constraints of available onboard computational capabilities. However as the electronics industry has developed more powerful processors, these limitations have now been largely removed, thus opening the door to the implementation of far more complex algorithms. Due to the necessity for any guidance algorithm to run in real-time, fully numerical optimization is still out of reach for an on-board implementation. Therefore there is still a large focus on the development of semi-analytical closed-form solutions.

One domain of major importance and research is how to increase the robustness for these guidance algorithms. This is especially important for new micro-launchers, such as RFA ONE which is currently being developed by Rocket Factory Augsburg (RFA), where limited testing on the hardware leads to larger uncertainties on the various performance characteristics of the rocket and a larger probability of partial failure. It should be noted that the constraints on the accuracy of orbital injection for the first demonstration flight are much looser than for a nominal flight with a proven launcher. This leads to the following research objective:

The aim of this research is to improve the existing exo-atmospheric closed loop guidance system at RFA by increasing the probability that it will successfully guide the rocket to orbit insertion in all cases where reaching orbit is feasible.

To better understand the context in which this research was performed, the following sections will provide some background information. First an overview of the company where this thesis research was executed is presented in Section 1.1. Then a definition of what is meant by guidance, especially in the context of a rocket, is given in Section 1.2. Next a brief description of the mission which will serve as the baseline scenario for which the guidance algorithms will be developed is specified in Section 1.3. Finally the structure of this thesis report will be laid out in Section 1.4.

1.1. Rocket Factory Augsburg

RFA is a startup located in Augsburg, Bavaria belonging to the ”new space” industry. While the definition of new space is still subject to a lot of discussion the main factor addressed by new space relates to the way the business case is carried out to satisfy two objectives: reducing cost and schedule of a typical space project. Clearly in order to succeed in these areas new methodology and processes must be applied in comparison to the more traditional way of conducting business in national agencies or large space corporations.

RFA is a startup strong of 220 members, counting the permanent staff and temporary staff like interns. It was founded in 2018 in Augsburg by a small team of engineers, many with previous experience in the rocket industry, working in stealth mode before appearing on the radar. Their objectives are both ambitious and challenging since RFA wants to design, develop and launch a new rocket aiming at drastically reducing the cost for placing a satellite in low earth orbit while simultaneously offering a better quality service to the customers. Not only the cost must be lower than a traditional launch on a typical Ariane 5, Vega or the future Ariane 6 but even more challenging lower than a SpaceX launch!

The objectives can only be achieved by working on two important domains in parallel; the first domain is the technology of the main engine and the second relates to the industrial production process. In the very early stage RFA designed and optimized a rocket main engine benefiting from the most efficient burn cycle possible: the staged combustion cycle. This engine, which benefits from the latest advances in manufacturing such as additive manufacturing, has already proven itself during multiple long-duration hot-fires and has recently been tested in a fully integrated second stage stack test. In parallel the team is designing the stages using as many standard components and parts to fully take advantage of the existing industrial complex available in Bavaria. For example even the first stage propellant tanks have been designed to be compatible with existing tooling used outside of the aerospace industry.

1.2. Guidance

Guidance Navigation and Control (GNC) is the department that is responsible for designing the system that will steer the rocket to space. They develop the GNC program that will be integrated into the flight computer software and they are also responsible for selecting and testing the Inertial Measurement Unit (IMU) and Global Navigation Satellite System (GNSS) receiver. The GNC program consists of three main parts as the name implies: guidance, navigation, and control. The navigation algorithm is responsible for taking the data from sensors such as the IMU and GNSS in order to precisely determine the state of the rocket. The guidance takes the current state of the rocket and finds the optimal path to reach the desired target orbit. The control logic ensures that the rocket follows this path by commanding the actuators such as the Thrust Vector Control (TVC) and Reaction Control system (RCS). A high level overview of the GNC system is shown in Figure 1.1.

Contrary to most other systems on the rocket such as the engines, it is not possible to test the GNC system on its own as it is intrinsically connected to the dynamics of the entire rocket. The GNC department therefore relies on many types of simulations to test the code. To enable this engineers have developed a high fidelity plant model of the rocket dynamics that allows them to simulate an entire flight. The combination of this plant model with the GNC program is the Functional Engineering Simulator (FES) which they can run both as Model In the Loop (MIL), where everything is virtual or Hardware In the Loop (HIL), where the GNC program runs on the flight computer connected to some of the existing real sensors.

As many of the rockets parameters have considerable uncertainties, especially for the first launch, and as the exact environmental conditions are always difficult to predict precisely, it is of paramount importance to test as many flight combinations as possible. The method used

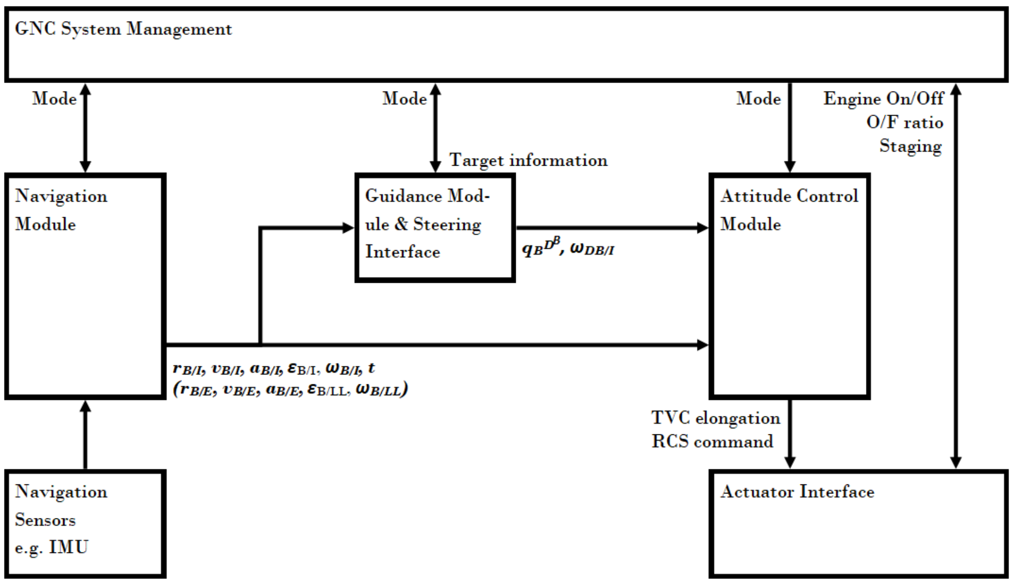

GNC System Management
Mode
Mode
Mode
Engine On/Off
O/F ratio
Staging
Target information
Navigation Module
Guidance Module & Steering Interface
$q_{B}{}^{D^{\beta}}, \omega_{DB/I}$
Attitude Control Module
$r_{B/I}, v_{B/I}, a_{B/I}, \varepsilon_{B/I}, \omega_{B/I}, t$
$(r_{B/E}, v_{B/E}, a_{B/E}, \varepsilon_{B/LL}, \omega_{B/LL})$
TVC elongation
RCS command
Navigation Sensors e.g. IMU
Actuator Interface

Figure 1.1: GNC block diagram [4]

for this purpose is called the Monte-Carlo method, where parameters are varied randomly according to some estimated uncertainty distributions. The analysis of these results can be fed back into the design process and can serve the verification activities.

In order to clearly define the bounds of this research, first we have to ask ourselves the question: What is guidance? One of the definitions of modern aerospace guidance is as follows:

"Guidance is about the determination of the maneuvering commands to steer the vehicle to fly a trajectory that satisfies the specified terminal/targeting condition as well as other pertinent constraints, and, if required, optimizes a defined performance." [5]

In particular for RFA ONE the guidance system is responsible for generating the attitude and attitude rate commands. These are then sent to the controller which transforms them into appropriate TVC and RCS commands. The guidance is therefore mainly concerned about the translational dynamics of the rocket limited to three degrees of freedom (3-DOF) and leaves the much faster rotational dynamics to the controller module. Due to the complexity of finding an optimal trajectory for the endo-atmospheric flight, that part is optimized offline and the open-loop guidance just tracks the reference trajectory. However from the second stage onward the aerodynamic effects become negligible and the much simpler exo-atmospheric optimal control problem is therefore solved onboard in a closed loop manner to ensure precise orbit insertion. The main focus of this study lies on the closed-loop part of the guidance. The inputs for the guidance module come from the navigation module, which estimates the translational and rotational state of the rocket, and the GNC state-machine, which defines the operating modes of the various modules.

1.3. Mission overview

The rocket that RFA is developing is called RFA ONE, which is a three stage rocket capable of lifting 1300 kg payload into a 500 km Sun-Synchronous Orbit (SSO)[1]. The primary launchpad will be Saxavord Spaceport on the Shetland Islands[2]. The first two stages largely share the

[1]https://www.rfa.space/launcher/
[2]https://www.rfa.space/rocket-factory-augsburgs-first-launch-to-take-place-from-saxavord-spaceport/

same engine technology and stage structures, leading to a fairly high thrust-to-weight ratio on the second stage. On the other hand the third stage has been optimized for long in-orbit operations and therefore has a much lower thrust-to-weight ratio. The first couple of flights will however mainly serve as a technology demonstration and will be done with slightly scaled down version, shown in Figure 1.2, which will only insert a 150 kg payload into orbit[3]. Moreover for this first flight the payload itself will primarily consist of small cube-sats which are themselves in-orbit demonstrators. There are therefore no stringent requirements on the orbital injection accuracy and the focus rather lies on getting to orbit in the first place. It is expected that the propellant of the third stage will be almost entirely depleted during this maiden flight, which means that the closed-loop guidance needs to be capable of dealing with very long thrust-arcs (60°). While coast arcs are quite common for rocket ascents into orbit, the current baseline trajectory doesn't include any.

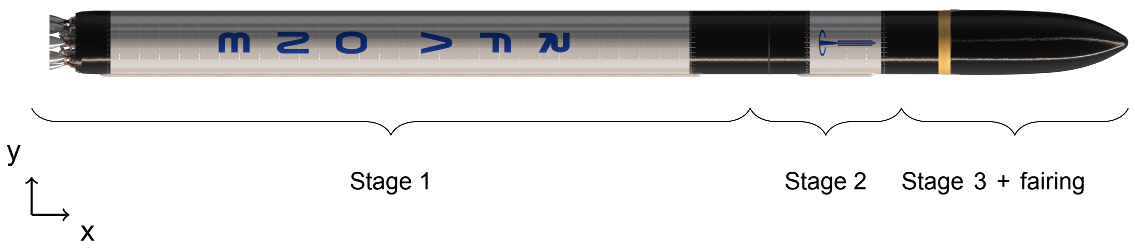


Figure 1.2: RFA ONE baseline configuration

1.4. Thesis outline

First an overview of the current state of the art for rocket guidance systems as well as the existing guidance system at RFA is given in Chapter 2. Chapter 3 then provides a brief summary of the reference frames and dynamics models used throughout this thesis and Chapter 4 goes through the theoretical background on which the guidance algorithms are based. The old guidance algorithm as well as the newly developed algorithms together with their derivations will then be presented in Chapter 5. The next three chapters contain the overall results, with Chapter 6 focusing on the performance of the algorithms, Chapter 7 the verification of the implementations, and Chapter 8 the sensitivity to changes in models or parameters. Finally the conclusions and recommendations are given in Chapter 9.

[3]`https://www.rfa.space/fully-booked-dlr-selects-seven-customers-for-rfas-inaugural-flight/`

2 State of the art and existing system

This chapter provides information on the state of the art of guidance systems found in literature. Then follows a brief overview of the existing guidance system at RFA that this thesis will try to improve. There the issues with the current algorithms will also be shown. Finally the research questions that this thesis aims to answer will be formulated at the end of this chapter.

2.1. Closed-loop guidance systems

This section will briefly summarize some of the findings on different closed-loop guidance methods that have been developed. They all have in common that they seek to solve the exo-atmospheric guidance problem. This involves finding the optimal values for unit-thrust direction $\hat{\mathbf{P}}$ and thrust magnitude T such that the cost function, Equation (2.4), is minimized while satisfying the equations of motion, Equations (2.1) to (2.3) and the terminal constraints Equation (2.5) [6].

$$\dot{\mathbf{V}} = a_T\hat{\mathbf{P}} + \mathbf{g}(\mathbf{r}), \quad a_T = \frac{T}{m} \tag{2.1}$$

$$\dot{\mathbf{r}} = \mathbf{V} \tag{2.2}$$

$$\dot{m} = \frac{T}{v_{ex}} \tag{2.3}$$

$$J = \int_{t_0}^{t_f} \dot{m}dt = \int_{t_0}^{t_f} \frac{T}{v_{ex}}dt \tag{2.4}$$

$$\boldsymbol{\psi}(\mathbf{r}(t_f), \mathbf{v}(t_f), t_f) = \mathbf{0} \tag{2.5}$$

The most common solution to this problem is found using optimal control theory for a linearized gravity model resulting in the co-states following Equation (2.6) and the unit-thrust direction then being given by Equation (2.7). Furthermore it can also be shown that thrust should either be maximized or set to zero, which implies a coast phase [6].

$$\mathbf{u} = \boldsymbol{\lambda}_V(t) = \hat{\boldsymbol{\lambda}}\cos\left(\omega(t - t_\lambda)\right) + \frac{\dot{\boldsymbol{\lambda}}}{\omega}\sin\left(\omega(t - t_\lambda)\right) \tag{2.6}$$

$$\hat{\mathbf{P}} = \frac{\mathbf{u}}{||\mathbf{u}||} \tag{2.7}$$

2.1.1. Powered Explicit Guidance

During the development of the Space Shuttle a new versatile guidance algorithm was developed that was capable of addressing all guidance modes ranging from ascent to abort as well as in-orbit manoeuvres. It was called the Powered Explicit Guidance (PEG) since it did not require a reference trajectory or even a descent initial guess. It used a predictor-corrector scheme in which the position and velocity integrals are simplified such that the steering parameters can be solved for analytically. Below is an overview of the key concepts. The equations where first developed by Jaggers [6] and the mechanization is well described by McHenry [7] and Brand [2]. This section will just give a brief overview of the algorithm. For a more in depth description and derivation, the reader is advised to read Section 5.2.1.

Two simplifying assumptions are made for the PEG. The first is that the thrust-arcs are small which means that the cosine and sine in Equation (2.6) can be linearized, as shown in Equation (2.8). The second is that the two steering vectors $\hat{\boldsymbol{\lambda}}$ and $\dot{\boldsymbol{\lambda}}$ are orthogonal to each other, which is a simplification of the transversality conditions which holds true for insertion into low eccentricity orbits. The resulting guidance law is essentially the vector form of the classical Linear Tangent Guidance (LTG) [8]. A visual representation is shown in Figure 2.1. The PEG revolves around finding the values for the three steering parameters ($\hat{\boldsymbol{\lambda}}$, $\dot{\boldsymbol{\lambda}}$, t_λ) such that the predicted position and velocity at cut-off satisfy the target constraints.

$$\mathbf{u} = \hat{\boldsymbol{\lambda}} + \dot{\boldsymbol{\lambda}}(t - t_\lambda) \tag{2.8}$$

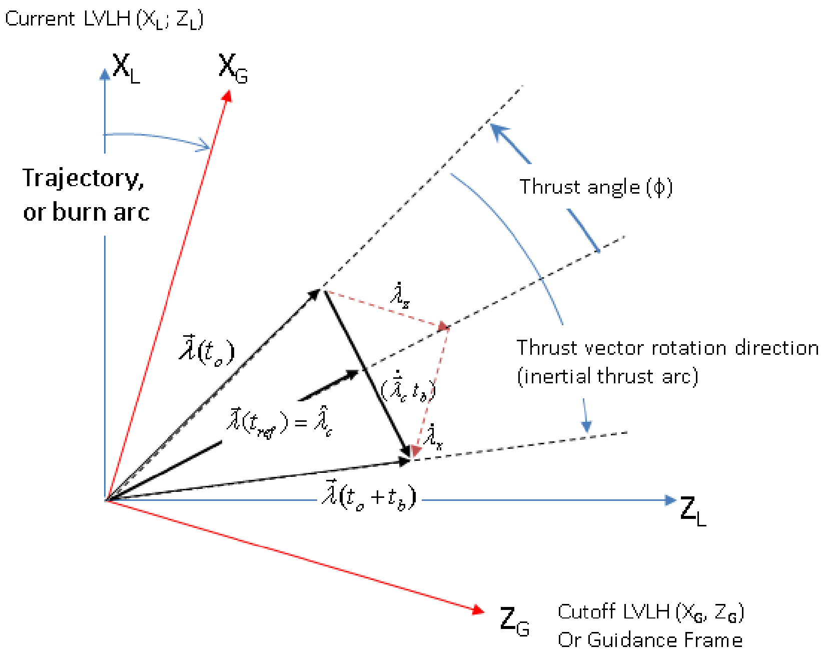

Current LVLH (X_L; Z_L)
X_L
X_G
Trajectory, or burn arc
Thrust angle (ϕ)
$\dot{\lambda}_z$
$\vec{\lambda}(t_o)$
Thrust vector rotation direction (inertial thrust arc)
$(\dot{\hat{\lambda}}_c\, t_b)$
$\vec{\lambda}(t_{ref}) = \hat{\lambda}_c$
$\dot{\lambda}_x$
$\vec{\lambda}(t_o + t_b)$
Z_L
Z_G
Cutoff LVLH (X_G, Z_G) Or Guidance Frame

Figure 2.1: PEG LTG Ascent Geometry [3]

To be able to evaluate the position and velocity integrals analytically, the unit-thrust direction defined in Equation (2.7) is approximated using the second order Taylor series expansion. This allows the integrals of the position and velocity increments due to the thrust to be expressed as a function of the steering parameters and a set of scalar integrals of the thrust acceleration. These acceleration integrals can be evaluated analytically in an efficient recursive manner [2]. After some simplification the steering parameters can then be solved for given values of the velocity and position integrals. One of simplifications is that t_λ is chosen such that the velocity integral is only a function of $\hat{\boldsymbol{\lambda}}$. The other simplification is to split off the quadratic terms into bias values which are not used when solving for the steering parameters. This leads to the linearized integrals $\mathbf{V}_{go}$ and $\mathbf{r}_{go}$, of which $\mathbf{V}_{go}$ is updated and $\mathbf{r}_{go}$ is re-computed every iteration [7].

Using the steering parameters the predicted final position and velocity can be computed analytically using the bias terms and approximated gravity integrals. Then the desired final position is obtained by correcting the predicted final position using the position constraints. Finally the desired velocity at cut-off is computed such that it satisfies the velocity constraints and $\mathbf{V}_{go}$ is corrected based on the difference between predicted and desired velocities [2].

2.1.2. SLS Guidance

The PEG algorithm was again selected as the baseline for the closed-loop guidance for SLS [9]. While it could be used with little modification on the SLS Block-1 variant due to its similar performance characteristics to the Space Shuttle, for the upcoming Block-1B variant a number of changes were required.

This is primarily due to the use of the Exploration Upper Stage (EUS) as the upper stage, which due to its lower thrust to weight ratio (TWR) caused much longer burn arcs. Whereas the burn arc for a typical Space Shuttle mission was 15°, for the ascent burn of the SLS with EUS it is around 35°. In some cases this would be even longer, with long in-orbit burns or engine-out scenarios reaching burn-arcs of 45°. This creates several convergence issues for the PEG, as some of the assumptions(such as a flat Earth), no longer hold [3]. The primary solution that was used to solve this issue was to divide the ascent arc into two segments with each sub-arc having its own set of target constraints. In this ”two target” approach the PEG would get an intermediate target during the Core Stage flight, which requires a flight performance reserve, and only receives the final target during the EUS flight. This improved convergence by shortening the length of each arc and reduced dispersion at the end of the Core Stage (CS) flight. Another method for improving the convergence is to apply a scaling factor to the correction of $\mathbf{V}_{go}$. The PEG is in essence equivalent to a Newton-Raphson root-finding scheme where the Jacobian is assumed to be the identity matrix. This assumption breaks down for long burn-arcs, resulting in over-corrections requiring many more iterations to converge. A scaling factor of 0.5 was empirically found to provide a more reliable and rapid convergence compared to the default logic [3].

In the rare event that the PEG still wouldn't converge, the initial versions would engage an attitude hold mode until it did converge. While this could be considered as the safest solution, it causes spikes in the pitch-rate when attitude hold is released as shown in Figure 2.2. As an alternative it is also possible to re-use the steering parameters from a previous call to the PEG, on the condition that it had converged before. As long as the predictions are sufficiently accurate the steering parameters can remain valid for some time. It essentially means that the guidance is temporarily functioning in a semi-open loop mode. This not only eliminates the pitch-rate spikes, but it has also been shown that orbit insertion can still be reasonably accurate even if the steering parameters are held constant until main engine cut-off [3].

Long burn-arcs also pose issues when constructing the turn-rate vector $\dot{\boldsymbol{\lambda}}$. This is best seen by taking a look at the in-plane geometry of the steering law used by the PEG, shown in Figure 2.1. As was shown in previous sections the PEG places the $\hat{\boldsymbol{\lambda}}$ vector in the same direction as $\mathbf{v_{go}}$. With $\hat{\boldsymbol{\lambda}}$ therefore being primarily defined by the velocity constraints, this leaves $\dot{\boldsymbol{\lambda}}$ to be found to satisfy the position constraints, primarily the altitude constraint. One of the assumptions made to simplify finding an analytical solution is the orthogonality between $\hat{\boldsymbol{\lambda}}$ and $\dot{\boldsymbol{\lambda}}$. This approximation of one of the transversality conditions is valid in most cases, especially for insertion into circular or near-circular orbits. However for the optimal solution of this ascent problem with no down-range position constraint, $\dot{\boldsymbol{\lambda}}$ shouldn't have any downrange component.

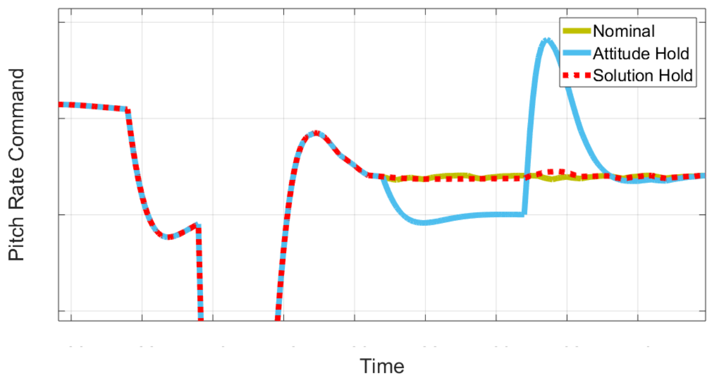


Figure 2.2: Pitch Rate Commands Around LAS Jettison with Failed PEG Convergence [10]

This means that for very low thrust-to-weight levels, i.e. long thrust arcs, two of the basic assumptions underpinning the PEG start to break down. The first is that for lower thrust-to-weight ratios $\hat{\boldsymbol{\lambda}}$ needs to point in a more radial direction relative to the cut-off reference frame in order to compensate for the gravity losses caused by the longer burn time. The result is that the component of $\dot{\boldsymbol{\lambda}}$ aligned with the radial direction gets proportionally smaller, which in turn leads to a larger overall magnitude of $\dot{\boldsymbol{\lambda}}$ to compensate for its diminished influence on the altitude constraint. This is further exacerbated by the long thrust arcs breaking the flat Earth assumptions, as the curvature in the trajectory also causes the $\hat{\boldsymbol{\lambda}}$ to point more radially with respect to the cut-off frame. While these issues could be addressed by modifying the PEG algorithm, this would compromise the algorithmic simplicity that made it so appealing. Instead, these issues are mitigated by adding safe-guards when constructing the turning rate vector, with the main purpose to limit its magnitude. These safe-guards remain active until either the thrust acceleration increases or the remaining thrust arc becomes short enough to approximate a flat Earth. Some of these were already developed during the Space Shuttle era, albeit in a more simpler form. It has been shown that these safe-guards only marginally increase the propellant consumption with respect to the optimal trajectory [3]. The main drawback of these methods is that the position constraints are partially relaxed while the limits are active, which leads to inaccurate burn-time predictions at the start of the flight, as illustrated in Figure 2.3.

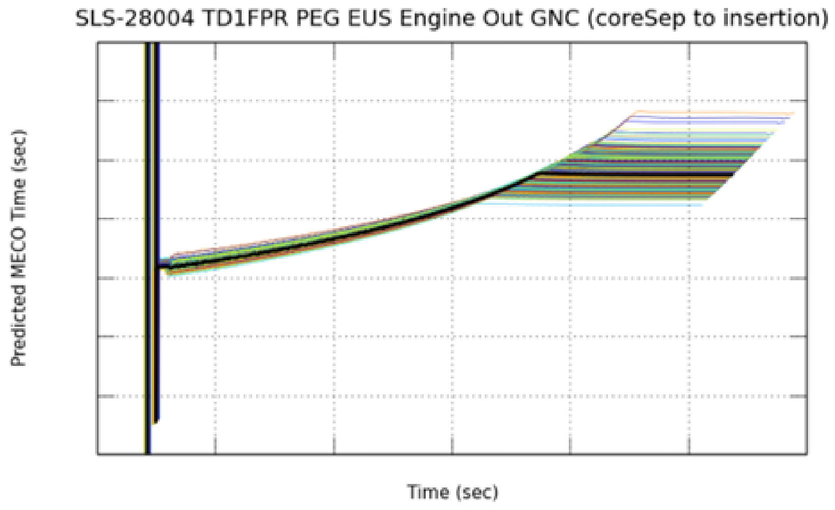


Figure 2.3: Predicted SECO-1 times for a single engine failure at the start of the EUS Ascent Burn [9]

Another improvement was the estimation of the so-called thrust factor, which could then be used to scale the mass-flow rate used in the PEG's predictor. The thrust factor is estimated using the sensed acceleration magnitude and the estimated mass. The measurement noise is then smoothed by applying a simple low-pass filter. The basic principle is that while in vacuum, the acceleration sensed by the IMU will be almost entirely caused by the thrust. One notable exception is when the body angular rates are high, which is typically the case at the start of the closed-loop guidance. Then the lever-arm between the Center of Mass (CoM) and IMU location will induce an additional acceleration. This can be mitigated by simply holding the thrust factor constant in this initial period. This method not only improves the predictions of the PEG, thereby improving performance, but it also provides implicit protection against different engine anomalies such as an engine out or stuck throttle valve [3].

In order to maximize the mission success probability and ensure a safe flight in every scenario, some contingencies where implemented to handle engine outs. This includes the use of two alternate orbit targets besides the nominal one, similar to the multiple abort modes on the Space Shuttle. The alternate targets were selected such that as many mission objectives could be satisfied as possible. The guidance would then select between the three targets based on pre-determined velocity thresholds at the instant of engine out. These velocity thresholds would be first determined using 3-DOF simulations and then validated using a 6-DOF Monte-Carlo analysis [10]. To further improve the performance in engine out scenarios the use of a pitch bias during the open-loop phase was also investigated. A second-degree polynomial was found to be sufficient for this purpose. While initially it was thought that the coefficients could be determined from the 3-DOF optimized trajectories, it was later found that a grid-search with the 6-DOF simulations yielded better results [11].

2.1.3. Other guidance systems

In recent years there hasn't been a lot of development on completely new closed-loop exo-atmospheric ascent guidance algorithms. Instead most of the research has been concentrated on other more complex problems such as endo-atmospheric ascent guidance [12, 13], re-entry guidance [14, 15, 16, 17], or in-orbit guidance [18, 19, 20]. While the additional constraints or dynamics involved in solving these problems would lead to unnecessary complexity of the algorithm if applied to the exo-atmospheric ascent case, it is still possible to obtain some hints for possible improvements to the exo-atmospheric algorithm. For example the optimization of coast-arcs for orbit transfers could be useful for insertion into higher altitude orbits. Furthermore some of the methods for ensuring a more robust convergence in highly constrained and hyper-sensitive scenarios could also be useful.

Delporte and Sauvinet [21] developed an alternative way of solving for the steering vectors of the PEG. They created an algorithm where the flight is split into multiple segments with constraints, such as target orbit or radar visibility, at the end of each segment. The unknowns vector is composed of the steering vectors and final time instead of $\mathbf{v}_{go}$. The thrust integrals are integrated in a more exact manner by keeping the normalizing factor, however this means that the steering parameters cannot be easily solved for as in the PEG. For the gravity integrals analytical expressions are obtained by expressing the gravity as a third order polynomial function of time. It still uses a predictor-corrector scheme, but instead of assuming that the Jacobian is the identity matrix it computes the sensitivity matrix analytically from the formulation of the thrust and gravity integrals. The steering vectors are then obtained by solving the system of equations.

Another interesting direction of research are the algorithms first developed by Dukeman [22, 23, 13] and then improved by Lu and Pan [24, 25]. While most of the initial research was focused on the guidance for a space-plane, including the endo-atmospheric guidance, the methods for solving the burn-coast-burn problem as a multiple-shooting root-finding problem are quite interesting. This allows them to be solved using standard robust root-finding algorithms such as Powells trust-region dog-leg method. What they all have in common is that the unknowns are the velocity and position co-state at the initial time as well as the free final time. For the multiple shooting version the state and co-state at the nodes as well as the coast end time are additional unknowns. There is also a fully analytical formulation for the state and co-state prediction, including an approximation of the effect of inverse square gravity on the co-state [26]. It differs from the PEG in that it uses a second order expansion of the sines and cosines and doesn't make the assumption of orthogonality of the two unknown steering vectors. This approximation can be improved by splitting the integration arc into multiple pieces. In this version it is also suggested to solve for the coast time using parabolic optimization in an outer loop. A mix of these algorithms has been developed as a replacement for the PEG for RFA ONE as described in Section 5.2.2.

Another advantage is the improved flexibility of the target constraints by treating the transversality conditions as an additional set of constraints for the root-finding problem instead of embedding them into the construction of the steering vectors. This is most useful when used in combination with a list of reduced transversality for any combination of missing target constraints [27]. Although Dukeman [13] suggests that it is more favorable to the user to solve for the transversality vectors numerically, as this would allow for easier switching constraints and allow a larger range of options.

During the development of SLS there were initially some doubts as to the suitability of the PEG, in large part due to the much longer thrust-arcs on the Block-1B variant with the EUS. A trade study was performed in which the PEG was compared to OPGUID, a competing algorithm that had initially been developed as an alternative to the IGM used for Apollo but which had eventually been shelved. While the exact details of OPGUID aren't available, from its description it seems to bear some resemblance to the algorithms developed by Dukeman and Lu in that it makes much less assumptions than the PEG. While it initially used a fourth order variable step-size integration scheme for the prediction, it was modified to perform the propagation analytically using segments of 100 seconds. During the trade-study several improvements were made to the PEG, many of which have been described in the previous section. While this made the PEG more robust it came at the cost of inaccurate burn-time predictions during the early part of the flight as well as requiring a performance reserve on the core stage. While OPGUID scored better in the performance related categories, in the end the PEG was selected in large part due to its flight heritage [9].

2.2. RFA baseline guidance system

The work presented in this thesis is built upon the existing guidance algorithm at RFA. This algorithm was based on the original formulation of the PEG for the Space Shuttle program [2]. This means that it didn't include any of the modifications that were made throughout the Space Shuttles lifetime or those developed for SLS, such as the turning rate limiters or the scaling factor in the corrector. Some other modifications specific to RFA were implemented instead. The two most important ones are the use of a numerical integrator for the prediction of the final state and the decision to drop the velocity bias term [4]. The first modification was mainly motivated by the availability of vastly more onboard computing power. For the long thrust arcs of RFA ONE this showed an improvement both in terms of insertion accuracy and in terms of performance. Furthermore by removing the semi-analytical formulation for the gravity integrals the algorithmic complexity was reduced and more complex gravity models could be more easily integrated. The second modification might seem somewhat contradictory if improved performance was the goal, since the steering parameters wouldn't get constructed using the correct values. However Stein claimed that dropping this term showed more accurate and robust performance [4], although this claim isn't supported by any numerical results or even a thorough analysis. This can be explained in part by the fact that not using the bias term would cause the PEG to partially relax both velocity and position constraints and $\mathbf{V}_{miss}$ wouldn't converge to zero. This had the effect of significantly shortening the predicted thrust arcs from $60°$ to $45°$, which indeed made it more robust. On top of that the initial thrust angles would also be a bit lower, which improved performance a bit. However of course the predicted time of insertion would be underestimated as long as the bias terms were still large, that is until the start of the third stage. In essence this modification had roughly the same affect as limiting the magnitude of the turning rate.

Most of these issues can be identified when looking at the pitch profiles from Monte-Carlo simulations, shown in Figure 2.4. During the first stage flight all the simulations follow an almost identical profile, as the guidance is in open-loop mode. However after first stage separation, which happens at $\sim 180\,s$, the guidance mode is switched to closed-loop and the first issues emerge. For some of the simulations the PEG fails to converge, which leads to nonsensical attitude commands being issued which causes the rocket to become unstable and eventually fall back to the ground. Even the ones that do converge are however not entirely without fault. It can be seen that the pitch at the start of the second stage shows very little dispersion, whereas at the end of the second stage at $\sim 700\,s$ the dispersion becomes quite large. This is caused by the lack of parameter estimation, which means that PEG can effectively only take into account the past performance variations through its effect on the velocity and position. As the difference between nominal and actual thrust acceleration becomes larger towards the end of the stage, so does the variation in pitch angle. After second stage separation the lack of parameter estimation is less of an issue as the flight path is dominated by gravity. However the previously accumulated errors continue to cause sub-optimal pitch angles in order to ensure insertion into a circular orbit. On the high end there are pitch angles of $45°$ which would lead to large steering and gravity losses. On the low end the pitch angle goes negative as it tries to reduce excess vertical velocity in order to circularize.

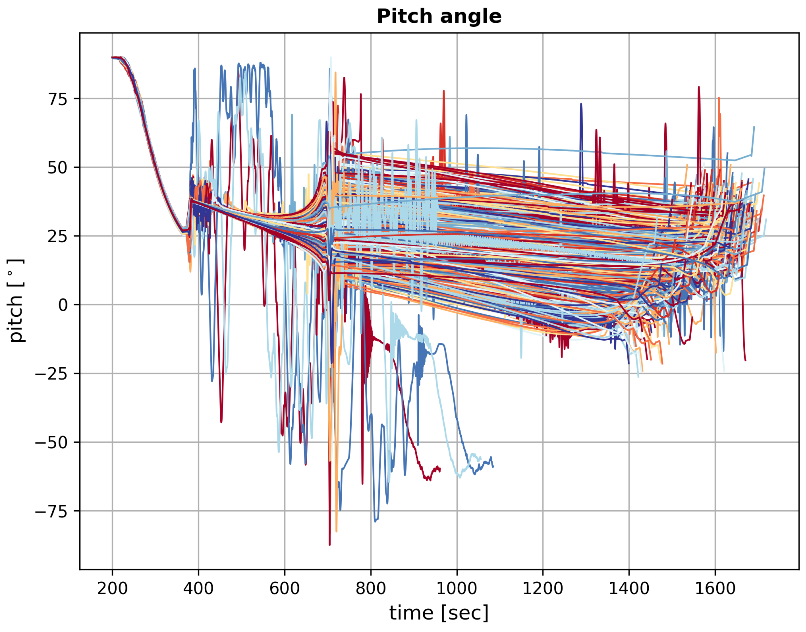


Figure 2.4: Pitch profile baseline Monte-Carlo results

Between the previously mentioned inefficient pitch profiles, sub-optimal staging times due to incorrect burn-time estimation, and generally small margin, the rocket fails to reach a stable orbit in a significant amount of simulations. As shown in Figure 2.5, while it does generally achieve or even exceed the desired apoapsis altitude, the periapsis altitudes are sometimes too low.

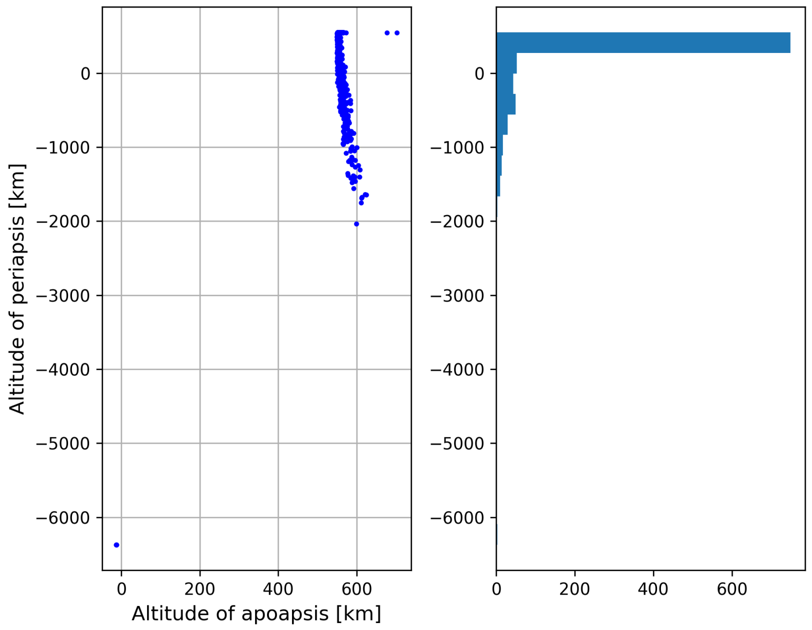


Figure 2.5: Final apoapsis and periapsis altitudes baseline Monte-Carlo results

2.3. Research Questions

As has been shown in the previous sections, the topic of exo-atmospheric closed-loop guidance has been the subject of study ever since the advent of rocket flight. However there are still some areas where improvements are possible. Namely on the topic of designing a guidance system for very long thrust-arcs. Most launchers up till now have used upper stages with relatively high TWRs, leading to short burn-arcs which considerably eases convergence. However with the selection of the very low TWR upper stage on RFA ONE, the long thrust arc causes the simplifying assumptions to break down and makes convergence more difficult. The solutions used to solve this issue thus far have either led to reduced performance, due to the necessity of flight performance reserves on intermediate stages, or in-accurate in-flight performance predictions, due to the partial relaxation of the constraints early in flight. A second point of research interest is how to deal with alternate target orbits in the presence of large parameter uncertainties. In previous research the on-board decision process has been based purely on the conditions at which some discrete event occurred. However this fails to take into account the performance of the remaining stages, of which a better estimate can be made during flight. Furthermore it fails to address the case of generally under-performing rocket with minimal performance reserves. This leads to the following research question:

What is required to create a robust exo-atmospheric closed-loop guidance system that can handle large parameter uncertainties and loose orbit insertion requirements?

This question can be broken down into three sub-question, each of them covering a specific area of improvement.

1. How can the convergence be improved for long thrust arcs?
   (a) Do different guidance laws yield better results than Linear Tangent Guidance?
   (b) Is it possible to have an algorithm with both good convergence and accurate predictions?
   (c) Can the two-target approach be used with minimal performance reserves?
2. How can the rocket performance parameters be estimated in-flight?
   (a) What measurements are best suited to perform parameter estimation?
   (b) What filtering techniques are best suited for parameter estimation?
3. How should the guidance system compensate for an under-performing rocket?
   (a) Can alternate target orbits be used without relying on discrete performance reducing events?
   (b) On what basis can the guidance system decide to go for one of the alternate target orbits?
   (c) Is it required to bias the open-loop trajectory?

3 Flight dynamics

This chapter will cover the underlying theory required to describe the motion of the rocket during the exo-atmospheric part of the flight, that is while it is outside of the atmosphere. First some useful reference frames are defined(Section 3.1) and then the applicable forces are discussed(Section 3.2) followed by the resulting equations of motion(Section 3.3). Finally a brief overview of orbital elements is given in Section 3.4.

3.1. Reference Frames

In order to describe the motion of any object, it is necessary to define the meaning of the objects state variables. This is done by defining a reference frame which has an origin and an orientation. For Cartesian reference frames this involves defining the origin and direction of three orthogonal unit vectors $\hat{\mathbf{i}}_x$, $\hat{\mathbf{i}}_y$, and $\hat{\mathbf{i}}_z$. Any non-scalar quantities, such as position or velocity, can then be expressed as a vector with three elements into those directions. While the Newtons laws of motion are only valid for inertial frames, it is often useful to define other non-inertial frames to aid in derivation or express forces in. Below is a list of reference frames which are useful for the purpose of exo-atmospheric ascent guidance as well as any relevant transformations.

3.1.1. Earth Centered Earth Fixed

For the purposes of locating an objects position with respect to the Earths surface one of the Earth Centered Earth Fixed (ECEF) frames can be used. These have their origin at the center of gravity of the Earth and rotate with the surface of the Earth making them non-inertial. Their axis are generally defined as follows [28]:

- $\mathbf{X}_R$ lies in the equatorial plane and points towards the prime meridian, which passes through Greenwich
- $\mathbf{Y}_R$ completes the right-handed system and therefore lies in the equatorial plane
- $\mathbf{Z}_R$ points north along Earths axis of rotation

The most widely used ECEF frame is the WGS84 frame, which amongst other things is used by the Global Positioning System (GPS). This is the primary reference frame used by the rockets navigation system. Furthermore this reference frame is used when computing the gravity acceleration with one of the more accurate models.

3.1.2. Earth Centered Inertial

For the purposes of simulation and derivation of guidance equations it is most useful to define the equations of motion with respect to an inertial reference frame. For a rocket ascent from the Earth the Earth centered inertial (ECI) reference frames are used. Their common point is that they all have their origin at Earths center of gravity, however they differ in the orientation of their axis. The most commonly used ECI frame is ECI-J2000 which is defined with the Earths mean Equator and mean Equinox at 12:00 Terrestrial Time on January 1st 2000, with axis defined as follows [28]:

- $\mathbf{X}_I$ lies in the equatorial plane and points towards the Vernal equinox
- $\mathbf{Y}_I$ completes the right-handed system and therefore lies in the equatorial plane
- $\mathbf{Z}_I$ points north along Earths axis of rotation

For the guidance it can be useful to define other inertial reference frames, such as ECI-T0. This frame is defined as the ECEF frozen at the instant of lift-off. There are a couple advantages to using this frame over ECI-J2000. Firstly it allows for the definition of reference trajectories that are independent of the time of launch, which is mostly useful for the open-loop guidance. However since switching the reference frames mid-flight could cause unforeseen consequences it is still used for the closed-loop guidance as well. As long as the longitude of the ascending node of the target orbit isn't constrained, there isn't any effect with regards to the definition of the target orbit constraints. A second benefit is that as previously mentioned the navigation system operates in the ECEF frame. The transformation between ECI-J2000 and ECEF can be quite complex due effects such as precession, nutation, and polar motion. On the other hand the transformation between ECI-T0 and ECEF is trivial as it only involves the rotation around a single axis, shown in Equation (3.1) where Ω_E is the Earths rotation rate and Δt is the time since launch.

$$\mathbf{T}_{ECEF}^{ECI-T0} = \mathbf{T}_3(-\Omega_E \Delta t) \tag{3.1}$$

3.1.3. Local Level

The local level frame, also called the North East Down frame, is a vehicle carried frame that often gives useful insights during analysis and is sometimes used for guidance purposes. Its main characteristic is that the XY-plane is the local horizontal plane with respect to an ellipsoidal Earth. The axis are then defined as [29]:

- $\mathbf{X}_L$ lies in the horizontal plane and points north
- $\mathbf{Y}_L$ completes the right-handed system, which means it lies in the horizontal plane and points east
- $\mathbf{Z}_L$ points along the local vertical downwards

While the pitch and roll angles of the rocket with respect to the local level frame are often useful quantities to analyze, the yaw angle is less informative especially when passing close to the poles. Furthermore the roll and yaw angles between the local level and body frames have a singularity when the rocket is vertical, such as on the launch-pad.

3.1.4. Body

In order to express the attitude of the rocket, the body frame is often used. The attitude and angular rates are then defined as the rotation of the body frame with respect to one of the inertial frames. The origin is at the rockets center of mass. Since the rocket is axisymmetric, the only well defined axis is the X-axis, which according to aerospace convention lies along the longitudinal direction of the rocket. To define the outboard axis the tower umbilical was used yielding the following definition [28]:

- $\mathbf{X}_B$ lies along the longitudinal axis of the rocket and is positive in the direction of motion
- $\mathbf{Y}_B$ points in the direction of the tower umbilical
- $\mathbf{Z}_B$ completes the right-handed system

3.1.5. Propulsion

One of the more uncommon frames is the propulsion frame, which is primarily defined by having the X-axis point along the thrust direction. While this frame might seem redundant for rockets due to its similarity with the body frame, it is quite important in the context of guidance. This is due to the thrust direction often being the main control variable in the closed loop guidance algorithms, which means that the output is usually the desired thrust direction and its time derivative. For a fully symmetrical rocket it would be sufficient to assume that the thrust direction coincides with the body x-axis. However rockets are not always fully symmetrical either by design, such as the Space Shuttle, uncertainty in the vehicle parameters, such as IMU misalignment, or due to an in-flight anomaly, such as an engine out. If this difference isn't accounted for there will be an undesired lateral acceleration which will lead to the trajectory diverging from the desired one.

Since the out-board axis aren't very relevant they can be chosen in a convenient direction. In order to make the propulsion frame consistent with the Local Level frame it can be defined as such [28]:

- $\mathbf{X}_P$ points in the desired direction of the thrust
- $\mathbf{Y}_P$ completes the right-handed system and is therefore perpendicular to the pitch plane
- $\mathbf{Z}_P$ lies within the pitch-plane and is positive downwards

While the definition above is fine for the purposes of closed-loop guidance, it has a singularity when the rocket is vertical, such as on the launch-pad. In that case it is better to define $\mathbf{Y}_P$ instead of $\mathbf{Z}_P$ and aligning it for example with the target orbit normal. Another useful property of this frame is that it is equivalent to the body frame with zero roll with respect to the local level. It can therefore form the basis for a roll program with the roll angle being defined with respect to this frame.

3.1.6. Cut-off (guidance)

Another uncommon frame is the cut-off reference frame, sometimes called the guidance frame. It is defined by the translational state of the rocket at orbit insertion. This is often the point in time at which target constraints are expressed so it can be useful to define some of the relevant quantities in this frame. Unless both position and velocity at cut-off are fully constrained, it is defined using a mix of target constraints and predicted state at cut-off which means its orientation can vary throughout flight and even between iterations of the guidance algorithm. The principal axis are defined as follows [2]:

- $\mathbf{X}_G$ lies within the orbital plane an is co-directional with the rockets position vector.
- $\mathbf{Y}_G$ is perpendicular to the orbital plane and points in the opposite direction to the angular momentum
- $\mathbf{Z}_G$ completes the right-handed system and therefore lies in the orbital and local horizontal planes and is positive in the direction of motion

3.2. Applicable Forces

This section will give a brief overview of the applicable forces for exo-atmospheric flight. While there are many more forces acting on the rocket throughout its ascent, they are mostly limited to the endo-atmospheric part of the flight. Furthermore for the guidance these forces only need to be modeled to a degree that the resulting guidance solution is close to optimal.

3.2.1. Gravity

One of the two primary forces acting on the rocket during exo-atmospheric flight is gravity. The simplest model for the gravity force is the central gravity model, where the Earth is assumed to be a perfect sphere with a homogeneous mass distribution. The acceleration vector under this assumption is then given by Equation (3.2) where μ is the standard gravitational parameter and $\mathbf{r}$ the position vector with respect to the center of the Earth.

$$\mathbf{g} = \frac{\mu}{||\mathbf{r}||^3}\mathbf{r} \tag{3.2}$$

If the ascent trajectory covers a large distance either due to the use of a long coast arc or due to a low thrust to weight ratio causing long thrust arcs, the gravity model above may no longer be sufficiently accurate. For this purpose more accurate gravity models exist, such as the WGS84-based model and zonal harmonics models. The WGS84 makes the same assumption of uniform mass distribution, but uses the WGS84 reference ellipsoid as the Earths shape. There exist several approximations of this model, although they lose their accuracy at higher altitudes which means the exact formulation would be required for the purposes of ascent guidance.

If further accuracy is desired it is also possible to model the gravity potential using spherical harmonics. This allows for better modeling of large surface features such as mountain ranges and uneven mass distribution. The gravity potential is then computed using Equation (3.3) [30] where θ is the geocentric latitude, ϕ the geographic longitude, R_e the mean Earth radius, P_{nm} Legendre polynomials, and J_n, C_{nm}, S_{nm} the model coefficients. The gravity acceleration is then obtain through differentiation of the gravity potential. While using the J_2 inside the guidance can be useful for the same reasons mentioned for the WGS84 model, other higher terms cause only a minor effect which means they only need to be included in the plant model of the simulation. In the end it comes down to a trade-off between computational efficiency and accuracy of the guidance predictions in which other sources of errors, such as numerical integration error, also need to be taken into account.

$$\begin{aligned} U(r,\theta,\phi) = -\frac{\mu}{r}\Bigg(1 + \sum_{n=2}^{\infty}\left(\frac{R_e}{r}\right)^n J_n P_{n0}(\sin(\theta)) \\ + \sum_{n=2}^{\infty}\sum_{m=1}^{n}\left(\frac{R_e}{r}\right)^n \left(C_{nm}\cos(m\phi) + S_{nm}\sin(m\phi)\right) P_{nm}(\sin(\theta))\Bigg) \end{aligned} \tag{3.3}$$

3.2.2. Thrust

The other major force acting on the rocket is the thrust force produced by one or multiple engines. The thrust can be computed using Equation (3.4), where the first term, which depends on the mass-flow rate $\dot{m}$ and specific impulse I_{sp}, is the thrust in vacuum and the second term is loss due to atmospheric pressure, which depends on the nozzle exit area A_e and the exit and ambient pressures p_e and p_0. For exo-atmospheric guidance only the vacuum thrust is considered as the pressure loss is negligible.

$$T = \dot{m} I_{sp} g_0 + (p_e - p_0) A_e \tag{3.4}$$

The mass-flow rates for the oxidizer and fuel can then be computed using Equation (3.5), where O/F is the oxidizer to fuel ratio. It might be required for the guidance to keep track of the oxidizer and fuel levels separately in order to compute accurate burn-times.

$$\dot{m}_{OX} = \dot{m}\frac{O/F}{O/F+1} \qquad \dot{m}_{FU} = \dot{m}\frac{1}{O/F+1} \tag{3.5}$$

3.2.3. Drag

The main focus of this work is on exo-atmoshperic flight, that is the portion of the flight in which aerodynamic forces are negligible. However the closed-loop guidance starts at the ignition of the second stage, which happens at an altitude of around 75 km for the current baseline mission profile. While the aerodynamic forces at this altitude are already very small, they can still have a measurable effect in the first seconds of the second stage flight. This is also caused by the relatively large angle of attack that results from the closed-loop guidance attitude commands. Even if this won't be included in the state dynamics, as will be shown in Section 5.3 some parts of the algorithm need to take the drag effect into account in order to function properly.

The drag force is usually given by Equation (3.6), in which C_D is the drag coefficient, S_{ref} the associated reference area, ρ the local atmospheric density, and V_A the airspeed, which if wind is neglected is equal to the velocity with respect to the ECEF frame [28]. C_D depends of the total angle of attack and the Mach number. The Mach number dependence is not of major relevance given that this portion of the flight occurs well into the hyper-sonic regime. If the temperature gradients are not too large, atmospheric air-density ρ can be approximated using a simple exponential model such as Equation (3.7), where ρ_0 is the air density at a reference altitude h_0 and H_s is a constant scale height [29].

$$D = C_D \frac{1}{2} \rho V_A^2 S_{ref} \tag{3.6}$$

$$\rho = \rho_0 e^{\frac{h-h_0}{H_s}} \tag{3.7}$$

3.3. Equations of Motion

The equations of motion presented in this section will only focus on what is essential to obtain a sufficiently accurate trajectory propagation inside the guidance for exo-atmospheric flight. For the purpose of simulating the real flight conditions taking into account all sorts of disturbances, RFA has already developed a high fidelity 6-DOF dynamic model. Since the model is already much more accurate than what is required to test the guidance algorithms, its further development is considered to be outside of the scope of this thesis.

For the purposes of an exo-atmospheric closed loop guidance algorithm it is sufficient to use a simple 3-DOF model. None of the typical constraints in exo-atmospheric flight are a function of the rotational state and given that rotational dynamics are so much faster than translational dynamics it can be assumed that the rocket instantly points in the direction that is commanded by the guidance. The equations of motion in 3-DOF with a thrust and gravity force in an inertial reference frame are then given by Equation (3.8) to Equation (3.10). In these equations the unit thrust direction $\hat{\mathbf{P}}$ is typically a control variable for the guidance. The thrust magnitude T is sometimes also a control or follows a predetermined function, in which case the mass m becomes a simple function of time. The gravity vector $\mathbf{g}(\mathbf{r})$ in the equations below represents some generalized gravity function which can be based on any of the models described in the previous section.

$$\dot{\mathbf{v}} = \frac{T}{m}\hat{\mathbf{P}} + \mathbf{g}(\mathbf{r}) \tag{3.8}$$

$$\dot{\mathbf{r}} = \mathbf{v} \tag{3.9}$$

$$\dot{m} = \frac{T}{I_{sp}g_0} \tag{3.10}$$

3.4. Orbital elements

As can be seen in the previous section the PEG requires the target constraints to be expressed in such a way that the desired position and velocity vectors can be constructed in the corrector. In the standard ascent mode this means that the velocity and position magnitudes as well as the desired flight path angle and unit vector normal to the target orbit need to be provided to the PEG [2]. However the target parameters provided to the guidance will generally be expressed in terms of orbital elements such as defined in Table 3.1 and shown in Figure 3.1 and Figure 3.2 [28]. For standard ascent this would require providing constraints on the semi-major axis, eccentricity, true anomaly, inclination, and longitude of the ascending node [27]. To convert between the two sets of constraints, there exist a number of relationships depending on the exact types of constraints. For example the radius and velocity are related to semi-major axis using Equation (3.11) and for insertion at periapsis or apoapsis the radius can be obtained using Equation (3.12) and Equation (3.13) respectively. The orbit normal vector points in the direction opposite to the angular momentum as defined in Equation (3.14) which in turn relates to the inclination using Equation (3.15) If some of the angles need to be solved for, such as finding the orbital plane that passes over a certain position, spherical trigonometry can be useful [31].

$$-\frac{\mu}{2a} = \frac{v^2}{2} - \frac{\mu}{r} \tag{3.11}$$

$$r_p = a(1-e) \tag{3.12}$$

$$r_a = a(1+e) \tag{3.13}$$

$$\mathbf{h} = \mathbf{r} \times \mathbf{v} \tag{3.14}$$

$$i = \frac{\hat{\mathbf{i}}_z\mathbf{h}}{||\mathbf{h}||} \tag{3.15}$$

a	Semi-major axis ($a > R_e$)
e	Eccentricity ($0 \le e < 1$)
i	Inclination ($0° \le i \le 180°$)
ω	Argument of pericentre ($0° \le \omega \le 360°$)
Ω	Longitude of the ascending node (LAN) ($0° \le \Omega \le 360°$)
θ	True anomaly ($0° \le \theta \le 360°$)

Table 3.1: Orbital elements definitions elliptical orbits [28]

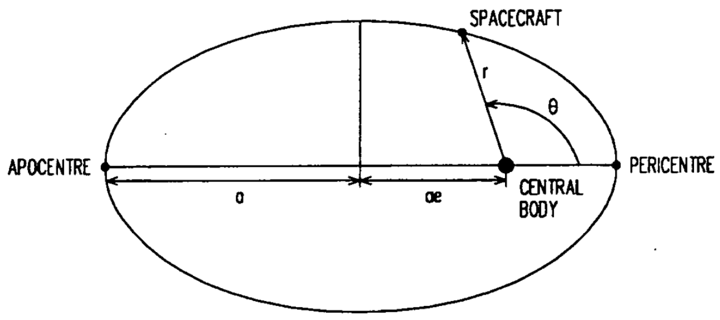


Figure 3.1: Definition of the semi-major axis, eccentricity, and true anomaly [28]

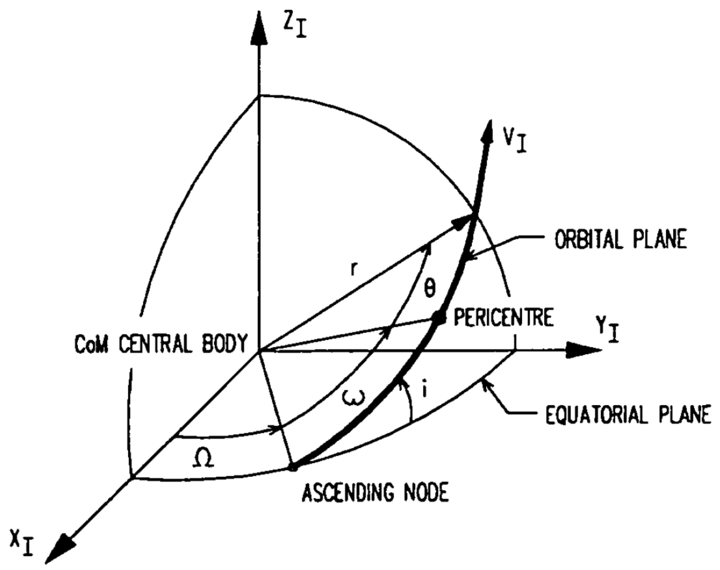


Figure 3.2: Definition the longitutude of the ascending node, argument of pericentre, and inclination [28]

4 Theoretical background

This chapter will present the final part of the theoretical background on which the algorithms in this thesis are built. First a brief overview of the guidance systems in non-deterministic environments is given. This is followed by the theory used for optimal control of dynamic system. Finally different methods for the estimation of parameters in dynamic systems in the presence of uncertainty are also outlined.

4.1. Stochastic guidance

The over-arching goal of the closed-loop guidance that will be developed is to maximize the probability for the rocket to reach orbit. Given that there are uncertainties on the various states and parameters of the rocket, the control problem is inherently stochastic. Contrary to deterministic problems, in order to solve stochastic control problems the cost function needs to be stated as the expected value of some function. Finding the optimal solution to a stochastic problem while taking into account all uncertainties can become quite complicated, given that the estimation and control processes become more tightly coupled. Fortunately under some conditions the optimal solution to stochastic problem will be given by connecting an optimal estimator with an optimal deterministic controller. This is called the certainty equivalence principle which is valid when the following conditions are satisfied [32, 33]:

1. The cost function has quadratic criteria.
2. The state dynamics are linear.
3. The measurements are linear in the state.
4. The estimator is a full-state estimator.
5. The state is propagated using a mathematical model and not with model replacement.
6. The noise signals corrupting the dynamics, measurement, and control are additive Gaussian white noise processes

While not all of these conditions are entirely full-filled for the ascent guidance case, as will be shown in Chapter 5 these assumptions are acceptable. Indeed all of the ascent guidance algorithms that can be found in literature are fully deterministic algorithms that implicitly use this principle. For the development of the guidance algorithm in this work it was therefore decided to approach the stochastic control problem as two separate problems: Creating a purely deterministic optimal guidance algorithm on one side that uses the optimal estimations from the navigation and vehicle parameter estimator on the other side.

4.2. Optimal control

The exo-atmospheric guidance problem involves determining a continuous set of controls $\mathbf{u}(t)$ which optimize a dynamic system, which is part of the *calculus of variations*. This can be formulated as minimizing a cost function, such as given in Equation (4.1), subject to dynamics constraints on the state $\mathbf{x}(t)$, given in Equation (4.2), with the initial state $\mathbf{x}(t_0)$ given and a set of end-point constraints given by Equation (4.3) [33]. This cost function consists of two parts: the first 'discrete' term is evaluated at the final time and the second 'continuous' term is evaluated over the the entire interval. The first constraint describes the dynamics of the system and the second constraint can be used to specify target conditions. While more generalized problem formulations exist, which include inequality constraints or path constraints, the formulation below is sufficient to cover the class of problems treated within this thesis.

$$J = \phi(\mathbf{x}(t_f), t_f) + \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t), t)dt \tag{4.1}$$

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) \tag{4.2}$$

$$\boldsymbol{\psi}(\mathbf{x}(t_f), t_f) = \mathbf{0} \tag{4.3}$$

The solution for this problem can be found with a method analogous to finite dimensional case for constrained optimization by adjoining the constraints to the cost function using a two set of Lagrange multipliers($\boldsymbol{\nu}$ and $\boldsymbol{\lambda}(t)$), which results in the augmented cost function Equation (4.4) [33, 34]. The continuous Lagrange multipliers are often called the co-state, given that, as will be shown later, they are also governed by a system of differential equations.

$$\bar{J} = \phi(\mathbf{x}(t_f), t_f) + \boldsymbol{\nu}^T\boldsymbol{\psi}(\mathbf{x}(t_f), t_f) + \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t), t) + \boldsymbol{\lambda}^T(t)(\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) - \dot{\mathbf{x}})dt \tag{4.4}$$

In order to find the set of necessary conditions for the optimum of the augmented cost function the first variation ($\delta\bar{J}$) is set to zero. For this it is useful to first define the *Hamiltonian*, Equation (4.5), for the continuous part and an auxiliary function Equation (4.6) for the discrete part [33].

$$H(\mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\lambda}(t), t) = L(\mathbf{x}(t), \mathbf{u}(t), t) + \boldsymbol{\lambda}^T(t)\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) \tag{4.5}$$

$$\boldsymbol{\Phi} = \phi + \boldsymbol{\nu}^T\boldsymbol{\psi} \tag{4.6}$$

The necessary conditions, often called the *Euler-Lagrange equations*, are then given by Equation (4.7) to Equation (4.9) in addition to Equation (4.2) and Equation (4.3). Equation (4.7) is a set of algebraic equations through which the control vector is obtained. The differential equations in Equation (4.8) define the dynamics of the co-state and the *tansversality conditions*, Equation (4.9), are a set of constraints on its final value.

$$\left(\frac{H}{\partial \mathbf{u}}\right)^T = \left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right)^T \boldsymbol{\lambda} + \left(\frac{\partial L}{\partial \mathbf{u}}\right)^T = \mathbf{0} \tag{4.7}$$

$$\dot{\boldsymbol{\lambda}} = -\left(\frac{H}{\partial \mathbf{x}}\right)^T = -\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)^T \boldsymbol{\lambda} - \left(\frac{\partial L}{\partial \mathbf{x}}\right)^T \tag{4.8}$$

$$\boldsymbol{\lambda}^T(t_f) = \left[\left(\frac{\boldsymbol{\Phi}}{\partial \mathbf{x}}\right)^T\right]_{t=t_f} = \left[\frac{\partial \phi}{\partial \mathbf{x}} + \boldsymbol{\nu}^T \frac{\partial \boldsymbol{\psi}}{\partial \mathbf{x}}\right]_{t=t_f} \tag{4.9}$$

Given that the transversality conditions above require solving for a set of unknown multipliers, $\boldsymbol{\nu}$, it can be easier to first reformulate them [26]. First a set of linearly independent vectors $\mathbf{y}_i$ is formed such that each of them satisfy Equation (4.10). For a problem with n states and m target constraints this will yield $n - m$ vectors. It can then be shown that Equation (4.11) is equivalent to Equation (4.9). Another way to view this is that the vector $\boldsymbol{\lambda} + \frac{\partial \phi}{\partial \mathbf{x}}$ must be perpendicular to the tangent plane of $\boldsymbol{\psi}(\mathbf{x_f}) = 0$. While the vectors $\mathbf{y}_i$ could be found by solving Equation (4.10), this last insight allows us to also find them by inspection.

$$\left[\frac{\partial \boldsymbol{\psi}}{\partial \mathbf{x}}\right]_{t=t_f} \mathbf{y} = \mathbf{0} \tag{4.10}$$

$$\mathbf{y}_i^T(\mathbf{x}(\mathbf{t_f}))\left[\boldsymbol{\lambda} + \frac{\partial \phi}{\partial \mathbf{x}}\right]_{t=t_f} = 0 \tag{4.11}$$

In case the final time isn't fixed there is an additional necessary condition given by Equation (4.12) to account for the additional unknown [34].

$$\left[\frac{\partial \phi}{\partial t} + H\right]_{t=t_f} = 0 \tag{4.12}$$

There exist different methods for solving the equations above. In case the state derivative function is sufficiently simple, it is possible to derive analytical expressions for the controls. The control therefore becomes a function of the initial state and co-state and this is called the indirect method as the Euler-Lagrange equations are satisfied indirectly. The optimal control problem therefore becomes a two-point boundary value problem which requires solving a system of non-linear equations. This can for example be solved using shooting methods with either one, single shooting, or more arcs, multiple shooting. For single shooting the number of unknowns and constraints is limited, but convergence can be difficult due to hyper-sensitivity of the final state to the initial conditions. This can be alleviated by splitting up the arc at the cost of additional unknowns at the intermediate nodes [35].

Obtaining the gradients of the state derivative function can sometimes be difficult or even impossible. In such a case the direct methods should be used instead where either the state, the control, or both are approximated and the parameters of the approximating function are solved for using non-linear programming [35]. These methods can be further divided into explicit methods and implicit methods depending on the nature of the relationship defining the state or control [34].

4.3. Optimal estimation

In-flight estimation of the parameters of the vehicle that are relevant to the guidance, such as thrust or mass, is one of the primary ways of improving the robustness of the closed-loop guidance system. These parameters can deviate from their nominal values for various reasons, such as a large initial uncertainty or in-flight anomalies. It was indeed already noted by Stein that parameter estimation was one of the main points of future improvements for the guidance of RFA ONE [4]. Some level of parameter estimation is being used for the SLS guidance [3] as was discussed in Section 2.1.2 and the estimation of certain critical rocket engine parameters for fault detection has also been proposed [36, 37].

4.3.1. Kalman Filter

In the case where the estimated state changes over time through some linear, stochastic, multistage process, then the sequential use of weighted least squares becomes the so-called Kalman filter [38, 33]. In this case the measurement equation is described by Equation (4.13) and the discrete transitions of the state are described by Equation (4.14), where $E(\mathbf{x}_0) = \overline{\mathbf{x}}_0$ and $E(\mathbf{w}_i) = \overline{\mathbf{w}}_i$.

$$\mathbf{z}_i = \mathbf{H}\mathbf{x}_i + \mathbf{v} \tag{4.13}$$

$$\mathbf{x}_{i+1} = \mathbf{\Phi}_i\mathbf{x}_i + \mathbf{\Gamma}_i\mathbf{w}_i \tag{4.14}$$

Furthermore the uncertainty on the initial state and process noise are given by Equation (4.15) and Equation (4.16) respectively.

$$E\left((\mathbf{x}_0 - \overline{\mathbf{x}}_0)(\mathbf{x}_0 - \overline{\mathbf{x}}_0)^T\right) = M_0 \tag{4.15}$$

$$E\left((\mathbf{w}_i - \overline{\mathbf{w}}_i)(\mathbf{w}_j - \overline{\mathbf{w}}_j)^T\right) = Q_i\delta_{ij} \tag{4.16}$$

The Kalman Filter consists of two main parts: propagation and update. The propagation of the state estimate and its uncertainty is performed at every time-step and is described by Equation (4.17) and Equation (4.18). Then for every step where measurements are available, the state and its covariance matrix are updated to take into account this new information using Equation (4.19) and Equation (4.20), where K_i, computed with Equation (4.21), is the Kalman gain matrix which represents the ratio between the uncertainty on state, given by $\mathbf{P}_i$, and the uncertainty in the measurements, given by $\mathbf{R}_i$. $\mathbf{H}_i$ simply serves as the transformation matrix between state and measurement. Furthermore it can be seen that the correction term in the update step is proportional to the difference between the estimated and actual measurements.

$$\overline{\mathbf{x}}_{i+1} = \mathbf{\Phi}_i\hat{\mathbf{x}}_i + \mathbf{\Gamma}_i\overline{\mathbf{w}}_i \tag{4.17}$$

$$\mathbf{M}_{i+1} = \mathbf{\Phi}_i\mathbf{P}_i\mathbf{\Phi}_i^T + \mathbf{\Gamma}_i\mathbf{Q}_i\mathbf{\Gamma}_i^T \tag{4.18}$$

$$\hat{\mathbf{x}}_i = \overline{\mathbf{x}}_i + \mathbf{K}_i(\mathbf{z}_i - \mathbf{H}_i\overline{\mathbf{x}}_i) \tag{4.19}$$

$$\mathbf{P}_i = \mathbf{M}_i - \mathbf{M}_i\mathbf{H}_i^T(\mathbf{H}_i\mathbf{M}_i\mathbf{H}_i^T + \mathbf{R}_i)^{-1}\mathbf{H}_i\mathbf{M}_i \tag{4.20}$$

$$\mathbf{K}_i = \mathbf{P}_i\mathbf{H}_i^T\mathbf{R}_i^{-1} \tag{4.21}$$

An overview of the whole process for a single step is shown in Figure 4.1. As stated previously the update step can only happen if measurements are available. If this is not the case then the state can still be predicted for new time-step by only applying the propagation step, with $\hat{\mathbf{x}}_i = \hat{\mathbf{x}}_i$ and $\mathbf{P}_i = \mathbf{M}_i$. Furthermore the act of measuring can only decrease the uncertainty on the state. Conversely propagation can only increase this uncertainty, as well as redistributing it between states if they are linked through the state transition matrix $\mathbf{\Phi}_i$ [33].

Most real-world processes are not discrete but continuous, possibly including continuous measurements. The noises in that case are white-noise processes. While there does exist a continuous version of the Kalman Filter [39], it can be approximated by the discrete multistage filter described above in order to implement it on a digital computer [33]. However when

$$
\begin{array}{lccccc}
 & & \overline{\mathbf{w}}_i & & \mathbf{z}_{i+1} & \\
 & & \downarrow & & \downarrow & \\
\text{mean:} & \hat{\mathbf{x}}_i & \longrightarrow & \overline{\mathbf{x}}_{i+1} & \longrightarrow & \hat{\mathbf{x}}_{i+1} \\
 & & \mathbf{Q}_i & & \mathbf{R}_{i+1} & \\
 & & \downarrow & & \downarrow & \\
\text{covariance:} & \mathbf{P}_i & \longrightarrow & \mathbf{M}_{i+1} & \longrightarrow & \mathbf{P}_{i+1}
\end{array}
$$

Figure 4.1: Diagram for a single step of the Kalman filter

doing this, the continuous noise covariance matrices $\mathbf{Q}$ and $\mathbf{R}$ need to be scaled as their discrete counterparts $\mathbf{Q}_i$ and $\mathbf{R}_i$ depend on Δt. More specifically the discrete noise matrices are inversely proportional to Δt [33].

4.3.2. Extended Kalman Filter

The Kalman filter described in the previous section is only directly applicable to processes with linear dynamics and measurement systems. However similarly to the static case, linearization around the nominal path or continuously linearizing around the new estimates can allow application of this linear filter to non-linear systems. However it should be noted that while for linear systems the Kalman filter gives the optimal estimates on the condition that the various noise matrices are accurate, this is not necessarily the case for non-linear systems. In fact it might even fail to converge entirely if the initial estimate is poor or disturbances so large that the linearization becomes invalid.

Suppose we have a multi-stage process with non-linear dynamics and measurements described in the general sense by Equation (4.22) and Equation (4.23).

$$\mathbf{x}_{i+1} = f_i(\mathbf{x}_i) + \mathbf{\Gamma}_i \mathbf{w}_i \tag{4.22}$$

$$\mathbf{z}_i = h(\mathbf{x}_i) + \mathbf{v}_i \tag{4.23}$$

It would then be reasonable to linearize these two equations using Equation (4.24) and Equation (4.25) [33].

$$\mathbf{H}_i \equiv \left.\frac{\partial h_i}{\partial \mathbf{x}_i}\right|_{\mathbf{x}=\mathbf{x}_i} \tag{4.24}$$

$$\mathbf{\Phi}_i \equiv \left.\frac{\partial f_i}{\partial \mathbf{x}_i}\right|_{\mathbf{x}=\mathbf{x}_i} \tag{4.25}$$

These two linearized matrices can then be used in Equation (4.18), Equation (4.20), and Equation (4.21) to propagate and update the uncertainty in the state. For the propagation and update of the state itself the non-linear dynamics measurement function can be directly used, see Equation (4.26) and Equation (4.27).

$$\overline{\mathbf{x}}_{i+1} = f_i(\hat{\mathbf{x}}_i) + \mathbf{\Gamma}_i \overline{w}_i \tag{4.26}$$

$$\hat{\mathbf{x}}_i = \overline{\mathbf{x}}_i + \mathbf{K}_i(\mathbf{z}_i - h(\mathbf{x}_i)) \tag{4.27}$$

4.3.3. Smoothing

The techniques discussed in previous sections have been mainly concerned with filtering and prediction, i.e. predicting the current state or future states given current measurements. However the resulting state estimate can sometimes still be somewhat noisy and oscillatory in nature. If the estimated parameters are then used to solve for the guidance solution this noise can propagate into the commanded attitude, especially if the guidance solution is sensitive to the estimated parameters. The closed-loop guidance algorithm itself can generally deal with such noise without any issue, given that it can generally converge even with a very poor initial guess. However problems might arise downstream of the guidance, such as the controller where the step-wise nature of the steering parameter update could cause undesirable effects. It might therefore be preferable to forego using more accurate estimates in favor of less noisy ones. This is where the third branch of optimal estimation comes into play, where current measurements are used to estimate a state in the past in a process called smoothing [33]. Such techniques have been proposed and tested in the past for the use of real-time rocket engine health monitoring [37, 36].

For the single-stage case the smoothing problem consist of estimating $\mathbf{x}_0$ and $\mathbf{w}_0$ given some measurement $\mathbf{z}_1$, that is to use information from time-step 1 to improve our estimates of the state at time-step 0 and its transition to the state at time-step 1. The so-called *smoothing* estimates that we obtain in this way are $\hat{\mathbf{x}}_{0/1}$ and $\hat{\mathbf{w}}_{0/1}$ to differentiate them from the *filtering* estimates $\hat{\mathbf{x}}_0$ and $\overline{\mathbf{w}}_0$. In order for $\hat{\mathbf{x}}_{0/1}$ and $\hat{\mathbf{w}}_{0/1}$ to be optimal estimates of $\mathbf{x}_0$ and $\mathbf{w}_0$ in the least-squares sense they must minimize the quadratic in Equation (4.28) while satisfying the state transition constraint given by Equation (4.29).

$$J = \frac{1}{2}(\mathbf{x}_0 - \hat{\mathbf{x}}_0)^T\mathbf{P}_0^{-1}(\mathbf{x}_0 - \hat{\mathbf{x}}_0) + \frac{1}{2}(\mathbf{w}_0 - \overline{\mathbf{w}}_0)^T\mathbf{Q}_0^{-1}(\mathbf{w}_0 - \overline{\mathbf{w}}_0) + \frac{1}{2}(\mathbf{z}_1 - \mathbf{H}_1\mathbf{x}_1)^T\mathbf{R}_1^{-1}(\mathbf{z}_1 - \mathbf{H}_1\mathbf{x}_1) \tag{4.28}$$

$$\mathbf{x}_1 = \mathbf{\Phi}_0\mathbf{x}_0 + \mathbf{\Gamma}_0\mathbf{w}_0 \tag{4.29}$$

It can be shown that the optimal estimates that minimize this quadratic are given by Equation (4.30) and Equation (4.31) [33], where $\hat{\mathbf{x}}_1 \equiv \hat{\mathbf{x}}_{1/1}$ is the filtered estimate discussed in previous sections obtained using Equation (4.19) and the proportionality matrices $\mathbf{C}_0$ and $\mathbf{B}_0$ are obtained using Equation (4.32) and Equation (4.33) respectively.

$$\hat{\mathbf{x}}_{0/1} = \hat{\mathbf{x}}_0 - \mathbf{C}_0(\overline{\mathbf{x}}_1 - \hat{\mathbf{x}}_1) \tag{4.30}$$

$$\hat{\mathbf{w}}_{0/1} = \overline{\mathbf{w}}_0 - \mathbf{B}_0(\overline{\mathbf{x}}_1 - \hat{\mathbf{x}}_1) \tag{4.31}$$

$$\mathbf{C}_0 = \mathbf{P}_0\mathbf{\Phi}_0^T\mathbf{M}_1^{-1} \tag{4.32}$$

$$\mathbf{B}_0 = \mathbf{Q}_0\mathbf{\Gamma}_0^T\mathbf{M}_1^{-1} \tag{4.33}$$

Furthermore it can be shown that the associated covariance matrices are Equation (4.34) and Equation (4.35).

$$\mathbf{P}_{0/1} = \mathbf{P}_0 - \mathbf{C}_0(\mathbf{M}_1 - \mathbf{P}_1)\mathbf{C}_0^T \tag{4.34}$$

$$\mathbf{Q}_{0/1} = \mathbf{Q}_0 - \mathbf{B}_0(\mathbf{M}_1 - \mathbf{P}_1)\mathbf{B}_0^T \tag{4.35}$$

While the relations above could be used directly, there are a number of potential numerical difficulties. The first one is the term $\mathbf{M}_1^{-1}(\bar{\mathbf{x}}_1 - \hat{\mathbf{x}}_1)$, as the elements of $\mathbf{M}_1^{-1}$ can be quite large while the elements of $\bar{\mathbf{x}}_1 - \hat{\mathbf{x}}_1$ can be quite small. The second one is the $\mathbf{M}_1^{-1}(\mathbf{M}_1 - \mathbf{P}_1)\mathbf{M}_1^{-1}$ where a similar issue might arise. Furthermore it is inconvenient to compute the inverse of $\mathbf{M}_1$. It is therefore preferable to use the identities in Equation (4.36) and Equation (4.37) [33].

$$\mathbf{M}_1^{-1}(\bar{\mathbf{x}}_1 - \hat{\mathbf{x}}_1) = -\mathbf{H}_1^T(\mathbf{H}_1\mathbf{M}_1\mathbf{H}_1^T + \mathbf{R}_1)^{-1}(\boldsymbol{z}_1 - \mathbf{H}_1\bar{\mathbf{x}}_1) \equiv \boldsymbol{\lambda}_0 \tag{4.36}$$

$$\mathbf{M}_1^{-1}(\mathbf{M}_1 - \mathbf{P}_1)\mathbf{M}_1^{-1} = \mathbf{H}_1^T(\mathbf{H}_1\mathbf{M}_1\mathbf{H}_1^T + \mathbf{R}_1)^{-1}\mathbf{H}_1 \equiv \boldsymbol{\Lambda}_0 \tag{4.37}$$

Using these identities the previous relationships for the estimates and covariance can be rewritten as shown in Equation (4.38) to Equation (4.41).

$$\hat{\mathbf{x}}_{0/1} = \hat{\mathbf{x}}_0 - \mathbf{P}_0\boldsymbol{\Phi}_0^T\boldsymbol{\lambda}_0 \tag{4.38}$$

$$\hat{\mathbf{w}}_{0/1} = \bar{\mathbf{w}}_0 - \mathbf{Q}_0\boldsymbol{\Gamma}_0^T\boldsymbol{\lambda}_0 \tag{4.39}$$

$$\mathbf{P}_{0/1} = \mathbf{P}_0 - \mathbf{P}_0\boldsymbol{\Phi}_0^T\boldsymbol{\Lambda}_0\boldsymbol{\Phi}_0\mathbf{P}_0 \tag{4.40}$$

$$\mathbf{Q}_{0/1} = \mathbf{Q}_0 - \mathbf{Q}_0\boldsymbol{\Gamma}_0^T\boldsymbol{\Lambda}_0\boldsymbol{\Gamma}_0\mathbf{Q}_0 \tag{4.41}$$

For the above single-stage smoothing algorithm to be most useful it needs to be extended to a multi-stage process. In this case the goal is to obtain the optimal estimate of $\mathbf{x}_i$ using all measurements $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_N$ for $i < N$. However compared to multi-stage optimal filtering there is more variety in how this is implemented. Simply extending the single-stage smoother to multiple time-steps would yield a smoother where starting from a fixed point in time previous states would be recursively estimated in a backwards sweep, obtaining progressively smoother results as you go further back in time. This requires computing the entire series of filtered states beforehand in a forward sweep [33]. An overview of this process is shown in Figure 4.2.

$$\begin{tikzcd}
\text{Forward sweep} & \hat{\mathbf{x}}_0 \arrow[r, "\downarrow\bar{\mathbf{w}}_0"] & \bar{\mathbf{x}}_1 \arrow[r, "\downarrow\boldsymbol{z}_1"] & \hat{\mathbf{x}}_1 \arrow[r] & \cdots & \bar{\mathbf{x}}_N \arrow[r, "\downarrow\boldsymbol{z}_N"] & \hat{\mathbf{x}}_N \\
 & \mathbf{P}_0 \arrow[r, "\downarrow\mathbf{Q}_0"] & \mathbf{M}_1 \arrow[r, "\downarrow\mathbf{R}_1"] & \mathbf{P}_1 \arrow[r] & \cdots & \mathbf{M}_N \arrow[r, "\downarrow\mathbf{R}_N"] & \mathbf{P}_N \\
\text{Backward sweep} & \hat{\mathbf{x}}_{0/N} & \hat{\mathbf{x}}_{1/N} \arrow[l, "\bar{\mathbf{x}}_1\downarrow"'] & \cdots & \hat{\mathbf{x}}_{N-1/N} \arrow[l] & \hat{\mathbf{x}}_{N/N} \arrow[l, "\bar{\mathbf{x}}_N\downarrow"'] & \\
 & \mathbf{P}_{0/N} & \mathbf{P}_{1/N} \arrow[l, "\mathbf{M}_1\downarrow"'] & \cdots & \mathbf{P}_{N-1/N} \arrow[l] & \mathbf{P}_{N/N} \arrow[l, "\mathbf{M}_N\downarrow"'] &
\end{tikzcd}$$

Figure 4.2: Diagram of multi-stage smoother

While this could be useful for the case where all measurements are processed at once, i.e. N is fixed, using this smoother in real-time with increasing N would require recomputing the smoothed estimate sequence for every time-step as well as storing the entire sequence of filter estimates. Other variants are therefore more useful such as a fixed-point smoother [33, 40], where i is fixed and N is increasing, or a fixed lag smoother [41], where N is increasing and $i = N - j$ for a constant j.

The fixed point smoother is relatively easy to implement using a simple recursion formula [33], as shown in Equation (4.42) and Equation (4.43). Of course this still requires the computation of $\hat{\mathbf{x}}_i$ and $\mathbf{P}_i$ which can be obtained by running a Kalman filter in parallel. This can be especially useful if the estimated states of interest are constant. It is also possible to implement a fixed-point fixed-lag smoother [36], were j overlapping fixed point smoothers are run in parallel each being initialized at a new fixed point. Then once a smoother has been updated j times it is terminated and a new one is initialized at the current time-step. This way every time-step there is a new smoothed estimate available with a fixed time delay.

$$\hat{\mathbf{x}}_{i/N} = \hat{\mathbf{x}}_{i/N-1} + \mathbf{P}_{i/N}\mathbf{H}^T\mathbf{R}_N^{-1}(\mathbf{z}_N - \mathbf{H}_N\hat{\mathbf{x}}_N), \qquad \hat{\mathbf{x}}_{i/i} = \hat{\mathbf{x}}_i \tag{4.42}$$

$$\mathbf{P}_{i/N} = \mathbf{P}_{i/N-1}(\mathbf{I} - \mathbf{P}_N\mathbf{H}_N^T\mathbf{R}_N^{-1}\mathbf{H}_N)^T, \qquad \mathbf{P}_{i/i} = \mathbf{P}_i \tag{4.43}$$

5 Guidance algorithm development

The research questions formulated at the end of Chapter 2 require the development of new or improved components for the guidance system. This chapter provides the rationales and derivations that were made throughout this development. First a high-level overview of the guidance is provided in Section 5.1, which introduces some key concepts and more clearly defines the scope of the development for this thesis. Section 5.2 provides derivations for both the old close-loop guidance system and the one that will replace it. This is followed by a discussion of the methods used for the vehicle parameter estimation in Section 5.3. Finally a simple algorithm for the in-flight selection of the optimal orbit target is presented in Section 5.4.

5.1. High-level overview

While this thesis only focuses on three components of guidance, they still have to operate within a larger system. An overview of the entire GNC system was already provided by Figure 1.1. A more detailed overview of just the guidance system is given in Figure 5.1. As can be seen, only three component fall within the scope of this thesis. The open-loop guidance, which is used during the endo-atmospheric part of the flight, is left untouched, as are the roll program and steering filter that transform the thrust direction commands into smooth attitude commands for the controller. The part this thesis focuses on can be seen as the core of the guidance and it is also capable of running standalone in a 3-DOF simulator. This is very useful for rapid testing of these components during development and the verification of the algorithms.

In order to better understand the derivations in the following sections it is important to introduce some key concepts. The first is that the flight of the rocket is divided into stages and phases. While the flight of a multi-stage rocket is often divided into the flight of the physical stages, this is not sufficient for the closed-loop guidance. The reasons are two-fold. First there might be some mass discontinuities throughout the flight of a stage, such as fairing release, and second the type of acceleration profile might differ throughout the stage, such as constant thrust in the beginning and constant acceleration at the end to limit g-loads. It is therefore necessary to further divide the stages into phases during which the mass is continuous and with each phase having a single smooth acceleration profile. For the maiden flight of RFA ONE there will be a total of four phases, with the second stage being divided in two by the fairing release. The goal of the vehicle parameter estimator is therefore to provide estimates of the relevant performance parameters throughout the exo-atmospheric flight for each of the remaining phases.

As was already discussed in Section 1.3, the current baseline for the maiden flight of RFA ONE is to perform a direct insertion into a circular Sun-synchronous orbit at an altitude of 550

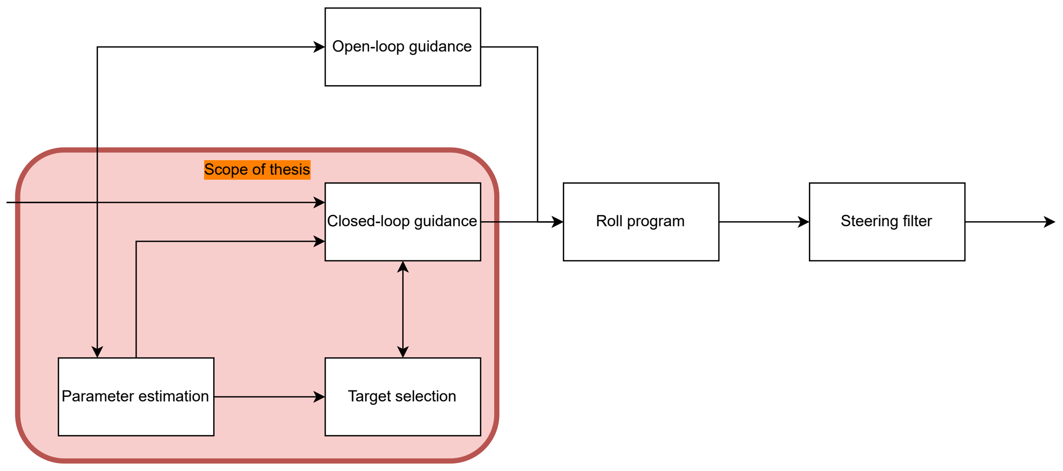


Figure 5.1: High-level overview of guidance system

km. This translates into a set of four constraints on the state at orbit insertion that need to be implemented in the algorithms. More specifically the in-plane geometry of the orbit is fully constrained as the velocity, altitude, and flight path angle are all fixed. In terms of orientation of the orbital plane, only the inclination is constrained. Crucially this means that the Longitude of the Ascending Node (LAN) and the argument of periapsis, equivalent to the down-range position, are not constrained.

5.2. Closed-loop guidance algorithm

This section will provide an in-depth derivation for two different closed-loop guidance algorithms. The closed-loop guidance is by far the most complex part of the guidance system. Its goals is to find the optimal thrust direction and associated angular rate that will bring the rocket to the desired target orbit. The closed-loop aspect comes from the fact that it periodically updates its internal steering parameters using the current position and velocity estimates from the navigation as well as new estimates of the vehicle parameters. In this way any estimation or modelling error that would lead to a deviation from the predicted trajectory can be corrected. First the PEG, which was already introduced in Chapter 2, will be presented in Section 5.2.1. This is followed by the new algorithm developed for this thesis in Section 5.2.2.

5.2.1. Powered Explicit Guidance

This section will give a more thorough derivation and description of the PEG as it was implemented at RFA. For an introduction to this classical guidance algorithm, the reader is referred to Chapter 2. As with most exo-atmospheric guidance system it aims to solve the problem as defined in Section 2.1 using optimal control theory, which is discussed in Section 4.2. While this algorithm was ultimately replaced by the algorithm described in Section 5.2.2, it will serve as a baseline reference for making future comparisons. It can also be useful to draw parallels between the two algorithms and the assumptions that are made.

5.2.1.1. Optimal steering law

In order to obtain the steering law, first the gravity function needs to be defined. For the purposes of the PEG the central gravity model is used, resulting in Equation (5.1) [6].

$$\mathbf{g}(\mathbf{r}) = -\omega^2 \mathbf{r}, \qquad \omega^2 = \frac{\mu}{r^3} \tag{5.1}$$

The resulting Hamiltonian then becomes Equation (5.2), where $\boldsymbol{\lambda}_r$, $\boldsymbol{\lambda}_V$, and λ_m are the

Lagrange multipliers for the position, velocity, and mass respectively.

$$H = \boldsymbol{\lambda}_r^T \mathbf{V} + \boldsymbol{\lambda}_V^T (a_T - \omega^2 \mathbf{r}) - \lambda_m (\frac{T}{v_{ex}}) - \frac{T}{v_{ex}} \tag{5.2}$$

Using Equation (4.8) it can be shown that the co-state follows the differential equations given by Equation (5.3) [6, 26], where $\mathbf{u}_r = \frac{\mathbf{r}}{r}$. This can be reformulated as Equation (5.4).

$$\begin{bmatrix} \dot{\boldsymbol{\lambda}}_r \\ \dot{\boldsymbol{\lambda}}_V \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{\lambda}}_V - 3\omega^2 (\boldsymbol{\lambda}_V \cdot \mathbf{u}_r)\mathbf{u}_r \\ -\boldsymbol{\lambda}_r \end{bmatrix} \tag{5.3}$$

$$\ddot{\boldsymbol{\lambda}}_V = -\omega^2 \boldsymbol{\lambda}_V + 3\omega^2 (\boldsymbol{\lambda}_V \cdot \mathbf{u}_r)\mathbf{u}_r \tag{5.4}$$

The second term in the equation above comes from the derivative of ω with respect to $\mathbf{r}$. If the changes in altitude throughout the flight are small compared to the overall radius, ω can be considered constant. This differential equation for the co-state then simplifies to Equation (5.5) [6]. This is equivalent to using a linear gravity model.

$$\ddot{\boldsymbol{\lambda}}_V = -\omega^2 \boldsymbol{\lambda}_V \tag{5.5}$$

The solution for the co-state is then given by Equation (5.6), where the unit-vector $\hat{\boldsymbol{\lambda}}$ and the rate vector $\dot{\boldsymbol{\lambda}}$ are the value of $\boldsymbol{\lambda}_V$ and its derivative at $t = t_\lambda$.

$$\mathbf{u} = \boldsymbol{\lambda}_V(t) = \hat{\boldsymbol{\lambda}} \cos(\omega(t - t_\lambda)) + \frac{\dot{\boldsymbol{\lambda}}}{\omega} \sin(\omega(t - t_\lambda)) \tag{5.6}$$

By using Equation (4.4) it can be shown that the unit-thrust vector is co-directional with $\mathbf{u}$ and is therefore given by Equation (5.7) [6].

$$\hat{\mathbf{P}} = \frac{\mathbf{u}}{||\mathbf{u}||} \tag{5.7}$$

In case of constraints on the final velocity and position, which is usually the case for ascent guidance, the only necessary transversality constraint is Equation (5.8). For insertion into circular or near-circular this can be approximated by $\mathbf{u} \cdot \dot{\mathbf{u}} \approx \boldsymbol{\lambda} \cdot \dot{\boldsymbol{\lambda}} = 0$ [6]. While there were some concerns that this assumption would lead to large performance losses in case of insertion into highly elliptical orbits, such as Geostationary transfer orbits (GTO), this class of orbits wasn't applicable for the Space Shuttle [42].

$$\dot{\mathbf{u}} \cdot \mathbf{N}_r - \mathbf{u} \cdot \mathbf{N}_V = 0, \quad \mathbf{N}_r = \mathbf{V} \times (\mathbf{r} \times \mathbf{V}), \quad \mathbf{N}_V = \mathbf{r} \times (\mathbf{r} \times \mathbf{V}) \tag{5.8}$$

If we assumed that $\omega = ||\dot{\boldsymbol{\lambda}}|| \equiv \dot{\lambda}$, then $\mathbf{u}$ becomes a unit-vector given by Equation (5.9) which makes the steering angle a linear function of time [6].

$$\mathbf{u} = \hat{\boldsymbol{\lambda}} \cos\left(\dot{\lambda}(t - t_\lambda)\right) + \frac{\dot{\boldsymbol{\lambda}}}{\dot{\lambda}} \sin\left(\dot{\lambda}(t - t_\lambda)\right) \tag{5.9}$$

If the thrust-arcs are short the curvature of the Earth has only a limited effect and it can therefore be assumed that ω approaches zero. This means that the simplification $\cos(\omega(t - t_\lambda)) \approx 1$ and $\sin(\omega(t - t_\lambda)) \approx \omega(t - t_\lambda)$ can be made yielding Equation (5.10). This is the linear tangent steering law that forms the basis of the PEG. The geometry of this steering law was shown in Figure 2.1 and is repeated in Figure 5.2 for convenience.

$$\mathbf{u} = \hat{\boldsymbol{\lambda}} + \dot{\boldsymbol{\lambda}}(t - t_\lambda) \tag{5.10}$$

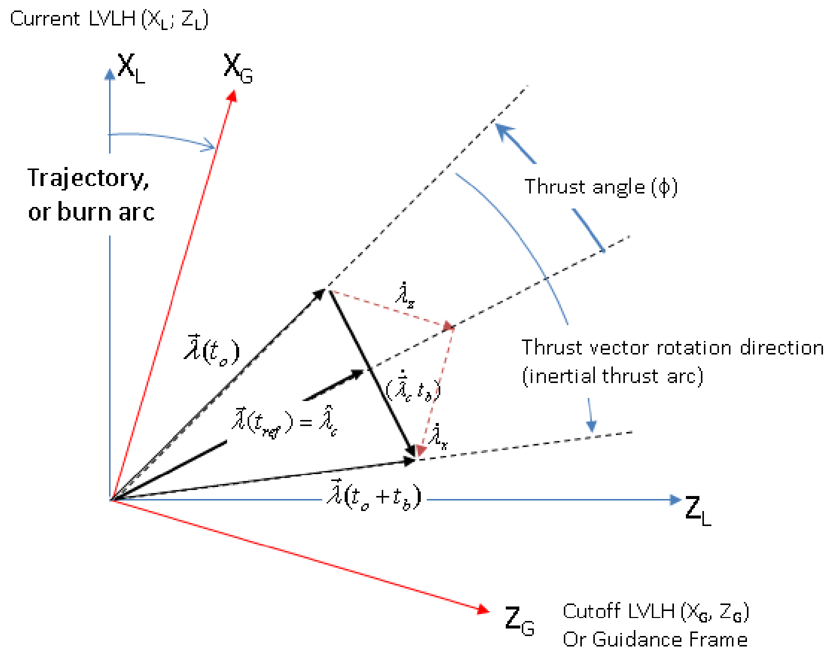

Current LVLH (X_L; Z_L)
X_L
X_G
Trajectory, or burn arc
Thrust angle (ϕ)
$\dot{\lambda}_z$
$\vec{\lambda}(t_o)$
Thrust vector rotation direction (inertial thrust arc)
$(\dot{\lambda}_c t_b)$
$\vec{\lambda}(t_{ref}) = \hat{\lambda}_c$
$\dot{\lambda}_x$
$\vec{\lambda}(t_o + t_b)$
Z_L
Z_G
Cutoff LVLH (X_G, Z_G) Or Guidance Frame

Figure 5.2: PEG LTG Ascent Geometry [3]

5.2.1.2. Velocity and position integrals

In order to satisfy the target constraints, the final desired state($\mathbf{V}_d$ and $\mathbf{R}_d$) needs to be formulated as a functions of the steering parameters $\hat{\boldsymbol{\lambda}}$ and $\dot{\boldsymbol{\lambda}}$. To start the final state is given as function of the state derivative and initial state in Equation (5.11) and Equation (5.12).

$$\mathbf{V}_d - \mathbf{V}_0 = \int_0^{t_{go}} \dot{\mathbf{V}} dt \tag{5.11}$$

$$\mathbf{R}_d - \mathbf{R}_0 = \int_0^{t_{go}} \int_0^{t} \ddot{\mathbf{R}} ds dt + \mathbf{V}_0 t_{go} \tag{5.12}$$

In order to derive analytical expression for these integrals the unit thrust direction, given by Equation (5.13), needs to be simplified. Specifically the normalization part is problematic, so a second order Taylor series expansion is used, see Equation (5.14). Inserting this back into the definition of thrust direction and dropping all terms above second order we now obtain Equation (5.15) [7].

$$\hat{\mathbf{P}} = \frac{\hat{\boldsymbol{\lambda}} + \dot{\boldsymbol{\lambda}}(t - t_\lambda)}{\sqrt{1 + \dot{\lambda}^2 (t - t_\lambda)^2}} \tag{5.13}$$

$$\frac{1}{\sqrt{1 + \dot{\lambda}^2 (t - t_\lambda)}} \approx 1 - \frac{1}{2}\dot{\lambda}^2 (t - t_\lambda)^2 \tag{5.14}$$

$$\hat{\mathbf{P}} = \hat{\boldsymbol{\lambda}} \left(1 - \frac{1}{2}\dot{\lambda}^2 (t - t_\lambda)^2\right) + \dot{\boldsymbol{\lambda}}(t - t_\lambda) \tag{5.15}$$

By combining the two integrals first with the equations of motion in Equation (2.1) and Equation (2.2) and then inserting Equation (5.15), we obtain the velocity and position integrals as a function of the steering parameters in Equation (5.16) and Equation (5.17).

$$\mathbf{V}_d - \mathbf{V}_0 = \int_0^{t_{go}} a\hat{\mathbf{P}} + \mathbf{G}dt = \int_0^{t_{go}} a_T\left(\hat{\boldsymbol{\lambda}}\left(1 - \frac{1}{2}\dot{\lambda}^2(t-t_\lambda)^2\right) + \dot{\boldsymbol{\lambda}}(t-t_\lambda)\right) + \mathbf{G}dt \tag{5.16}$$

$$\mathbf{R}_d - \mathbf{R}_0 - \mathbf{V}_0 t_{go} = \int_0^{t_{go}}\int_0^{t} a_T\left(\hat{\boldsymbol{\lambda}}\left(1 - \frac{1}{2}\dot{\lambda}^2(t-t_\lambda)^2\right) + \dot{\boldsymbol{\lambda}}(t-t_\lambda)\right) + \mathbf{G}dsdt \tag{5.17}$$

In order to solve for the steering parameters, let us first assume that we can evaluate the integrals, by assuming a_T and t_{go} are known, given in Equation (5.18).

$$\begin{aligned} L &= \int_0^{t_{go}} a_T dt & S &= \int_0^{t_{go}}\int_0^{t} a_T dsdt \\ J &= \int_0^{t_{go}} a_T t dt & Q &= \int_0^{t_{go}}\int_0^{t} a_T t dsdt \\ H &= \int_0^{t_{go}} a_T t^2 dt & P &= \int_0^{t_{go}}\int_0^{t} a_T t^2 dsdt \\ \mathbf{V}_{grav} &= \int_0^{t_{go}} \mathbf{g}dt & \mathbf{R}_{grav} &= \int_0^{t_{go}}\int_0^{t} \mathbf{g}dsdt \end{aligned} \tag{5.18}$$

By inserting the velocity integrals above into Equation (5.16) we now have the desired velocity as a function of the steering parameters, as seen in Equation (5.19) where $\mathbf{V}_{thrust}$ is given in Equation (5.20).

$$\mathbf{V}_d - \mathbf{V}_0 = \mathbf{V}_{thrust} + \mathbf{V}_{grav} \tag{5.19}$$

$$\mathbf{V}_{thrust} = \hat{\boldsymbol{\lambda}}\left(L - \frac{1}{2}\dot{\lambda}^2(H - 2t_\lambda J + t_\lambda^2 L)\right) + \dot{\boldsymbol{\lambda}}(J - Lt_\lambda) \tag{5.20}$$

The same can be done for the position integrals resulting in Equation (5.21) where $\mathbf{R}_{thrust}$ is given by Equation (5.22).

$$\mathbf{R}_d - \mathbf{R}_0 - \mathbf{V}_0 t_{go} = \mathbf{R}_{thrust} + \mathbf{R}_{grav} \tag{5.21}$$

$$\mathbf{R}_{thrust} = \hat{\boldsymbol{\lambda}}\left(S - \frac{1}{2}\dot{\lambda}^2(P - 2t_\lambda Q + t_\lambda^2 S)\right) + \dot{\boldsymbol{\lambda}}(Q - St_\lambda) \tag{5.22}$$

5.2.1.3. Steering parameters

At this stage in the derivation let us assume the final state($\mathbf{V}_d$ and $\mathbf{R}_d$) is fully defined by the target constraints. Furthermore the initial state($\mathbf{V}_0$ and $\mathbf{R}_0$) is given by navigation. The problem therefore becomes finding the values for $\hat{\boldsymbol{\lambda}}$, $\dot{\boldsymbol{\lambda}}$, and t_λ such that the Equation (5.20) and Equation (5.22) are satisfied.

In order to make these equations easier to solve, it is convenient to choose t_λ such that the term with $\dot{\boldsymbol{\lambda}}$ vanishes from the velocity integral. This implies Equation (5.23), which means t_λ is given by Equation (5.24)

$$J - Lt_\lambda = 0 \tag{5.23}$$

$$t_\lambda = \frac{J}{L} \tag{5.24}$$

Inserting Equation (5.24) into Equation (5.20) we obtain Equation (5.25). This equation however still doesn't allow us to easily solve for $\hat{\boldsymbol{\lambda}}$, and is therefore split into the linear velocity integral(Equation (5.27)) depending only on $\hat{\boldsymbol{\lambda}}$ and a bias term accounting for the steering losses.

$$\mathbf{V}_{thrust} = \hat{\boldsymbol{\lambda}}\left(L - \frac{1}{2}\dot{\lambda}^2(H - t_\lambda J)\right) \tag{5.25}$$

$$\mathbf{V}_{thrust} = \mathbf{V}_{go} - \mathbf{V}_{bias} \tag{5.26}$$

$$\mathbf{V}_{go} = \hat{\boldsymbol{\lambda}}L \tag{5.27}$$

From Equation (5.27) two important insights can be obtained. First $\hat{\boldsymbol{\lambda}}$ is co-directional with $\mathbf{V}_{go}$ and can therefore be obtained by normalizing it, see Equation (5.28). The second is that L is simply the magnitude of $\mathbf{V}_{go}$, see Equation (5.29). This will allow us to find the value for t_{go} using $\mathbf{V}_{go}$.

$$\hat{\boldsymbol{\lambda}} = \frac{\mathbf{V}_{go}}{||\mathbf{V}_{go}||} \tag{5.28}$$

$$L = ||\mathbf{V}_{go}|| \tag{5.29}$$

The position integral can be similarly split into $\mathbf{R}_{go}$, given in Equation (5.31), and a bias term.

$$\mathbf{R}_{thrust} = \mathbf{R}_{go} - \mathbf{R}_{bias} \tag{5.30}$$

$$\mathbf{R}_{go} = \hat{\boldsymbol{\lambda}}S + \dot{\boldsymbol{\lambda}}(Q - St_\lambda) \tag{5.31}$$

$\dot{\boldsymbol{\lambda}}$ can therefore be computed using Equation (5.32), where $\mathbf{R}_{go}$ is obtained using Equation (5.33).

$$\dot{\boldsymbol{\lambda}} = \frac{\mathbf{R}_{go} - \hat{\boldsymbol{\lambda}}S}{(Q - St_\lambda)} \tag{5.32}$$

$$\mathbf{R}_{go} = \mathbf{R}_d - \mathbf{R}_0 - \mathbf{V}_0 t_{go} - \mathbf{R}_{grav} + \mathbf{R}_{bias} \tag{5.33}$$

Since there is no constraint on the position in the down-range direction, the down-range component of $\mathbf{R}_{go}$ is also free. In order to enforce the orthogonality between the two steering vectors($\hat{\boldsymbol{\lambda}} \cdot \dot{\boldsymbol{\lambda}} = 0$), the scalar relationship in Equation (5.34) is used.

$$\hat{\boldsymbol{\lambda}}\mathbf{R}_{go} = S \tag{5.34}$$

The previously obtained value for $\mathbf{R}_{go}$ can therefore be decomposed into two components. The first lies within the cut-off plane which depends only on the target constraints and is given by Equation (5.36). The second essentially depends on the burn-time and is obtained using Equation (5.37).

$$\hat{\mathbf{i}}_z = \frac{\mathbf{R}_d}{||\mathbf{R}_d||} \times \hat{\mathbf{i}}_y \tag{5.35}$$

$$\mathbf{R}_{go,xy} = \mathbf{R}_{go} - (\mathbf{R}_{go} \cdot \hat{\mathbf{i}}_z)\hat{\mathbf{i}}_z \tag{5.36}$$

$$R_{go,z} = \frac{S - \hat{\boldsymbol{\lambda}}\mathbf{R}_{go,xy}}{\hat{\boldsymbol{\lambda}}\hat{\mathbf{i}}_z} \tag{5.37}$$

Putting these two components together in Equation (5.38), we obtain the corrected value for $\mathbf{R}_{go}$ which is then used in Equation (5.32). This correction will eventually trend towards zero as the algorithm converges and $\mathbf{R}_d$ is adjusted for burn-time in the corrector, see Equation (5.54).

$$\mathbf{R}_{go} = \mathbf{R}_{go,xy} + R_{go,z}\hat{\mathbf{i}}_z \tag{5.38}$$

5.2.1.4. Acceleration integrals

In order to evaluate the thrust integrals used in the previous sub-section, the thrust acceleration as a function of time is needed. It is convenient to change the parameters from T, m_0, and $\dot{m}$ to v_{ex} and τ as seen in Equation (5.39), where τ is given by Equation (5.40). It is assumed here that these two parameters are known for all stages.

$$a_T = \frac{T}{m} = \frac{v_{ex}\dot{m}}{m_0 - \dot{m}t} = \frac{v_{ex}}{\tau - t} \tag{5.39}$$

$$\tau = \frac{m_0}{\dot{m}} \tag{5.40}$$

Next the thrust integrals are evaluated for a multi-stage rocket. There exist closed-form solutions for these integrals, as derived in [6] and also shown in [2], which can be computed in a recursive manner. While the version of the PEG for the Space Shuttle had to evaluate these integrals for both constant thrust and constant acceleration phases, RFA ONE has only constant thrust phases which simplifies the equations a bit. The thrust integrals per phase are given in Equation (5.41) to Equation (5.45), which can then be summed together to obtained the thrust integrals over the whole flight. In the equations below L, J, and H are the integrals up to and including the $i-1th$ phase. H can be obtained directly using Equation (5.46).

$$L_i = v_{ex,i}\ln\left(\frac{\tau_i}{\tau_i - t_{b,i}}\right) \tag{5.41}$$

$$J_i = L_i\tau_i - v_{ex,i}t_{b,i} + L_i t_{go,i-1} \tag{5.42}$$

$$S_i = -J_i + t_{b,i}L_i + Lt_{b,i} \tag{5.43}$$

$$Q_i = S_i(\tau_i + t_{go,i-1}) - \frac{1}{2}v_{ex,i}t_{b,i}^2 + Jt_{b,i} \tag{5.44}$$

$$P_i = Q_i(\tau_i + t_{go,i-1}) - \frac{1}{2}v_{ex,i}t_{b,i}^2\left(\frac{1}{3}t_{b,i} + t_{go,i-1}\right) + Ht_{b,i} \tag{5.45}$$

$$H = t_{go}J - Q \tag{5.46}$$

5.2.1.5. Predictor

Now that the steering parameters have been obtained using the simplified velocity and position integrals, a more accurate prediction of the final state needs to be made. In the version of the PEG used in the Space Shuttle this prediction was done mostly analytically [2]. First the bias terms $\mathbf{V}_{bias}$ and $\mathbf{R}_{bias}$ are evaluated using the newly obtained $\hat{\boldsymbol{\lambda}}$ and $\dot{\boldsymbol{\lambda}}$, thereby obtaining the more accurate thrust integrals $\mathbf{V}_{thrust}$ and $\mathbf{R}_{thrust}$. Then the gravity integrals are computed using the Conic State Extrapolation routine, by constructing a special coasting trajectory around the powered trajectory such that they have similar gravity integrals.

While the method above worked quite well for the Space Shuttle, its limited accuracy especially for longer thrust arcs posed a problem for RFA ONE [4]. Fortunately the on-board computational power has vastly improved since the 1980s, which allowed the predictor step to be replaced by a simple yet more accurate numerical integration scheme. It is shown in state space format in Equations (5.47) to (5.51), where $\mathbf{f_T}$ and $\mathbf{g}$ are respectively the thrust and gravity accelerations. As an added benefit it is now also much easier to incorporate more complex force models, such as the J2 effect in the gravity force.

$$\mathbf{x} = \begin{bmatrix} \mathbf{R} & \mathbf{V} & \mathbf{R}_{thrust} & \mathbf{V}_{thrust} & \mathbf{R}_{grav} & \mathbf{V}_{grav} \end{bmatrix}^T \tag{5.47}$$

$$\mathbf{u} = \begin{bmatrix} \mathbf{f_T} & \mathbf{g} \end{bmatrix}^T \tag{5.48}$$

$$\mathbf{A} = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \end{bmatrix} \tag{5.49}$$

$$\mathbf{B} = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{I}_{3\times3} & \mathbf{I}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} \end{bmatrix} \tag{5.50}$$

$$\dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{Bu} \tag{5.51}$$

5.2.1.6. Corrector

The last step of the PEG cycle is to use the predicted final state to apply corrections to $\mathbf{V}_{go}$ and $\mathbf{R}_d$ in order to drive the predicted final state to the desired final state. The corrector outlined below is most useful for direct insertion into a near circular orbit, but there are many other ways to formulate this step depending on the applicable target constraints.

In this corrector $\mathbf{V}_{go}$ is the only persistent state which is adjusted, whereas everything else, including $\mathbf{R}_d$, is recomputed for every iteration. The first step is to project the predicted position into the desired orbital plane to satisfy the lateral position constraint, as shown in Equation (5.52).

$$\mathbf{R}'_d = \mathbf{R}_p - (\mathbf{R}_p \cdot \hat{\mathbf{i}}_y)\hat{\mathbf{i}}_y \tag{5.52}$$

Then the predicted position is re-scaled to the desired magnitude, shown in Equation (5.53) and Equation (5.54), in order to satisfy the radial position constraint. By re-computing the desired position in this fashion, the length of the burn-arc is automatically driven to correspond to the required burn-time.

$$\hat{\mathbf{i}}_x = \frac{\mathbf{R}'_d}{||\mathbf{R}'_d||} \tag{5.53}$$

$$\mathbf{R}_d = R_d \hat{\mathbf{i}}_x \tag{5.54}$$

Using the newly re-computed desired position, the cut-off frame can be constructed, thereby allowing the construction of the desired velocity vector, see Equation (5.56). This satisfies the constraints on velocity magnitude, lateral velocity, and flight-path angle (γ_d).

$$\hat{\mathbf{i}}_z = \hat{\mathbf{i}}_x \times \hat{\mathbf{i}}_y \tag{5.55}$$

$$\mathbf{V}_d = V_d \begin{bmatrix} \hat{\mathbf{i}}_x & \hat{\mathbf{i}}_y & \hat{\mathbf{i}}_z \end{bmatrix} \begin{bmatrix} \sin\gamma_d \\ 0 \\ \cos\gamma_d \end{bmatrix} \tag{5.56}$$

Finally the error in final velocity, Equation (5.57), is used to update $\mathbf{V}_{go}$, see Equation (5.58). c is a scaling factor which was set to 1 for standard ascent modes on the Space Shuttle. If there was a coasting arc between cut-off and the target orbit c would be set to some value smaller than 1 depending on the length of the coasting arc in order to avoid corrections that oscillate and over-correct from one iteration to the next.

$$\mathbf{V}_{miss} = \mathbf{V}_p - \mathbf{V}_d \tag{5.57}$$

$$\mathbf{V}_{go,i+1} = \mathbf{V}_{go,i} - c\mathbf{V}_{miss} \tag{5.58}$$

For RFA ONE the long thrust arcs can cause significant convergence issues if c is set to 1 when biases are enabled. As mentioned in Section 2.2 by setting both biases to zero the convergence becomes more reliable, albeit at the cost of large errors in the predicted time to insertion. In order to reliably converge to an accurate solution a two-phase homotopy approach can be used. First the PEG is run without biases until $\mathbf{V}_{miss}$ reaches a plateau and then biases are enabled to continue converging to a zero $\mathbf{V}_{miss}$. During the first phase c can be safely set to 1, however when biases are enabled it needs to be set to a much smaller value. In order to still have a rapid convergence when close to the solution, c can be increased once the magnitude of $\mathbf{r}_{miss}$, as defined by Equation (5.59), falls below a certain threshold.

$$\mathbf{r}_{miss} = \mathbf{R}_p - \mathbf{R}_d \tag{5.59}$$

5.2.2. Quadratic Powered Explicit Guidance

As was previously shown in Section 2.2, the PEG shows some deficiencies when applied to long thrust arcs such as is the case for RFA ONE. The primary issues were the difficult convergence and the sub-optimal solutions. While some improvements to the PEG have been proposed, they were not achieving satisfactory results in all cases. Instead of continuing to try to fix an algorithm that is based on assumptions that are fundamentally broken for RFA ONE, it was decided to abandon it and start over with a new algorithm. The development of this new algorithm presented here was done for this thesis; it makes fewer assumptions and is constructed to be more robust. While this comes at the cost of an increase in computational complexity, it is still very much capable of running in real-time on current flight hardware. It is built upon a number of existing algorithms found in literature. Most parts are based on the semi-analytical single-shooting algorithm developed by Lu [26], however it was extended to a multiple-shooting form [24]. Another change is in the constraints, where the formulation of Pan [27] was used. Furthermore the new algorithm turns out to be similar to the OPGUID algorithm that was considered as an alternative to the PEG for SLS. Given that one of its defining features is the quadratic approximation of the unit-thrust direction, it was decided to name this algorithm the Quadratic Powered Explicit Guidance (QPEG).

5.2.2.1. Optimal ascent

Similarly to the PEG, the QPEG is based on optimal control theory, see Section 4.2, applied to the exo-atmospheric rocket ascent problem. The problem statement is therefore very similar, with the goal being to minimize a generic performance index as given by Equation (5.63) while subject to the equations of motion given by Equations (5.60) to (5.62). This performance index can be used to formulate a range of different optimization problems, such as minimum-propellant or minimum-time [26].

$$\dot{\mathbf{r}} = \mathbf{V} \tag{5.60}$$

$$\dot{\mathbf{V}} = \mathbf{g}(\mathbf{r}) + \frac{T}{m(t)}\hat{\mathbf{P}}(t) \tag{5.61}$$

$$\dot{m} = -\frac{T}{v_{ex}} \tag{5.62}$$

$$J = \phi(\mathbf{r}_f, \mathbf{V}_f, t_f) + \epsilon \int_0^{t_f} \frac{T}{v_{ex}} dt \tag{5.63}$$

For this first part of the derivation we will use a linear gravity model, as given by Equation (5.64) where $\overline{r}$ is an average radius over the trajectory. This approximation leads to the same result as when we neglect the second term of Equation (5.4) in the derivation of the PEG. When the change in radius over the trajectory is small, such as an ascent to Low Earth Orbit (LEO), the errors with respect to the inverse-square model are minimal [26]. A method for reducing the effects of this assumption will be shown in Section 5.2.2.2.

$$\mathbf{g}(\mathbf{r}) = -\frac{\mu}{r^3}\mathbf{r} \approx -\frac{\mu}{\overline{r}^3}\mathbf{r} \equiv -\overline{\omega}^2\mathbf{r} \tag{5.64}$$

The first major difference with the PEG is the use of normalization of the equations of motions. This is done to improve numerical conditioning by ensuring all quantities are close to unity. Distances are normalized by a reference length R_0 such as the Earth radius, accelerations by the corresponding gravity, $g_0 = \sqrt{\mu/R_0^3}$, velocities by $\sqrt{R_0 g_0}$, and time by $\sqrt{R_0/g_0}$. One of the nice benefits of this formulation is that $\mu = 1$ in non-dimensional form. The new redefined equations of motion with respect to non-dimensional time are then given by Equations (5.65) to (5.68) [26]. From this point on only the non-dimensional quantities will be used in this section.

$$\mathbf{r}' = \mathbf{V} \tag{5.65}$$

$$\mathbf{V}' = -\omega^2\mathbf{r} + \frac{T}{m(t)g_0}\hat{\mathbf{P}}(t) \tag{5.66}$$

$$m' = -\frac{T}{v_{ex}}\sqrt{\frac{R_0}{g_0}} \tag{5.67}$$

$$\omega = \sqrt{\left(\frac{R_0}{\overline{r}}\right)^3} \tag{5.68}$$

Based on optimal control theory the Hamiltonian for this problem is given by Equation (5.69), where $\mathbf{p}_r \in R^3$, $\mathbf{p}_V \in R^3$, and p_m are the co-states for $\mathbf{r}$, $\mathbf{V}$, and m respectively, and $L =$

$\epsilon \frac{T}{v_{ex}} \sqrt{\frac{R_0}{g_0}}$. Similarly to the PEG, the co-states satisfy the differential Equation (5.70) and the unit-thrust is given by Equation (5.71) [26].

$$H = \mathbf{p}_r^T \mathbf{V} + \mathbf{p}_V^T \left(-\omega^2 \mathbf{r} + \frac{T}{m(t) g_0} \hat{\mathbf{P}}(t) \right) - p_m \frac{T}{v_{ex}} \sqrt{\frac{R_0}{g_0}} - L \tag{5.69}$$

$$\begin{bmatrix} \mathbf{p}_r' \\ \mathbf{p}_V' \end{bmatrix} = \begin{bmatrix} \partial H / \partial \mathbf{r} \\ \partial H / \partial \mathbf{V} \end{bmatrix} = \begin{bmatrix} \omega^2 \mathbf{p}_V \\ -\mathbf{p}_r \end{bmatrix} \tag{5.70}$$

$$\hat{\mathbf{P}} = \mathbf{1}_{p_V} = \frac{\mathbf{p}_V}{||\mathbf{p}_V||} \tag{5.71}$$

It is now possible to formulate an analytical solution for the co-state equations. Using the definitions in Equation (5.72), the co-state is given as a function of time by Equation (5.73), where I_3 is a 3 x 3 identity matrix [26]. $\mathbf{p}_{V_0}$ and $\mathbf{p}_{r_0}$ are unknown initial conditions for the co-state. This result is very similar to the PEG, however the solution is explicitly expanded around the initial time instead of an arbitrary time t_λ.

$$\boldsymbol{\lambda}(t) = \begin{bmatrix} \mathbf{p}_V(t) \\ -\mathbf{p}_r(t)/\omega \end{bmatrix} \quad \boldsymbol{\lambda}_0 = \begin{bmatrix} \mathbf{p}_{V_0} \\ -\mathbf{p}_{r_0}/\omega \end{bmatrix} \tag{5.72}$$

$$\boldsymbol{\lambda}(t) = \begin{bmatrix} \cos(\omega(t-t_0)) I_3 & \sin(\omega(t-t_0)) I_3 \\ -\sin(\omega(t-t_0)) I_3 & \cos(\omega(t-t_0)) I_3 \end{bmatrix} \boldsymbol{\lambda}_0 \equiv \Phi(t-t_0) \boldsymbol{\lambda}_0 \tag{5.73}$$

Using the linear gravity model it is also possible to obtain a closed-form solution for the state dynamics. First we define two integrals of the thrust acceleration vector Equations (5.74) and (5.75). Then, using definitions in Equation (5.76), the state is given as a function of time by Equation (5.77), where Γ is given by Equation (5.78).

$$\mathbf{I}_c(t, t_0) = \int_{t_0}^{t} \mathbf{1}_{p_V}(\zeta) \cos(\omega \zeta) \frac{T}{m(\zeta) g_0} d\zeta \in R^3 \tag{5.74}$$

$$\mathbf{I}_s(t, t_0) = \int_{t_0}^{t} \mathbf{1}_{p_V}(\zeta) \sin(\omega \zeta) \frac{T}{m(\zeta) g_0} d\zeta \in R^3 \tag{5.75}$$

$$\mathbf{x}(t) = \begin{bmatrix} \mathbf{r}(t) \\ \mathbf{V}(t)/\omega \end{bmatrix}, \quad \mathbf{x}_0 = \begin{bmatrix} \mathbf{r}_0 \\ \mathbf{V}_0/\omega \end{bmatrix}, \quad \mathbf{I}(t, t_0) = \begin{bmatrix} \mathbf{I}_c(t, t_0) \\ \mathbf{I}_s(t, t_0) \end{bmatrix} \tag{5.76}$$

$$\mathbf{x}(t) = \boldsymbol{\Phi}(t-t_0) \mathbf{x}_0 + \Gamma(t) \mathbf{I}(t, t_0) \tag{5.77}$$

$$\Gamma(t) = \frac{1}{\omega} \begin{bmatrix} \sin(\omega(t) I_3 & -\cos(\omega(t) I_3 \\ \cos(\omega(t) I_3 & \sin(\omega(t) I_3 \end{bmatrix} \tag{5.78}$$

In order to obtain a fully analytical formulations for the state propagation, it is required to approximate some of the factors in the integrals. Let us first define new variable x, given by Equation (5.79), which we assume for now to be much smaller than one. This corresponds to a time of flight of around 800 seconds. While this assumption clearly doesn't hold for RFA ONE, it will be shown later that segmenting the flight into small arcs allows us to use the method described below for each arc while still obtaining very accurate results.

$$x = \omega(t - t_0) \tag{5.79}$$

First the unit-thrust direction is approximated using a second-order Taylor series expansion. Expanding both factors separately we obtain Equation (5.80) and Equation (5.81). Combining them and dropping all higher order terms yields Equation (5.82), where $\mathbf{1}_{p_{V_0}}$, $\boldsymbol{\eta}_1$, and $\boldsymbol{\eta}_2$ are given by Equations (5.83) to (5.85).

$$\frac{1}{||\mathbf{p}_V||} = \frac{1}{p_{V_0}} + \frac{1}{p_{V_0}}\left(\frac{\mathbf{p}_{V_0}^T\mathbf{p}_{r_0}}{\omega p_{V_0}^2}\right)x + \frac{1}{2p_{V_0}}\left(1 - \frac{p_{r_0}^2}{\omega^2 p_{V_0}^2} + 3\left(\frac{\mathbf{p}_{V_0}^T\mathbf{p}_{r_0}}{\omega p_{V_0}^2}\right)^2\right)x^2 \tag{5.80}$$

$$\mathbf{p}_V(t) = \mathbf{p}_{V_0} - \frac{\mathbf{p}_{r_0}}{\omega}x - \frac{\mathbf{p}_{V_0}}{2}x^2 \tag{5.81}$$

$$\mathbf{1}_{p_V} = \frac{\mathbf{p}_{V_0}}{p_{V_0}} + \omega(t-t_0)\boldsymbol{\eta}_1 + \omega^2(t-t_0)^2\boldsymbol{\eta}_2 \tag{5.82}$$

$$\mathbf{1}_{p_{V_0}} = \frac{\mathbf{p}_{V_0}}{p_{V_0}} \tag{5.83}$$

$$\boldsymbol{\eta}_1 = \left(\frac{\mathbf{p}_{V_0}^T\mathbf{p}_{r_0}}{\omega p_{V_0}^2}\right)\mathbf{1}_{p_{V_0}} - \frac{\mathbf{p}_{r_0}}{\omega p_{V_0}} \tag{5.84}$$

$$\boldsymbol{\eta}_2 = \frac{1}{2}\left(\left(\frac{\mathbf{p}_{V_0}^T\mathbf{p}_{r_0}}{\omega p_{V_0}^2}\right)^2 - \frac{p_{r_0}^2}{\omega^2 p_{V_0}^2}\right)\mathbf{1}_{p_{V_0}} + \left(\frac{\mathbf{p}_{V_0}^T\mathbf{p}_{r_0}}{\omega p_{V_0}^2}\right)\boldsymbol{\eta}_1 \tag{5.85}$$

Let us now define the integrals of the scalar thrust acceleration for a single-stage vehicle in Equations (5.86) to (5.88) which all have analytical solutions [26]. These integrals are very similar to the acceleration integrals used in the PEG, although the QPEG doesn't require evaluating any double integrals. It should be noted that Equation (5.88) differs from Equation 29 in [26], as the original equation was missing a square on m_0.

$$L = \frac{T}{g_0}\int_0^{t_{go}} \frac{1}{m_0 - |m'|\zeta}d\zeta = \frac{T}{|m'|g_0}\ln\left(\frac{m_0}{m_0 - |m'|t_{go}}\right) \tag{5.86}$$

$$M = \frac{T}{g_0}\int_0^{t_{go}} \frac{\omega\zeta}{m_0 - |m'|\zeta}d\zeta = \frac{\omega T}{|m'|^2 g_0}\left(m_0\ln\left(\frac{m_0}{m_0 - |m'|t_{go}}\right) - |m'|t_{go}\right) \tag{5.87}$$

$$N = \frac{T}{g_0}\int_0^{t_{go}} \frac{\omega^2\zeta^2}{m_0 - |m'|\zeta}d\zeta = \frac{\omega^2 T}{|m'|^3 g_0}\left(m_0^2\ln\left(\frac{m_0}{m_0 - |m'|t_{go}}\right) - |m'|m_0 t_{go} - \frac{1}{2}|m'|^2 t_{go}^2\right) \tag{5.88}$$

Then, by using the approximations $\cos\omega\zeta \approx 1 - \frac{1}{2}(\omega\zeta)^2$ and $\sin\omega\zeta \approx \omega\zeta$, we can now rewrite Equation (5.74) and Equation (5.75) into Equation (5.89) and Equation (5.90) respectively, again dropping all terms of order larger then 2.

$$\mathbf{I}_c(t_{go}) = L\mathbf{1}_{p_{V_0}} + M\boldsymbol{\eta}_1 + \left(\boldsymbol{\eta}_2 - \frac{1}{2}\mathbf{1}_{p_{V_0}}\right)N \tag{5.89}$$

$$\mathbf{I}_s(t_{go}) = M\mathbf{1}_{p_{V_0}} + N\boldsymbol{\eta}_1 \tag{5.90}$$

Using the above definitions for the thrust integrals, the state dynamics solution Equation (5.77) can now be rewritten to yield Equation (5.91) and Equation (5.92) for the position and velocity at cut-off respectively. It should again be noted that Equation (5.92) differs from Equation 34 in [26] as the first N should be replaced by $N/2$.

$$\begin{aligned}\mathbf{r}(t_{go}) &= \mathbf{r}_0 \cos\omega t_{go} + \frac{\mathbf{V}_0}{\omega}\sin\omega t_{go} \\ &+ \frac{1}{\omega}\left(\left(L\sin\omega t_{go} - \frac{N}{2}\sin\omega t_{go} - M\cos\omega t_{go}\right)\mathbf{1}_{pV_0}\right. \\ &\left.+ (M\sin\omega t_{go} - N\cos\omega t_{go})\boldsymbol{\eta}_1 + \boldsymbol{\eta}_2 N\sin\omega t_{go}\right)\end{aligned} \tag{5.91}$$

$$\begin{aligned}\mathbf{V}(t_{go}) &= -\mathbf{r}_0\omega\sin\omega t_{go} + \mathbf{V}_0\cos\omega t_{go} \\ &+ \left(L\cos\omega t_{go} - \frac{N}{2}\cos\omega t_{go} + M\sin\omega t_{go}\right)\mathbf{1}_{pV_0} \\ &+ (M\cos\omega t_{go} + N\sin\omega t_{go})\boldsymbol{\eta}_1 + \boldsymbol{\eta}_2 N\cos\omega t_{go}\end{aligned} \tag{5.92}$$

The co-state at cut-off, which will be required for the transversality conditions, is given by Equation (5.93) and Equation (5.94). Equation (5.94) contains another correction with respect to Equation 36 in [26], as the $\frac{1}{\omega}$ would have meant $\mathbf{p}_r(0) \neq \mathbf{p}_{r_0}$.

$$\mathbf{p}_V(t_{go}) = \mathbf{p}_{V_0}\cos\omega t_{go} - \mathbf{p}_{r_0}\frac{\sin\omega t_{go}}{\omega} \tag{5.93}$$

$$\mathbf{p}_r(t_{go}) = \mathbf{p}_{V_0}\omega\sin\omega t_{go} + \mathbf{p}_{r_0}\cos\omega t_{go} \tag{5.94}$$

5.2.2.2. Linear gravity compensation

While the above formulation for the state and co-state can work rather well for short arcs, the linear gravity assumption starts to cause significant errors when used on longer arcs. This means that when a flight includes a long coast arc or if the burn-time is very large, such as is the case with RFA ONE, the optimality of guidance solution will suffer. The effect is largest in co-state, which determines the optimal thrust direction, rather than on the state dynamics directly. Fortunately, it is possible to largely compensate the effect of the linear gravity assumption using a simple correction that barely increases computational complexity [26, 18].

First we go back to the original inverse gravity model Equation (5.95).

$$\mathbf{g}(\mathbf{r}) = -\frac{1}{r^3}\mathbf{r} \tag{5.95}$$

From this the co-state dynamics are now given by Equations (5.96) and (5.97), a similar result to what was obtaining in the derivation of the PEG.

$$\mathbf{p}_r' = \frac{1}{r^3}\mathbf{p}_V - \frac{3\mathbf{p}_V\mathbf{r}}{r^5}\mathbf{r} \tag{5.96}$$

$$\mathbf{p}_V' = -\mathbf{p}_r \tag{5.97}$$

It is only now than we make the simplifying approximation in Equations (5.98) and (5.99), where $\boldsymbol{\kappa}_0$ is defined in Equation (5.100). The first approximation is equivalent to assuming a linear gravity model from the start.

$$\frac{1}{r^3} \approx \frac{1}{r_0^3} \equiv \omega^2 \tag{5.98}$$

$$\frac{3\mathbf{p}_V\mathbf{r}}{r^5}\mathbf{r} \approx \frac{3\mathbf{p}_{V_0}\mathbf{r}_0}{r_0^5}\mathbf{r}_0 \equiv \omega^2\boldsymbol{\kappa}_0 \tag{5.99}$$

$$\boldsymbol{\kappa}_0 = \frac{3\mathbf{p}_{V_0}\mathbf{r}_0}{r_0^2}\mathbf{r}_0 \tag{5.100}$$

The co-state Equation (5.96) now becomes Equation (5.101). It is clear that $\boldsymbol{\kappa}_0$ acts as a compensation for the linear gravity assumption.

$$\mathbf{p}_r' = \omega^2\mathbf{p}_V - \omega^2\boldsymbol{\kappa}_0 \tag{5.101}$$

Using the differential equation above, the analytical formulation for the co-state at cut-off can be modified to yield Equations (5.102) and (5.103) that replace Equations (5.93) and (5.94) respectively.

$$\mathbf{p}_r(t_{go}) = \mathbf{p}_{V_0}\omega\sin\omega t_{go} + \mathbf{p}_{r_0}\cos\omega t_{go} - \boldsymbol{\kappa}_0\omega\sin\omega t_{go} \tag{5.102}$$

$$\mathbf{p}_V(t_{go}) = \mathbf{p}_{V_0}\cos\omega t_{go} - \mathbf{p}_{r_0}\frac{\sin\omega t_{go}}{\omega} + \boldsymbol{\kappa}_0(1-\cos\omega t_{go}) \tag{5.103}$$

This can also be written as Equation (5.104), from which it can be seen that $\boldsymbol{\kappa}_0$ acts as a bias term to the original solution.

$$\boldsymbol{\lambda}(t) = \begin{bmatrix} \cos(\omega(t-t_0))I_3 & \sin(\omega(t-t_0))I_3 \\ -\sin(\omega(t-t_0))I_3 & \cos(\omega(t-t_0))I_3 \end{bmatrix}\boldsymbol{\lambda}_0 + \begin{bmatrix} (1-\cos(\omega(t-t_0)))I_3 \\ \sin(\omega(t-t_0))I_3 \end{bmatrix}\boldsymbol{\kappa}_0 \tag{5.104}$$

It can also be shown that when the additional terms are included in the Taylor series expansion of the unit-thrust direction, it now becomes Equation (5.105) where $\tilde{\boldsymbol{\eta}}_2$ is given by Equation (5.106). From this it becomes clear that $\boldsymbol{\kappa}_0$ acts as a second-order compensation on the unit-thrust direction. By replacing $\boldsymbol{\eta}_2$ with $\tilde{\boldsymbol{\eta}}_2$ in Equations (5.91) and (5.92) the effect of linear gravity compensation can be included in the prediction of the final state as well.

$$\mathbf{1}_{p_V} = \mathbf{1}_{p_{V_0}} + \omega(t-t_0)\boldsymbol{\eta}_1 + \omega^2(t-t_0)^2\tilde{\boldsymbol{\eta}}_2 \tag{5.105}$$

$$\tilde{\boldsymbol{\eta}}_2 = \boldsymbol{\eta}_2 + \frac{1}{2}\frac{\boldsymbol{\kappa}_0}{p_{V_0}} - \frac{1}{2}\left(\frac{\mathbf{p}_{V_0}^T\boldsymbol{\kappa}_0}{p_{V_0}^2}\right)\frac{\mathbf{p}_{V_0}}{p_{V_0}} \tag{5.106}$$

5.2.2.3. Multi-phase vehicles

The previously defined acceleration integrals where only valid for single-stage vehicles. However the exo-atmospheric flight of RFA ONE involves two stages, the second and the third stage, with the second stage further being split into two phases by the fairing release. However similarly to the PEG it is not difficult to extend these integrals for a multi-phase vehicle [26]. It is assumed that every stage before the last one will burn out completely, with only the last phase varying in length. It is therefore assumed that the burn-times for all phases except the last one are known, the burn-time of the last phase being given by Equation (5.107). Furthermore it is assumed that the other vehicle parameters are known for all phases.

$$t_{b,n} = t_{go} - \sum_{i=1}^{n-1} t_{b,i} \tag{5.107}$$

The scalar thrust acceleration integrals can now be split into the multiple phases. Furthermore it is convenient to re-formulate the vehicle parameters in terms of τ and v_{ex} instead of T, m_0, and m'. In addition to being the parameters that will be provided by the vehicle parameter estimator, this change also ensures that all parameters in the algorithm are normalized. The integrals for each stage are then given by Equation (5.108).

$$L_i = v_{ex,i} \ln\left(\frac{\tau_i}{\tau_i - t_{b,i}}\right) \tag{5.108}$$

$$M_i = \omega v_{ex,i} \left(\tau_i \ln\left(\frac{\tau_i}{\tau_i - t_{b,i}}\right) - t_{b,i}\right) \tag{5.109}$$

$$N_i = \omega^2 v_{ex,i} \left(\tau_i^2 \ln\left(\frac{\tau_i}{\tau_i - t_{b,i}}\right) - \tau_i t_{b,i} - \frac{1}{2} t_{b,i}^2\right) \tag{5.110}$$

In order to account for the non-zero start time of the successive phases it is necessary to apply the correction given by Equations (5.111) and (5.112), where the start time of each phase is given by Equation (5.113) with $t_0 = 0$.

$$\bar{M}_i = \omega t_{i-1} L_i + M_i \tag{5.111}$$

$$\bar{N}_i = \omega^2 t_{i-1}^2 L_i + 2\omega t_{i-1} M_i + N_i \tag{5.112}$$

$$t_k = \sum_{i=1}^{k} t_{b,i} \tag{5.113}$$

The total integrals can then simply be obtained by summing over all the phases using Equations (5.114) to (5.116). These can then be directly used within the state propagation in place of the single-stage equivalents. This method can also easily allow for the addition of fixed-length coasting arcs by simply setting $v_{ex,i}$ to zero, $t_{b,i}$ to the length of the coasting arc, and τ_i to some arbitrary large value.

$$L = \sum_{i=1}^{n} L_i \tag{5.114}$$

$$M = \sum_{i=1}^{n} \bar{M}_i \tag{5.115}$$

$$N = \sum_{i=1}^{n} \bar{N}_i \tag{5.116}$$

5.2.2.4. Trajectory segmentation

As previously mentioned the approximation used for deriving the closed-form solution is only valid for small arcs. In order to obtain accurate predictions of the state and co-state at cut-off it is therefore necessary to split the trajectory into multiple segments. It is then sufficient to ensure that for each segment $\omega(t_{i+1} - t_i)$ is much smaller than one. Another advantage of segmenting the propagation is that the effects of the gravity model approximation is lessened. Instead of having a single average gravity for the entire trajectory, it is now possible for ω

and $\boldsymbol{\kappa}_0$ to be computed for each segment [9]. Contrary to what is suggested by Lu [26], this segmentation cannot simply be performed by splitting up phases into several virtual phases and using the multi-phase approach described in the previous section. The reason for this is that the analytical formulations of the scalar thrust acceleration integrals are already exact. Instead the main source of inaccuracies for long arcs are the second-order approximations for $\mathbf{1}_{p_V}$, $\cos(\omega t)$, and $\sin(\omega t)$ in the vector integrals $\mathbf{I}_c$ and $\mathbf{I}_s$. Solving this essentially requires restarting the integration for each segment, using as initial conditions the final values of the previous segment. By recursively calling the propagation function, as shown in Algorithm 1 where $\mathbf{y}$ is defined by Equation (5.117) and *propagateSingle* is the single segment propagation discussed in previous sections, the added code complexity is kept to a minimum.

$$\mathbf{y} = \begin{bmatrix} \mathbf{V} \\ \mathbf{r} \\ \mathbf{p}_V \\ \mathbf{p}_r \end{bmatrix} \tag{5.117}$$

Algorithm 1 Recursive segmented state and co-state propagation

```
function propagateMultiple($\mathbf{y}_0$,$t_0$,$t_{go}$)
    if $(t_{go} - t_0) > \Delta t_{max}$ then
        $\mathbf{y}_1 \leftarrow$propagateSingle($\mathbf{y}_0$,$t_0$,$t_0 + \Delta t_{max}$)
        $\mathbf{y}_f \leftarrow$propagateMultiple($\mathbf{y}_1$,$t_0 + \Delta t_{max}$,$t_{go}$)
    else
        $\mathbf{y}_f \leftarrow$propagateSingle($\mathbf{y}_0$,$t_0$,$t_{go}$)
    end if
    return $\mathbf{y}_f$
end function
```

5.2.2.5. Automatic gravity averaging

As mentioned in the previous section, one of the main sources of propagation errors is the use of the linear gravity model. The errors are particularly large if a single constant value for ω and $\boldsymbol{\kappa}$ is used. Fortunately the multi-segment approach discussed in the previous section already goes a long way towards minimizing these errors. Having separate gravity constants for each segment allows for a better approximation of the inverse square model [9]. The issue still remains on how to compute the gravity constants for each segment. One obvious choice would be to use the values of $\mathbf{r}$ and $\mathbf{p}_V$ at the start of each segment. However since the altitude tends to mostly increase along an ascent trajectory, this method would lead to an over-estimation of the gravity. It would therefore be preferable to find some estimate for the average values. There exist numerous methods in literature for obtaining these average values, such as a predictor-corrector method [13]. The method implemented for this work relies on propagating the initial state and co-state on a coasting arc with a duration of $(t_{go} - t_0)/2$. Propagation on a coasting arc can be done with greatly simplified equations and therefore doesn't add much computational complexity. This allows evaluating the gravity constants at a point that is close to the true mid-point of the powered segment. In the case of RFA ONE this method offers particularly good results for the third stage since the low TWR means that the trajectory is dominated by gravity and the coasting arc doesn't deviate much from the powered arc. While other more precise methods were also considered, they were discarded as the coast-arc method proved to be sufficiently accurate. In a way this method shows some similarity to the analytical method for computing gravity integrals in the PEG [2].

5.2.2.6. Constraints

Up until this point in the derivation, the focus has mainly been on developing a sufficiently accurate predictor of the state and co-state at the endpoint of an optimal trajectory. However the state at orbit insertion is of course not entirely free, but limited to a sub-space of R^6 by the target orbit constraints. While in theory there could be up to six target constraints, corresponding to the six states, this would require continuous engine throttling [6] which isn't possible on the maiden flight of RFA ONE. Therefore only target orbits with up to five constraints will be considered. For a set of m constraints the final state will have $6-m$ degrees of freedom, which means that an additional $6-m$ transversality conditions will be required to ensure that the state at cut-off is optimal. Lastly the free final time requires the addition of another transversality condition. This means that there will be a total of seven end-point constraints for the seven unknowns $(\mathbf{p}_{V_0}, \mathbf{p}_{r_0}, t_{go})$.

The set of target constraints that will be used for most of the flight is very similar to the one used in the PEG. It consists of constraints on the final altitude, velocity, orbital inclination, and flight path angle. In terms of orbital elements this is equivalent to constraining the semi-major axis, eccentricity, inclination, and true anomaly at cut-off. This similarity does not mean that this is the only set of possible constraints. On the contrary, the use of a standard root-finding problem formulation instead of a predictor-corrector scheme means that in theory any set of constraints could be used. The main difference with the PEG is that instead of constraining the entire orbital plane, only inclination needs to be constrained. The baseline mission is to perform a direct insertion into a circular orbit. The flight path angle is therefore always set to zero and the velocity is pre-computed using Equation (3.11). Insertion into elliptical orbits are also possible with this method, however insertion would be constrained to either apoapsis or periapsis. A free-attachment constraint, i.e. free true anomaly at insertion, would likely be more optimal, but those constraints are singular for circular orbits requiring additional complexity [43]. The constraints are implemented using Equations (5.118) to (5.121), where $\mathbf{1}_N$ is the unit-vector pointing north and the starred variables are the target values.

$$s_1 = \mathbf{r}_f^T\mathbf{r}_f - r_f^{*2} = 0 \tag{5.118}$$

$$s_2 = \mathbf{v}_f^T\mathbf{v}_f - v_f^{*2} = 0 \tag{5.119}$$

$$s_3 = \mathbf{1}_N^T(\mathbf{r}_f \times \mathbf{v}_f) - ||\mathbf{r}_f \times \mathbf{v}_f|| \cos i^* = 0 \tag{5.120}$$

$$s_4 = \mathbf{r}_f^T\mathbf{v}_f - r_f^* v_f^* \sin\gamma_f^* = 0 \tag{5.121}$$

As previously mentioned, the use of four target constraints means that there are two degrees of freedom in the state at insertion. By using the method for reduced tranversality conditions discussed in Section 4.2, the two transversality conditions are formed by finding the vectors corresponding to those degrees of freedom. With the target constraints above the longitude of the ascending node and argument of perigee are free. Using the standard set of equations developed by Pan [27], these two conditions are Equations (5.122) and (5.123), where $\mathbf{h}_f$ is the angular momentum at insertion.

$$s_5 = (\mathbf{p}_{r_f} \times \mathbf{r}_f + \mathbf{p}_{V_f} \times \mathbf{v}_f)^T\mathbf{h}_f = 0 \tag{5.122}$$

$$s_6 = (\mathbf{p}_{r_f} \times \mathbf{r}_f + \mathbf{p}_{V_f} \times \mathbf{v}_f)^T\mathbf{1}_N = 0 \tag{5.123}$$

Normally a free final time would require the additional transversality condition Equation (5.124). This constraint can cause significant numerical difficulty if the L term in the Hamiltonian isn't scaled correctly. It can however be shown that if certain requirements are met, this condition is always satisfied. More specifically the terminal constraints at insertion need to be stationary in the absence of thrust. This is for example satisfied if the constraints are expressed in terms of orbital elements or in case of the set of constraints above when inserting into a circular orbit or at one of the apsis of an elliptical orbit [24]. Since there are there now only six equations for seven unknowns, one of the unknowns could be eliminated. It is however not obvious which one to eliminate, so Equation (5.124) is replaced by Equation (5.125) instead which can be trivially be solved. By forcing the norm of the co-states to be unity the numerical conditioning of the root-finding problem is improve [26].

$$H(t_{go}) = 0 \tag{5.124}$$

$$s_7 = \mathbf{p}_f^T \mathbf{p}_f - 1 = 0 \tag{5.125}$$

As the rocket moves closer to the point of insertion and t_{go} becomes smaller, the ability to control the final position is reduced. In other words, a change in final position requires more drastic manoeuvres. This means that keeping the altitude constraint Equation (5.118) until orbit insertion could cause serious issues, such as failure to converge to a solution or large changes in commanded attitude. It is therefore typical to remove this constraint when close to insertion, which was also the case for the PEG. For the QPEG this is handled by replacing the altitude and velocity constraints with a single constraint on the semi-major axis given by Equation (5.126). The switch between the two sets of constraints happens when t_{go} passes below a certain threshold. Of course the new degree of freedom requires an additional transversality condition, which is given by Equation (5.127).

$$\frac{1}{2}\mathbf{v}_f^T \mathbf{v}_f - \frac{1}{r_f} + \frac{1}{2a^*} = 0 \tag{5.126}$$

$$-\mathbf{p}_{V_f} \mathbf{v}_f + \mathbf{p}_{r_f} \mathbf{r}_f r_f v_f \sqrt{\frac{2}{r_f} - \frac{1}{a^*}} = 0 \tag{5.127}$$

5.2.2.7. Multiple-shooting

The methods described in previous sections could be used as is in a single-shooting formulation. The unknowns would be the initial co-state vectors $\mathbf{p}_{V_0}$ and $\mathbf{p}_{r_0}$ as well as t_{go}, yielding seven unknowns. The system of equations is then formed by the m target orbit constraints, $6 - m$ transversality conditions, and the constraint on the magnitude of the co-state. This non-linear system of equations could then be solved by any root-finding algorithm. This formulation is quite similar to the PEG, however the transversality conditions are now explicitly included and all the unknowns are included instead of being re-computed as part of the predictor-corrector scheme. Similarly to the PEG, trying to solve the QPEG using single-shooting will also lead to convergence issues due to the long low TWR third stage burn. To solve this a multiple shooting method was implemented where the trajectory is split into multiple arcs. The obvious locations for the interior nodes are the cut-off and ignition events [24]. This is visualized for a burn-coast-burn trajectory in Figure 5.3, which has two interior nodes. Since the baseline mission scenario doesn't include a coast phase between the second and third stage, we will consider only two arcs with the interior node placed at second stage cut-off. Under the multiple-shooting formulation each interior point adds twelve unknowns to the system, namely the initial state and co-state at the start of each new arc, denoted by $\mathbf{y}^+$ in Figure 5.3. The corresponding twelve new equations are the continuity conditions between the propagated values of $\mathbf{y}$ at the end of an arc and the initial conditions at the start of the next arc. At the patch point of the second and third stage this then gives Equation (5.128).

$$\mathbf{y}_{S2,f} - \mathbf{y}_{S3,0} = 0 \tag{5.128}$$

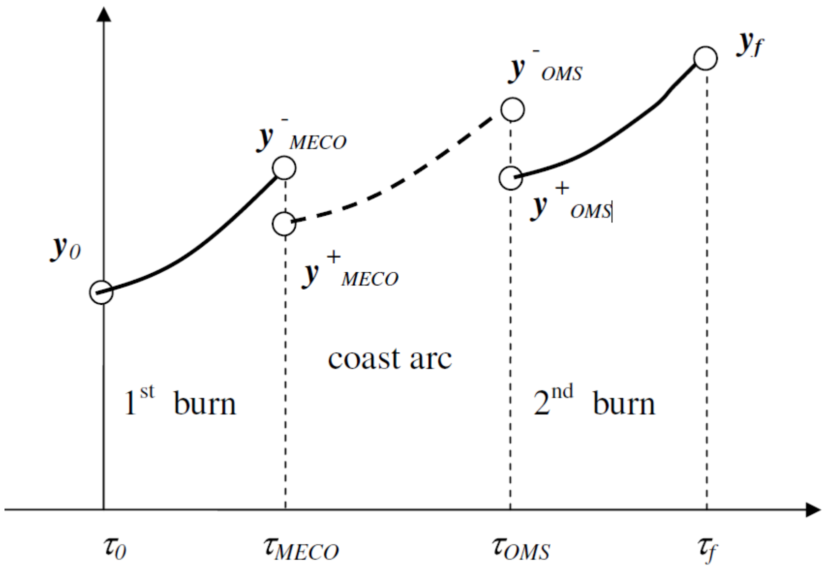


Figure 5.3: Multiple-shooting formulation for optimal exo-atmospheric ascent with coast [24]

The full system now has 19 unknowns, as shown in Equation (5.129) and 19 non-linear equations. Although a large number of unknowns usually increases complexity, solving the system of equations is made easier by the decrease in non-linearity. While the full system of equations could be used during the entire flight, it is computationally inefficient to keep the interior node once the second stage is depleted. Therefore the QPEG switches back to the single-shooting formulation during the third stage burn. If two arcs are not sufficient to achieve the desired convergence characteristics, it is always possible to introduce an additional arc by splitting the third stage flight in two sub-arcs.

$$\mathbf{x} = \begin{bmatrix} \mathbf{p}_{V_{S2,0}} \\ \mathbf{p}_{r_{S2,0}} \\ \mathbf{V}_{S3,0} \\ \mathbf{r}_{S3,0} \\ \mathbf{p}_{V_{S3,0}} \\ \mathbf{p}_{r_{S3,0}} \\ t_{go} \end{bmatrix} \tag{5.129}$$

5.2.2.8. Bias term

While the QPEG algorithm presented thus far can already be considered as suitable, it still has a few deficiencies when run for the baseline mission scenario. As will be shown in Section 6.1.5 the resulting thrust direction vectors are satisfactory, but the angular rate commands show some artifacts caused by the analytical approximations. Furthermore the predicted time to insertion only converges to the true value at the end of the second stage flight. All these issues could be solved by simply replacing the analytical propagator with a numerical one. Integrating

Equations (5.65), (5.66), (5.102) and (5.103) using a variable-step size integration scheme can indeed yield results closer to the exact solution. However due to limits on the available computational power it isn't feasible to use numerical integration for the entire root-finding process. It was therefore chosen to use a hybrid solution where the numerical integration is only used to compute a bias vector once the root-finding solver has converged. This bias vector is the difference in the constraint values computed using the numerical and analytical methods. On the next guidance cycle the bias vector is added to the constraint vector during the root-finding process. In this way the solution vector converges to the exact solution in just a few guidance cycles and the aforementioned issues are largely resolved. This method of biasing the targets is similar in concept to the *higher-order LTVC* routine used in the guidance of the Orion spacecraft [20]. The numerical integrator can also be used to incorporate more precise force models, such as adding the J_2 term in the gravity model.

It should be noted that this hybrid approach can have some down-sides. Other than the small added cost in computational effort, it can potentially introduce some instabilities. For the solution vector to converge to a stable solution, the change in the bias vector from one cycle to the next needs to be small. This is similar to the assumptions made for the $\mathbf{V}_{bias}$ and $\mathbf{r}_{bias}$ terms in the PEG. For this to be the case the analytical solution has to be sufficiently close to the numerical one. Fortunately this condition is satisfied as long as t_{go} doesn't become too large.

5.2.2.9. Closed-loop guidance

Using the QPEG algorithm for closed-loop guidance is relatively straight-forward and bears a lot of similarity to the PEG. While the guidance itself runs at a high rate to provide smooth commands to the controller, the QPEG algorithm is only executed once every two seconds. On the first cycle the unknown vector is initialized by using the initial guesses $\mathbf{p}_{V_0} = \mathbf{V}_0/||\mathbf{V}_0||$, $\mathbf{p}_{r_0} = \mathbf{0}$, and $t_{go} = \sum_i^n t_{b,i}$ as suggested by Lu [26]. The state and co-state values at the interior node are obtained by propagating the current state to the second stage cut-off. An alternative would be to use the reference trajectory. The initial state is set to the current normalized navigation state. Then Powell's dog-leg trust-region method is used to solve the root-finding algorithm. On each iteration first the state and co-state are propagated to the end of each arc and then the constraint values are computed. Once it has converged, the commanded thrust direction and attitude rate can be computed using Equations (5.130) and (5.131) where the co-states are given by Equations (5.102) and (5.103) with t_{go} as the time since the last update. In order to reduce the number of iterations the unknown vector is propagated using Equations (5.96) and (5.97) and $dt_{go}/dt = -1$ before calling the QPEG again. Finally the stage three engine shut-down command is sent once t_{go} reaches zero.

$$\hat{\mathbf{P}}(t) = \frac{\mathbf{p}_V(t)}{||\mathbf{p}_V(t)||} \tag{5.130}$$

$$\omega_{B/I}(t) = \frac{\mathbf{p}_V(t) \times \mathbf{p}_r(t)}{||\mathbf{p}_V(t)||^2} \tag{5.131}$$

While the QPEG has been built from the ground up to be more robust and have less convergence failures than the PEG, convergence of the root-finding algorithm is still not always guaranteed. As will be shown in Section 6.1.1 these convergence failures can still happen for some cases with extremely low performance, such as multiple engine outs. There should therefore be some protections to ensure that these cases are handled correctly, otherwise erroneous attitude commands could be generated leading to unnecessarily early flight termination. It should be reminded that the maiden flight of RFA ONE is primarily a technology demonstration of the various rocket systems. While achieving the nominal orbit would definitely be the best scenario, any degraded orbit would still be regarded as a mission success. Furthermore even if no orbit is reachable, the rocket should still fly a stable trajectory such that all the stages can be tested by burning out until depletion.

The first step is to identify a convergence failure and prevent the resulting guidance solution from being used to generate commands. This can be done by simply putting a limit on the number of iterations for the root-finding algorithm and only using solutions that converged before hitting this limit. The mitigation for non-convergence can then be split into two cases:

1. The QPEG has never converged before, so no previous solution is available.
2. The QPEG had converged on a previous guidance cycle, which means a previous solution is available.

For the first scenario there is nothing that the QPEG can do to generate attitude commands. Instead a status flag is set to error, which signals to the guidance to fall back to another method. This could be for example going into attitude hold, flying a gravity turn, or switching back to open-loop by following the reference trajectory. This last option is what was implemented, given that it tends to provide the closest trajectory to the optimum. On every update cycle of the QPEG a new attempt at convergence is made. The idea being that convergence is generally correlated with time to insertion, so there is a chance it will still successfully converge at some future point in time.

In the second scenario, not all is lost. Thanks to the high accuracy of the predictor and the in-flight estimation of the vehicle parameters, see Section 5.3, the guidance solutions are generally close to the true optimum and don't need to be updated much after initial convergence. As will be shown in Section 6.4 the only point where the solution requires a significant update is at the start of the third stage. For the other parts of the flight an old solution can simply be propagated forward in time, using the same mechanism for reducing iterations, without incurring significant performance losses. In essence this method is equivalent to the solution hold that has been proposed for the PEG on SLS [10]. Given that this situation is still not ideal, the status flag is set to warning and it tries to re-converge on every update cycle of the QPEG. On non-convergence the bias term is also reset to zero, with the rationale being that a large bias can sometimes be the cause of the non-convergence.

5.3. Parameter estimation

As was noted in Section 4.3, the estimation of the vehicle parameters used by the guidance can help improve its robustness. While the navigation provides accurate estimates of the current translational and rotational state of the vehicle, these are not sufficient to perform accurate predictions of the vehicles future state. More specifically, the guidance needs to be able to predict the future thrust acceleration profile. This requires knowledge of the acceleration parameters, v_{ex} and $\tau = \frac{m}{\dot{m}}$, for each phase as well as the length of each phase, t_b. The importance of these parameters becomes evident from the thrust acceleration integrals used by both the PEG and QPEG, see Sections 5.2.1.4 and 5.2.2.3.

The rocket is outfitted with many sensors, some of which can be used for estimating the vehicle parameters. These range from sensors providing information about the entire system, such as the IMU, to sensors measuring more specific quantities, such as level sensors in the propellant tanks and pressure sensors in the engine. The IMU can provide instantaneous acceleration data in the form of velocity increments over a small integration time. The IMU measures all acceleration acting on it except for the gravity. In vacuum it will therefore primarily sense the acceleration caused by the thrust. The navigation then provides a sensed acceleration signal that has been corrected for measurement biases. It should however be noted that this signal still contains the measurement noise, which can be modeled as white noise. It should also be noted that the sensed acceleration represents the acceleration at the IMU location and not the rockets CoM. Therefore any acceleration with respect to the CoM, caused for example by vibration or high angular rates, will also be sensed. The sensed acceleration magnitude is being used for parameter estimation on SLS [3]. The level sensors could provide valuable propellant mass estimates, which could be used in turn to estimate t_b. However the sensor design is still to be finalized and it is questionable if they will be reliable enough to be used for parameter estimation, or if they will just be used to issue engine cut-off commands. Similarly the engine pressure sensors could be used to provide information about the thrust, but they are prone to failure either due to freezing or clogging up with combustion products.

In order to properly model the estimated parameters and provide reasonable initial uncertainty estimates it is important to understand their behaviour. The exhaust velocity v_{ex} is likely to be the parameter with the least uncertainty. Given that it primarily depends on the nozzle geometry and to a lesser amount on combustion efficiency, it should be rather well known from engine tests and it should also change very little during flight. Furthermore the pressure loss term should be negligible since only vacuum flight is considered. The only case where it would significantly deviate from the nominal value is if the engine suffers some kind of partial failure damaging some of the components. In such a scenario the engine could still provide close to the nominal thrust but v_{ex} would be lower. However such a failure would likely only occur during engine start-up. Another minor source of uncertainty is the RCS gas consumption, as the slight increase in total mass-flow paired with unchanged thrust would result in a small decrease in effective v_{ex}. The other two parameters, τ and t_b, are very similar to each other. As can be seen in Equations (5.132) and (5.133) they both provide information about the current mass of the vehicle. These two parameters will have a larger uncertainty, mainly due to the uncertainties in propellant mass and mass-flow rate. If it is assumed that the $\dot{m}$ is constant, then they should both decrease linearly with time with a predefined slope. However $\dot{m}$ may not entirely be constant throughout the flight, for example due to temperature and pressure changes at the pump inlets. The main difference between these two parameters is that as will be shown later τ is observable using just the sensed acceleration, while t_b is not.

$$\tau = \frac{m_{dry} + m_{propellant}}{\dot{m}} \tag{5.132}$$

$$t_b = \frac{m_{propellant}}{\dot{m}} \tag{5.133}$$

5.3.1. A simple linear Kalman Filter

For the initial version of the parameter estimator it was decided to just use the sensed acceleration from the IMU. In vacuum it is nearly equal in magnitude to the thrust acceleration, given by Equation (5.134). Simply taking the magnitude of the sensed acceleration introduces a positive bias due to the measurement errors in the side-ways directions. This bias is proportional to $(\epsilon_{meas}/a_T)^2$, which means the sensor noise causes a negligible bias since the sensor noise is much smaller than the thrust acceleration. However vibration could cause a significant bias if the sample frequency of the velocity increment is large compared to the lower bound of the vibration spectrum. An alternative would be to use the measured acceleration along a single axis. This would however require knowledge of the thrust direction in the body frame, which varies over time due to constantly changing TVC deflections. For this initial version only the magnitude will be used as the issues it poses will be ignored for now.

Combining all the measurements from different points in time while also taking into account the dynamics of the parameters requires some sort of filter. No filter designed to estimate all the required parameters was found in public literature. For example the thrust factor algorithm used for the Space Shuttle and SLS, which uses a simple first-order low-pass filter, requires a separately computed mass estimate [3]. It was therefore decided to develop a new parameter estimation algorithm using the Kalman filters described in Section 4.3. Equation (5.134) could be used directly as the measurement with τ and v_{ex} as states, however there are two problems. The first minor issue is that it is non-linear, which would require usage of the potentially unstable extended Kalman filter. The second more important issue is that in this form v_{ex} is not observable if a single measurement is used per update. For v_{ex} to be observable multiple measurements need to be batched into a single update, which increases the algorithmic complexity and decreases robustness to unexpected step-changes in the states.

$$a_T(t) = \frac{T}{m_0 - \dot{m}t} = \frac{v_{ex}}{\tau_0 - t} = \frac{v_{ex}}{\tau(t)} \tag{5.134}$$

Fortunately both of these issues can be solved using a simple change in variables. By defining the state as Equation (5.135) and using the reciprocal of the acceleration, $z = 1/a_T$ as the measurement we obtain a linear measurement equation while maintaining linear dynamics. The first state, x_1, represents the current estimate for $1/a_T$ and the second state, x_2, the rate of change of x_1.

$$\mathbf{x} = \begin{bmatrix} \frac{\tau}{v_{ex}} & \frac{1}{v_{ex}} \end{bmatrix}^T \tag{5.135}$$

Under the assumption of a constant mass-flow rate the dynamics of the system are then given by Equation (5.136) where Δt is the time since the previous guidance cycle.

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \begin{bmatrix} \frac{t_i - t_{i+1}}{v_{ex}} \\ 0 \end{bmatrix} \quad \rightarrow \quad \mathbf{\Phi} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \tag{5.136}$$

The measurement equation is then defined by Equation (5.137), where t_i is the current time and t_z is the time of the measurement. When the measurements are not batched, but used to update the state as soon as they are received, t_i will be equal to t_z which means z_i does not directly depend on x_2. This is logical given that it isn't possible to estimate both states using a single measurement. Instead the observability of x_2 is provided by the off-diagonal term in the state-transition matrix which will introduce a correlation between the two states over time. This property can be seen in Equation (5.138), where $\mathcal{O}$ is clearly of full rank which proves that all states are observable [33].

$$z_i = \frac{1}{a_T(t_z)} = \frac{\tau}{v_{ex}} + \frac{1}{v_{ex}}(t_z - t_i) = \mathbf{Hx}, \qquad \mathbf{H} = \begin{bmatrix} 1 & t_z - t_i \end{bmatrix} \tag{5.137}$$

$$\mathcal{O} = \begin{bmatrix} \mathbf{H} \\ \mathbf{H\Phi} \end{bmatrix} = \begin{bmatrix} 1 & t_z - t_i \\ 1 & \Delta t + (t_z - t_i) \end{bmatrix} \tag{5.138}$$

Using the reciprocal of the acceleration as the measurement causes some difficulties when defining the measurement noise covariance matrix $\mathbf{R}$. Strictly speaking this transformation means that the errors in the measurement can no longer be represented by additive Gaussian white noise. However if the errors are small with respect to the thrust acceleration, which was already assumed earlier in this derivation, the errors on the reciprocal can be found using a linear approximation in Equation (5.139). This means that R, given by Equation (5.140), becomes dependent on the current acceleration.

$$\frac{1}{a_T + N(0, \sigma_{noise})} \approx \frac{1}{a_T} - N\left(0, \frac{\sigma_{noise}}{a_T^2}\right) \tag{5.139}$$

$$\mathbf{R} = \left(\left(\frac{1}{a_T(t_i)}\right)^2 \sigma_{noise}\right)^2 \tag{5.140}$$

The current acceleration value could be obtained in a number of ways. For example the nominal value could be computed from nominal values of τ and v_{ex} or the current measured value could be used. Instead it was chosen to use the currently estimated acceleration using Equation (5.141). This then yields Equation (5.142).

$$\tilde{a}_T(t_i) = \frac{\tilde{v}_{ex}}{\tilde{\tau}(t_i)} = \frac{1}{x_{1,i}} \tag{5.141}$$

$$\mathbf{R} = \left(x_{1,i}^2 \sigma_{noise}\right)^2 \tag{5.142}$$

The parts that still need to be defined for a linear Kalman filter are the initial state $\mathbf{x}_0$ estimate and its uncertainty $\mathbf{P}_0$ as well as the process noise $\mathbf{Q}$. For $\mathbf{x}_0$ the nominal values can easily be pre-computed from the key parameters of contributing sub-systems. The uncertainties are a bit more difficult to obtain, given the multitude of contributing random distributions and not all transformations being linear. Currently a mix of uniform and normal distributions is used to model the various parameters of the rocket. It is therefore not even guaranteed that the initial states are even normally distributed. In order to not be too dependent on the specific definition of the vehicle parameters, $\mathbf{P}_0$ is computed by taking a large number of random samples for the vehicle parameters using the same function used for Monte-Carlo simulations and computing the covariance matrix from those samples. Determining the process noise isn't as straight-forward since it needs to cover a broad range of model inaccuracies. In the current process model it was assumed that both v_{ex} and $\dot{m}$ are constant. If they are not constant, which is likely going to be the case to some extent, then a process noise is required to ensure that the estimates of the state do not diverge from the true states as the filter becomes overly confident. On the other hand if it is set too high the state uncertainty will stay large and the estimates will be quite noisy. In practice this means that the values for the process noise will have to be determined experimentally as a trade-off between the ability to track the true states and minimizing the noise in the estimates. Getting the best of both worlds could be achieved by using a smoother, such as described in Section 4.3.3.

The parameter estimator discussed up to this point works best when there are no large discontinuities in the vehicle parameters. While with the right process noise it can handle some discontinuities, it will take some time to re-converge to the new value. This could be acceptable for unforeseen and otherwise undetected changes in the vehicle parameters. There are however a few events where we know in advance that such discontinuities will occur. These are the mass jettisons, such as fairing separation, or changes in the propulsion parameters, such as the use of a different engine after staging. To handle these discontinuities the same phases that were originally defined for the thrust acceleration integrals can be re-used, since a phase was defined as having constant propulsion parameters and continuous mass. Given that these phase changes are consecutive to commands issued by the GNC system, the parameter estimator is always aware of when a phase change is about to happen. The only uncertainty is the delay between the command being issued and the system response. The actions taken by the parameter estimator depend on the type of discontinuity. In case of staging where there is both a large change in mass and a switchover to a new engine, the states after the phase change will show very little correlation with the ones before and the filter is simply reset with a new initial guess. If the phase change is due to a small mass jettison or a change in throttle setting, then it would be a waste to throw out the current state estimates. Instead the relevant states can be modified either through addition or scaling and the state uncertainty is artificially amplified to account for the unknown magnitude of the change.

5.3.2. Disturbances

As previously mentioned there are a number of disturbances that could cause the parameter estimator to diverge if not handled correctly. These disturbances are differentiated from the measurement noise in that they act more like a one-sided bias to the measurement. This means they can't simply be accounted for by the measurement uncertainty, as that assumes a zero-mean additive noise. Fortunately most of these disturbances are directly or indirectly correlated to other inputs provided to the guidance by the navigation system.

5.3.2.1. Angular rates and acceleration

One of the largest sources of bias in the acceleration is due to the relative movement of the IMU with respect to the center of mass of the vehicle. The IMU is located on the third stage at the top of the rocket giving it both a longitudinal and lateral offset. This means that any movement, either translational or rotational, relative to the center of mass will induce an additional acceleration on top of the acceleration of the vehicle itself. However the rather simplified measurement equation used for the parameter estimation is only meant to model the acceleration of the entire vehicle. Especially rotation around the Y- and Z-axis can cause significant acceleration along the X-axis, as shown by Equation (5.143), which causes large errors since it is roughly aligned with the thrust direction. Given that the IMU will always be located above the CoM throughout the flight, any pitch- or yaw-rate will have the effect of decreasing the sensed acceleration. Trying to correct the measured acceleration for this disturbance would require estimating the position of the CoM, which would add quite a bit of algorithmic complexity while adding a large degree of uncertainty. This issue was also encountered during the development of the guidance for SLS [3]. They noted that large attitude changes primarily occurred at the start of a phase, such as when switching from open-loop to closed-loop guidance. Their solution was therefore to simply not use any of the acceleration measurements during the first part of a phase. In our case in addition to implementing a parameter hold at the start of each phase, there is also an upper limit placed on the measured angular rate for an acceleration measurement to be considered valid. This limit was conservatively chosen to ensure that the relative acceleration caused by angular rates will never exceed the sensor noise even when the distance between IMU and CoM is maximum. While angular acceleration can also cause measurement errors, the induced acceleration, given by Equation (5.144), will always be in a lateral direction which is less of an issue.

$$\mathbf{a}_{\omega} = \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}_{IMU/COM}) \tag{5.143}$$

$$\mathbf{a}_{\dot{\omega}} = \dot{\boldsymbol{\omega}} \times \mathbf{r}_{IMU/COM} \tag{5.144}$$

5.3.2.2. Aerodynamic drag

Up to this point it has been assumed that the parameter estimation is only used during exo-atmoshperic flight and aerodynamic forces have been neglected. However during the first part of the second stage they are small but not entirely negligible. As can be seen in Figure 5.4, while the drag acceleration at ignition is much smaller than the thrust acceleration it is still an order of magnitude larger than the sensor noise level. It can also be seen that the drag acceleration approximately follows an exponential decay curve and quickly becomes irrelevant.

Similarly to angular rates, the aerodynamic drag will also cause a negative bias on the measured acceleration which could cause the estimated parameters to diverge. The parameter hold that was implemented at the start of the second phase also helps mitigates the effects of drag, but it might have to be excessively long to account for cases where staging occurs at a lower altitude than expected. Instead it was chosen to apply a correction to the acceleration measurement.

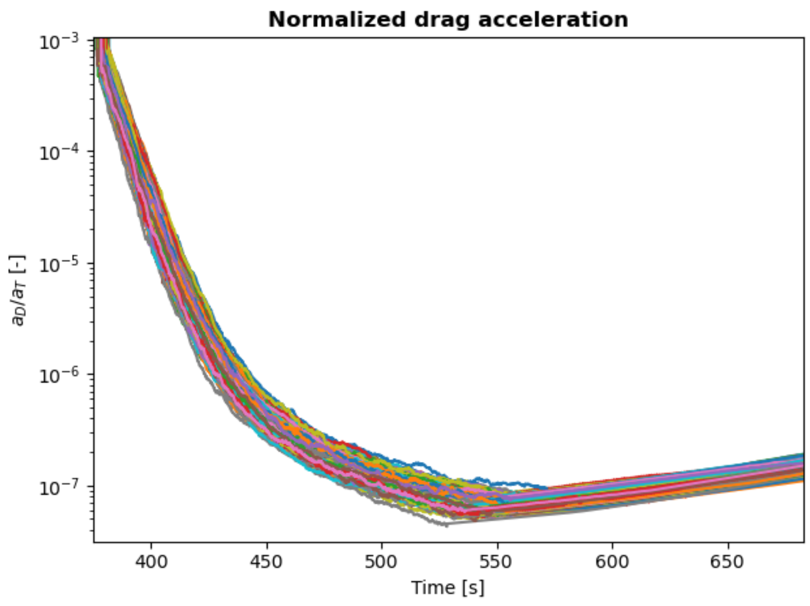


Figure 5.4: Normalized drag acceleration during S2 flight

In order to correct the acceleration measurement, first an estimate of the aerodynamic drag has to be obtained. As was already noted the drag acceleration seems to exhibit an exponential decay. This can be explained by examining the behaviour of the various contributing factors, shown in Equation (5.145) [28]. The largest variation in drag will come from the atmospheric density ρ, which for a constant temperature will have an exponential decay as a function of altitude. The airspeed V_A stays relatively constant in the first part of the second stage flight due to the TWR being close to one. The drag coefficient C_D is also relatively constant as the angle of attack doesn't change much after the initial pitch-up. Finally the effect of the decrease in mass is small due to the total mass still being large at the start of the stage. The behaviour of the drag acceleration will therefore by dominated by the change in atmospheric density. This can be seen in Figure 5.5, where despite the variation in altitude at stage separation the dispersion in terms of acceleration magnitude is small and the slopes are all very similar.

$$a_D = \frac{D}{m} = \frac{\frac{1}{2}\rho V_A^2 C_D S_{ref}}{m} \tag{5.145}$$

Given the findings above, it was decided to use the approximation given by Equation (5.146) to estimate the drag acceleration as a function of the altitude h. In this approximation $a_{drag,0}$ and h_0 are the values of the drag and altitude respectively at stage two ignition and H is the exponential decay coefficient. These parameters can either be obtained from the reference trajectory or by fitting them to the results of Monte-Carlo simulations. Using altitude instead of time as the independent parameter provides better accuracy in off-nominal scenarios where

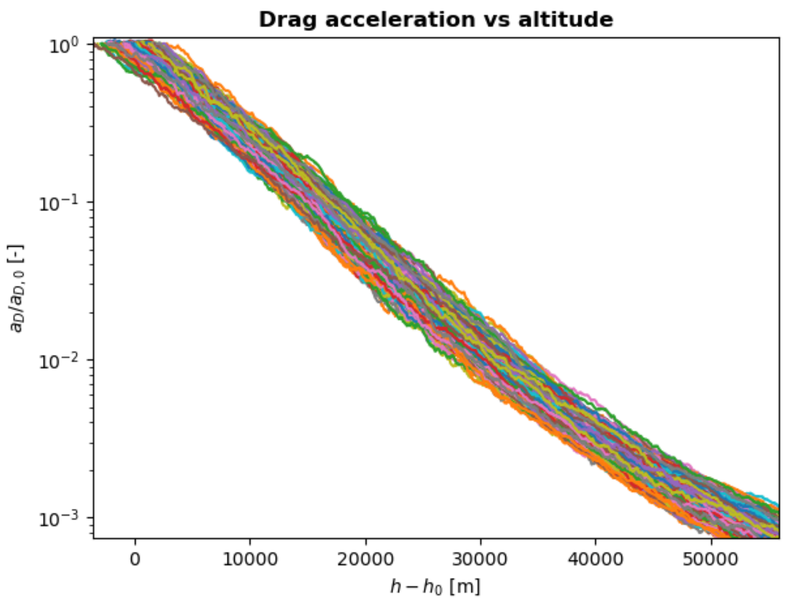


Figure 5.5: Drag acceleration dispersion against altitude

the altitude at staging can be significantly different. The estimated drag is then added to the measured acceleration.

$$a_{drag} = a_{drag,0} \cdot 10^{\frac{h-h_0}{H}} \tag{5.146}$$

In order to account for the errors introduced by the simplified drag estimation, the measurement uncertainty is increased. The increase in uncertainty will be proportional to the drag acceleration itself. In order to be very conservative the proportionality factor has been set to one. The measurement noise σ_{noise} in Equation (5.142) is therefore now defined by Equation (5.147).

$$\sigma_{noise} = \sigma_{noise,IMU} + a_{drag} \tag{5.147}$$

5.3.3. Burn-time estimation

As was explained previously, the guidance requires three parameters for each phase: v_{ex}, τ, and t_b. The Kalman Filter described above can only estimate the first two parameters. The burn-time on the other hand is not directly observable from the sensed acceleration, because inert mass and propellant mass are indistinguishable from each other. In order to make the burn-time observable, an additional sensor would be required to measure the propellant levels. However even without any additional sensors it is still possible to obtain an improved estimate of the burn-time using just τ. As can be seen in Equations (5.132) and (5.133) t_b and τ differ only by the presence of the dry mass in the numerator, which for rocket stages is much smaller than the propellant mass. Furthermore the uncertainty on the dry mass is smaller than the uncertainty on the propellant mass and mass-flow rate, because the structures can easily be weighed before launch. The only large source of uncertainty in the dry mass is the amount of residual propellant at engine cut-off. This means that τ and t_b exhibit a high degree of correlation, as will be demonstrated in Section 6.2.2. Given an estimate for τ, it is therefore possible to obtain a better estimate for the total burn-time of a stage than the nominal value. This estimate is computed using Equation (5.148), where the zero subscript indicates that these values are evaluated at ignition. If it is assumed that the mass-flow rate is constant, the values at ignition simply differ from the current values by the time since ignition. If however there are large changes in mass-flow during flight, such as due to a change in throttle setting, this method might yield worse estimates than the nominal values because it erroneously retroactively assumes that the new mass-flow applies to the whole stage flight time.

$$t_{b,0} = \frac{\tau_0}{\tau_{0,nom}} t_{b,0,nom} \tag{5.148}$$

5.4. Insertion into alternate orbits

The previous sections have mainly dealt with the closed-loop guidance for a nominal flight. The parameter estimation techniques can be used to handle uncertainty in the performance parameters, as long as the available performance is not significantly less than expected. However as demonstrated by the recent launch of Starship[1], a fully nominal flight is far from a guarantee, especially for the first flight. It is in fact quite rare for the first launch of a rocket, especially for a young company, to be entirely successful. Furthermore due to budget constraints it might not always be possible to test every single part of the rocket as extensively as we could wish. It is therefore quite possible that some sort of anomaly might occur during the first launch of RFA ONE, such as an engine out.

Nevertheless the guidance system should still be able to handle these kind of partial failures while ensuring that the mission objectives are satisfied as much as physically possible. The worst thing that could happen from a guidance perspective would be if the algorithms went unstable and started issuing hazardous commands that would only worsen the situation. This topic has already been studied extensively for other launchers, including for SLS [10, 11] where the ambitious target of bringing humans to the Moon and to Mars require that it can safely complete its mission, even with an in-flight mishap. It is often these off-nominal scenarios that create the most challenging and complex guidance problems, as illustrated by the many abort modes on the Space Shuttle [44], of which the Abort to Orbit (ATO) was required once [45], which in part drove the modularity of the PEG such that it could support the variety of target constraints [2]. While the first flight of RFA ONE does not have to take into account as many safety requirements, it should still try to obtain as much information as possible on the structural and propulsion behaviour and a stable flight is therefore of paramount importance, even if it fails to reach orbit. As will be shown in Section 6.3, not doing anything and continuing towards the nominal target can lead to objectively worse outcomes compared to targeting a degraded orbit.

While almost all the literature focuses on handling discrete detectable events, such as engine out, this might not be the only cause of a failed orbit insertion on RFA ONE. The very limited performance margins combined with a relatively large uncertainty on the parameters contributing to the performance means that even for a seemingly nominal flight the third stage could run out of propellant before orbit insertion. It is therefore not sufficient to only take action if one of these events occurs, but instead the orbit selection algorithm should evaluate the feasibility of successfully achieving orbit insertion using broader criteria. After reviewing the literature on this topic, it was decided to develop a new algorithm to perform the orbit selection.

While the impact of an engine out is rather high, it should still be viewed as a low probability event. Any changes made to the flight software should therefore be kept to the minimum required to prevent a catastrophic failure. As each additional code path will require additional testing and potentially introduce new points of failure, it is undesirable to add a lot of complexity.

[1]`https://spacenews.com/starship-lifts-off-on-first-integrated-test-flight-breaks-apart-minutes-later/`

Under no circumstances should these additions affect an otherwise nominal flight. This was the overall design philosophy behind the engine out (EO) implementation in the guidance for SLS [10] and will also serve as a guiding principle in the following development.

5.4.1. Specification of target orbits

The over-arching goal of the this function is to maximize the probability of mission success. Clearly defining mission success in such a way that it can be formulated as a mathematical function is however not a trivial matter. The objective of the maiden flight of RFA ONE is first and foremost to demonstrate to the world the capabilities of the launcher, ideally by reaching a stable orbit. While the nominal mission involves the insertion of a 150 kg payload into a 500 km orbit, it would still be considered a great success if the third stage completes just a few orbits before re-entry. Even if achieving orbit is not possible, all the stages should still burn until depletion and staging should occur as planned, such that as many of the systems can be tested and the telemetry relayed to the ground. It is therefore of paramount importance that the rocket remains stable and a loss of control should be avoided at all costs.

One way of specifying the orbital targets is to provide a list targets ranked in order of preference. The first target would be the nominal one and the other targets would each be designed in such a way as to maximize the number of mission objectives that are satisfied with the available performance. For example for Artemis I this involved the selection of two alternate sets of target constraints at Main Engine Cutoff (MECO) [11], differentiated by their energy requirements. If an engine out occurs close to lift-off the so called Low Energy Alternate MECO Target (AMT-LO) would be used where the rocket would insert into LEO. During the middle part of the flight the High Energy Alternate MECO Target (AMT-HI) is used where the Orion capsule is inserted into a highly elliptical orbit such that the heat-shield could be tested in a high-speed re-entry. Finally if an engine out would occur in the last part of the flight, SLS would Press to MECO (PTM) and simply insert into the nominal target orbit. For RFA ONE the alternate targets could be a series of progressively lower altitude orbits, either circular or elliptical, with as last resort a high altitude sub-orbital flight that demonstrates the capabilities of the second and third stages while ensuring a stable down-link to the ground-stations. This method means that all the decision making of which orbit is more preferable can be done before launch and it also allows for pre-computing threshold triggers for each target.

Another option would be to define a merit function and define a range of acceptable target orbits. One possible merit function could be for example to maximize the periapsis altitude with a fixed burn-time. The bounds would then ensure that the periapsis doesn’t end higher than the nominal one. The guidance would then try to optimize the target parameters during flight. This method could allow for more flexibility and allow the guidance to better react to unforeseen circumstances than a list of predefined target orbits, but it would come at the cost of increased algorithmic and computational complexity for the guidance. It would also be more difficult to verify the implementation given that the number of test-cases would become very large. Furthermore proving that operational constraints, such as the disposal of spent stages, are satisfied would be complicated.

5.4.2. Selection of the target orbit

From the two methods for specifying the target orbits mentioned in the previous section it was decided to implement the first one, given that it had some flight heritage. However as discussed previously the selection process would still have to be different from what was found in literature due to the constraints applicable to the maiden flight of RFA ONE. Instead of only acting on the occurrence of specific failures, it has to work for any kind of under-performance. Furthermore it has to take action as early as possible to maximize the range of viable options.

As the flight progresses the cost of switching target increases, to the point where lowering the target altitude might no longer yield a decrease in burn-time.

The method of velocity thresholds used for orbit selection in case of engine outs on SLS could be adapted to the environment of RFA ONE. There could for example be some minimum thresholds on velocity or specific energy that need to be satisfied at the start of the second and third stage for each target. The guidance would then only change its target at those specific moments based on the current vehicle state. However this method would likely lead to overly conservative target selection in order to ensure successful insertion with the large uncertainties on the vehicle parameters. The velocity triggers would essentially have to be chosen for a worst case scenario. Moreover an excess of propellant on the second stage would cause a lower velocity at the end of the first stage while not necessarily decreasing overall vehicle performance.

Fortunately the guidance will have more information available to make a better selection. The vehicle parameter estimator discussed in Section 5.3 provides improved estimates of the performance parameters of the currently active stage. The QPEG then provides reasonably accurate estimates of the total burn-time required to reach orbit. This can be compared against the estimated available total burn-time to compute the estimated margin. This margin can be used as a feasibility metric to score all the target orbits. Not only does this take into account the current vehicle state but also the expected future performance. The orbit selection is then simply performed by finding the first target in the list whose margin is above a configurable minimum margin threshold. This minimum margin can for example account for uncertainties in the estimated burn-time or ensure propellant reserve after insertion. It can also be varied throughout the flight such that a higher uncertainty during the second stage can be compensated by a larger margin. Another useful property is that during a nominal flight the estimated margin should remain constant if all parameters are estimated perfectly. The margin for the current orbit can be computed at no added cost given that it is provided by the closed-loop guidance solution. A significant change in the margin, either negative or positive, can therefore be used as a signal that some of the inputs have changed in an unexpected manner and that the orbit selection needs to be re-evaluated. Conversely if the margin remains unchanged it can safely be assumed that the currently selected orbit remains the best candidate without having to explicitly evaluate all the other targets. Given that the flown trajectory is fuel-optimal for the currently selected target and other points along the trajectory, it should not be fuel-optimal for the other targets. The margins for the other targets should therefore only decrease during flight, or at least not increase more than the margin of the currently selected target.

6 Results

This chapter will show and discuss the results of various analyses that were performed on the new guidance algorithms. The main focus is the performance improvements provided by the new algorithms as well as some of their individual features. Section 6.1 will go over the simulation results of the QPEG in a 3-DOF environment with perfect knowledge of the vehicle parameters. Then the performance of the parameter estimation is discussed in Section 6.2. After that the target selection algorithm combined with the two other guidance components is analysed in Section 6.3. Finally the performance of the guidance as a whole when integrated with the rest of the GNC system in a 6-DOF simulator is presented in Section 6.4.

6.1. Closed-loop guidance system

In this section the performance improvements of the QPEG over the PEG will be analysed followed by a deeper dive into the benefits of some of the individual features of the QPEG. In order to isolate just the behaviour of the closed-loop guidance a 3-DOF simulator will be used in which the only forces are gravity and thrust. Furthermore in the Monte-Carlo analysis the exact vehicle parameters will be provided to the guidance, which would be equivalent to having a perfect parameter estimator. It should also be noted that staging will occur based on the pre-defined burn-times, although the guidance can cut off the thrust whenever it determines that the desired target orbit has been reached.

6.1.1. Comparison between PEG and QPEG

First we will compare various results given by the QPEG to the PEG data. The results were generated using the same initial conditions and vehicle configuration, so any difference can be entirely attributed to the differences between the two algorithms. Initially we will just look at a single simulation for a vehicle with fully nominal parameters. The first point of comparison is the pitch angle with respect to the local horizontal plane, shown in Figure 6.1. This represents, together with the yaw angle, the primary output command of both algorithms and it determines the shape of the trajectory. It is immediately obvious that there are a number of differences. The first point is that as expected the initial pitch angle for the QPEG is smaller than the one for the PEG. This confirms the expectation that the initial thrust angle for the PEG is indeed larger than the optimum [3]. Another point of interest is that the pitch profile of the QPEG looks smoother then the one for the PEG. The kink that happens at second stage separation is gone as is the step in the middle of the third stage. The only discontinuity in pitch for the QPEG happens towards the end of the flight when the position constraint is released. The pitch angle at the end of the flight, when the rocket is circularizing its orbit, is also lower and is therefore better aligned with the final velocity vector. The sharp increase in pitch at the end is caused by the rocket going into attitude hold mode once it has reached the target orbit and is of no importance for this comparison.

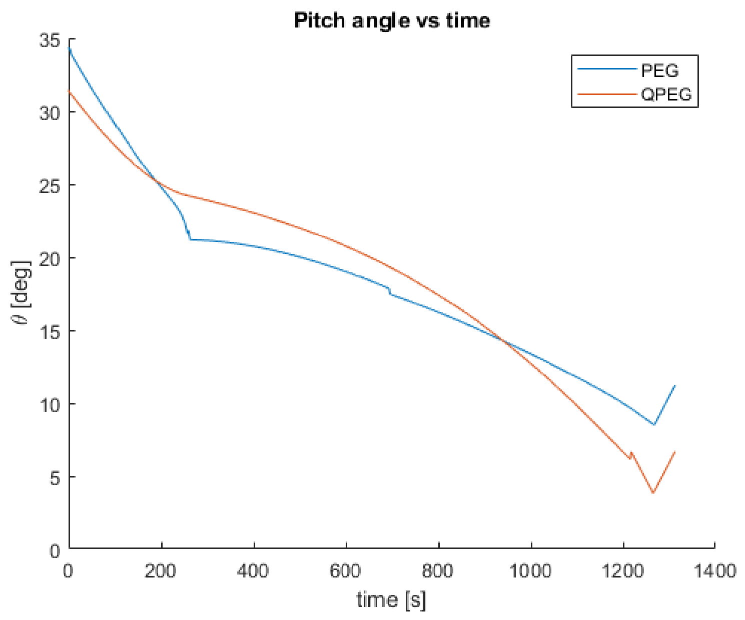


Figure 6.1: Comparison pitch angle PEG and QPEG

Other than giving attitude commands the closed-loop guidance also provides predictions of the time and the state at orbit insertion or other points of interest. These auxiliary outputs can be used for a variety of purposes within the GNC system. Especially the remaining time to insertion, t_{go}, is of special interest to the guidance system, since it will be used to compute the estimated burn-time margins. Since t_{go} decreases over time, it is more useful to analyze the predicted time of insertion, $t_{insertion} = t + t_{go}$, which should remain constant if the predictions are accurate and the solution optimal. The error in the predicted time of insertion is shown in Figure 6.2 for both the PEG and QPEG. Whereas this error is persistent for the PEG during much of the second stage flight, for the QPEG it goes almost immediately to zero. The non-zero error on the first cycle of the QPEG is due to the constraints bias not being initialized yet. The change in $t_{insertion}$ over time for the PEG essentially indicates a discrepancy between the trajectory computed at the beginning of the second stage and the one actually flown in closed-loop. Since the PEG used here is implemented with a numerical propagator which should yield small prediction errors, this again hints at the initial solution not being optimal. It should also be noted that even without the bias term the error of the QPEG is smaller than the one from the PEG.

Of course any improvements in accuracy should not come at the cost of the algorithm no longer satisfying the real-time requirements. Fortunately the number of iterations is very similar for both algorithms, as shown in Figure 6.3. This is true for the initial convergence on the first guidance cycle as well as for each subsequent update of the steering parameters. It should be noted that this cannot be used to draw definitive conclusions on the performance since the computation time per iteration will be different.

While analyzing a single sample at a time can give many useful insights, in order to be able to draw broader conclusions on the performance differences between the two algorithms it is necessary to analyze the behaviour under a large number of combinations for the input parameters. For this we will first look at the conditions for which the two algorithms are able

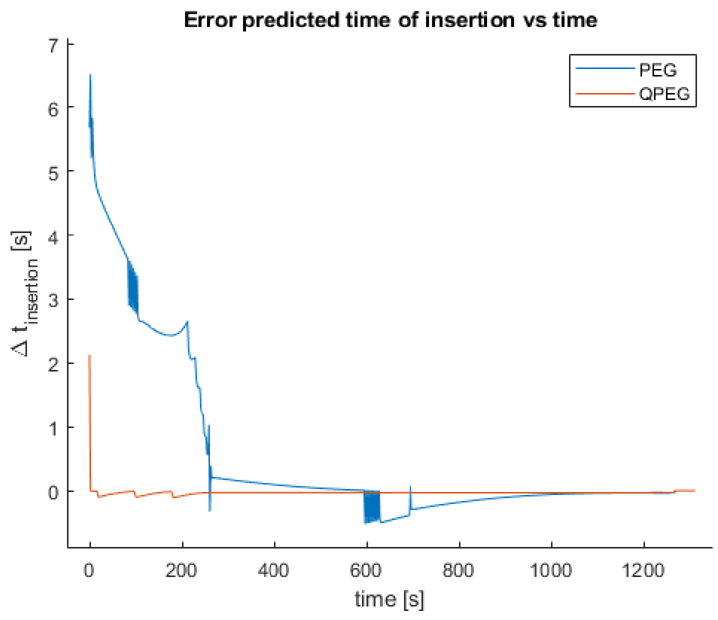


Figure 6.2: Comparison predicted time of insertion error PEG and QPEG

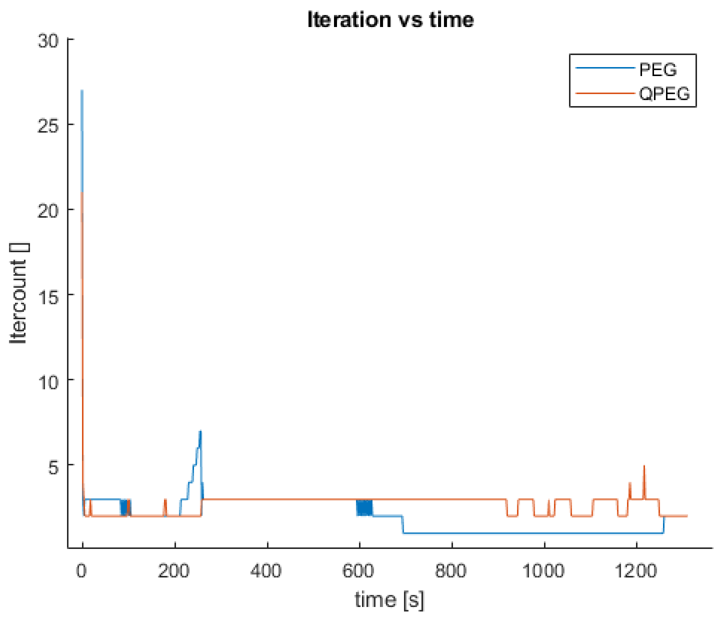
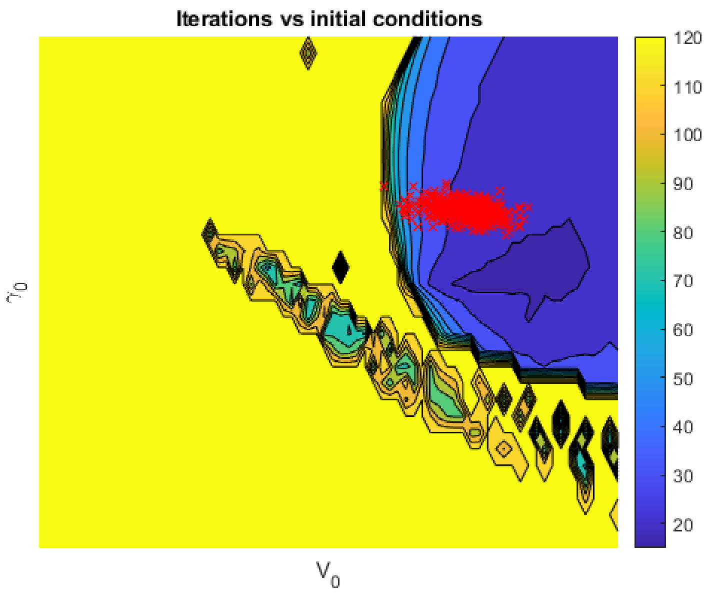


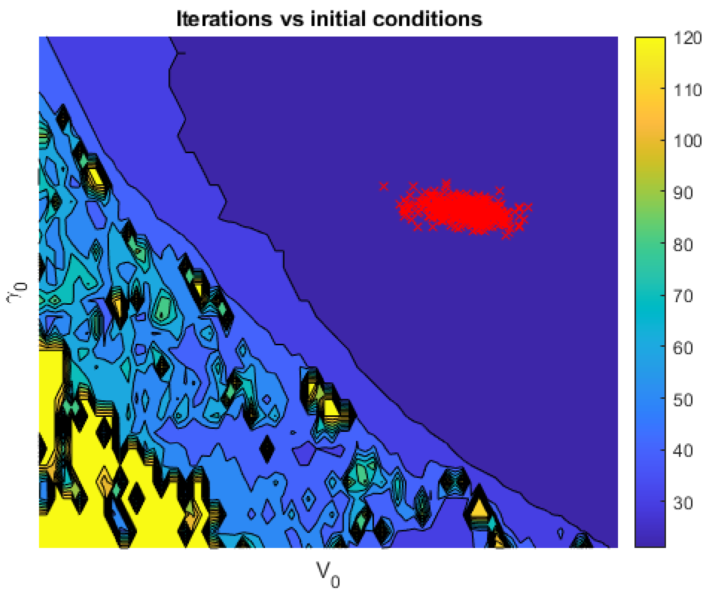
Figure 6.3: Comparison iterations per update PEG and QPEG

to converge to a solution. It is in particular the position and velocity of the rocket at second stage ignition, when the closed-loop guidance is initialized, which are most important. The reason is that of the three inputs (target orbit, vehicle parameters, state vector) to the closed-loop guidance, at initialization both the target orbit and vehicle parameters will be the nominal values loaded into the flight computer. This leaves the state vector estimated by the navigation as the only input with some uncertainty.

The number of iterations required for convergence as a function of the initial velocity and flight-path angle can be seen in Figure 6.4 for the PEG and QPEG respectively. All the yellow areas correspond to combinations where the iteration limit was reached, which means that the algorithm didn't converge. The dispersion of these two parameters for a nominal flight is also shown for reference. The dispersion of the out-of-plane components and the altitude at the end the first stage has little influence on the convergence. It is immediately obvious that for a nominal flight the QPEG is practically guaranteed to rapidly converge to a solution, whereas the PEG had a non-negligible probability of failure. It should be noted that for this analysis the maximum number of iterations was set to 120. In general the two algorithms either converge rapidly or fail to converge to a solution even with a very large number of iterations.



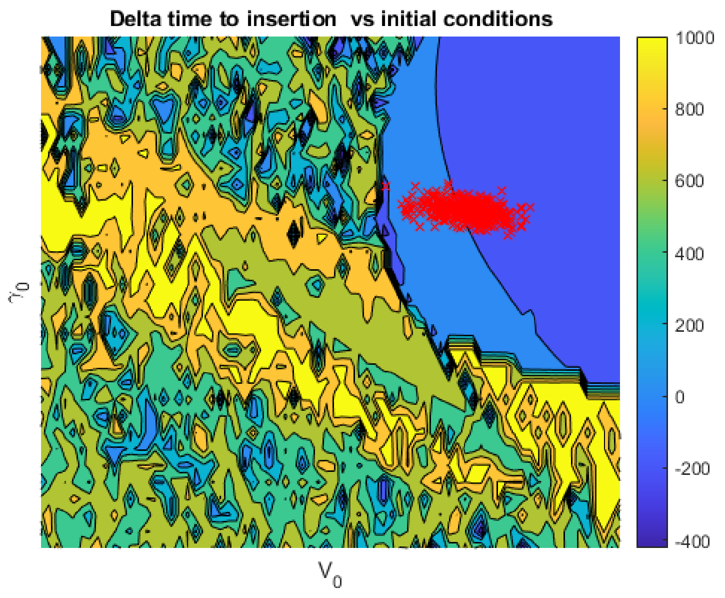
(a) PEG



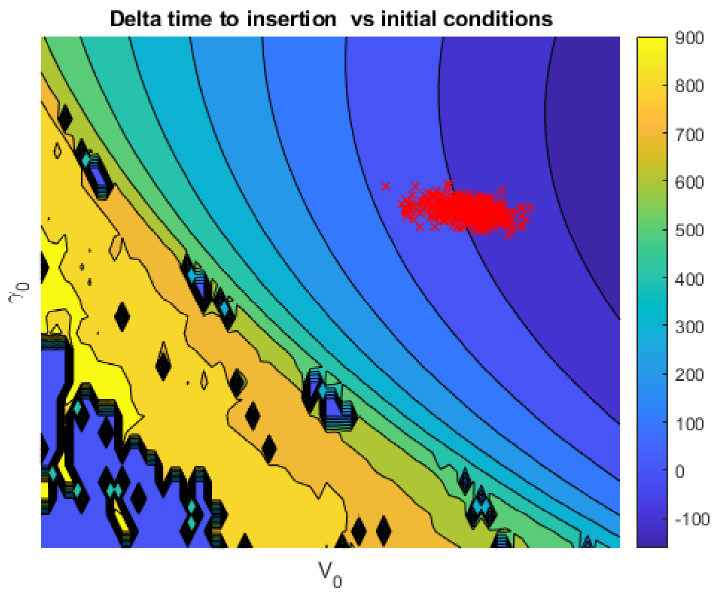
(b) QPEG

Figure 6.4: Number of iterations required for initial convergence

Another important observation is that for both the PEG and QPEG there is a strong correlation between t_{go}, shown in Figure 6.5, and the number of iterations required for initial convergence. This can be explained by the fact that the final state is more sensitive to initial guesses for a larger t_{go}. It should also be noted that some of the solutions that were obtained with the PEG were invalid, as can be seen in Figure 6.5a. This is likely due to the lack of any explicit convergence criteria on the position error in the PEG. Another interesting observation is that whereas the PEG struggles to converge for low initial flight-path angles, the QPEG has no such limitation. Nonetheless the QPEG still has some limits on the initial conditions where it can successfully converge.



(a) PEG



(b) QPEG

Figure 6.5: Predicted time to insertion after initial convergence

There is some odd behaviour for longer burn-times where some combinations of initial conditions cause a convergence failure in the QPEG, whereas all neighbouring combinations do converge. While the sensitivity of the final state to the initial conditions does seem to become very large in those cases, the exact cause has not been identified. Fortunately those limits correspond to flight-times which are much longer than the available burn-time, which means those trajectories would not be of any use. Non-convergence should therefore also be used as a trigger to evaluate alternate target orbits. If it is ever needed to achieve convergence for these very long flight-times, the third-stage burn could be split up into two or more arcs by adding more multiple-shooting nodes.

While improving the convergence for long burn-arcs was the main reason for changing the closed-loop guidance algorithm, a secondary objective was to improve the optimality of the trajectory. In order to analyze this a Monte-Carlo analysis was run with identical conditions for both algorithms, such that differences could be compared for each sample. The effects on the burn-time margin can be seen in Figure 6.6. It should still be noted that only the results where both algorithms had converged are shown here. It therefore excludes all the samples which have a very long burn-arc, which are also the cases where the PEG would have had sub-optimal performance. While the overall distribution for the two algorithms looks almost identical, when taking the difference for each sample it becomes clear that switching from the PEG to the QPEG always has a positive effect on the margins. For the vast majority of cases the improvement isn't very large, which seems to indicate that the PEG was already close to optimal.

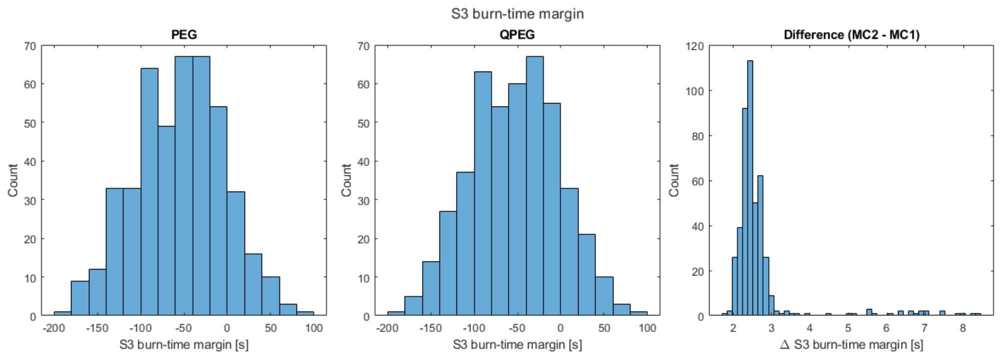


Figure 6.6: Comparison burn-time margin PEG and QPEG

6.1.2. Single-shooting vs Multiple-shooting

One of the main defining features of the QPEG is the usage of a multiple-shooting method instead of a single-shooting method. The main driver for this decision was to improve the convergence for long thrust arcs. As was shown in Section 6.1.1, the QPEG is indeed able to converge for a much larger range of initial conditions than the PEG. The question still remains how the different changes contributed to this improvement. For this purpose a convergence analysis was run for the QPEG where it was forced to run in the single-shooting mode that is normally only used during the third stage flight. Every other part of the QPEG was kept the same as in Section 6.1.1. The results are shown in Figure 6.7, which again shows the number of iterations required for convergence and the dispersion of initial conditions. When comparing this to Figure 6.4b it can be seen that the boundary where a convergence failure is guaranteed is indeed a bit closer to the nominal point. On the other hand the region where convergence is guaranteed remains largely unchanged, mostly because the thin region of non-convergence is still present in the same location. This phenomenon is therefore likely caused by something other than the number of arcs and the cause is still unknown. Another point of interest is that even in single-shooting mode QPEG is able to converge for a much larger range of initial conditions than the PEG. The improved convergence characteristics of the QPEG are therefore mainly coming from other features of the QPEG, such as the use of a more robust root-finding method. While the effects of the multiple-shooting method are not as large as initially expected, it was still decided to continue using it since it has almost no drawbacks. In the future it can allow for some extensions to the algorithm such as specifying an initial guess for the interior point which could improve convergence. The difference in computational effort is small since the state still needs to be propagated over the same time-span, simply divided in two parts, and the additional constraints are trivial to compute. The only added effort comes from the increased size of the linear system, 19 instead of 7 unknowns, that needs to be solved by the root-finding algorithm on each iteration.

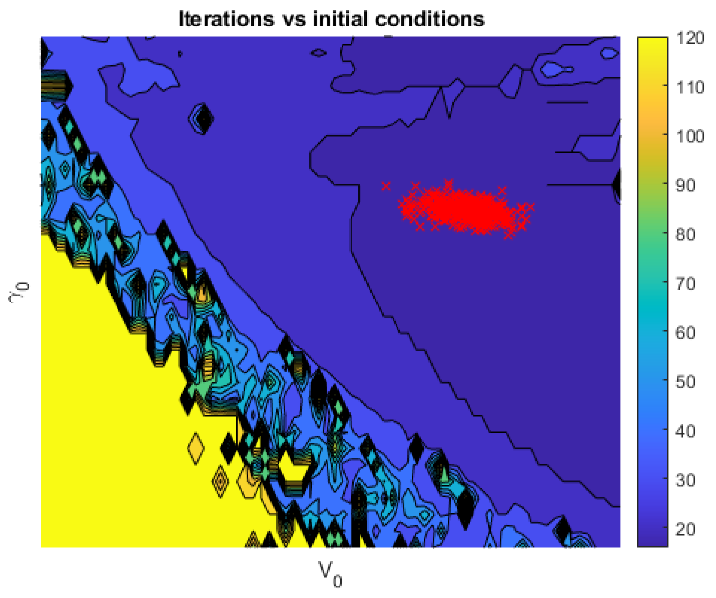


Figure 6.7: Number of iterations required for initial convergence QPEG single shooting

6.1.3. Linear gravity compensation

The compensation for the linear gravity approximation through the use of the added $\boldsymbol{\kappa}$ term was one of the suggestions made by Lu [26]. However its effect wasn't clearly quantified, especially for long thrust-arcs such as required for RFA ONE. Therefore a comparative analysis was set up to measure the effect of omitting the $\boldsymbol{\kappa}$ term. In both cases the use of constraint biasing was disabled such that only the analytical propagator would be used. The effects on the total burn-time can be seen in Figure 6.8, which shows that the linear gravity compensation offers a substantial improvement in performance. This can be explained by the drastically different pitch profile, shown in Figure 6.9. The linear gravity compensation is therefore essential to obtain an optimal pitch profile.

6.1.4. Gravity averaging

Another feature of the QPEG that seeks to improve the accuracy of the predictions for long thrust arcs is the gravity averaging method. Instead of evaluating ω and κ at the start of each segment, they are evaluated in the middle of a coast arc with the same initial conditions. This mostly solves the over-estimation of the gravity force that would otherwise occur for ascent trajectories. In order to measure its effect a comparative analysis similar to Section 6.1.3 was set up with the same baseline configuration. The results can be observed in Figure 6.10, which

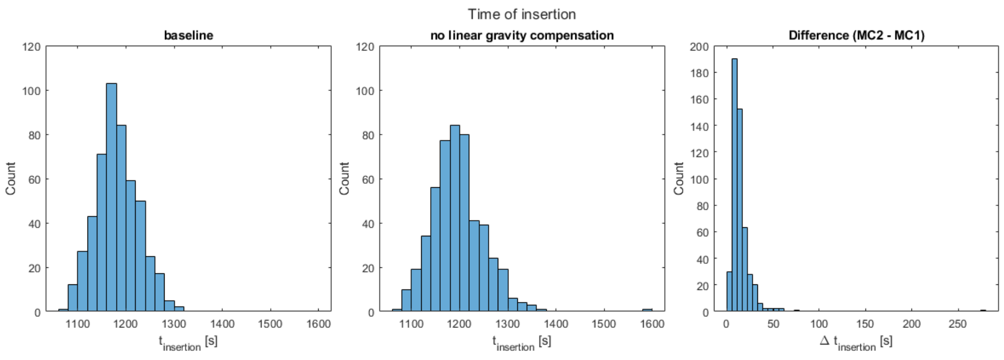


Figure 6.8: Comparison time to insertion linear gravity compensation

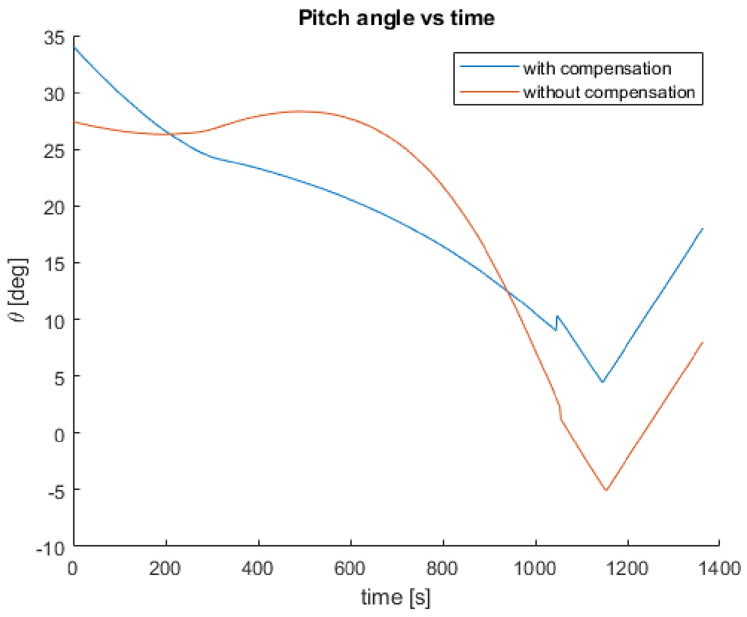


Figure 6.9: Comparison pitch angle linear gravity compensation

shows that the effect on the time to insertion is small but definitely positive. The benefits of gravity averaging are more substantial for the estimated time to insertion in case no constraint biasing is used, as shown in Figure 6.11. If constraint biasing is used there are still benefits to having the analytical solution be as close as possible to the numerical one, which will be demonstrated in the next section.

6.1.5. Constraint biasing

Biasing the constraint values with the difference between the analytical and numerical methods can in theory provide the accuracy of the numerical method while only evaluating it once per guidance cycle. However after an initial evaluation the difference in terms of burn-time proved to be negligible, as shown in Figure 6.12. This means that from a pure performance point of view the analytical propagation is sufficiently accurate for the resulting trajectories to be equally optimal to the numerical method. Furthermore the constraints bias can cause some additional convergence failures when the analytical and numerical solutions are too far apart. This effect can be seen in Figure 6.13, which shows the number of iterations on the second call to the QPEG, i.e. the first call with a non-zero bias vector. The number of cases where convergence fails is slightly increased compared to Figure 6.4b. While a failure to converge

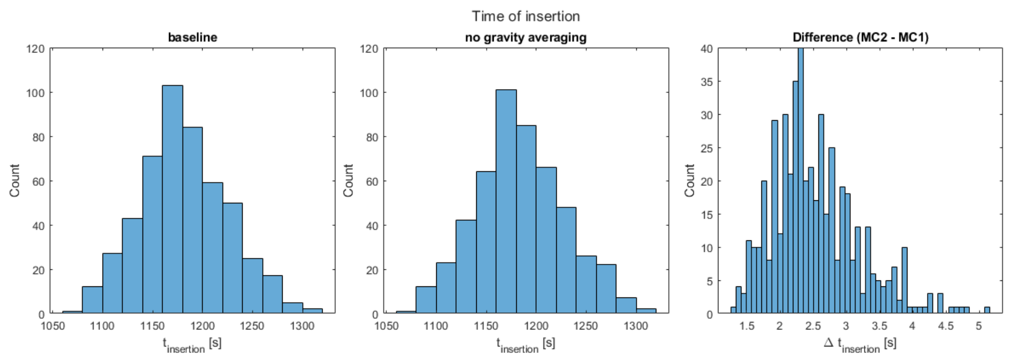


Figure 6.10: Comparison time to insertion gravity averaging

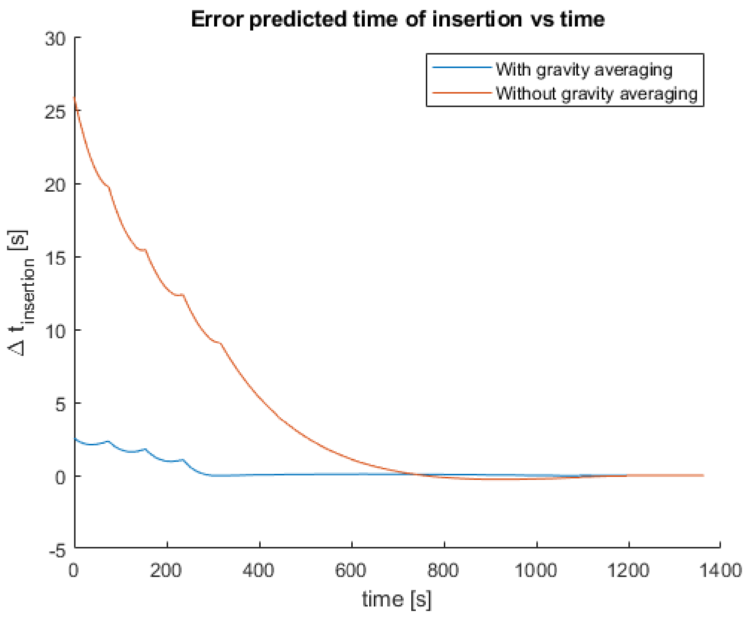


Figure 6.11: Comparison predicted time of insertion error gravity averaging

on the second call has less impact, since the previously found solution can be re-used, it is still undesirable. After each non-convergence the bias is reset, which means that QPEG will alternate between converging without the bias and not converging with the bias until the bias becomes sufficiently small.

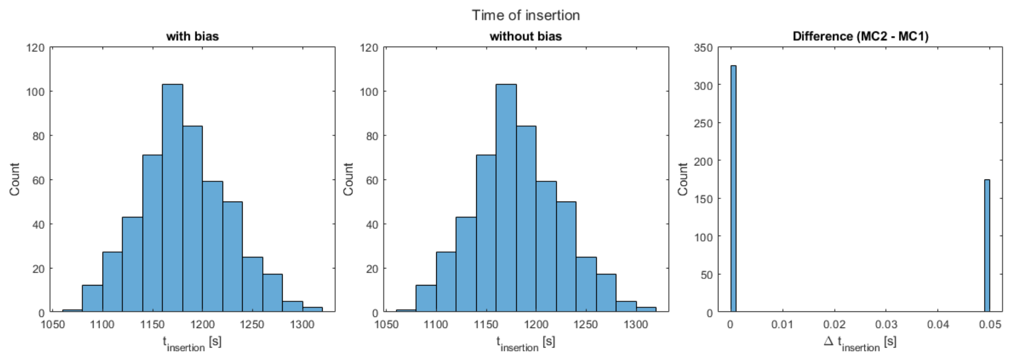


Figure 6.12: Comparison time to insertion constraint biasing

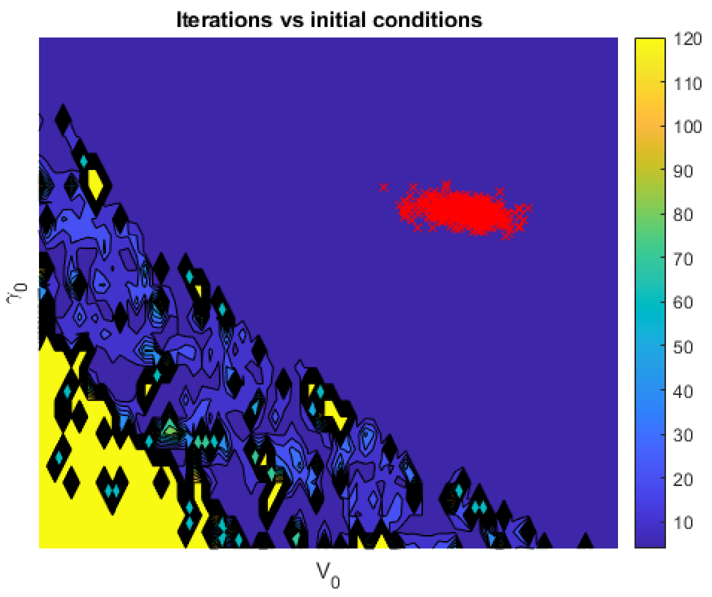


Figure 6.13: Bias values with fixed segment lengths

It can also be interesting to analyze the values of the biases themselves and their behaviour over time. This is shown in Figure 6.14 for the analytical propagation method described in Algorithm 1. One of the first features that we observe is the step-wise nature with which they converge. This is due to the segments of the analytical propagation having a fixed length except for the last segment. The biases are a result of the total propagation of all the segments on an arc, with the error of each segment being related to its size. Therefore as t_{go} becomes smaller only the last segment shrinks in size and only the error for that segment decreases. Once the last segment becomes much smaller than the others the propagation error will reach a plateau. It isn't until the segment count is reduced by one and the next segment starts shrinking that the error again starts to decrease. This makes the global convergence behaviour just slightly better than linear. Nevertheless both the biases associated with the continuity constraints at Second-stage Engine Cutoff (SECO) and the terminal constraints converge to zero. The jump in bias values towards the end of the third stage is due to the release of the position constraint with biases not being used for the new constraint set. As will be shown in Section 6.4, the slow convergence of the biases can cause issues towards the end of the flight where the bias values can start oscillating from one guidance cycle to the next. This can eventually cause the QPEG to converge to incorrect solutions or fail to converge all-together. In an attempt to resolve this issue the segmentation algorithm was revised to enforce a minimum number of segments in addition to the maximum segment length. This has the effect that once the number of segments reaches this minimum the biases start to converge to zero much more rapidly, as shown in Figure 6.15, since all segments are being reduced in size simultaneously.

Another interesting aspect of the bias values is that they show in which part of the state vector and constraints the analytical propagator introduces the most error. It can be seen that for the second stage biases the error in $\mathbf{p}_r$ is by far the largest followed by $\mathbf{V}$ and $\mathbf{p}_V$. The biases for r on the other hand are much smaller, likely due to the short integration time. For the terminal constraints it is the flight-path angle constraint that has the largest error, followed by the velocity magnitude constraint and free down-range distance transversality condition. Due to the longer integration time the altitude constraint also has a larger error. The two constraints related to the orbital plane orientation are much smaller, which is likely caused by the relatively small yaw angles.

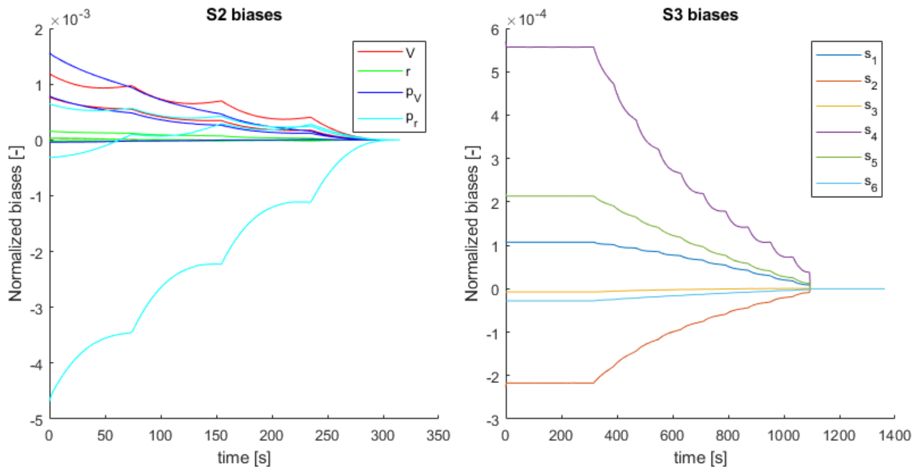


Figure 6.14: Bias values with fixed segment lengths

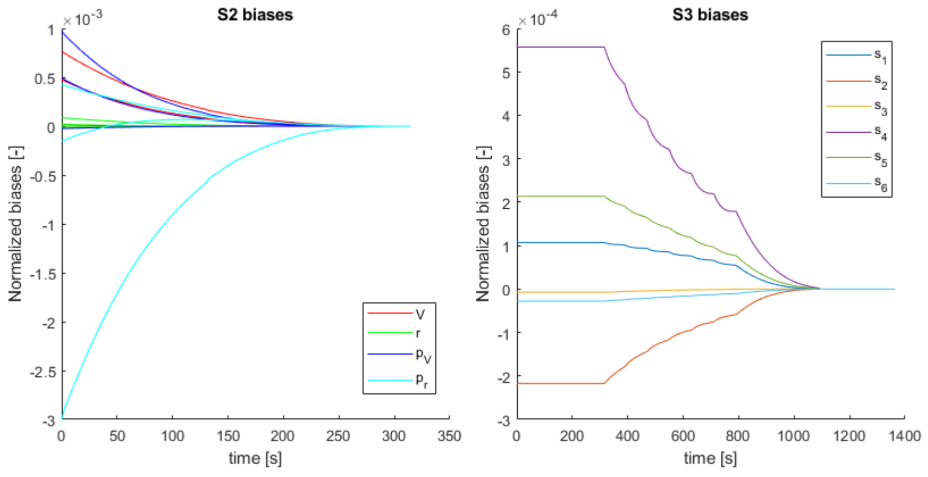


Figure 6.15: Bias values with variable segment lengths

6.2. Parameter estimation

This section will focus on the performance of the vehicle parameter estimator under ideal conditions. To this end it will first be tested using synthetically re-created acceleration measurements. In these tests only measurement noise and a limited number of disturbances will be used. After that it will be integrated with the QPEG to evaluate the impact of the improved vehicle parameters accuracy on the performance of the guidance.

6.2.1. Estimation of τ and v_{ex}

The initial version of the vehicle parameter estimator only directly provides estimates for v_{ex} and τ. However the states used in the Kalman filter are $\frac{\tau}{v_{ex}}$ and $\frac{1}{v_{ex}}$. These two quantities are not of much physical importance themselves, but they can readily be transformed into τ and v_{ex} which are then used by the closed-loop guidance. This non-linear transformation does however mean that there is no exact solution for obtaining the uncertainty matrix of v_{ex} and τ.

It can however be approximated using the first-order approximation given by Equation (6.1), where $\mathbf{P}_x$ is the state uncertainty of the filter and $\mathbf{P}_y$ is the uncertainty of the original vehicle parameters.

$$\mathbf{P}_y = \begin{bmatrix} \frac{1}{x_2} & -\frac{x_1}{x_2^2} \\ 0 & -\frac{1}{x_2^2} \end{bmatrix} \mathbf{P}_x \tag{6.1}$$

First we will look at the performance of the filter under somewhat ideal conditions replicating the second stage flight. That is the only noise added to the acceleration signal is the measurement noise and simplified aerodynamic drag. For this test the parameters τ and v_{ex} were varied each time with standard deviations of $10s$ and $1m/s$ respectively, which are somewhat representative of the uncertainty of these parameters. The Kalman filter was initialized with an equivalent initial uncertainty on the state. The results for the estimation error and uncertainty over time are shown in Figures 6.16 and 6.17. As can be seen the error on τ gets reduced very rapidly by more than an order of magnitude after which it continues becoming smaller as the acceleration increases. On the other hand with the initial uncertainty on v_{ex} being much smaller the error stays mostly constant at the beginning and only starts decreasing once the error on τ becomes sufficiently small and the signal to noise ratio increases due to increasing acceleration. It does however eventually become much smaller than if the nominal value had been used, so including v_{ex} in the filter also has benefits for a nominal flight. The brief increase in the uncertainties and errors of τ around 50 s is caused by the fairing release and subsequent parameter hold. Finally it can also be seen that the errors stay consistent with the estimation uncertainty.

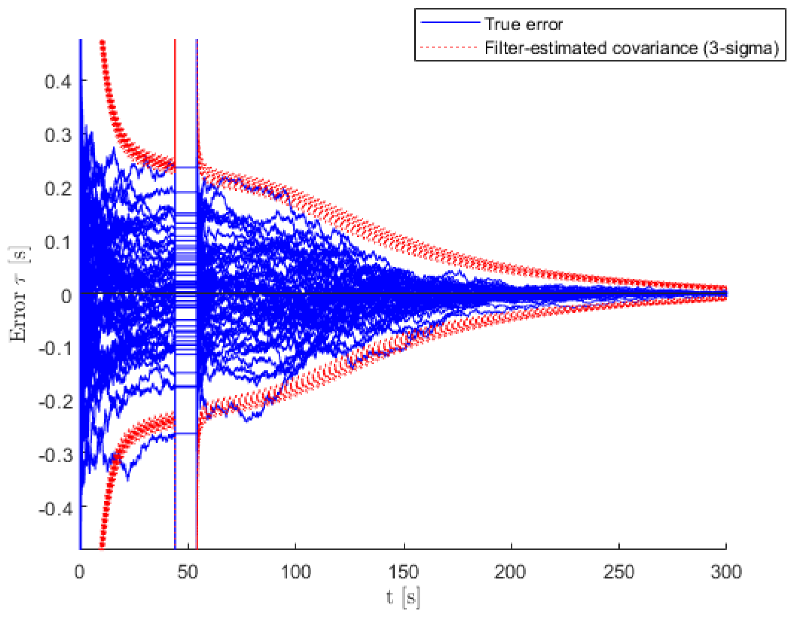


Figure 6.16: Estimation error τ under ideal conditions for stage two

In order to assess the effect of this simple estimation scheme on the performance of the closed-loop guidance two Monte-Carlo analyses were performed using nominal parameters and estimated parameters in the QPEG. The same method of using the same initial conditions and vehicle parameters was repeated. The effects on the burn-time margins are shown in Figure 6.18, were it can be seen that on average the use of the estimated parameters improved margins compared to using nominal parameters. The increase in performance can be quite

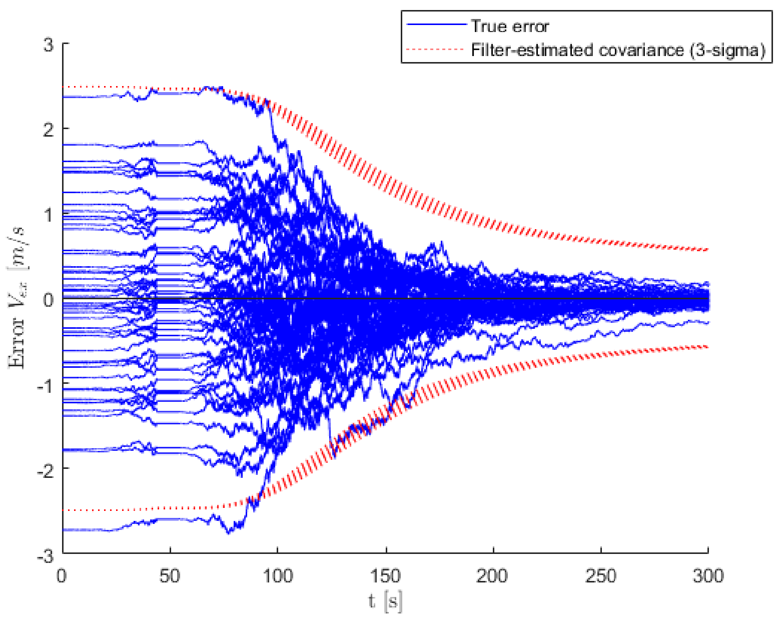


Figure 6.17: Estimation error v_{ex} under ideal conditions for stage two

substantial, however there are also some cases where the performance is decreased. Of the different vehicle parameters, the change in performance is most correlated to τ on the second stage as shown in Figure 6.19. As one would expect the largest performance improvements are obtained when τ is far from the nominal value. The cases with decreased performance occur when τ is just a bit off-nominal and are caused by incorrect estimates of the burn-time. This will be further discussed in Section 6.2.2.

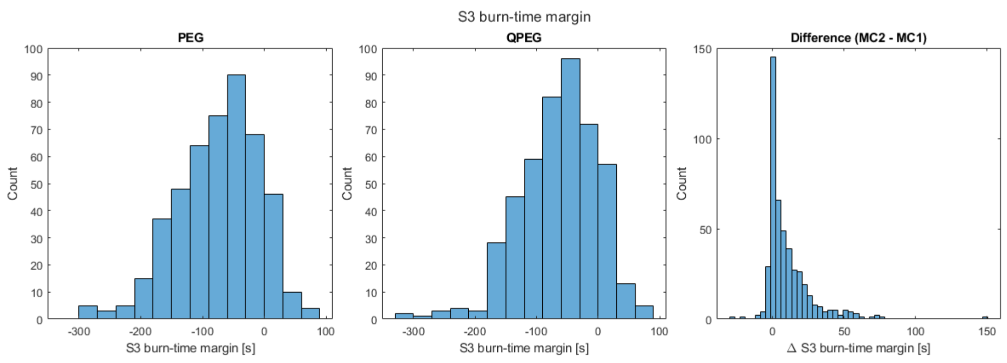


Figure 6.18: Comparison burn-time margin nominal and estimated parameters

6.2.2. Burn-time estimation

In addition to v_{ex} and τ, it is also necessary to provide the burn-time, t_b, for each phase to the closed-loop guidance. Since it isn't observable using just the sensed acceleration, it had to be estimated some other way. By exploiting the correlation between τ and t_b it is possible to obtain a reasonable estimate for t_b, as we can see in Figure 6.20. It shows the dispersion of t_b and τ for the second and third stages. The colors indicate whether the performance is improved (blue) or reduced (purple). This method is of course not perfect and the estimate of t_b will have an error that is an order of magnitude larger than the error for τ. There are however certain conditions in which using the nominal vehicle parameters yields better performance

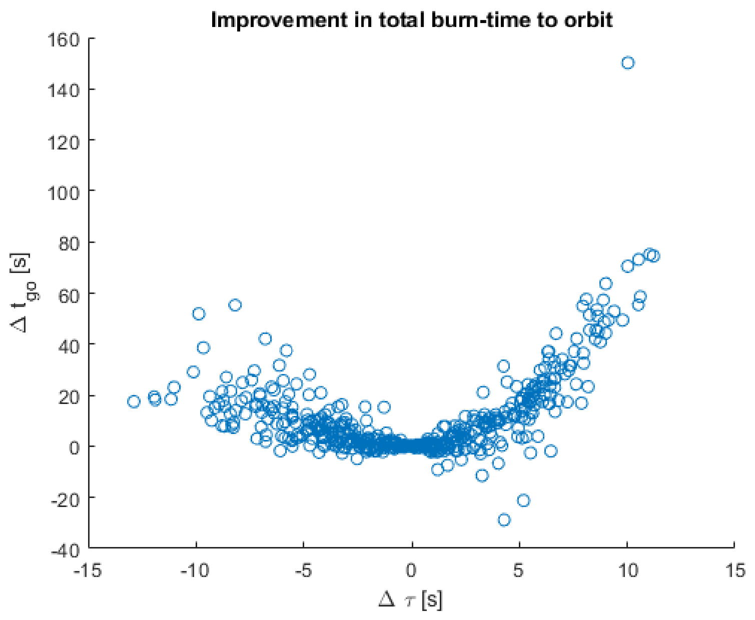


Figure 6.19: Burn-time improvement vs S2 τ dispersion

than using the estimated parameters. From Figure 6.20a it is clear that those cases are relatively close to the nominal vehicle, but where the ratio t_b/τ deviates from the nominal one. It is for those cases that an improved estimation method for t_b, possibly by using more sensors, could benefit the performance. Fortunately the fact that these cases are relatively close to the nominal vehicle means a sub-optimal algorithmic performance has less impact due to higher performance margins. From Figure 6.20b it is clear that that the correct estimation of the stage 3 parameters shows little correlation with the overall performance improvement, which is to be expected since at that point in the flight the trajectory is already largely fixed and the burn-time of the last stage isn't used by the QPEG.

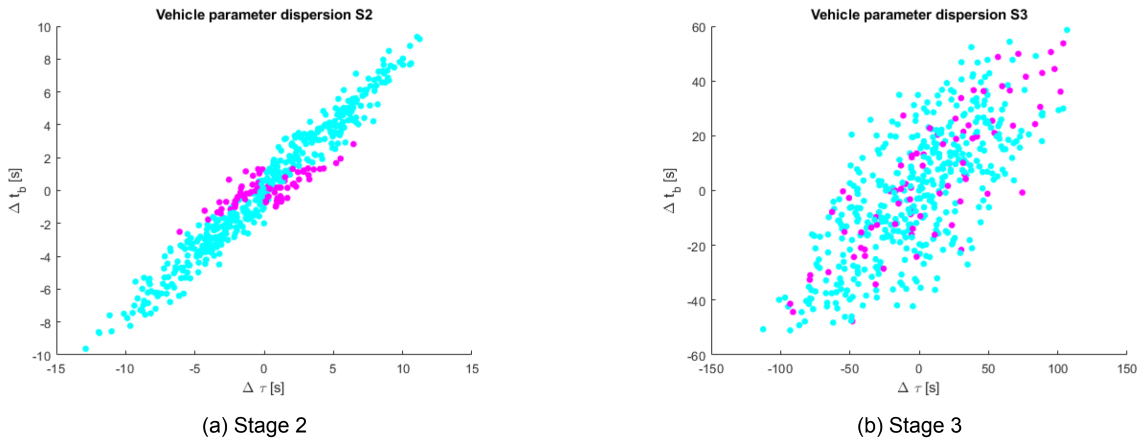


Figure 6.20: Dispersion of τ an t_b at ignition classified by performance

As shown before, the vehicle parameter estimation using just acceleration measurement is far from perfect and can lead to substantial errors on the estimated stage burn-times. This in turn can cause the trajectories to be sub-optimal thereby reducing performance. In order to quantify this performance loss, which is equivalent to the performance that could be gained by improving the parameter estimation, the performance delta was computed between using estimated parameters and exact parameters. The ideal estimator used for this is the same that was already used throughout Section 6.1. The results of this analysis are shown in Figure 6.21. In the overwhelming majority of cases the performance difference is negligible and it is clear that the current rudimentary parameter estimation already reduces the performance loss by an order of magnitude. Of course effects that have been neglected thus far, such as non-constant mass-flow and imperfect Oxidizer to Fuel ratio (O/F) control, could still cause this performance loss to increase.

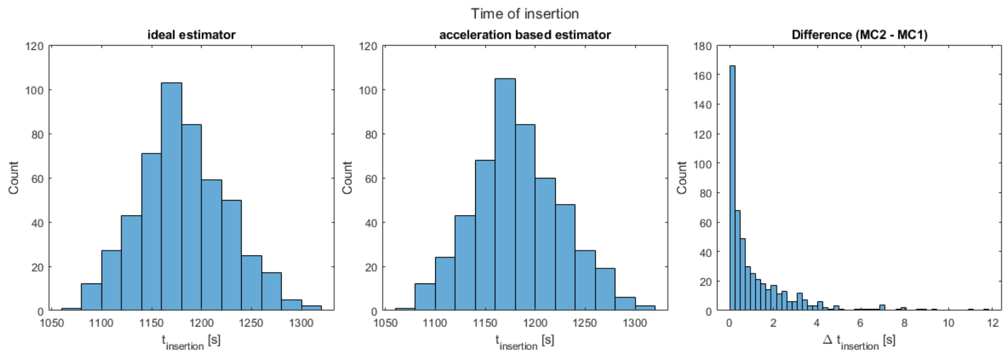


Figure 6.21: Comparison time to insertion estimated and ideal parameters

6.3. Insertion into alternate orbits

All three improvements to the guidance system developed in this thesis will be combined to assess their effect on the probability of reaching the desired orbit. The main focus will lie on the last feature, which is the target selection based on the output of the QPEG and parameter estimation. First a preliminary analysis will show the impact of not using any alternate target orbits and how much could be gained in terms of orbit altitude if the maximum feasible target was always selected. Then the in-flight target selection algorithm will be tested using various combinations of target orbits and desired margins. The chosen combinations in this thesis are largely arbitrary and mostly serve to illustrate the functionality of the algorithm in different configurations.

6.3.1. Preliminary analysis

Some of the reasons for changing the target orbit mid-flight were already listed in Section 5.4. Preemptively lowering the target orbit altitude could benefit overall mission success. In order to further illustrate this, some preliminary analyses were performed on the behavior without orbit re-targeting and the maximum feasible orbit altitudes that could be reached if the guidance had perfect knowledge of the vehicle state and parameters.

The primary issue with sticking to the nominal target orbit when there isn't enough performance to reach it is the disproportionate impact on the periapsis altitude. The reason for this is shown in Figure 6.22, where it can be seen that the apoapsis altitude is already at the target value mid-way through the third stage flight, so in the last part of the burn almost all the energy goes towards raising the periapsis. This means that an under-performing vehicle, where the engine cuts off before orbit insertion, will essentially perform an insertion at the apoapsis of an elliptical orbit. Insertion at apoapsis is usually less efficient than an insertion in the same orbit at periapsis. This can be seen in Figure 6.23 which shows the burn-time margins for different target orbits with a nominal vehicle and identical initial conditions at the start of the second stage. It can also be seen that if the goal is to maximize the periapsis altitude, which minimizes orbital decay, then an insertion into a lower circular orbit should be preferred. Not only does it minimize the loss in periapsis altitude, but it also results in a larger orbital energy. The decision to lower the target orbit should however be taken as early in the flight as possible in order to reduce losses. As is also shown in Figure 6.22 the apoapsis altitude at the beginning of the third stage is already close to the target orbit altitude, so trying to insert into a much lower altitude will incur large steering losses.

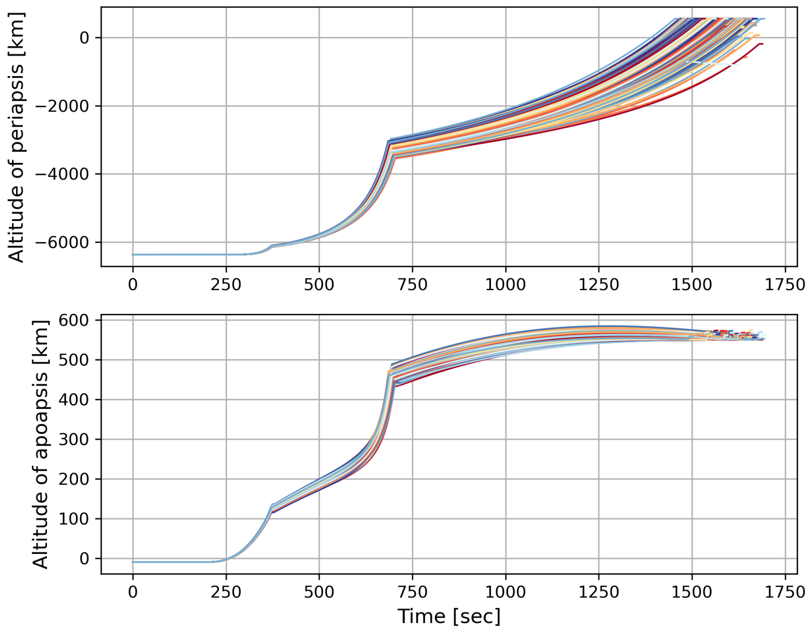


Figure 6.22: Instantaneous apoapsis and periapsis during flight

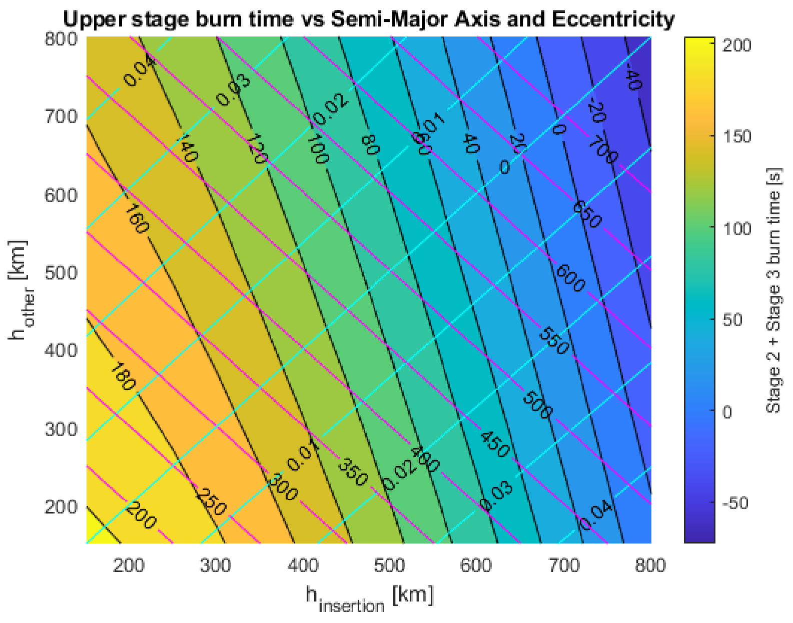


Figure 6.23: Burn-time margin vs target orbit

6.3.2. Baseline scenario

In the baseline scenario for these analyses it was decided to use a circular orbit at an altitude of $700\,km$ as the target. The reason for this increased target altitude compared to the previously described nominal mission is that the nominal performance margin was large enough to enable the rocket to successfully insert in almost all the cases. This would have made the alternate targets largely useless. In order to make the analyses more interesting the margins had to be reduced. While this could have been achieved by using a lower performing first stage, which would be the case with engine outs, the higher target altitude was chosen instead as it would allow for a larger range of alternate targets. The resulting periapsis altitudes are shown in Figure 6.24, where it can be seen that a successful insertion is possible in a majority of cases but for some cases the periapsis altitude is negative.

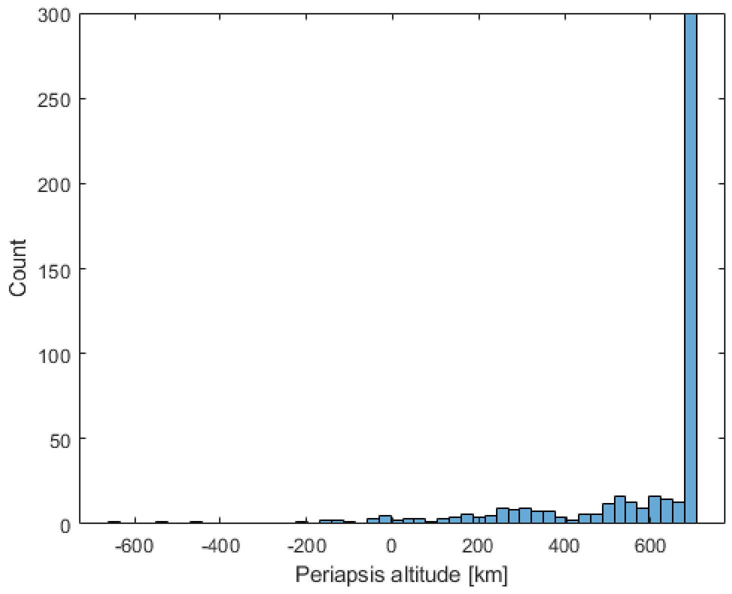


Figure 6.24: Periapsis altitude dispersion without orbit re-targeting - 700 km target altitude

In order to set reasonable expectations for what the guidance shall achieve, the maximum feasible orbit altitude was also computed. As shown in Figure 6.25 it should be possible to reach a stable orbit in every case. This scenario represents a guidance that has perfect knowledge of all vehicle parameters and it is therefore the best that could be achieved using the orbit re-targeting algorithm.

6.3.3. Single alternate orbit

For the first test case of the orbit re-targeting there will only be a single alternate orbit. Given that the nominal target is already a low Earth orbit, there isn't much margin to gain by lowering the target altitude. So the alternate target was set to a circular orbit with an altitude of 200 km as a sort of bare minimum to stay orbital for a short amount of time. For this scenario the required burn-time margins during the second and third stage were set to 60 s and 30 s respectively to account for the increased uncertainty on total vehicle performance during the second stage.

The results for the periapsis altitude can be seen in Figure 6.26. It is clear that the rocket is now able to successfully insert into a circular orbit in all cases. However whereas previously in the majority of the cases the rocket was able to insert into a 700 km orbit, now in most cases the rocket inserts into the minimal 200 km orbit. So the increased probability of reaching the

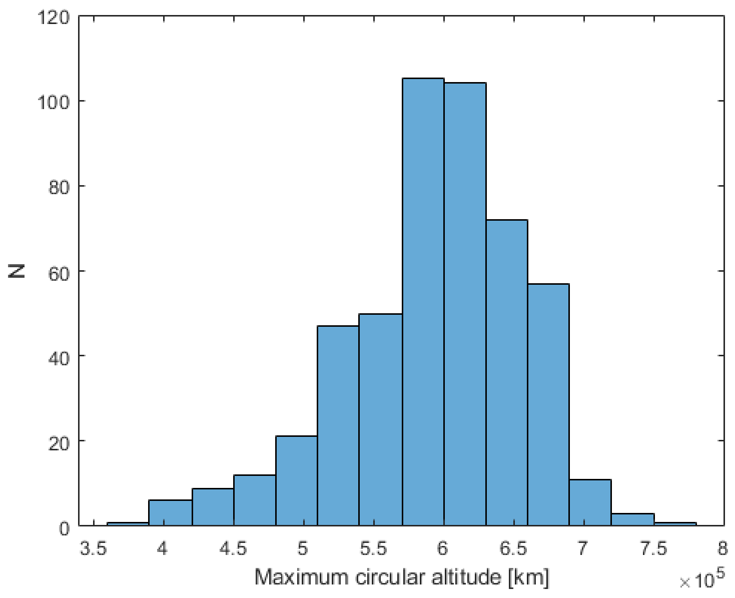


Figure 6.25: Maximum feasible circular orbit altitude

minimum target came at the cost of a decreased probability of reaching the nominal target. This ratio however is not a direct consequence of the general method, but rather the result of the somewhat conservative margins that were chosen. If insertion into a circular orbit is less critical and it is just the periapsis altitude that matters, then the required margins could be relaxed a bit. As shown in Figure 6.27, the estimated margins at insertion are indeed almost always above the required minimum. The couple of cases where it is below the required minimum are caused by either an over-estimated stage two burn-time or lower stage three performance. In those cases the guidance would target the nominal orbit during the second stage flight and after second stage separation the cost of re-targeting to the lower orbit would be higher then continuing to the nominal one. From this it is clear that the logic that a lower altitude orbit requires less propellant is not universally true throughout the flight. For the guidance to be able to re-target at the start of stage three the alternate orbits need to be sufficiently similar to previously targeted orbit. Despite all that the true margins were still positive for all cases. Given the large amount of remaining propellant the third stage might even be able to subsequently raise its orbit back to the nominal one after initial insertion. However given that re-ignition of the third stage engine is not currently considered for the first flight of RFA ONE, this option will not be explored further in this thesis.

6.3.4. Multiple alternate orbits

In an effort to alleviate some of the limitations that were identified previously, the next test will include a larger amount of target orbits. The nominal orbit and minimum orbits are still set to 700 km and 200 km respectively, but intermediate orbits at an interval of 100 km have now been added bringing the total to six target orbits. The required margins were kept the same as for the previous test. In reality the different alternate target orbits would not necessarily be spaced out equally, but would likely be chosen to minimize the probability of interference with other space objects. The final selection will result from the work performed by the mission analysis department and falls outside of the scope of this thesis.

The results for this test are presented in Figure 6.28 in the same format as the previous test. It is immediately clear that the rocket is able to achieve insertions into higher altitude

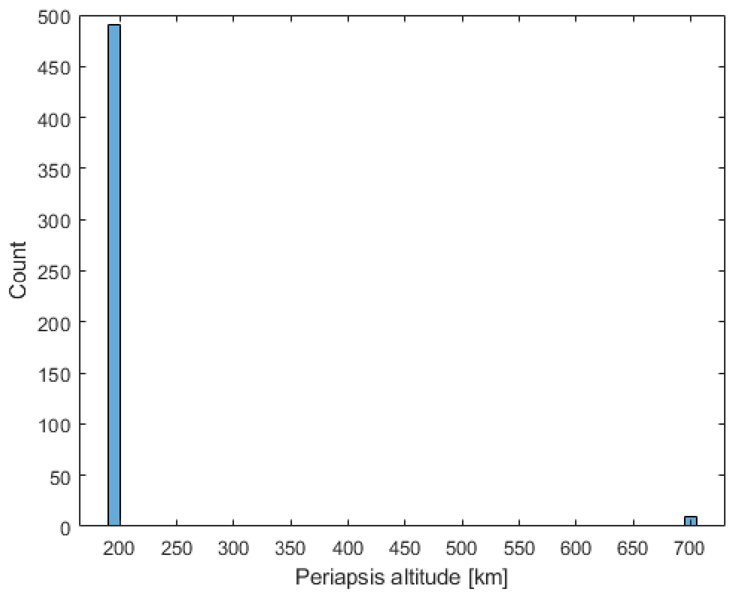


Figure 6.26: Periapsis altitude dispersion with a single alternate orbit

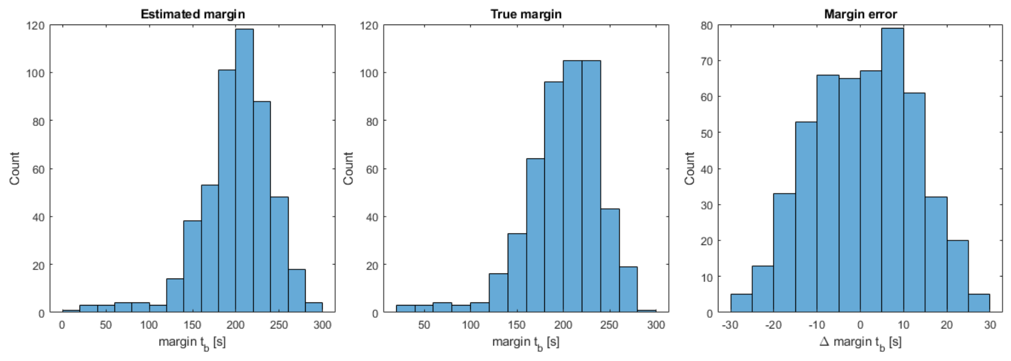


Figure 6.27: Burn-time margins with a single alternate orbit

orbits than the minimum, with the peak now being at 600 km. Even the number of insertions into the nominal orbit has increased, which is likely due to the guidance initially targeting one of the higher intermediate orbits during the second stage and then switching to the nominal target at the start of the third stage. All these improvements are achieved while still maintaining a 100 % success rate for insertions into circular orbits. In terms of margins, shown in Figure 6.29, the overall spread has decreased somewhat but it remains significantly larger than the increase in burn-time required to go into the next higher orbit, as determined using Figure 6.23. Furthermore there are still some cases where the estimated margin is below the imposed minimum. Solving this issue might require increasing the margin during the second stage or providing a finer mesh of target orbit altitudes.

A potential limiting factor for the number of targets is that the maximum number of iterations per guidance cycle can become quite large. Currently the target search always starts at the top of the list and is all done within a single guidance cycle. In case the guidance was targeting a lower altitude, a target search during the third stage will likely cause convergence failures of the QPEG for multiple targets before finding one that is feasible.

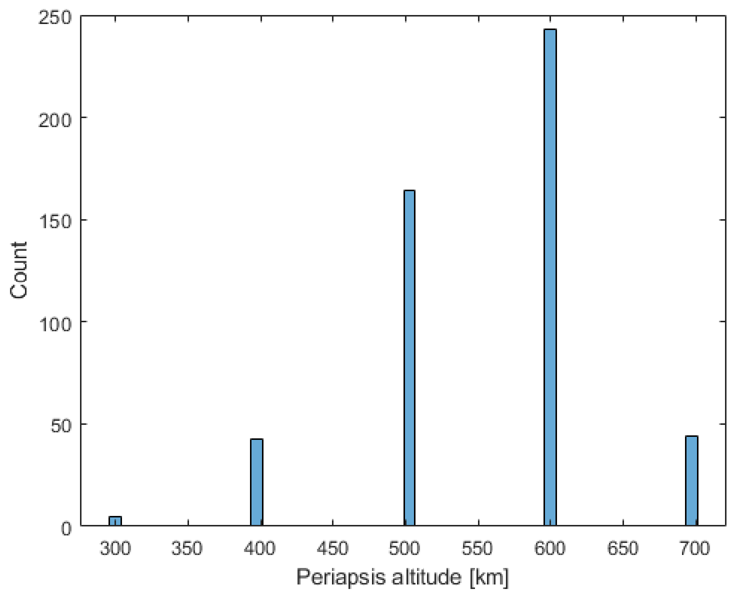


Figure 6.28: Periapsis altitude dispersion with multiple alternate orbits

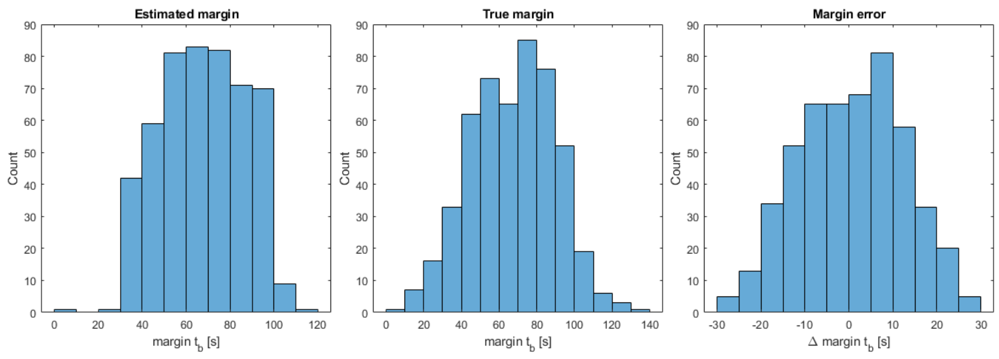


Figure 6.29: Burn-time margins with multiple alternate orbits

6.3.5. Reduced S2 margins

In the following test the impact of the margin limits was investigated. Until now the required margin was set to a higher value during the second stage than during the third stage under the assumption that this could help mitigate the increased uncertainty on the remaining vehicle performance. In this test however the required minimum margin will be set to a constant value of 30 s, which will allow the guidance to target higher orbits during the second stage flight. The targets remain unchanged compared to the previous test.

The effect of this change is that the probability of reaching higher orbits is greatly increased, as shown in Figure 6.30, with twice as many simulations reaching the nominal orbit. By allowing for a more opportunistic target selection during the second stage, the rocket will likely be closer to the optimal trajectory at SECO allowing for more of them to insert into the maximum feasible orbit. By looking at the margin in Figure 6.31 it can be seen that guidance more effectively uses the available propellant as the maximum available margin at insertion is reduced a bit. While it could be expected that the reduced required margins during the second stage could lead to cases where the guidance initially aims for a too high an orbit, eventually leading to failed insertion into any circular orbit, this isn't actually the case. On the contrary the minimum margins don't seem to have significantly changed compared to Figure 6.29. It can therefore be concluded that an optimal selection of the required margins can significantly improve performance with little observable negative effects.

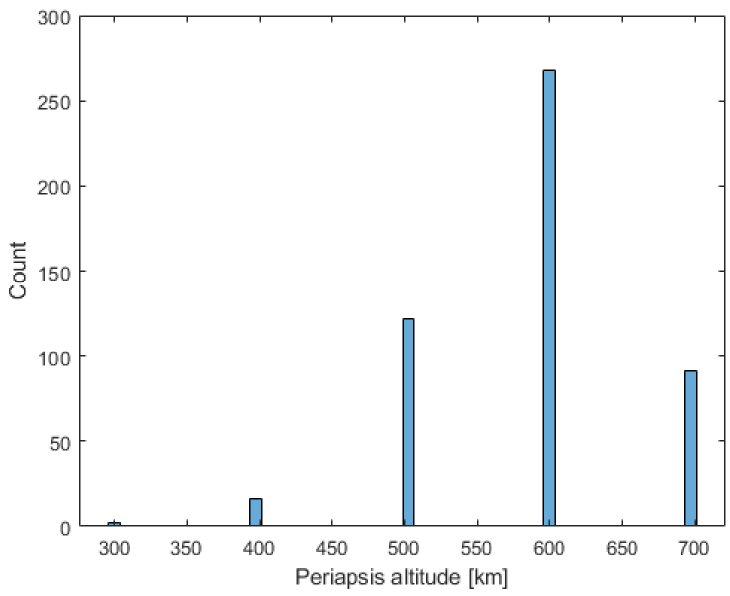


Figure 6.30: Periapsis altitude dispersion with multiple alternate orbits and reduced margins S2

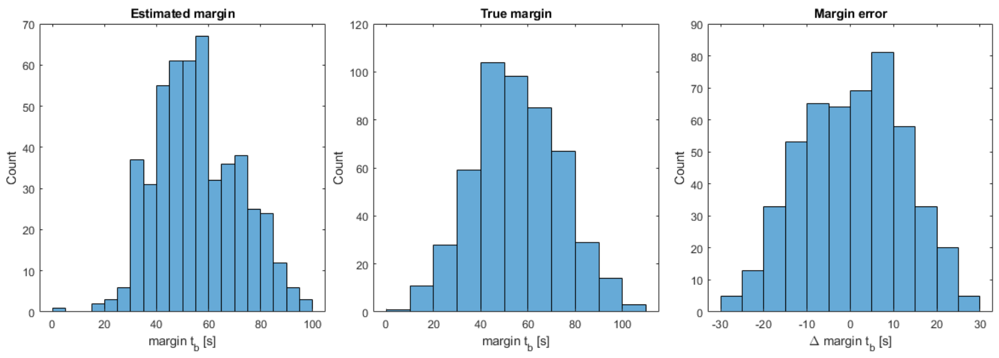


Figure 6.31: Burn-time margins with multiple alternate orbits and reduced required margins S2

6.3.6. Zero margins

As was demonstrated in the previous section, it is possible to achieve higher altitudes by lowering the required minimum margins. However previously the margins were still set such that the rocket would always insert into a circular orbit. If the mandatory requirement of achieving a circular orbit is relaxed, the margins could be reduced even more. In this analysis the required margins were set to zero for all phases. The resulting periapsis dispersion is shown in Figure 6.32, where it can be seen that the number of simulations where the rocket inserts into the nominal orbit has been greatly increased and is close to the theoretical maximum. The minimum achieved periapsis altitude has also been increased compared to the previous configurations. However as expected this comes at the cost of the final orbit being slightly eccentric in a small amount of cases, which is also evident from the negative margins shown in Figure 6.33. Nevertheless, using minimal required margins seems to be best way to maximize the periapsis altitude.

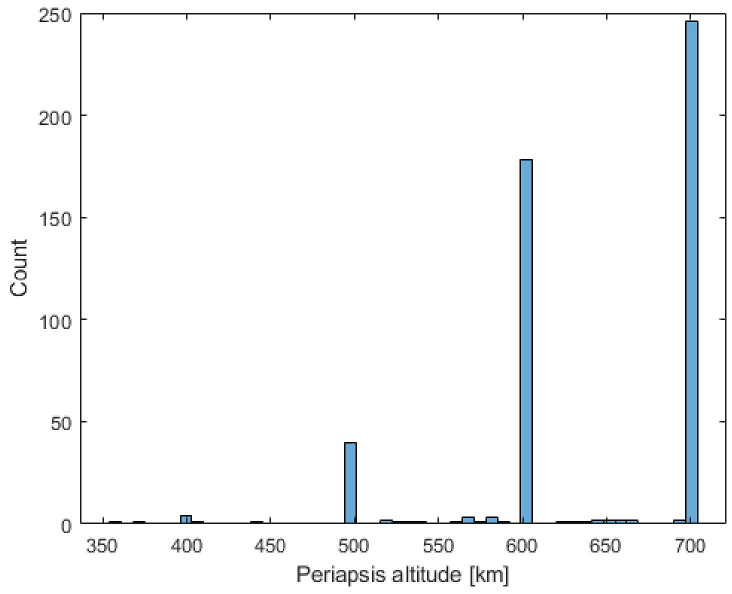


Figure 6.32: Periapsis altitude dispersion with multiple alternate orbits and zero margins

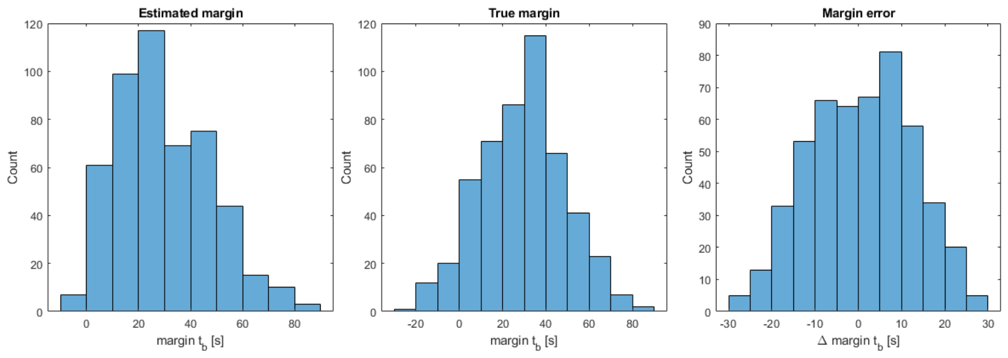


Figure 6.33: Burn-time margins with multiple alternate orbits and zero margins

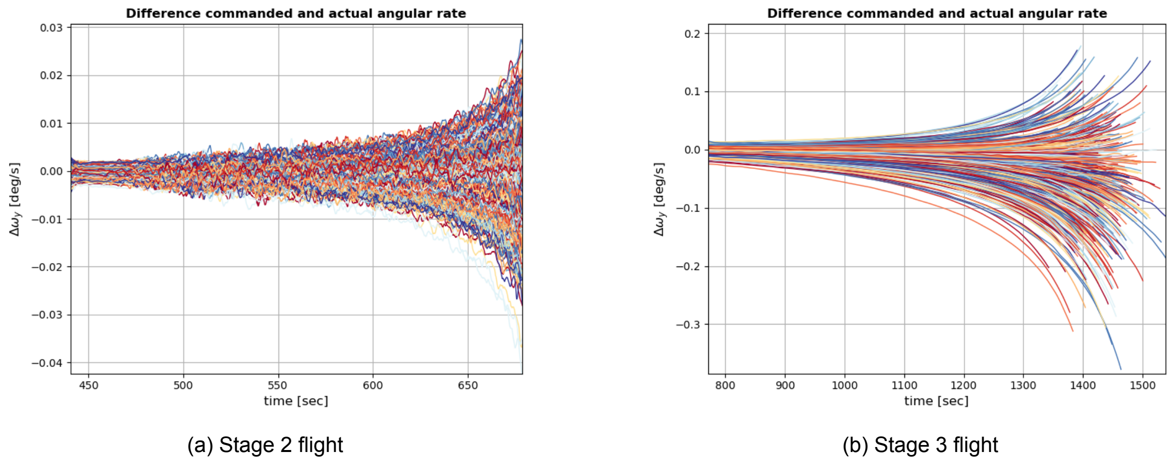
6.4. Integrated system performance

All of the results that have been shown so far were obtained by running separate components of the guidance or a combinations of them in a simplified 3-DOF environments. While this allowed for more specific and rapid testing, the performance shown so far are likely better than if it was running in a real-world environment. Though the ultimate test will be the first launch of RFA ONE, the GNC department has developed several tools that allow them to verify the performance of the GNC system. The most important tool is the Functional Engineering Simulator (FES) which combines the complete GNC software with a highly detailed 6-DOF plant model. For the purposes of the guidance this means it will be submitted to a wide range of perturbations representative of the real flight. The drawbacks are that with such a complex model the run-times are much larger, more than an order of magnitude compared to the 3-DOF environment, and the conditions that the guidance is subjected to are more difficult to control. While it is possible to disable some of the dynamics models and sensor noises, the overall control remains limited. Furthermore due to the many different closed-loop systems acting in concert it can often be difficult to isolate the source of a specific issue. Another drawback is that the possibilities for interactive debugging are much more limited and internal states are only available if they are included in the guidance output. Due to time-constraints neither the J_2

gravity model nor the variable segment lengths were implemented in the version of the QPEG used for the following tests.

6.4.1. Command consistency

The closed-loop guidance has two outputs through which it commands the vehicle attitude. These are the thrust direction, which will define the commanded pitch and yaw angles, and the angular rates, which can be used as a feed-forward in the steering filter or controller. Since they both get computed from the same guidance parameters they should in theory be consistent, that is the derivative of the thrust direction between guidance cycles should equal the commanded angular rate. In practice however the closed-loop guidance will continuously adjust the guidance parameters in order to correct for disturbances. So any difference between the predicted and real dynamics will lead to inconsistencies in the two guidance commands. An inconsistent input to the controller could make it less stable. Fortunately these inconsistencies do get filtered out by the steering filter, so the controller isn't directly affected by them. The magnitude of the inconsistencies depends a lot on the type and magnitude of the disturbance. Figure 6.34 shows the difference between the input and output angular rates of the steering filter in the FES which in a sense gives a measure of the cumulative effect of all unmodeled dynamics and poorly estimated parameters. The different colors in this graph and other graphs from the FES only serve to more easily identify individual lines. Clearly the angular rate difference has a tendency to grow throughout the flight and reaches a maximum at orbit insertion. This is likely due to the fact that errors early in the flight require much smaller corrections as the effect can be integrated over a longer period of time. It should however be noted that there is little correlation between command differences of the two stages, which is somewhat expected as the disturbances are likely caused in part by the stage construction. It should also be noted that during the third stage there is a bias towards a negative difference, which means that the actual angular rate systematically increases more rapidly than the commanded one. This hints at a systematic error in the modelled dynamics. Some of the disturbances that lead to theses inconsistencies will be analyzed in more detail in Section 8.2



(a) Stage 2 flight (b) Stage 3 flight

Figure 6.34: Difference between commanded angular rates and derivative of commanded thrust direction

6.4.2. Monte-Carlo results

The first batch of 500 Monte-Carlo simulations was done using the QPEG with parameter estimation but without orbit re-targeting. It would therefore only target a single circular target orbit of 550 km, similar to what the PEG previously did.

The results for the orbital altitudes, inclination, and longitude of the ascending node (LAN) are shown in Figures 6.35 to 6.37 respectively. The most striking result when comparing to those shown in Section 2.2 is the large decrease in the number of simulations where the rocket fails to reach as stable orbit. While it was expected that there would be some improvement, as was shown in previous section, the magnitude of the improvement far exceeds that observed in the 3-DOF environment. Upon further inspection it was discovered that the improved performance can be largely attributed to a reduction in the dispersion of the residual propellants on the second stage. This was caused by the parameter estimation providing more accurate estimates of t_b to the GNC manager, which improved the timing of the engine cut-off commands. For the orbital inclination the accuracy is comparable to what was previously obtained with the PEG. The vast majority of cases end up well inside the required accuracy bounds, with the few cases outside the bounds correlated to the ones where propellant was depleted before orbit insertion. The LAN on the other hand shows a much larger dispersion compared to the PEG. This is caused by the fact that the QPEG doesn't constrain the LAN at all. While the PEG didn't have any explicit constraints on the LAN either, it has to internally choose some value in order to construct the orbit plane normal $\hat{\mathbf{i}}_y$. This value was chosen based on the position at MECO, which has relatively little dispersion. By allowing this wider dispersion the QPEG is wasting less propellant in trying to correct it.

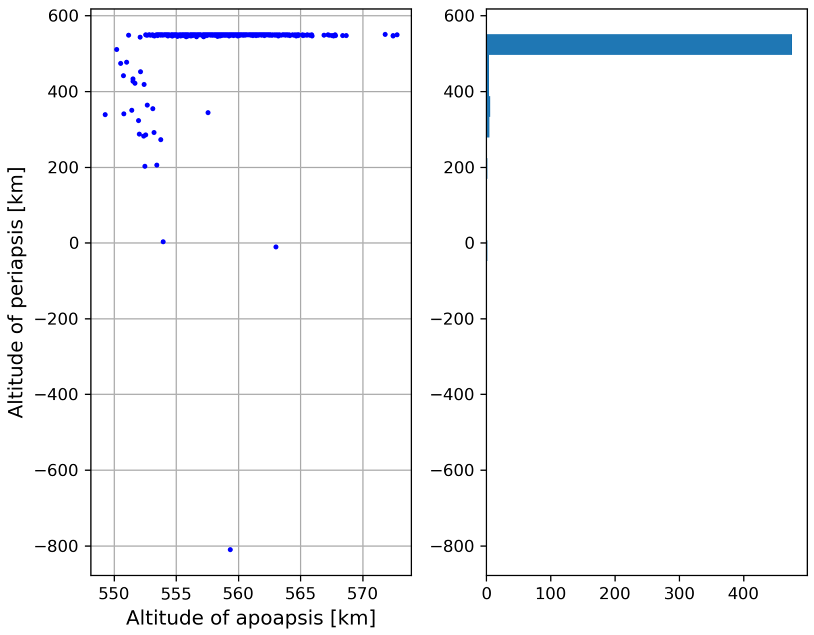


Figure 6.35: Apoapsis and periapsis dispersion single target orbit in the FES

The last point of interest is that there seems to be a bias towards slightly over-shooting the target orbit altitude for the apoapsis. This same bias can also be seen when looking at the semi-major axis for the simulations with successful insertions, shown in Figure 6.38. Given that the altitude at insertion is actually correct, it means that the velocity at insertion is too high which would be caused by issuing the engine shut-off command too late. Three factors where found to contribute to this:

- The engine cut-off command issued by the guidance came a bit too late
- There is a delay between the engine cut-off command being sent and the engine starting to shut-down during which it continues to produce full thrust
- The residual thrust during the shut-down transient adds a little bit of ΔV

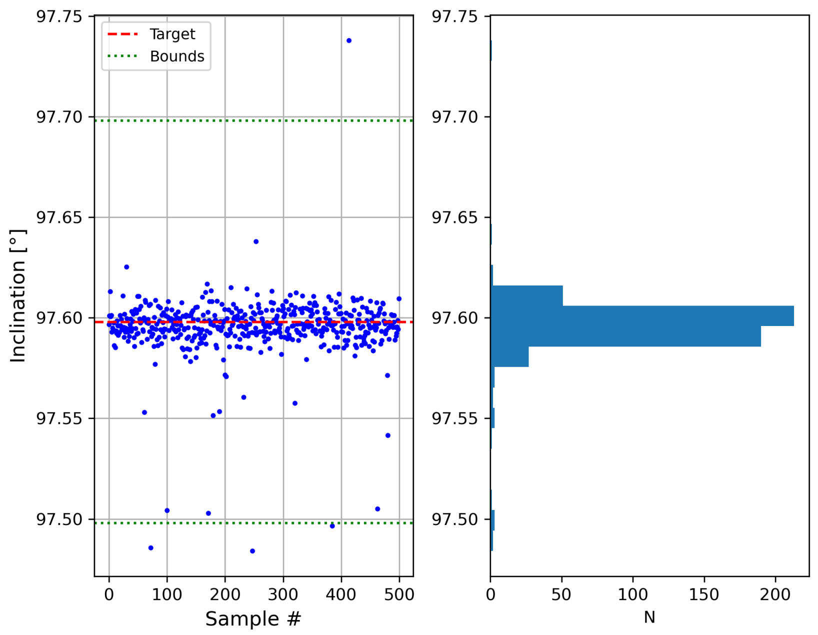


Figure 6.36: Orbital inclination dispersion in the FES

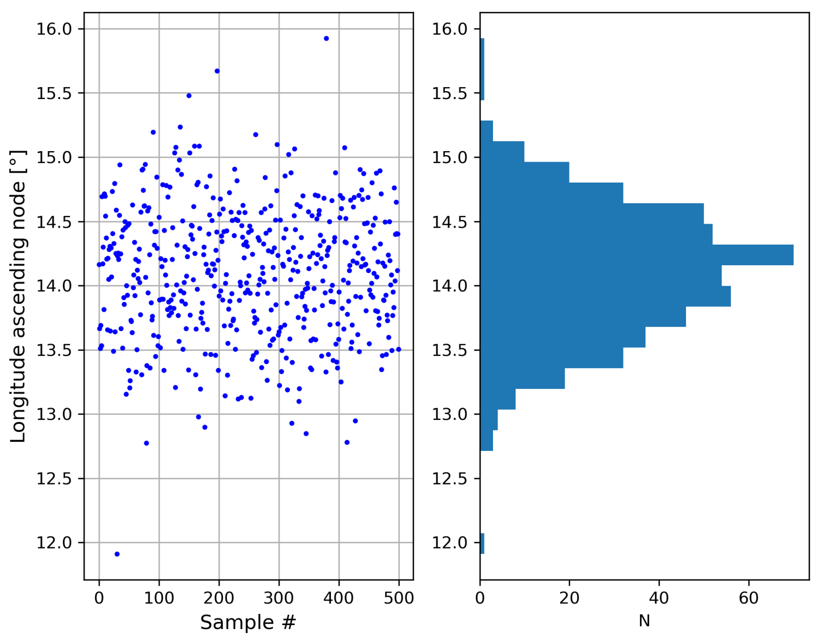


Figure 6.37: Dispersion longitude of the ascending node in the FES

The first point is caused by a combination of stopping to update the QPEG in the last ten seconds before insertion to avoid instabilities and model inaccuracies in the propagation, which lead to a small error in t_{go}. This could be solved by sending the engine shut-off command based on the current orbital energy and its time rate of change instead of t_{go} as described by Dukeman [13]. The second and third point could be addressed in the guidance by computing the average impulse generated during the engine shut-down delay and transient. By biasing the time of engine cut-off it can be ensured that the velocity after the transient is at the target value. While there would still be some dispersion due to uncertainties in these delays, the mean semi-major axis should at least be closer to the target value. It should also be noted that the delay used in this simulation had an exaggerated uncertainty.

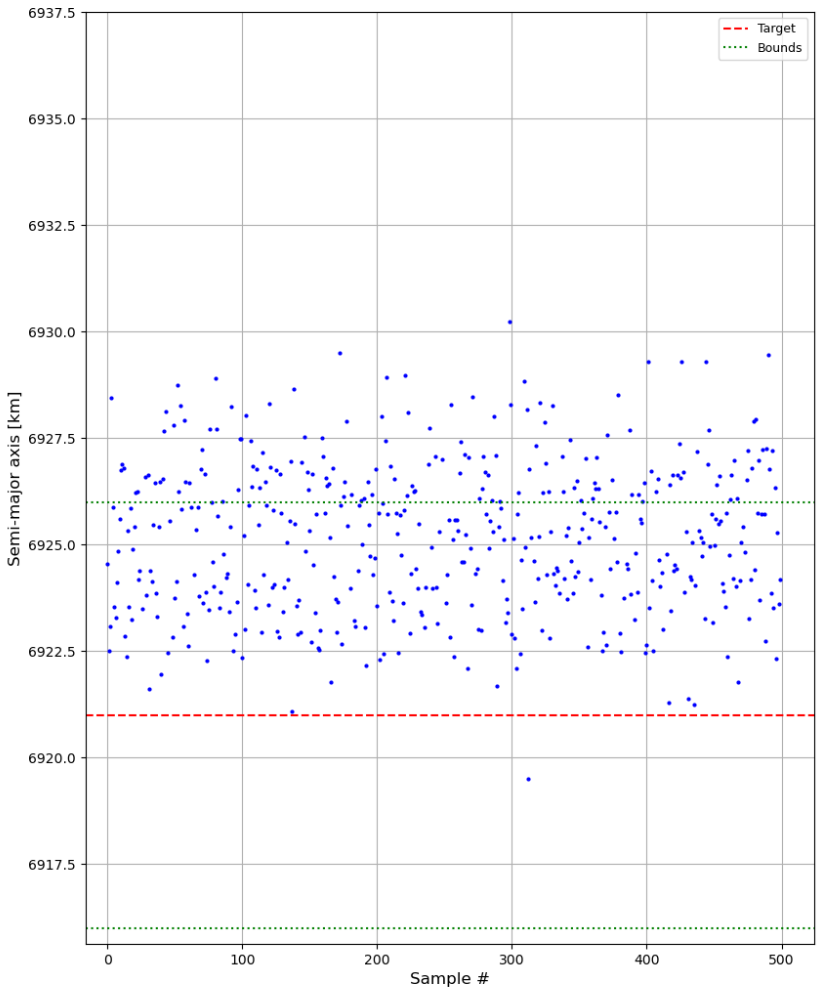
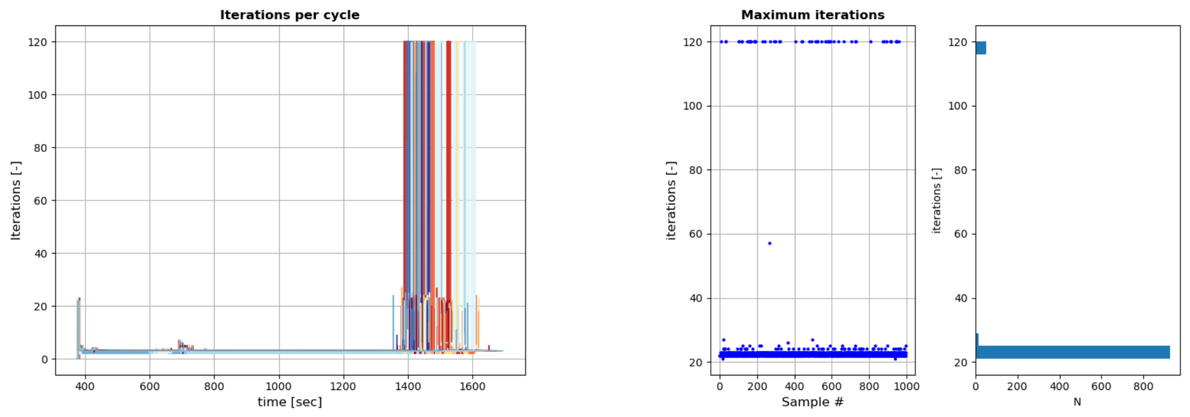


Figure 6.38: Dispersion of the semi-major axis successful orbit insertion in the FES

From the previous results it is already clear that the QPEG was able to converge in all cases, as none of them had a catastrophic failure like we have seen with the PEG. This is further confirmed by the number of iterations required by the QPEG throughout the flight, which is shown in fig. 6.39. The number of iterations required for initial convergence seems to be very consistent and well below the maximum. After that it requires only a few iterations during each guidance cycle to compensate for any errors in its predictions. The first exception to this occurs after second stage separation, where a couple more iterations are required due to the uncertainties involved in engine cut-off and staging. Towards the end of the flight there were some simulations were the QPEG failed to converge for a short period of time. Upon further inspection this would happen just before the altitude constraint release and the QPEG always re-converged right after the constraint release. For those simulations the bias values were abnormally large and didn't go to zero as expected. Instead they started to oscillate from one guidance cycle to the next, which seems to be the cause for the convergence failures. This issue seemed to be fixed by using variable segments in the propagator, as proposed in Section 6.1.5, although this would need to be confirmed by further analysis.

In order to assess the performance of the vehicle parameter estimator, its internal state was logged over the course of each simulation. The estimated values for both v_{ex} and τ were then



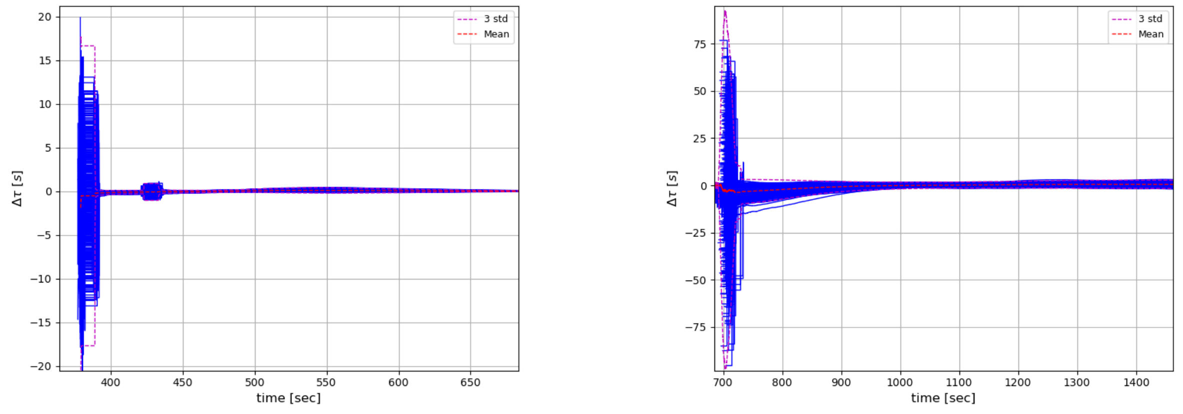
(a) Iterations over time

(b) Maximum iterations

Figure 6.39: Iterations per cycle of the QPEG in the FES

compared to the actual values computed from the thrust force, mass-flow, and mass. Due to the controller not having been tuned for the exact vehicle configuration used in this simulation, some attitude oscillations were observed during the second stage. Due to their detrimental effect on the parameter estimation the worst cases have been removed from the graphs used for this analysis, although the remaining simulations still contained some oscillations. The results including the worst cases will be shown in Section 7.4.

The results for the estimation error in τ are shown in Figure 6.40, which also shows the mean error and the 3-σ interval. For the initial period of each phase no measurements are used, so the parameters are left at their nominal values. These errors are therefore representative of the case when no parameter estimation is done. As soon as the parameters start being updated with measurements the error in τ decreases very rapidly, similarly to what was observed in the simplified testing environment. While the fairing release temporarily increases the error in τ, the filter quickly re-converges once the parameter hold ends.

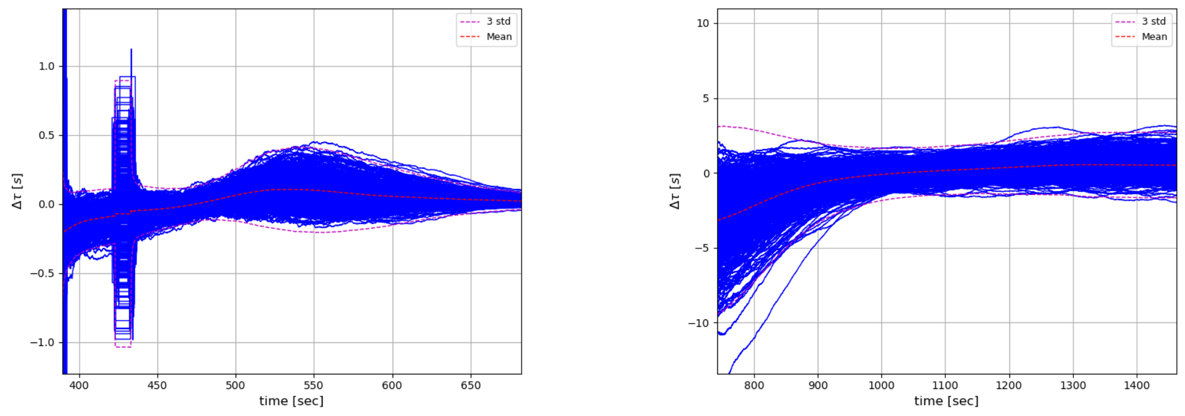


(a) Stage 2 flight

(b) Stage 3 flight

Figure 6.40: τ estimation error in the FES

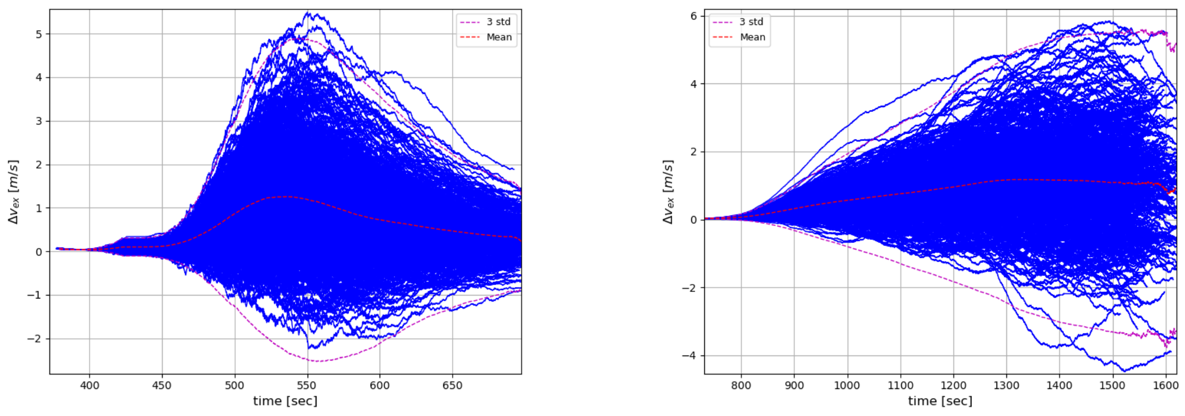
Re-scaling the y-axis to focus on the results after initial convergence, as shown in Figure 6.41, we get more insight into the continued algorithm behaviour. It clearly doesn't continue to steadily converge towards zero like in the simplified test environment. Instead there seems to be an initial negative bias for both the second and third stage. During the second stage the errors even start growing again around the mid-point before converging back towards the end of the stage. This is likely related to the previously mentioned attitude oscillations. During the third stage the errors just seem to converge to a non-zero steady-state.



(a) Stage 2 flight (b) Stage 3 flight

Figure 6.41: τ estimation error after initial convergence

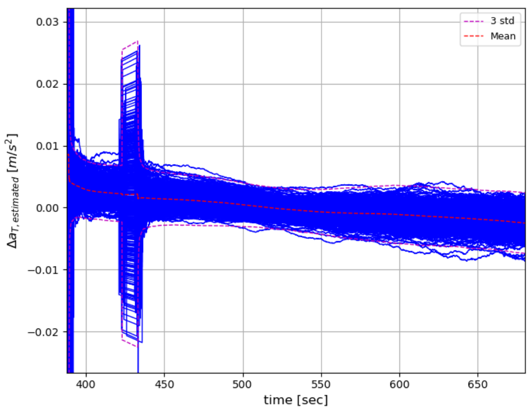
The estimation error in v_{ex} is shown in Figure 6.42. It should be noted that in the current uncertainty configuration of the FES no uncertainty is applied to v_{ex}, hence all the errors start at zero. It would therefore be expected that the errors oscillate a bit around the zero point with a magnitude proportional to the estimated state uncertainty. However for both stages the errors start growing to magnitudes much larger than even the initial uncertainty. For the second stage it eventually starts converging back to zero around the same time as τ. During the third stage however the growth in the errors is seemingly unbounded.



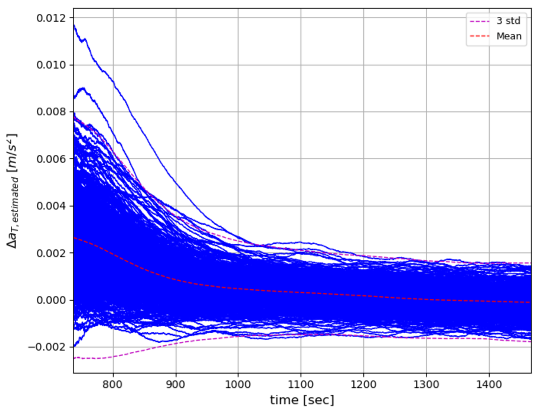
(a) Stage 2 flight (b) Stage 3 flight

Figure 6.42: v_{ex} estimation error in the FES

Despite what the estimation errors for τ and v_{ex} seem to suggest, the vehicle parameter estimator is still able to predict the instantaneous thrust acceleration with a high degree of accuracy. As can be seen in Figure 6.43, the errors seem to converge to some steady state that is smaller than the sensor noise level with a mean that is close to zero. For both stages there seems to be a bias in the initial estimates, which is likely caused by inconsistent configuration of the propellant burned during engine start-up. So it seems that the parameter estimator is reasonably able to estimate the current vehicle acceleration, but given the larger errors in v_{ex} it has more difficulty making accurate predictions of how it will evolve over time. While this will have an impact on the optimality of the guidance solution, eventually the closed-loop will correct for any accumulated errors.



(a) Stage 2 flight



(b) Stage 3 flight

Figure 6.43: Estimation error of the thrust acceleration in the FES

By combining the predicted time of insertion with the estimated total burn-time, the guidance is able to obtain an estimation for the performance margins. This is a crucial part of the target selection algorithm used when there are one or more alternate target orbits. It is therefore important to assess whether this estimation is sufficiently accurate. The error in the estimated margin throughout the flight is shown in Figure 6.44. It is clear that the guidance is capable of getting a reasonable estimate already during the second stage, as the errors aren't much larger than during the third stage. Furthermore, the margins remain rather constant during the flight, with the only large jump occurring at the second stage separation. This was another assumptions made for the target selection algorithm, as it only re-triggers the target selection process on a significant change in the estimated margin.

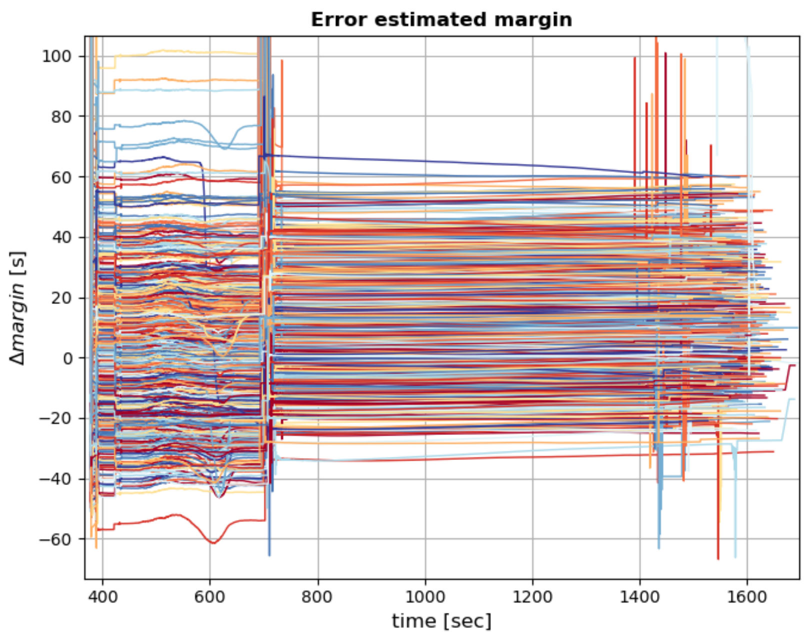
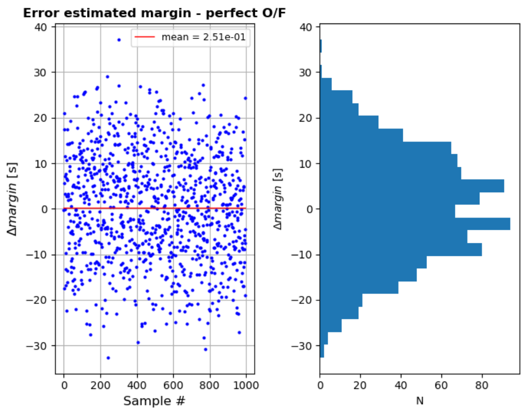


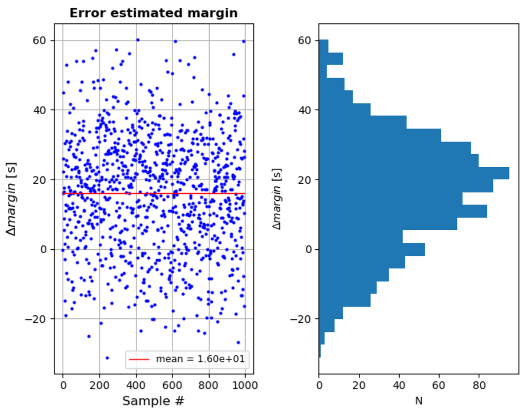
Figure 6.44: Error estimated margin in the FES

The error of the estimated margin at insertion is shown in Figure 6.45 for both an ideal

O/F on the third stage and a scenario where one propellant would deplete before the other. The differences between the two is the method for computing the reference third stage burn-time, with the former lumping all propellant together as done by the guidance and the latter computing separate burn-times for fuel and oxidizer and using the smallest of the two. Due to the simpler design of the stage 3 engine, the second scenario is more realistic. A non-ideal O/F could be caused either by uncertainties in propellant loading or engine performance. As can be seen the mean error for the ideal case is negligible, which means that the errors from the parameter estimator do not introduce any significant bias. The amount of dispersion seems to be consistent with the uncertainty in t_b for a fixed τ seen in Figure 6.20b. For the case with a non-ideal O/F the margins tend to be overestimated and the distribution is a bit skewed. In order to compensate for this the required margins during target selection would have to be increased.



(a) Ideal O/F stage 3



(b) Real O/F stage 3

Figure 6.45: Final error estimated margin in the FES

In order to verify that the orbit selection algorithm still function correctly in the FES, two sets of alternate target orbits where set up. The first more classical case involved just a single alternate target orbit at 200 km. For this scenario the minimum margins were set slightly negative in order to make the guidance a bit more opportunistic. In the second case additional targets were added between 200 km and 550 km in 50 km increments. For this case the required margins were set to zero in order to make it more conservative. For both scenarios the target orbits were ranked by their altitude.

The altitude profiles for the first scenario are shown in Figure 6.46. It is clear that the target down-selection happened at multiple points throughout the flight, although for the vast majority it was at the start of either the second or third stage. For the first group the down-selection occurred right after the initialization of the QPEG before parameter estimation had even started, which means it was purely based on the initial state after staging. There was one simulation where the target changed in the middle of the second stage, which was therefore likely triggered by the parameter estimation. For the second large group the target change happened at the start of the third stage, which means that an incorrectly estimated stage two burn-time was likely the cause. For all these simulations it is clear that the effect in terms of altitude doesn't become immediately visible, as the changes in attitude take some time to integrate into altitude differences.

The resulting orbit altitudes after engine cut-off can be seen in Figure 6.47. While the number of runs where the vehicle successfully inserted into one of the target orbits increased compared to the single target scenario, there are still a few cases where a stable orbit isn't achieved. Some of the runs where the guidance tried to insert into a 550 km orbit and failed

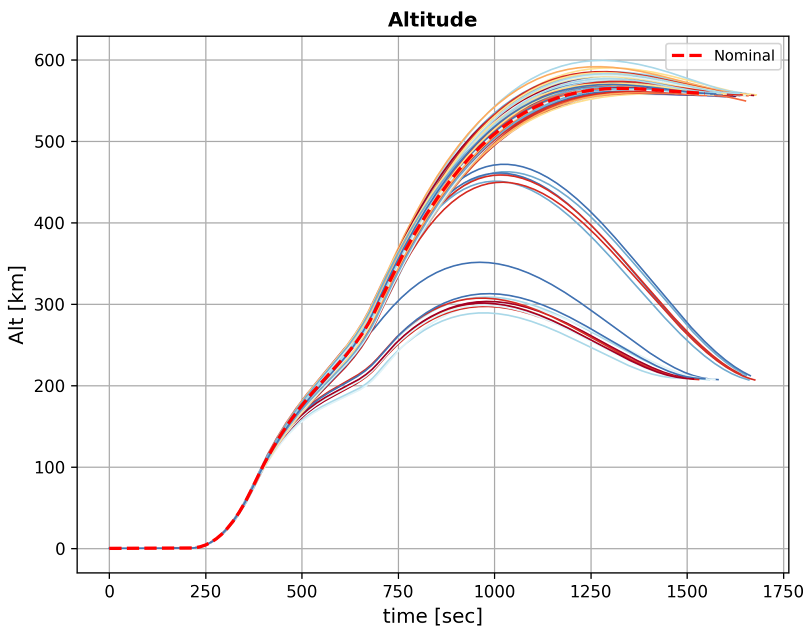


Figure 6.46: Altitude profile with two target orbits in the FES

now see the guidance successfully inserting into the back-up target. However the six cases where insertion into the back-up target fails were unexpected. The explanation is that these correspond to the runs that changed target at the start of the third stage. At this point the apoapsis altitude was already well above the back-up target altitude, as can be seen in Figure 6.46. This means that a large pitch correction was required to circularize at the new target altitude, with the rocket having an extreme angle of attack near insertion which lead to very large steering losses. Additionally while the burn-time margin for the lower orbit was larger, there is no such guarantee for the periapsis altitude. So a higher periapsis altitude might well have been achieved by sticking to the nominal target orbit. This is a prime example of why it is so important to make the target selection as early as possible in the flight and why accurate estimates of the second stage vehicle parameters are essential. In order to avoid this undesirable behaviour the required margins during the second stage flight could be increased to force a more conservative target selection. The decrease in the margin threshold after staging would make a change to a lower altitude target less likely to occur.

The results for the second scenario, in which a total of eight targets were provided to the guidance, is shown in Figure 6.48. With this configuration the probability of reaching a stable orbit is greatly increased, with only two runs re-entering the atmosphere. Furthermore in this batch the single outlier with a negative periapsis altitude was identified to correspond to a simulation which had significant attitude instabilities contributing to the loss of performance. It can also be seen that the guidance only used the nominal and first back-up. The small altitude difference between those two targets means that only small corrections are required to change between the two, even if the target change occurs during the third stage burn. Thus offering alternate targets close to the original one can be very beneficial.

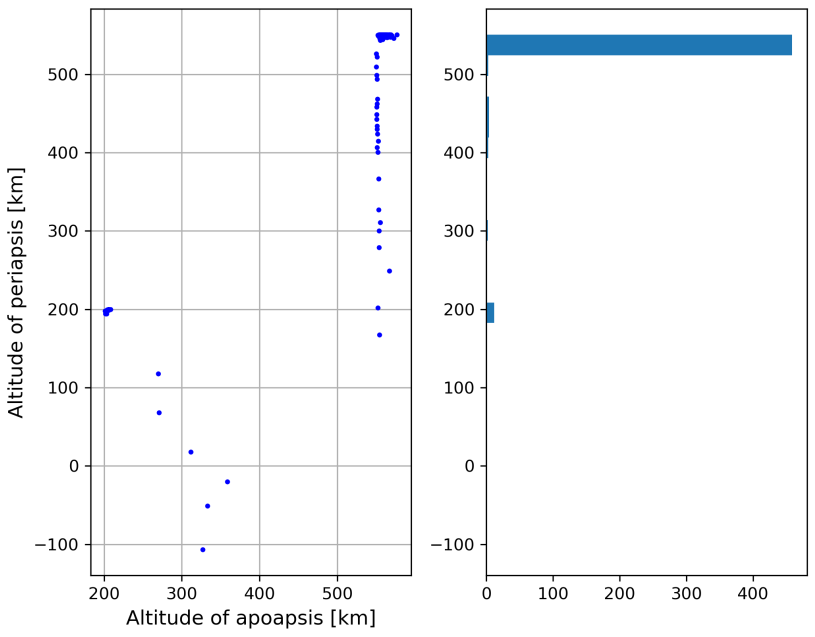


Figure 6.47: Apoapsis and periapsis dispersion two target orbits in the FES

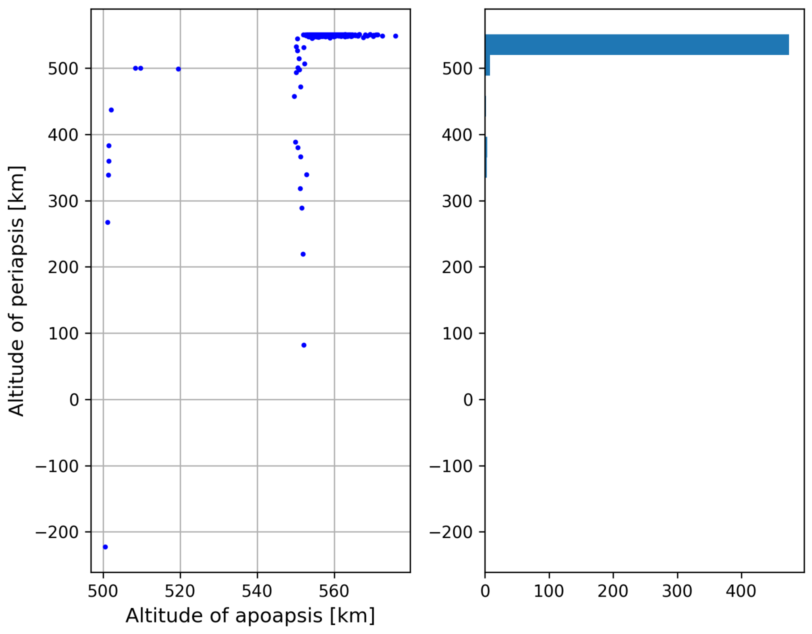


Figure 6.48: Apoapsis and periapsis dispersion eight target orbits in the FES

7 Verification and validation

In order to ensure that the guidance algorithm will perform its task in flight successfully, it will have to go through the process of verification and validation. This involves testing that each part of the system works as it is supposed to do and that the guidance successfully integrates with the rest of the GNC and flight computer software [46]. It will of course be impossible to perform a real-life test of the guidance system without launching the rocket itself, but using software simulations it should still be possible to get a high level of confidence that the system will behave in a satisfactory manner. The sections in this chapter follow the level of integration, starting with unit-testing discrete functions in Section 7.1. Then the major components of the guidance are verified in Sections 7.2 and 7.3. This is followed by tests of the entire guidance system in stressing scenarios in Section 7.4. Finally the capability of the guidance algorithms to run in a real-time environment is demonstrated in Section 7.5.

7.1. Unit-tests

The first part of verification involves unit-testing the individual software components and functions. By splitting the code into small pieces with a well defined function it often becomes possible to analytically derive simple test cases. By designing the inputs in such a way that each code path is covered, it is possible to already catch the majority of the bugs in the code at this early stage. Furthermore these unit-tests can easily be integrated into an automated pipeline which ensures that there is no regression when making changes to the code. This helps catch some of the potential unforeseen consequences when making changes to a piece of code at a later time. A brief overview of the subset of unit-tests in the guidance that were implemented as part of this thesis is given below.

7.1.1. Test-cases closed-loop guidance

The following list of test-cases were implemented for the QPEG:

- The propagation of the unknowns vector that happens at the start of each guidance cycle is verified by using the analytical solutions for the co-state.
- The accuracy of the analytical propagator is verified by comparing its results to those of the numerical propagator.
- Interior-point continuity constraints are tested by giving it two identical vectors verifying it returns the zero vector.

- The full target orbit constraints were tested by constructing a state vector from the orbit parameters and the associated optimal co-state. The test passed if it returns the zero vector.
- The reduced target orbit constraints were tested in a similar manner, with the exception that a different optimal co-state was used.

All test above were successful.

7.1.2. Test-cases parameter estimation

The following unit-tests verify different aspects of the parameter estimation in an automated fashion:

- The correct computation of the nominal values of τ, v_{ex}, and t_b for each stage from the component based vehicle definition was verified by comparing it the model provided by Systems Engineering.
- The propagation part of the Kalman filter was tested by initializing it at t_1 and then requesting the parameters at t_2. The difference for t_b and τ compared to their initial values was then verified to be equal to $t_2 - t_1$ and it was verified that v_{ex} remained unchanged.
- It was verified that both the state and state covariance are reset to the correct values when the parameter estimator was set to a phase that corresponds to a different stage.
- It was verified that only τ and its uncertainty were changed when the parameter estimator was set to a new phase within the same stage.
- The ability of the parameter estimator to compensate for drag disturbance was tested by simulating measured accelerations with off-nominal parameters for the drag acceleration and verifying that the errors for τ and v_{ex} were within acceptable bounds.
- The ability of the parameter estimator to reject measurements during periods of high angular rates was tested by simulating measured accelerations with angular rates above the limit values and verifying that the errors for τ and v_{ex} were still at the nominal values.
- The ability of the parameter estimator to tolerate a delay between a staging command being sent and its effect on the measured acceleration was tested by simulating the measured accelerations that would occur in such a scenario and verifying that the errors for τ and v_{ex} were still at the nominal values.
- It was verified that the error induced by sensor noise on τ and v_{ex} was within acceptable bounds.

All of the tests above were also successfully executed.

7.2. Verification of the parameter estimation

While some performance results for the parameter estimation have already been shown in Chapter 6, it useful to provide an objective measure of the Kalman filter's performance. Furthermore when deriving the filter some assumptions were made on the dynamics and the measurement noise, so it is important to verify that the performance is consistent with the underlying theory.

It is important to verify the consistency of the filter, which means verifying that the estimated state converges to the true state. If the estimation errors become unacceptably large it is said to be diverging. In order to assess the performance of the filter it is required to define some criteria to quantify this consistency, two of which will be used in this thesis [47]:

- The state error should have a zero mean
- The state error covariance should be equal to the one given by the filter

Both require many realizations of the state error, which means that they will be evaluated using the results of Monte-Carlo simulations. The first test will be on the *Normalized Estimation Error Squared* (NEES), defined by Equation (7.1). If the filter is consistent $\epsilon(k)$ is chi-square distributed with n_x degrees of freedom, equal to the number of states, with an expected value given by Equation (7.2)

$$\epsilon(k) = \delta\mathbf{x}(k)\mathbf{P}(k)^{-1}\delta\mathbf{x}(k) \tag{7.1}$$

$$E[\epsilon(k)] = n_k \tag{7.2}$$

The average NEES from N Monte-Carlo simulations is then given Equation (7.3).

$$\bar{\epsilon}(k) = \frac{1}{N}\sum_{i=1}^{N}\epsilon^i(k) \tag{7.3}$$

The hypothesis H_0 that the state estimation errors are consistent with the covariances calculated by the filter can then be accepted if Equation (7.4) is true, where r_1 and r_2 are defined by the $1-\alpha$ probability region of a chi-square distribution with Nn_x degrees of freedom normalized by N [47].

$$\bar{\epsilon}(k) \in [r_1, r_2] \tag{7.4}$$

If the state estimation error has a significant bias, the above test is likely to fail even if the variability of the error is small. Since there is only a single test for the entire state vector, it could be caused to fail by a bias on only a single component. In that case the second test can be used to look at the *Normalized Mean Estimation Error* (NMEE), defined by Equation (7.5) for each component j.

$$\mu_j(k) = \frac{1}{N}\sum_{i=1}^{N}\frac{\delta x_j^i(k)}{\sqrt{P_{jj}(k)}} \tag{7.5}$$

If the filter is consistent μ_j is distributed $\mathcal{N}(0, 1/N)$. The hypothesis that the true mean of the estimation error is zero can then be accepted if Equation (7.6) holds, where r is defined by the $1-\alpha$ probability region of a zero-mean normal distribution with a standard deviation of $1/\sqrt{N}$ [47].

$$\mu_j(k) \in [-r, r] \tag{7.6}$$

These two methods were applied to the FES results for the parameter estimation, where the simulations with excessively high angular rates have been filtered out. First we can look at the NMEE, shown in Figure 7.1, since the mean values being close to zero is a prerequisite to the NEES test being valid. Considering that this batch contains 500 samples, the acceptance interval would have to be computed using a standard deviation of $1/\sqrt{500} = 0.0447$. Clearly this not the case for any reasonable value of α, which means that both states contain systematic biases. It is therefore no surprise that the average NEES, shown in fig. 7.2, is also outside of any reasonable acceptance bounds. It can therefore be concluded that the filter greatly underestimates its estimation errors. A possible solution for this issue would be to at least re-tune it to take into account the larger degree of uncertainties caused by perturbations that were initially neglected. For some of these perturbations it might also be necessary to extend the dynamics model of the filter and add additional states.

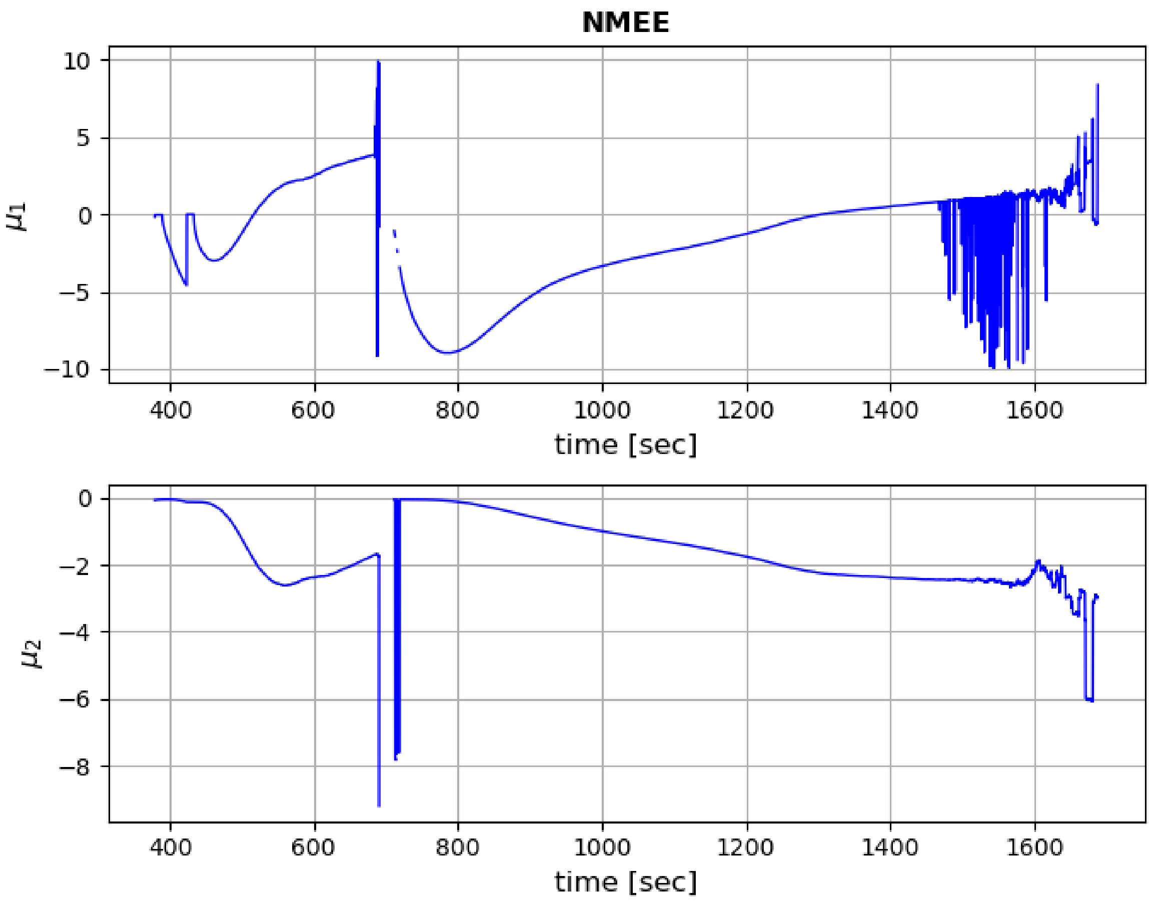


Figure 7.1: Normalized mean estimation error parameter estimation in the FES

7.3. Comparison closed-loop guidance to reference trajectory

The closed-loop guidance can very well converge to a solution that seems satisfactory, however there is no guarantee that this solution is a global optimum. For example the PEG and QPEG converge to different solutions leading to differences in performance. In order to validate the optimality of the trajectory produced by the closed-loop guidance algorithm, it can be compared to the reference trajectory generated by a proven trajectory optimization software. In order to perform this comparison, the dynamics model and the constraints used to generate the reference trajectory should be set up such that they resemble the ones used for guidance simulations as close as possible. Then the commanded pitch and yaw profiles can be compared as well as the total time to orbit insertion. Under the assumption that the optimization controls and grids were set up correctly when generating the reference trajectory, the solution it finds should be an optimum. Furthermore by making some adjustments to the initial guess, such as trying to replicate the solution found by the closed loop guidance, it can be ruled out

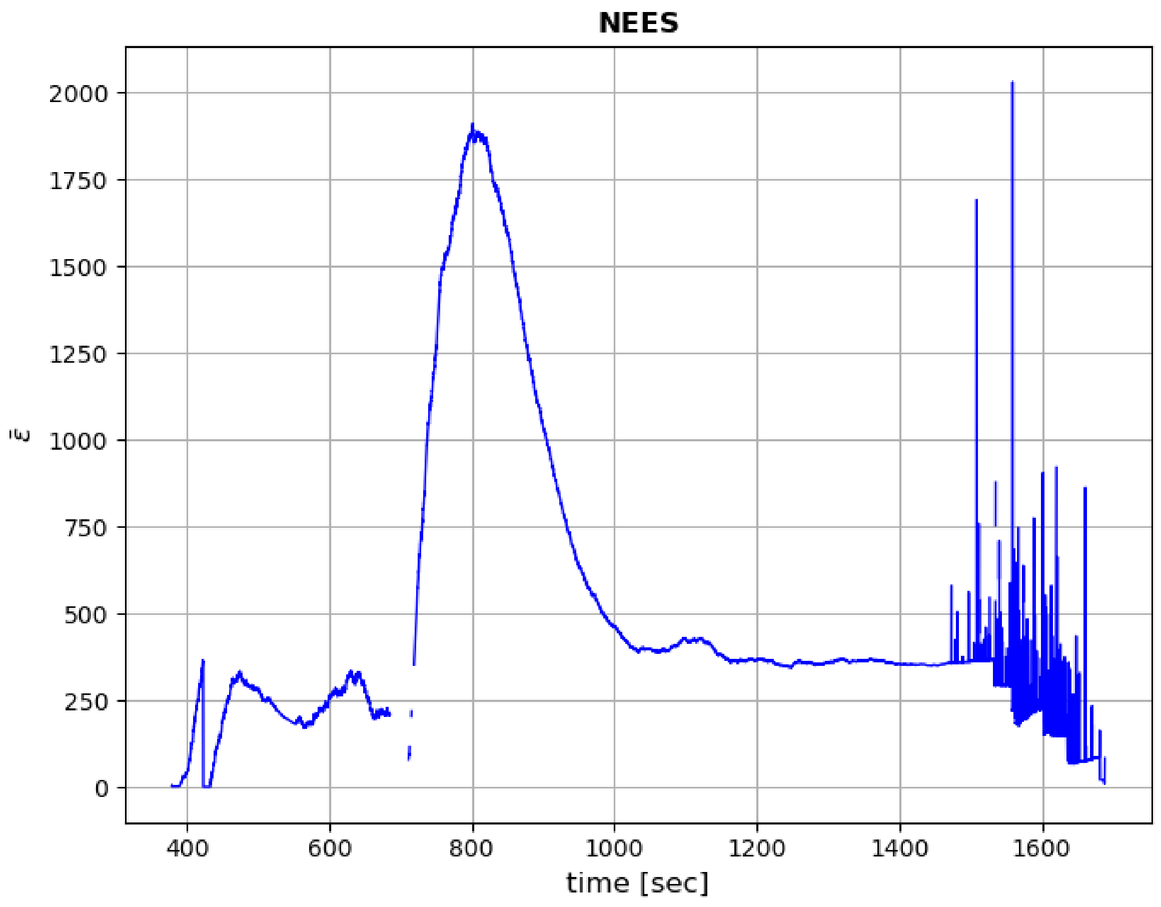


Figure 7.2: Average normalized estimation error squared parameter estimation in the FES

that no better optimum exists within the immediate vicinity and that it is a global optimum. The performance of the closed-loop guidance is expected to be close to the one found for the reference trajectory. If there is a significant difference, this could point towards a general deficiency in the algorithm. In order to prove this the test could be redone with a scenario that is more favorable to the closed-loop guidance by better satisfying the assumptions, such as having a higher TWR. In previous research done at RFA such a difference was indeed found, although the investigation into its origins was largely inconclusive [4].

7.3.1. Setup

The trajectory selected for this comparison is the baseline mission scenario, in which a three-stage rocket performs a direct insertion into a 550 km circular orbit. During the second stage there is a fairing release and between the second and third stages there is a coast arc of $5\ s$ to represent staging. This brings the total number of phases to five:

1. First stage
2. Second stage with fairing
3. Second stage without fairing
4. Coast
5. Third stage

The comparison will not be performed on phase 1, as the QPEG is not suitable for the atmospheric flight of the first stage. While an atmosphere and aerodynamics model was used for the reference trajectory, it will not be relevant for this test as only the exo-atmospheric part of the flight is simulated. To simulate the trajectory with the QPEG the simplified 3-DOF simulator will be used as it most closely matches the dynamics used to generate the reference trajectory. It was further ensured that both the vehicle configuration and gravity model, which includes the J_2 term, were identical.

7.3.2. Results

Due to the large amount of results, only the most relevant ones will be presented here. The first comparison was performed by initializing the QPEG at the start of phase 2, which is normally where the closed-loop guidance would start. The resulting pitch profiles and their difference are shown in Figure 7.3. Apart from a difference at the start of the second stage and at the very end of the third stage, the two pitch profiles are basically identical. The difference in pitch at the start of the second stage is due to a continuity constraint being enforced for the controls of the reference trajectory, similarly to the QPEG. However the reference trajectory shown here didn't start at the second stage, but at lift-off so the pitch angle at the start of the second stage needs to be identical to the pitch angle at the end of the first stage, a limitation which is not imposed on the QPEG. Furthermore the controls in trajectory optimizer were approximated as linear splines and the control grid used was rather coarse, with only a single segment for phase 2. Another potential source of differences is that for reference trajectory the fairing release constraints are actively enforced such that the time of fairing release can be optimized. This might cause it to go for a slightly more lofted trajectory than the QPEG if everything else is equal. The differences at the end are due to the position constraint release used in the QPEG, which is of course not present in the reference trajectory, and due to the simulation of the QPEG continuing for a short time after orbit insertion.

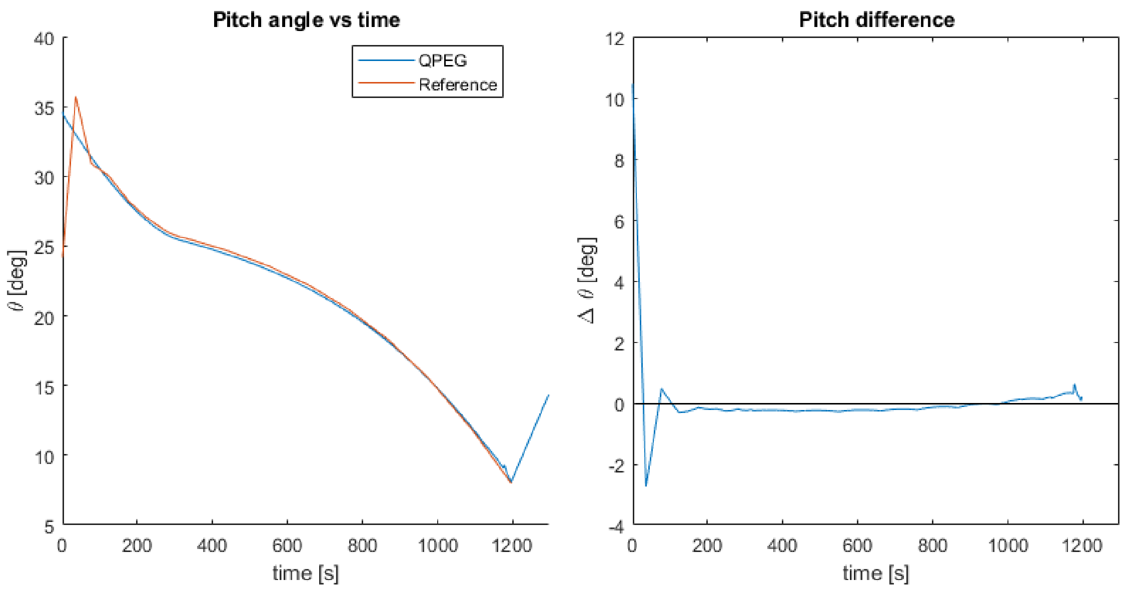


Figure 7.3: Comparison pitch between QPEG and reference trajectory, starting at phase 2

An additional point of comparison is the altitude profile, shown in Figure 7.4. With a maximum difference of just over $2\ km$, the differences here are minor as well. The altitude and velocity profiles were used in part to ensure that the different force models were consistent. As a final point it should be noted that in this particular configuration the QPEG performed slightly better than the reference trajectory, with a difference of just over $1\,s$, although this can mostly be attributed to the previously mentioned additional constraints used in the trajectory optimizer.

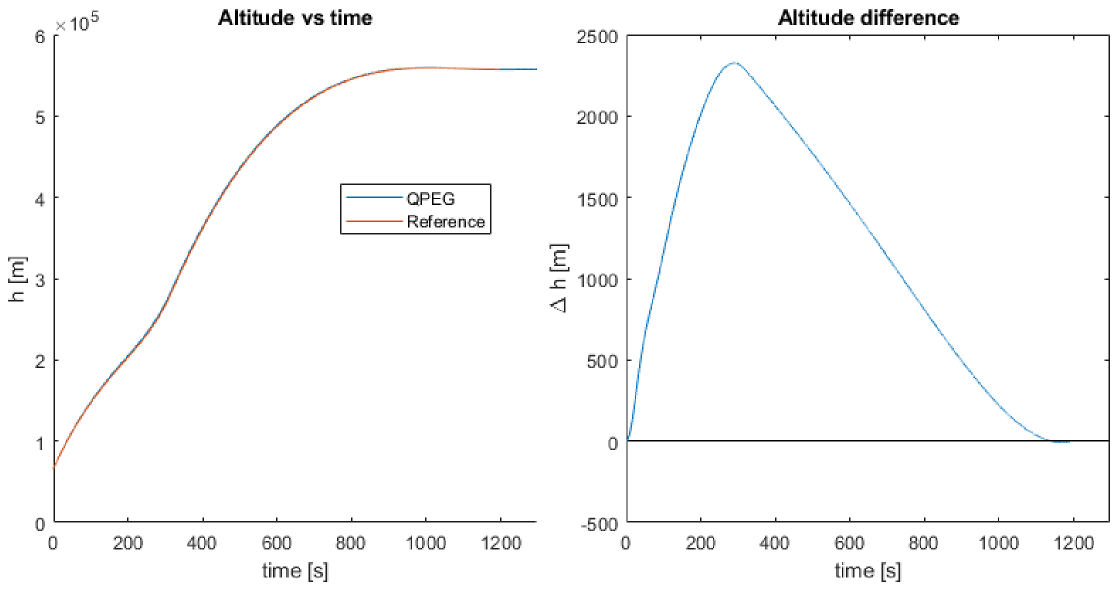


Figure 7.4: Comparison pitch between QPEG and reference trajectory, starting at phase 2

In order to eliminate the effects of the initial discrepancy in the pitch angle, additional comparisons were performed by initializing the QPEG at later stages of the flight. The results for phases 3 and 5 can be seen in Figures 7.5 and 7.6 respectively. By starting the QPEG right after fairing release most of the pitch-up discrepancy is avoided, while still including the majority of the second stage flight. It therefore still includes the challenging unbalanced thrust acceleration profile of a high TWR second stage and a low TWR third stage. Despite this, the difference in pitch angle is negligible for most of the flight.

When starting the QPEG at third stage ignition these differences become even smaller. The errors at this point are most likely caused by the use of linear splines for control parametrization in the trajectory optimizer. In this case the difference in performance has also vanished and the differences in velocity and position are negligible.

7.4. Robustness in stressing scenarios

In order to test the entire guidance software it can either be run in a stand-alone fashion with a simplified 3-DOF model or it can be integrated with the rest of the GNC code in the FES. The first approach allows for isolating just the behavior of the guidance while the latter is essential for finding the complex interactions of the different sub-systems. Since there is some uncertainty in the vehicle parameters and environmental conditions it isn't sufficient to just run tests with the nominal configuration. For this purpose a framework for running Monte-Carlo simulations has been developed within the GNC team. Monte-Carlo analysis is a commonly used technique to test the robustness of a design and verify that it satisfies requirements in a statistical manner [48]. As an alternative it is also possible to generate a smaller set of worst-case scenarios using a combination of Design of Experiments grids and a Maximum Likelihood Estimation process [49]. This would allow for stressing the system in a computationally more efficient manner.

The scenario that will most stress the guidance is different for each of its components. For the closed-loop guidance it has already been shown that the number of iterations required for

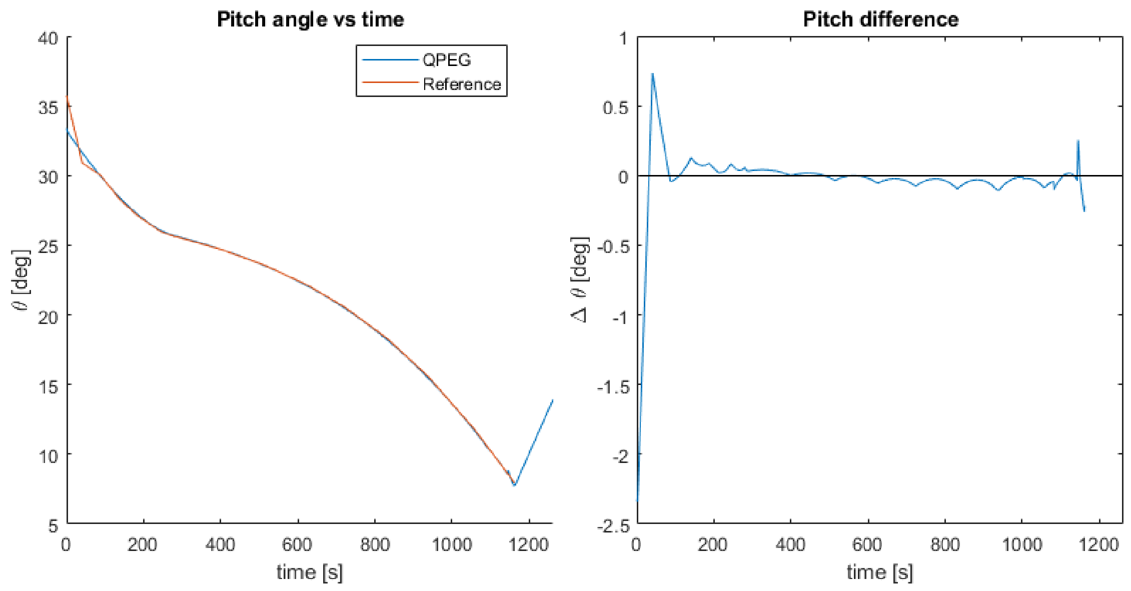


Figure 7.5: Comparison pitch between QPEG and reference trajectory, starting at phase 3

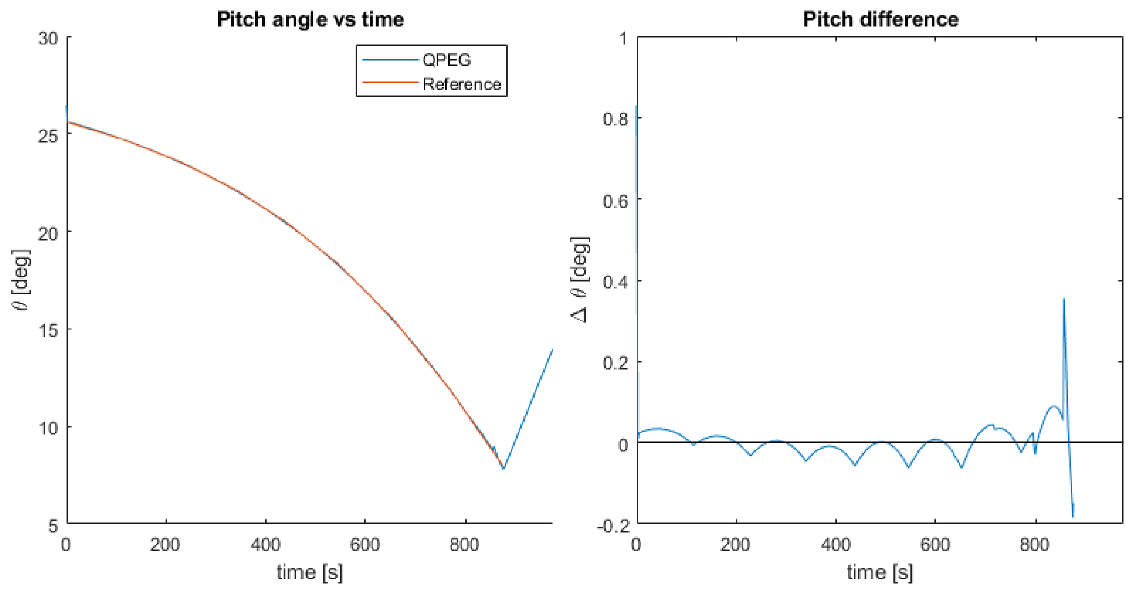


Figure 7.6: Comparison pitch between QPEG and reference trajectory, starting at phase 5

convergence is tightly coupled to the total time to insertion. In general a long time to insertion will be caused by a low velocity at stage two ignition, which is the case if there are one or more engine outs on the first stage. In order to verify that the guidance always converges, the initial conditions should therefore be set to those encountered after one or more engine outs. For the parameter estimator the challenges largely come from all the disturbances affecting the sensed acceleration. Modelling all these in the isolated simulator would be too complex, and data from the FES should therefore be used. Possible sources of unexpected high disturbances include starting the second stage at a lower altitude, which would lead to higher drag, and instabilities in the controller, causing large angular rates throughout the flight.

In order to stress the QPEG it was chosen to test a configuration where a single engine would fail at a random time during the first stage flight. This case was chosen to answer a mandatory operational requirement of the maiden flight. The rocket should be able to tolerate an engine out at any point during the first stage and continue its mission to the best of its ability. This means that a stable flight for the remaining stages is required and if physically possible it should also insert into some stable orbit. While an extensive performance analysis, similar to the one done for SLS [11], has yet to be performed, it can already be verified that the QPEG can handle such cases. For the requirement of a stable flight the QPEG would ideally always converge to a solution for all the resulting initial conditions. As shown in Figure 7.7 this is indeed the case. Despite the much larger dispersion of initial conditions compared to a nominal flight, it is still well within the region where convergence is guaranteed. In the unlikely but critical case where two or more engines fail the QPEG might not be able to always converge for the nominal target orbit, but it would not be possible to reach the nominal orbit anyway.

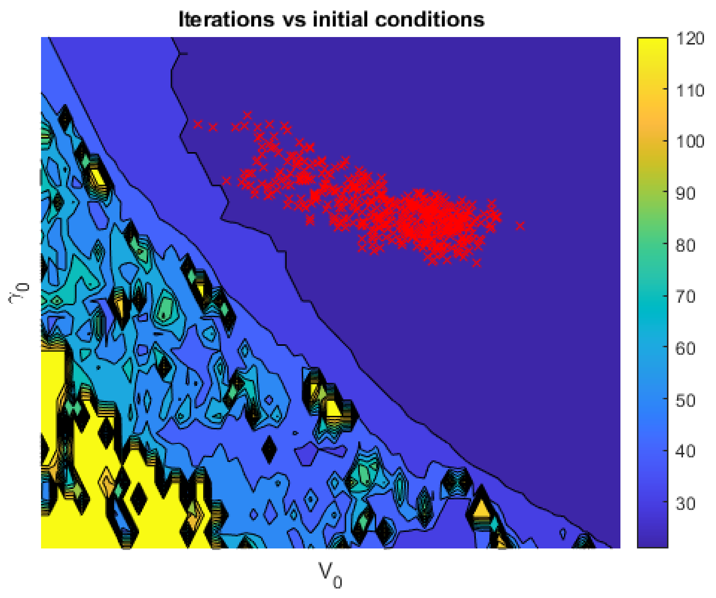


Figure 7.7: Convergence of the QPEG for single engine out on the first stage

The second challenge for the closed-loop guidance is that the rocket should still try to insert into some form of stable orbit despite the reduced performance. The periapsis dispersion when using just a single target orbit is shown in Figure 7.8. Clearly the probability of reaching the nominal orbit is significantly reduced and a larger amount of simulations fail to insert into a stable orbit. This represents the case where no corrective measures are taken in regards to throttling, open-loop pitch biasing, or target selection. However the automatic target selection algorithm developed as part of this thesis was specifically meant to be able to resolve some of these issues. When re-running those simulations using the six target configuration that was previously tested in Section 6.3, the periapsis distribution shown in Figure 7.9 was obtained. While there are still some cases where the rocket doesn't reach a stable orbit, their number has been greatly reduced. Furthermore in the vast majority of cases the rocket now inserts into one of the predefined circular orbits instead of some unpredictable elliptical orbit.

As previously mentioned the results for the parameter estimator performance in Section 6.4 specifically excluded some simulations that contained unexpectedly high angular rates throughout the second stage. While these instabilities will likely disappear after fine tuning the attitude controller, it is still interesting to analyze the behaviour of the parameter estimation in such a scenario. The estimation errors of v_{ex} and τ for all simulations are shown Figure 7.10. Clearly

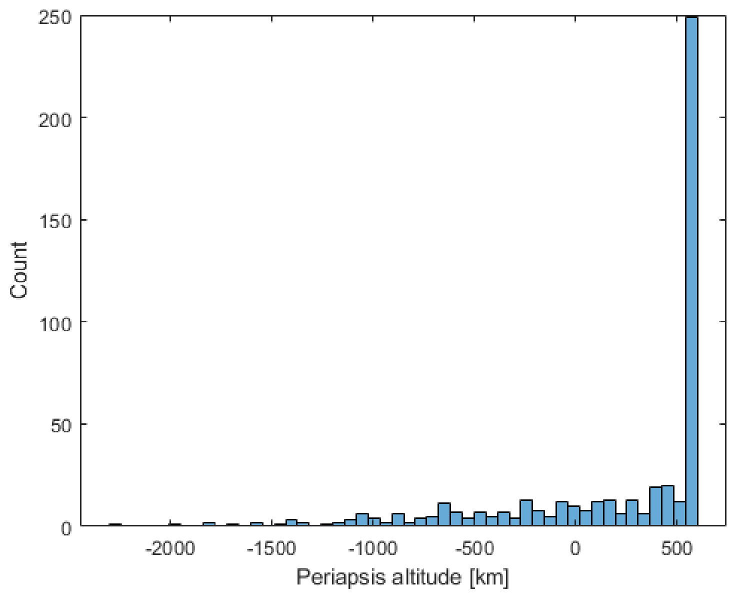


Figure 7.8: Periapsis dispersion for single engine out on the first stage with a single target

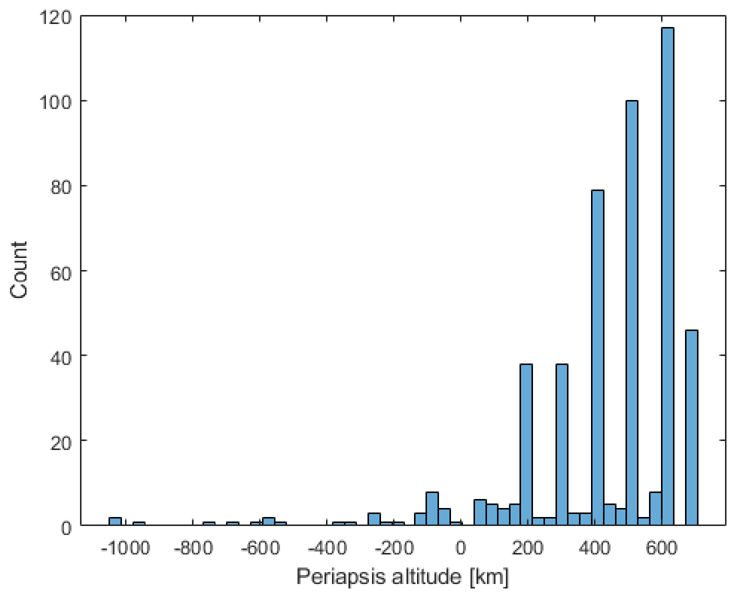
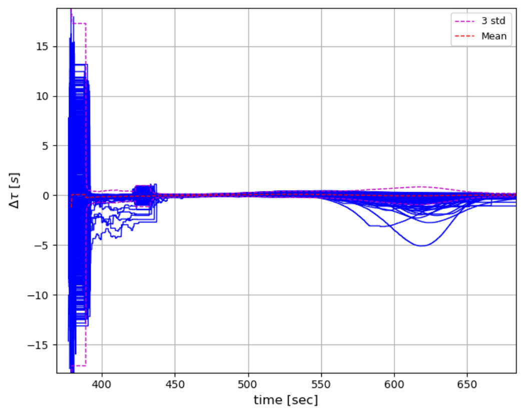
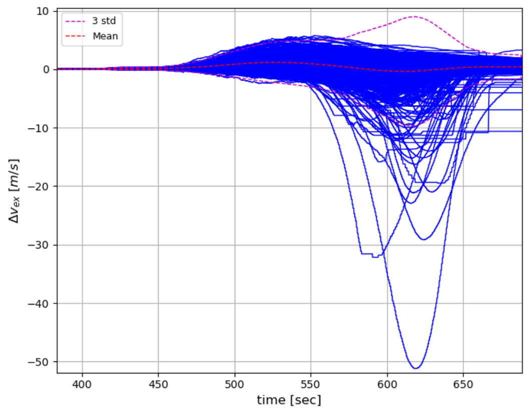


Figure 7.9: Periapsis dispersion for single engine out on the first stage with multiple target

the estimation errors are much larger for both parameters during the periods of high angular rates. The safety measure to discard acceleration measurements when the angular rates exceed a predefined threshold becomes active in some cases by essentially cutting off the peaks. As the attitude oscillations diminish towards the end of the stage, the estimated parameters start to converge back to the true values again. It is interesting to note that the couple of cases where there are large errors in τ at the start of the stage, those errors vanish after the fairing release presumably thanks to the artificially increased state uncertainty. So while high angular rates can cause fairly large estimation errors, they are still smaller than the initial uncertainty. Furthermore once the perturbations stop, the errors slowly converge back to zero.



(a) Stage 2



(b) Stage 3

Figure 7.10: Estimation error of the thrust acceleration in the FES

7.5. HIL testing

As a final validation step the guidance will be run with the rest of the GNC software on the flight computer. In order to enable this RFA has a Hardware In the Loop (HIL) setup which includes not only the flight computer, but also a real-time simulator for the flight dynamics and sensor outputs. With this test configuration the effects of delays and real-time scheduling on the flight hardware can be analyzed and quantified. Contrary to the navigation and control sub-systems, the guidance has little to no direct interaction with hardware components. It is however still essential to check that the algorithm can run smoothly within the allocated time-slot and does not perturb the real-time nature of the other components. In timing benchmarks it was verified that initial convergence of the QPEG with 22 iterations took around half the time allocated for the fast control loop. Under ideal conditions it could therefore run at the same rate as the controller. However when taking into account that in case of non-convergence the number of iterations will be significantly higher, the guidance parameter update has to run in a separate slower loop in order to guarantee that it doesn’t interfere with the timing of the controller. As a point of comparison the QPEG was also run with the numerical propagator replacing the analytical one, which resulted in a five-fold increase in execution time. This clearly demonstrates the advantage of the analytical formulation.

8 Sensitivity analysis

For many of the results presented in the previous two chapters some simplifications were made to the dynamics and not all parameters had uncertainty applied to them. This was especially true for the 3-DOF simulations. While the FES has far fewer simplifications, it has the issue that many different dynamics models and parts of the GNC system can cause perturbations, which makes it difficult to trace the source of some phenomena. This chapter will therefore analyze a couple of perturbation sources to quantify their isolated effect on the guidance performance.

8.1. Parameter estimation

For the vehicle parameter estimation the simplified models ignored most perturbations, and only the atmospheric drag has been considered. However as was shown in Section 6.4, the performance in the FES was clearly much worse than in the simplified testing environment. Therefore the parameter estimator will be submitted to test cases involving a wider range of perturbations in the isolated test environment.

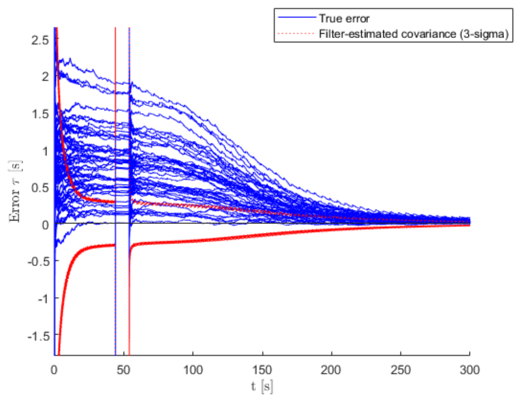
8.1.1. RCS gas consumption

One of the inaccuracies in the model used to test the parameter estimation on its own was the assumptions that the propellant mass-flow through the engines was the only source of mass changes. However during the second and third stages there is another source of small mass change coming from the RCS thrusters used for roll control. It happens in short pulses at semi-regular intervals and this mass-flow have no effect on the net thrust. This results in small step-changes in τ, which can be modelled using Equation (8.1). There is also an effective reduction of v_{ex}, as shown by Equation (8.2).

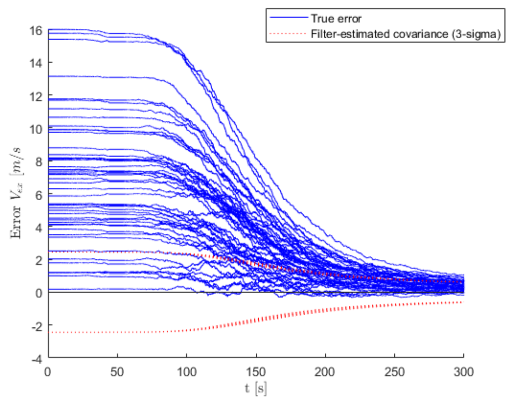
$$\tau = \tau_0 - t - floor(t/\Delta t_{RCS})\Delta\tau_{RCS}, \qquad \Delta\tau_{RCS} = \frac{\Delta m_{RCS}}{\dot{m}_{engine}} \tag{8.1}$$

$$v_{ex,eff} = \frac{T_{engine}}{\dot{m}_{engine} + \frac{\Delta m_{RCS,tot}}{t_b}} \tag{8.2}$$

After modifying the model using Equation (8.1) the parameter estimator was ran using the same configuration parameters that were used before. The results for the estimation errors are shown in Figure 8.1 where both the magnitude and interval of the RCS pulses was varied randomly within the expected range. Clearly both parameters don't behave as they did in the ideal case. For v_{ex} the initial error is greater than the initial uncertainty and is biased to one side due to the lower effective v_{ex}. Despite the inconsistent uncertainties, v_{ex} is still able to converge to a rather small error. The error in v_{ex} also translates to an error in τ, which even after it has converged is still inconsistent with the filter estimated uncertainties due to the saw pattern caused by the RCS pulses. All this means that while the filter is able to estimate the instantaneous acceleration quite reliably, future accelerations will not be as accurate during the first half of the stage.



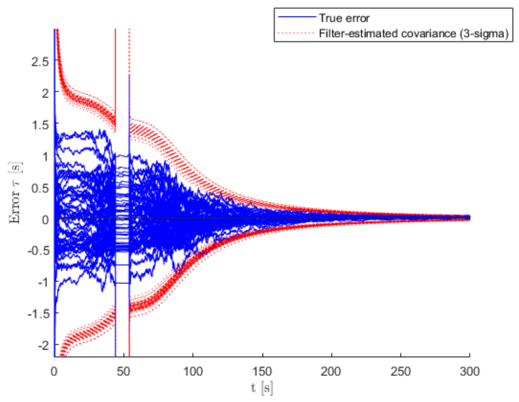
(a) τ estimation error



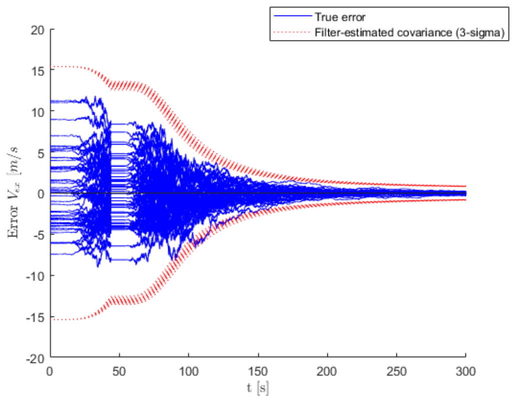
(b) v_{ex} estimation error

Figure 8.1: Effect of RCS gas consumption estimation errors, $\overline{\Delta\tau}_{RCS,tot} = 0.5s$

In order to mitigate these issues a number of changes are required. For τ the process noise should be increased such that it doesn't become overconfident and the filter estimated variance stays consistent with the true errors. For v_{ex} the initial estimates shouldn't be reduced to take into account average RCS gas consumption and the initial uncertainties increased to take into account the variability in the RCS usage from one flight to another. The results after the implementation of these changes are shown in Figure 8.2. The changes make the filter consistent again as the bias is mostly removed and state uncertainties no longer converge faster than the errors. The only negative point compared to the results shown in Section 6.2 is the increased uncertainties during the first part of the stage, which will also have an effect on the uncertainties of t_b and t_{go}. Fortunately towards the end of the stage the errors on both parameters converge to reasonable values. So while not explicitly modelled by the dynamics of the filter, RCS gas consumption can still be handled by simply tuning the filter correctly.



(a) τ estimation error



(b) v_{ex} estimation error

Figure 8.2: Effect of RCS gas consumption estimation errors after re-tuning, $\overline{\Delta\tau}_{RCS,tot} = 0.5s$

8.1.2. Non-constant thrust

Another source of potential model error is the assumption of a constant thrust force in vacuum. This assumption was made for both the isolated test environments and the FES. However there is no guarantee that this will actually be the case. Changes in thrust could be caused for example by changes in the O/F ratio requiring different set-points for both the O/F and throttle valves. Furthermore varying pump inlet pressures and temperatures could also affect the turbine speed. Recent engine test campaign results showed that the thrust exhibited some variability throughout the full burn duration.

It was therefore decided to test the vehicle parameter estimator using a steadily increasing mass-flow. This was modelled by making a slight adaption to the computation of τ to account for the linear change in the mass-flow, as shown in Equation (8.3) where $\delta_{\dot{m}}$ is the normalized thrust drift. Using this model the acceleration will increase not only due to the increasing thrust, but also due to the more rapidly decreasing mass. The filter was left unchanged for this test.

$$\tau = \frac{m_0 - \dot{m}_0 t - \frac{1}{2}\ddot{m}t^2}{\dot{m}_0 + \ddot{m}t} = \frac{\tau_0 - t - \frac{1}{2}\delta_{\dot{m}}t^2}{1 + \delta_{\dot{m}}t}, \qquad \delta_{\dot{m}} = \frac{\ddot{m}}{\dot{m}_0} = \frac{\dot{T}}{T_0} \tag{8.3}$$

The results for the estimation errors of τ and v_{ex} are shown in Figures 8.3 and 8.4 respectively for a relatively small drift. It is clear that both parameters start diverging at around $100\,s$, which is precisely when the estimation error on v_{ex} would start converging to zero. Although the estimates for the instantaneous acceleration are still rather accurate, albeit with a small lag, the precision of future thrust acceleration levels are worsened. For larger values of $\delta_{\dot{m}}$ the filter performance only continues to decrease. In order to make the filter consistent again, a rather large increase to the process noise of $\frac{\tau}{v_{ex}}$ is required. This has the effect of making v_{ex} much less observable, which means its estimate will not deviate much from the nominal value.

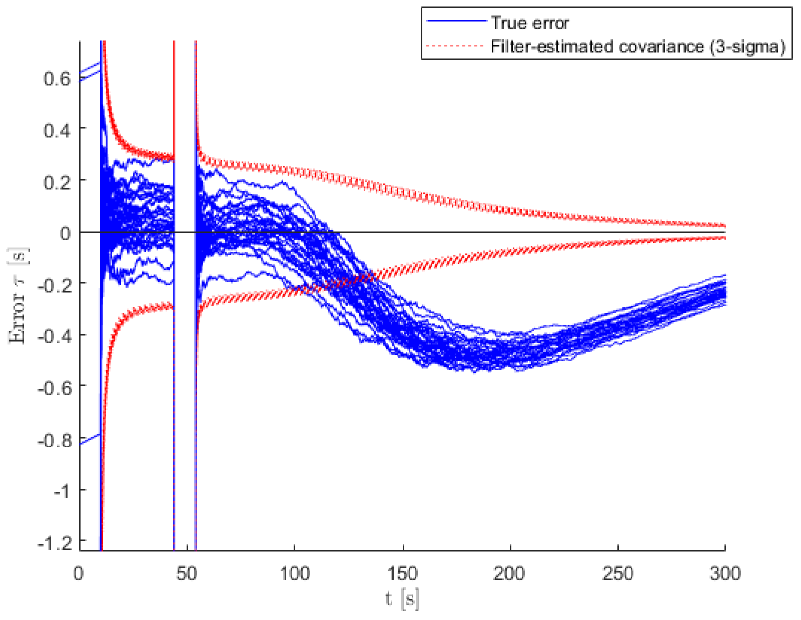


Figure 8.3: Effect of thrust drift on τ error, $\delta_{\dot{m}} = 10^{-5}$

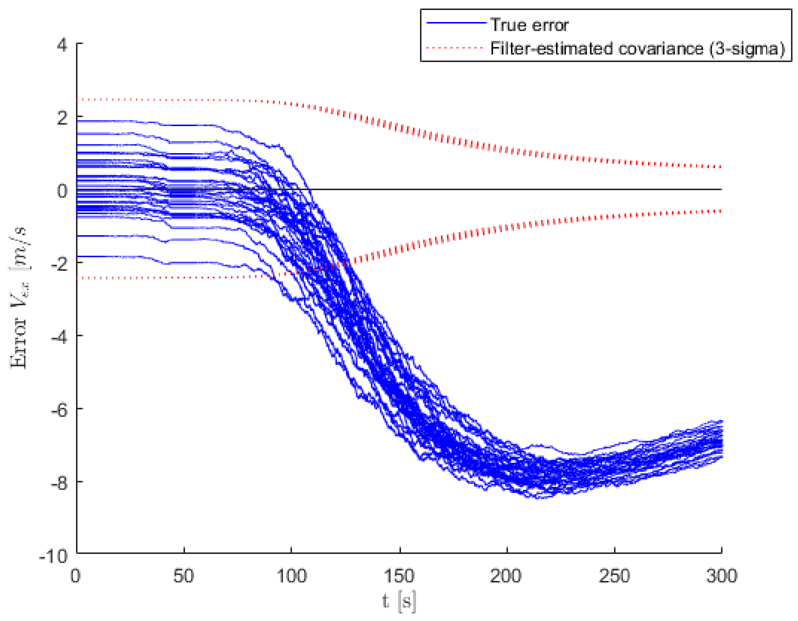


Figure 8.4: Effect of thrust drift on v_{ex} error, $\delta_{\dot{m}} = 10^{-5}$

8.2. Closed-loop guidance

While the performance of the QPEG didn't suffer as much in the FES compared to the parameter estimation, there were still some undesirable behaviours that were not present in the 3-DOF simulations. In these 3-DOF simulations it had been assumed that the other parts of the GNC system would function perfectly. This section will therefore take a closer look at what happens if there are errors either on the input or output side of the closed-loop guidance.

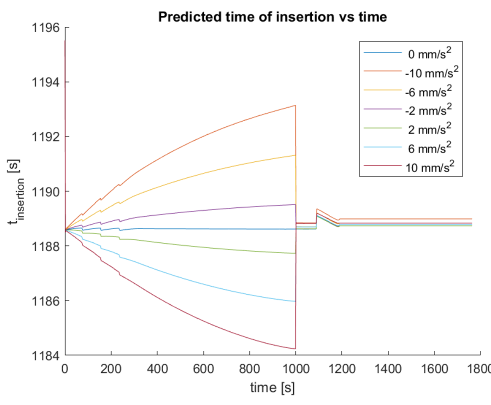
8.2.1. Navigation errors and discontinuities

The QPEG requires three main inputs in order to determine the optimal trajectory: the target constraints, the current vehicle state, and the vehicle parameters. The latter two allow it to make a prediction of the vehicle state at some future point in time. Errors in the vehicle parameters have already been extensively covered as estimating these was one of the main topics of this thesis. Thus far errors in the estimated velocity and position have been mostly neglected. However in a fully nominal flight these errors are likely to remain insignificant from a guidance perspective with expected errors at insertion being less then $1\,m$ in terms of position and $0.1\ m/s$ in terms of velocity. This is due to the hybrid navigation system which corrects these errors using regular GNSS measurements [50]. During the first and second stages the velocity errors will be a bit higher due to the vibrations induced by the engines. Even though these errors are included in the FES, their effect will be much smaller compared to other perturbations. However in the event that the GNSS signal is lost during flight the navigation reverts back to pure inertial navigation which relies only on the IMU. The rocket position and velocity errors will then start growing due to errors originating for example from the sensor biases or the attitude. Another source of concern is what happens when the GNSS signal is reacquired and the estimated states almost instantly snap back to the true state. If this happens close to insertion the guidance could command some excessively large and risky manoeuvres to correct the errors. It should however be noted that the scenarios simulated in this section were designed specifically to find the limits of the QPEG and are very unlikely to occur.

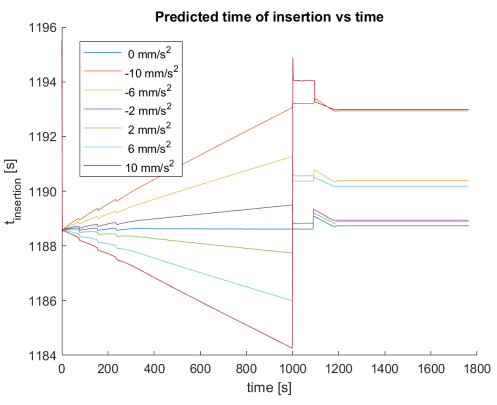
In order to simulate long GNSS dropouts for the guidance in the 3-DOF test environment, a constant acceleration bias was applied in a local-level frame aligned with the trajectory. This results in a linear drift for the velocity and quadratic drift for the position error. A bias on the order of $10\ mm/s^2$ was determined to be representative of the level of errors, which over the course of $1000\ s$ will result in a $10\ m/s$ velocity error and $5\ km$ position error. The GNSS dropout is simulated from the start of the second stage until some predetermined time close to insertion. If the GNSS signal is not reacquired before insertion the navigation error would roughly translate to an error in the final orbit insertion accuracy.

Combinations of three different GNSS re-acquisition times ($800\ s$, $1000\ s$, $1100\ s$) were tested for errors in either the radial, along-track, or cross-track. With a nominal insertion time of $1189\ s$ and the altitude constraint being released $100\ s$ before insertion, this will cover the reaction of the QPEG for re-acquisition both before and after the altitude constraint release. Given the large number of test combinations and associated results only the most interesting ones will be shown.

During the GNSS dropout the most visible effect is on the predicted time of insertion, as shown in Figure 8.5. For both radial and along-track cases the errors grows quasi-linearly to a couple seconds . Given the similar magnitude of the effect for both directions, it seems like the velocity error plays a more dominant role compared to the position error. For cross-track error, the effect on $t_{insertion}$ is more than an order of magnitude smaller. After the GNSS signal is re-acquired the QPEG re-converges to the correct solution with a small number of iterations, which means that $t_{insertion}$ now gives an accurate prediction. For the along-track case the accumulated performance penalty is at most less than a second, but it primarily affects the cases with a negative drift, i.e. where the velocity was under-estimated. For the cases where the velocity was over-estimated the performance penalty was negligible. For the radial case the performance penalty is a bit more substantial and is roughly of equal magnitude to the error in the predicted time of insertion during the dropout. Again cases with a positive drift seem to fare better than the negative ones. So even if the GNSS signal is lost for a limited amount of time during the ascent it can have a lasting effect.



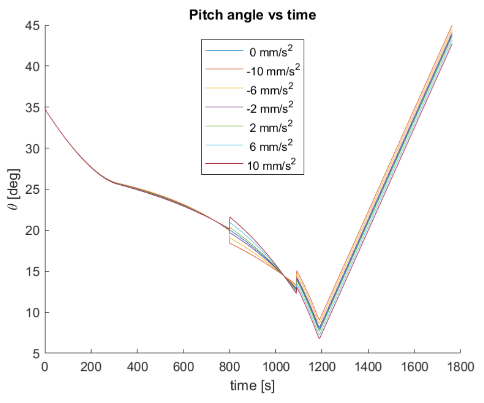
(a) Along track



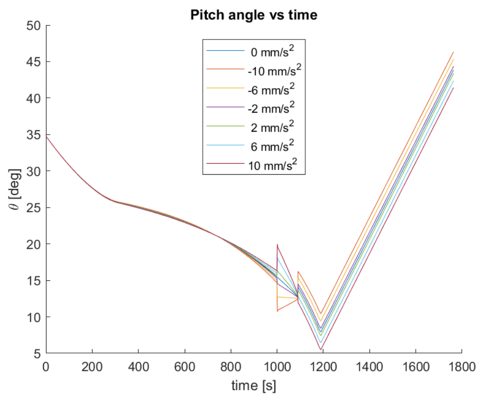
(b) Radial

Figure 8.5: Effect of navigation error on $t_{insertion}$, $t_{GNSS} = 1000\ s$

In terms of commanded attitude the effect during the dropout is negligible for the along-track and cross-track cases, as shown in Figures 8.6 and 8.7 respectively. This is consistent with the tiny performance losses that were observed for them. On the other hand for the radial case, shown in Figure 8.8, the commanded pitch angle starts to diverge from the nominal one. Once the navigation error snaps back to zero, the new guidance solution found by the QPEG can result in quite a large jump in pitch angle. This jump is largest for the radial error cases and also becomes larger the closer it is to the time of insertion. The latter can be explained by the fact that it has less time to correct for the error, so the amplitude of the corrective actions need to be larger. Meanwhile the former is likely because the radial position, which is the altitude, is constrained whereas the along-track, or down-range, position is not. Given that this jump seems to be proportional to the navigation error, it could be expected that much larger errors would result in larger jumps as the QPEG tries to force the vehicle to the originally targeted orbit. In such a case it might be more advantageous to abandon the original target and circularize at the currently optimal altitude. Optimal insertion into the original orbit would likely require adding a coast phase.

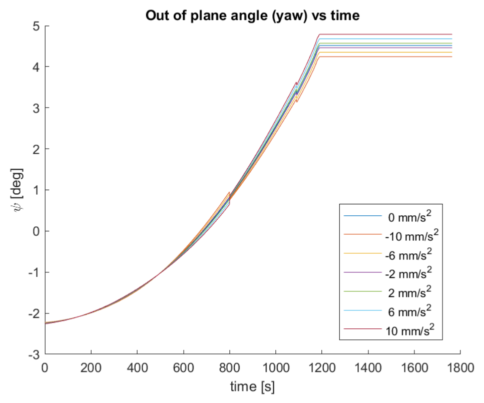


(a) $t_{GNSS} = 800\ s$

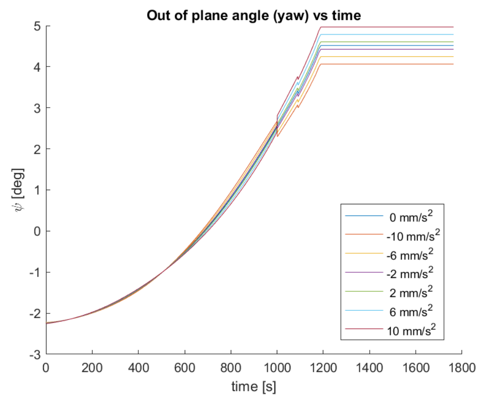


(b) $t_{GNSS} = 1000\ s$

Figure 8.6: Effect of along-track navigation error on pitch angle



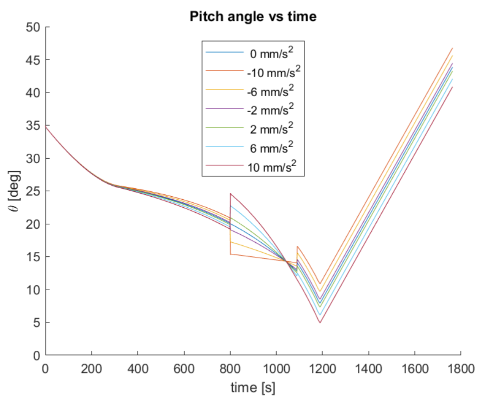
(a) $t_{GNSS} = 800\ s$



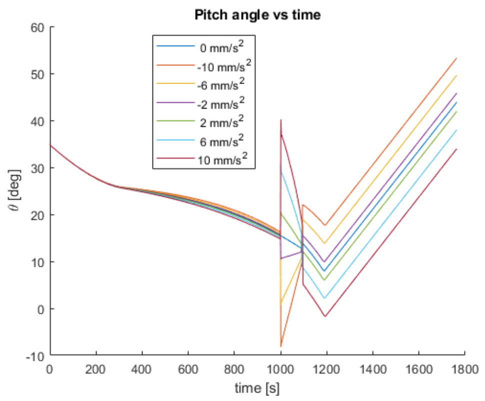
(b) $t_{GNSS} = 1000\ s$

Figure 8.7: Effect of cross-track navigation error on yaw angle

Another interesting aspect is the impact on the insertion accuracy if the GNSS signal is only obtained after the altitude constraint has been released. This can be seen in the altitude profiles for radial navigation errors in Figure 8.9. When t_{GNSS} is before the constraint switch, QPEG forces the rocket back to the target altitude, which is why the jump in pitch is so large. If however t_{GNSS} is after the switch the rocket will more or less stay at that altitude, with the target velocity being adjusted to maintain the same semi-major axis. The impact on the eccentricity at insertion is displayed in Figure 8.10b, which shows that a larger drift in general leads to larger eccentricity. For reference Figure 8.10a shows the case where the altitude error is corrected before insertion, with the remaining eccentricity error being caused by the time-step of the simulation. In a way the altitude constraint release already has the effect of automatically changing the target orbit to a more optimal one to avoid performing a huge correction manoeuvre in the last couple of minutes before insertion.

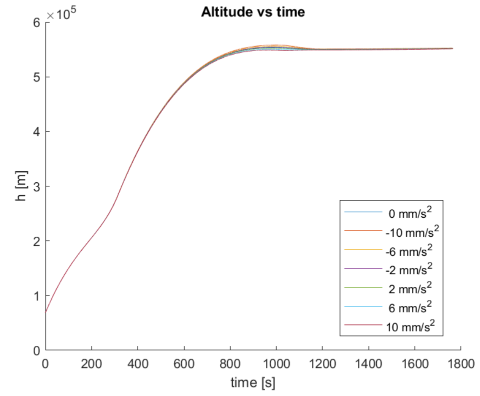


(a) $t_{GNSS} = 800\ s$

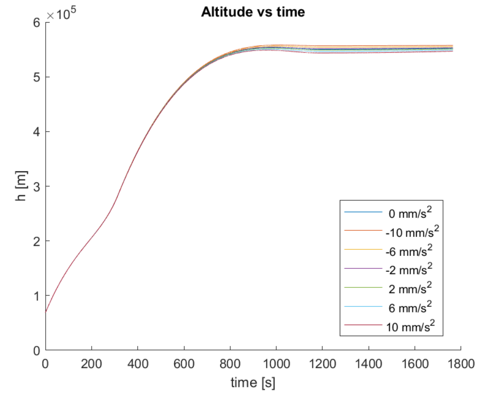


(b) $t_{GNSS} = 1000\ s$

Figure 8.8: Effect of radial navigation error on pitch angle

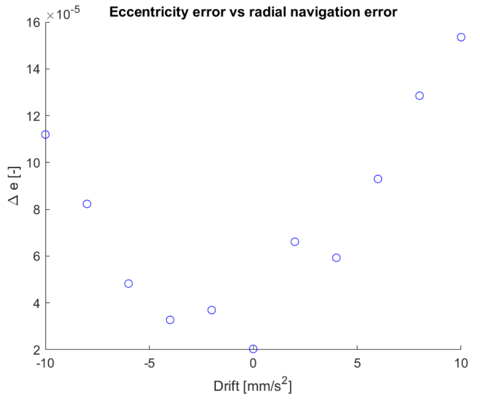


(a) $t_{GNSS} = 1000\ s$

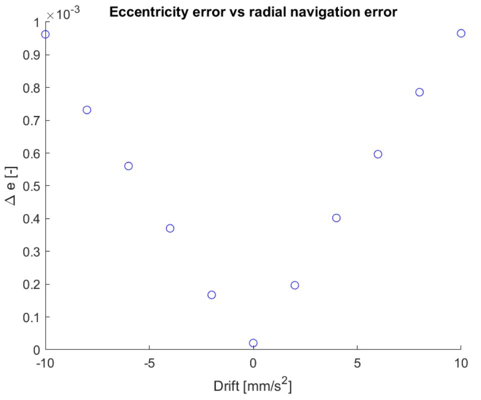


(b) $t_{GNSS} = 1100\ s$

Figure 8.9: Effect of radial navigation error on altitude



(a) $t_{GNSS} = 1000\ s$



(b) $t_{GNSS} = 1100\ s$

Figure 8.10: Effect of radial navigation error on eccentricity at insertion

8.2.2. Thrust misalignment

One of the simplifications that has been made thus far in the isolated guidance environments is that the thrust force was always perfectly aligned with the commanded thrust direction. One of the main reasons for this is that the guidance currently lacks any sort of direct correction for a thrust misalignment. Furthermore while the estimation of the thrust angles with respect to the body frames could at some point be part of the vehicle parameter estimator, the actual correction would likely happen in the roll-program component, where the commanded attitude is constructed, which is considered outside the scope of this thesis. Of course thrust misalignment was already part of the many perturbations in the FES, presented in Section 6.4, but it is difficult to pin certain behaviours to specific causes with such a wide range of uncertainties. Therefore this section will take a deeper look into the effects on the guidance caused by thrust misalignment if it is not compensated.

First we need to more precisely define what is meant by a thrust misalignment. The current guidance algorithm makes the assumptions that the body x-axis is aligned with the thrust vector when constructing the commanded attitude. This assumption would be valid under some ideal conditions. There are however many reasons why this would not be the case. The first obvious reason is if the vehicle is not perfectly axisymmetric throughout the entire flight, of which the Space Shuttle is a prime example. On RFA ONE this is also the case to some extent due to an asymmetric tank configuration on the third stage. The Space Shuttle guidance algorithm therefore took the estimated thrust direction into account when constructing the commanded attitude [7]. Even if the vehicle is fully symmetric by design, manufacturing and assembly tolerances could still result in the CoM or the engine gimbal point being off-center, which would result in the controller converging to a non-zero steady-state deflection for the TVC. In addition to that a misalignment between stages or between the IMU and the rest of the vehicle will contribute to the problem.

For this sensitivity analysis the quasi-6-DOF simulator for the entire guidance system was used, since it offered a convenient way of including the thrust misalignment. This means that both the attitude and angular rate will be continuous due to the steering filter even without the presence of the full 6-DOF dynamics. The roll angle was set to zero such that rotations around the vehicle's y- and z-axes would correspond to the local vertical and horizontal planes respectively. A constant misalignment was applied throughout the whole flight around either the body y- or z-axes. While this means that the results combine the effects of a misalignment during the second and third stages, which would normally be independent, the separate contributions can still be identified and using a non-constant misalignment would have added quite some complexity. The tests were performed using an otherwise nominal vehicle configuration for the baseline mission scenario with only the part of the flight with closed-loop guidance being simulated.

In the first test the thrust misalignment was varied in a range from $-5°$ to $+5°$ around the body y-axis, which corresponds to the pitch or vertical plane. The angles given represent the rotation from the engine reference frame to the body reference frame. The effects of these misalignments on the pitch angle throughout the flight can be seen in Figure 8.11. As expected the initial pitch angles are unaffected by the thrust misalignment, since it hasn't had an impact on position and velocity yet. However the pitch angles start to diverge quite rapidly during the second stage as the closed-loop guidance adjusts its attitude command in a direction opposite to the thrust deflection in order to compensate for the slow drift in position and velocity from the initially predicted trajectory. The yaw angle on the other hand is not impacted. A maximum in this divergence is reached at the end of the second stage, with the pitch angles converging back to the nominal one during the third stage. However the most interesting difference occurs when the altitude constraint is released. For the nominal case it was previously observed that the change in constraints produced a small jump in the pitch angle, which in this test is almost entirely smoothed out by the steering filter. However it seems that in case of a thrust misalignment this jump is greatly amplified, which is similar to what was seen for some of the results from the FES. This suggests that thrust misalignments in the pitch plane affect the trajectory in such a way that the altitude constraint significantly impacts the optimal pitch angle.

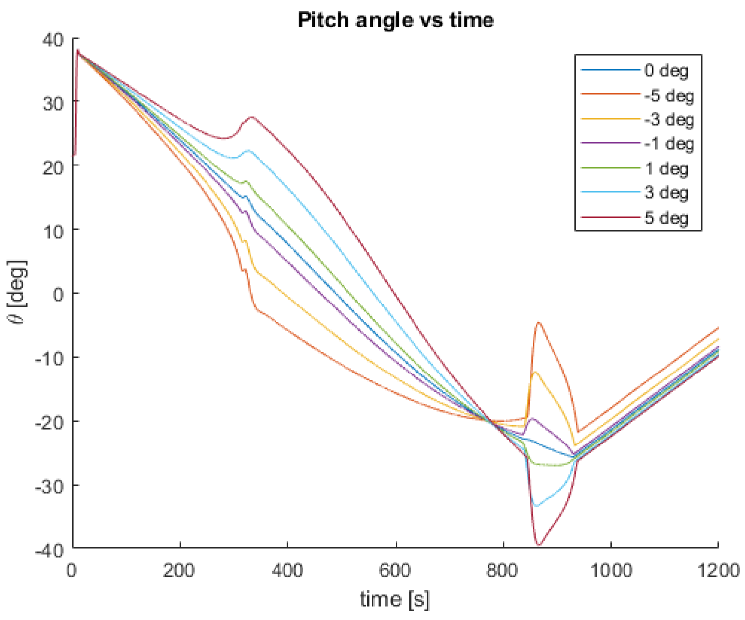


Figure 8.11: Effect of vertical thrust misalignment on pitch

Another variable which is strongly affected is the predicted time of insertion, shown in Figure 8.12. Given that it is usually a rather accurate prediction, as demonstrated by the nominal case, it can serve as a good indicator of the accumulated performance penalty. As expected most of the performance loss occurs during the second stage due to its much higher TWR, which in a more realistic scenario would be compensated by a smaller thrust misalignment. Furthermore it can be observed that the altitude constraint release is also paired with a decrease in the t_{go}, which again suggests that the deviation from the nominal trajectory means that the altitude constraint causes a larger performance impact. Another notable discrepancy is the difference between the positive and negative angles, with negative angles having a slightly larger impact on performance.

Finally the time of insertion, which is equal to the total burn-time, has been plotted against the thrust deflections in Figure 8.13a. The relationship between the two is close to quadratic for the relatively small angles used in this test. It is therefore not surprising that when plotted against the cosine of the deflection angle in Figure 8.13b, which is proportional to the thrust in the body x direction, the relationship is almost perfectly linear. The asymmetry between positive and negative angles is however again visible.

In a second test, thrust deflections in a range of $-5°$ to $+5°$ were applied around the body z-axis. Due to the zero roll angle this means the deflections are purely in the lateral plane. The effects on the yaw-angle can be seen in Figure 8.14, which has some similarities to the pitch angle in the previous test. However the change of the yaw angle compared to the nominal trajectory is smaller, and there are no discontinuities when the altitude constraint is released since the lateral constraints remain unchanged. It should be noted that similarly to the previous test, the effect on the perpendicular angle is negligible.

When looking at the total time to insertion, shown in Figure 8.15, the effects of lateral deflections are almost identical to vertical deflections. The main difference is that now there is barely any difference between the negative and positive sides. Furthermore the performance loss seems to follow the average of the two sides in Figure 8.12.

Again the total burn-time has been plotted against be thrust misalignment angles. As can be seen in Figure 8.16 the impact on performance follows the same trends as it did for vertical

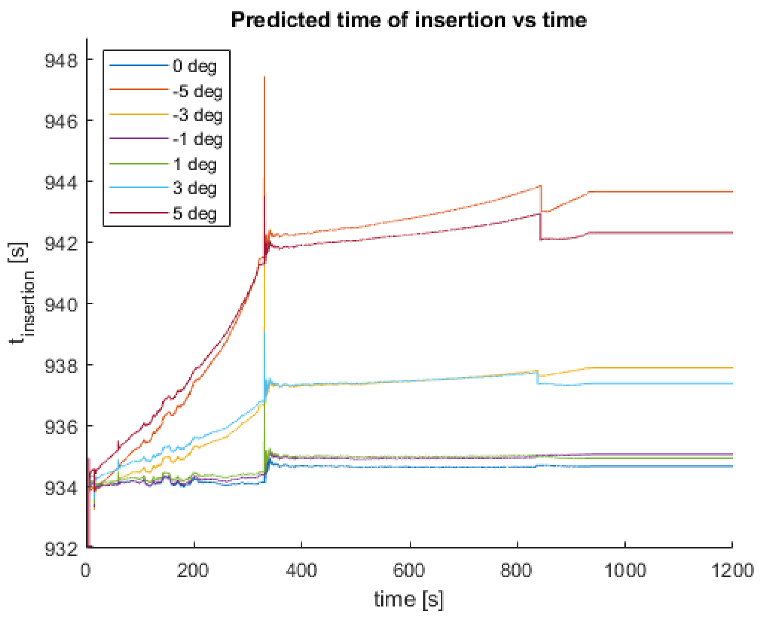
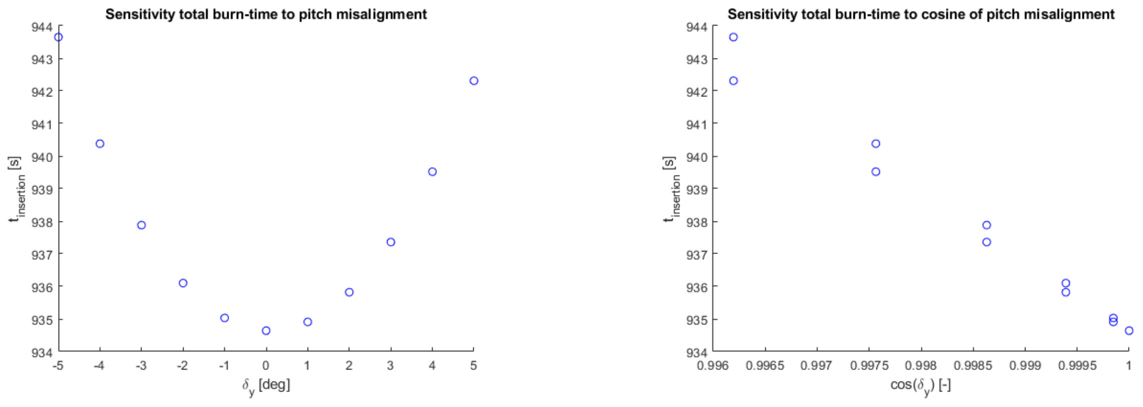


Figure 8.12: Effect of vertical thrust misalignment on predicted insertion time



(a) Angle (b) Cosine of angle

Figure 8.13: Sensitivity of total burn-time to vertical thrust misalignment

deflections, with the losses now being seemingly independent of the direction.

To conclude it has been shown that thrust misalignments that are not directly corrected by the guidance will still be indirectly corrected due to their influence on the position and velocity. The delay in its effects does however cause some performance penalties, which are mostly independent of the direction of the deflection. The thrust deflection primarily has an impact on the attitude angle in the same plane, with little to no effect on the perpendicular plane. There are however some significant differences in the effects on attitude angles between vertical and lateral deflections, as vertical deflections cause an undesirable discontinuity when the altitude constraint is released. It should be noted that this analysis only considered relatively small angles and that larger angles might cause some higher order effects.

8.2.3. Gravity model

Another simplification that was made for the isolated guidance simulation environments was to use a spherical gravity model. In this way it would match the gravity model used inside the

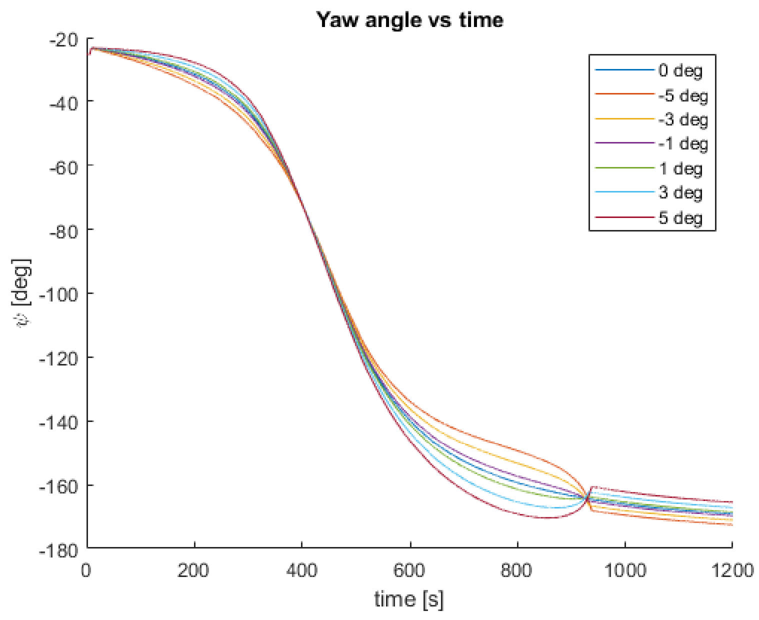


Figure 8.14: Effect of lateral thrust misalignment on yaw

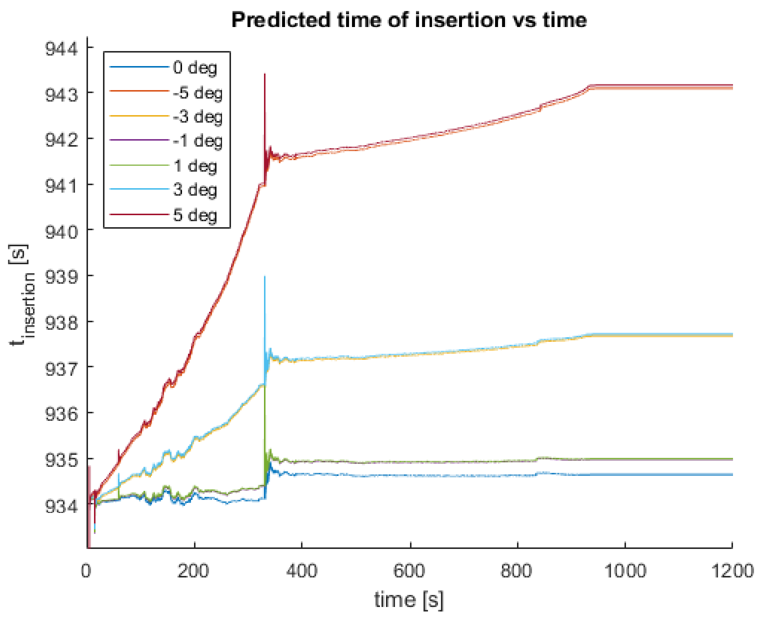
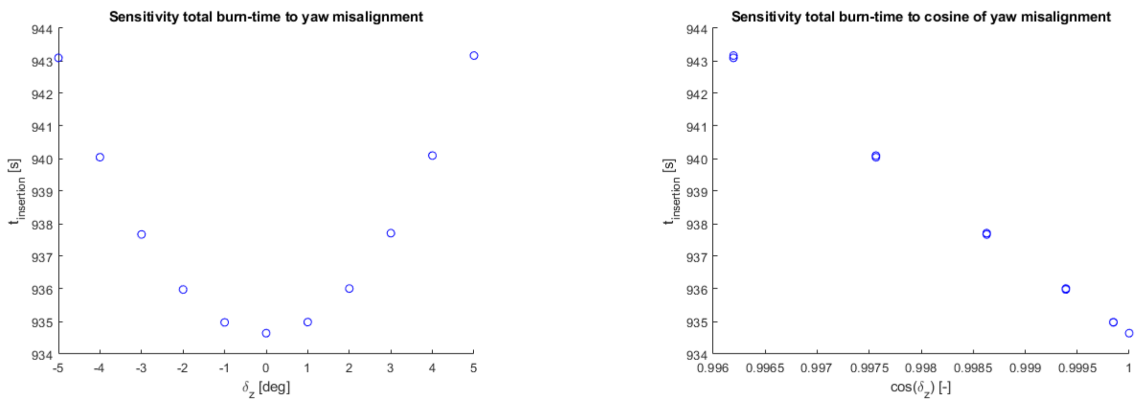


Figure 8.15: Effect of lateral thrust misalignment on predicted insertion time

QPEG. Of course the FES uses a more accurate gravity model, but its effects are difficult to isolate from all the other perturbations. Therefore the impact of a more precise gravity model was assessed in the 3-DOF simulator for the guidance. The WGS84-based model was used for this test since the ellipsoidal gravity should account for the majority of the inaccuracies. The effects on the pitch angle are shown in Figure 8.17. The change during the second stage is negligible, likely due to the thrust being the dominant force. However during the third stage, where the thrust is much smaller than the gravity, the pitch angle with the WGS84-based model is lower. This is caused by the lower gravity experienced near the poles during a polar launch, which means that less vertical thrust is required to reach the desired altitude. It is



(a) Angle (b) Cosine of angle

Figure 8.16: Sensitivity of total burn-time to lateral thrust misalignment

also interesting to see that the discontinuity in pitch angle at the altitude constraint release just before orbit insertion is amplified compared to the spherical gravity case. This is similar to what was observed for thrust misalignments in the vertical plane, which suggests that any prediction error for the vertical motion is likely to cause this phenomenon.

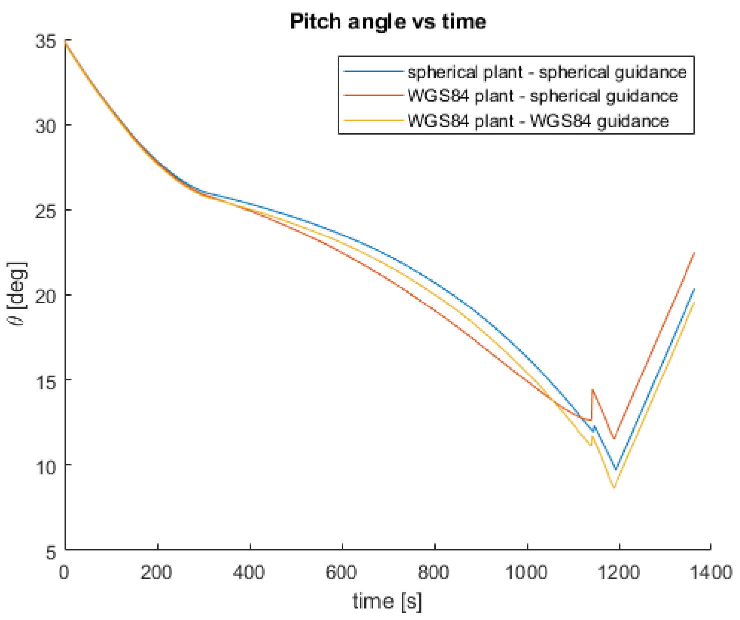


Figure 8.17: Effect of gravity model on pitch angle

The predicted time of insertion, shown in Figure 8.18, is also interesting to look at. During the second stage the differences are again negligible, however shortly after the third stage ignition the predicted time of insertion starts to decrease at a rather constant rate. This means that in its current form the QPEG is going to have a bias towards overestimating the total required burn-time for polar launches from high latitudes, while it would likely underestimate it for equatorial launches from low latitudes. Furthermore it is likely that this contributes to the offset in the semi-major axis shown in Figure 6.38, since the small error in the gravity integral accumulated during the final guidance parameter hold will be unaccounted for. In order to mitigate this the QPEG can be modified to include the improved gravity model. In the numerical propagator this would be as simple as re-using the same formulation used in the plant model, with just the gravity constant having to be normalized correctly. While the analytical propagator could also be adapted, this is unnecessary as the constraints bias is sufficient. This modification was implemented as part of the comparison to the reference trajectory discussed in Section 7.3, and the result can also be seen in Figures 8.17 and 8.18. The main advantage is the improved predicted time of insertion, which is again constant throughout the flight. Additionally there is also a small improvement in terms of performance and the pitch discontinuity when switching constraints is diminished.

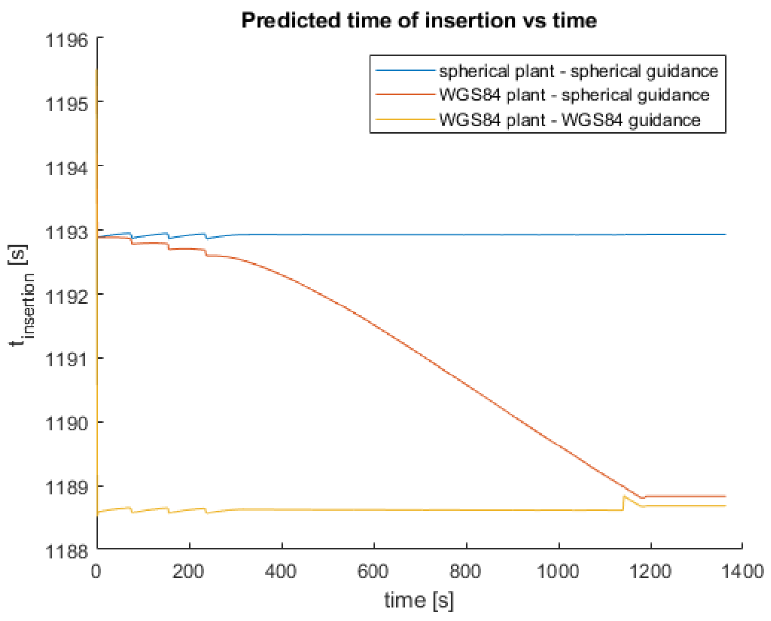


Figure 8.18: Effect of gravity model on predicted insertion time

9 Conclusion and recommendations

The aim of this research was to improve the existing exo-atmospheric closed loop guidance system at RFA by increasing the probability that it will successfully guide the rocket to orbit insertion in all cases where reaching orbit should be feasible. Throughout this report several improvements to the existing algorithms were developed and analyzed to determine their efficacy. The main conclusions on the performance improvements will be presented in Section 9.1 and some recommendations based on identified deficiencies will be given in Section 9.2.

9.1. Conclusions

The main research question raised at the start of this thesis was:

What is required to create a robust exo-atmospheric closed-loop guidance system that can handle large parameter uncertainties and loose orbit insertion requirements?

In order to answer this question three components of the closed-loop guidance were studied in great detail. The first one solves the optimal control trajectory problem in order to determine the desired attitude commands. The second was an algorithm to estimate during the flight the vehicle performance parameter relevant to the guidance. The last component combined the information provided by the other two components to select the optimal target orbit. From this we formulated three research sub-questions, each focusing on one of these components.

Sub-question 1: *How can the convergence be improved for long thrust arcs?*

One of the primary issues with the existing closed-loop guidance algorithm was that it sometimes failed to converge to a solution, which would have catastrophic consequences for the rocket. The reason for this was that it used an algorithm, called the Powered Explicit Guidance (PEG), which was originally developed for the Space Shuttle that made several assumptions to allow it to run on the computers of the time. Due to the short thrust arcs involved (15°) the curvature of the Earth could be neglected in the derivation of the control law and a simple predictor corrector scheme offered rapid convergence. When applied to RFA ONE however these assumptions broke down due to the long thrust arcs (60°) and the low TWR of its third stage. Some mitigations to these issues had already been developed for SLS, but they were not entirely suitable for RFA ONE. Therefore a new closed-loop guidance algorithm was designed to solve the problem at a more fundamental level, called the Quadratic Powered Explicit Guidance (QPEG). It combines multiple aspects found in recent literature and some new features were developed as part of this thesis.

It was shown that not only is the QPEG capable of converging to a solution for a much wider range of initial conditions, but it also offers some small performance benefits. The QPEG can rapidly converge to a solution for all nominal flight scenarios as well as for scenarios with reduced performance due to a single engine out on the first stage. While there still exist conditions where a solution cannot be obtained, those correspond to burn-times which are infeasible for RFA ONE and would therefore be of little use. Most of the improvements in terms of convergence are due to the algorithm now being formulated as a standard root-finding problem, which can be solved using robust algorithms like Powell's trust-region dogleg method. The use of a multiple-shooting formulation with a node placed at the end of the second stage instead of a single-shooting method further improved convergence. In the event of non-convergence the fall-back to previous solutions can now largely prevent undesirable consequences.

Another important point is that the improvements in convergence could not come at the expense of a loss in performance. It has been shown that the QPEG offers a minor reduction of burn-time compared to the PEG. Furthermore it was found that the solution obtained using the QPEG is functionally identical to the reference trajectory created using an industry-proven general trajectory optimizer for the ascent mission studied in this thesis. The linear gravity compensation term proved to be essential in obtaining the optimal trajectory, while other aspects such as constraint biasing and gravity averaging had little to no impact on total burn-time.

A final benefit of the closed-loop guidance is the auxiliary data it can provide in addition to the attitude commands. The predicted time to insertion was of particular interest as it forms a core part of the target selection algorithm. One of the issues of the PEG was that with its baseline implementation it would consistently underestimate the time to insertion and most of the methods to improve its convergence would worsen these estimates. The QPEG however is able to provide accurate predictions even with its analytical propagator and if constraint biasing is used the prediction errors vanish entirely on the condition that the input parameters are accurate and no perturbation, such as thrust misalignment or navigation error, is present.

Sub-question 2: *How can the rocket performance parameters be estimated in-flight?*

The second major problem with the guidance that this thesis aimed to solve was the estimation of the rocket parameters used by the closed-loop guidance. The relevant parameters are the engine exhaust velocities v_{ex}, the mass to mass-flow rate ratios $\tau = \frac{m}{\dot{m}}$, and the stage burn-times t_b. Together these three parameters uniquely define the acceleration profile for each stage. The existing system did not perform any in-flight estimation and simply used the nominal values loaded into the flight computer. The consequence was that any uncertainty in the thrust or mass parameters would make the predictions of the closed-loop guidance inaccurate and therefore lead to a sub-optimal trajectory. Of course any errors would eventually be compensated so there was little impact on insertion accuracy, but the required total burn-time was still increased. For this thesis it was decided to implement a very simple parameter estimation algorithm that only relied on the acceleration measured by the IMU. A linear Kalman filter was then used to compute estimates of v_{ex} and τ, with estimates for t_b obtained using its correlation with τ. The residual amount of drag at the start of the second stage was compensated using a simple exponential model.

It was shown that under ideal conditions the parameter estimator could rapidly reduce the initial error in τ and throughout the stage flight estimates for both v_{ex} and τ would be further refined. Providing these improved estimates to the QPEG also yielded on average significant performance improvements, especially for the most off-nominal cases. There were however some configuration close to the nominal one which saw a slight decrease in performance compared to using the nominal parameters, which was attributed to the larger estimation error for t_b. It was also shown that using appropriate tuning the parameter estimator could handle some perturbations like RCS gas mass consumption, although other perturbations like a slow drift in the thrust caused unavoidable minor errors.

When used in the FES, the performance of the parameter estimator was significantly degraded due to all the additional perturbations. In particular high angular rates or accelerations would cause the estimated states to diverge significantly. Even for rather nominal flights the errors in the states would be underestimated by the filter. Nevertheless the errors in the estimated parameters were still smaller compared to just using the nominal values, especially for the instantaneous acceleration. The predicted burn-time margins obtained from combining the previously mentioned predicted time to insertion with the estimated total burn-time was found to be reliable and constant indicator of the performance margins. The main limiting factor on its accuracy was the error in the burn-time due to it not being directly observable using just acceleration measurements.

Sub-question 3: *How should the guidance system compensate for an under-performing rocket?*

The last major improvement developed as part of this thesis was a system that would allow the guidance to autonomously change the target orbit if it determined that reaching the nominal orbit isn't feasible. The main difficulty here was that due to the small performance margins and high vehicle parameter uncertainties this wouldn't just be required for off-nominal scenarios, such as a first-stage engine out, but might also be used on a seemingly nominal flight. The algorithm that was implemented for this task in some ways resembles the engine out logic used for SLS. In particular they both make use of a ranked list of specific target orbits. In this way the specification of target orbits can be done during pre-flight mission analysis using arbitrarily complex criteria. However instead of using velocity thresholds at discrete events, such as engine outs, the target selection algorithm used for this thesis continuously monitors the aforementioned predicted burn-time margin. If it falls below a predefined minimum threshold it iterates through the list of targets to find the best feasible orbit. Furthermore this search can be re-triggered throughout the flight if the predicted margin changes substantially. As soon as the target is changed the QPEG is able to re-converge without any issues.

It was demonstrated that this algorithm was flexible enough to adapt to missions ranging from one target to eight targets; however in theory nothing prevents the algorithm from using even more targets, the limiting factor being the computational time per guidance cycle. It is capable of both down-selecting at any point in time if the predicted margin suddenly decreases, but it can also go back to a higher target altitude if the margin increases sufficiently. Furthermore the configurable margin thresholds allow the guidance to be more opportunistic or conservative based on the mission requirements and it can also allow reserving propellant for in-orbit operations, e.g. for a de-orbit manoeuvre. It should be noted that this algorithm is especially suited to maximizing the probability of reaching one of the predetermined orbits. If maximizing the final orbit altitude is the goal, then this algorithm can sometimes lead to sub-optimal results. It was tested for a Monte-Carlo batch with single engine outs on the first stage; although the probability of reaching a stable orbit was increased, the rocket still failed to reach orbit for a small number of cases.

9.2. Recommendations

During this research numerous points of improvement were identified. Many of them were not further explored either because they fell outside the scope of this thesis or due to time constraints. Some of them will be addressed at some future point in the development of the guidance system at RFA. The most interesting ones will be presented below.

Coast arc

The QPEG presented here can only be used for missions with a fixed coast duration. However this limits its flexibility to adapt to unforeseen circumstances. The primary reason it wasn't further explored is that the baseline mission used in this thesis didn't feature any coast arcs, even though it might be more efficient to use one. It was highlighted by Lu that the performance is not very sensitive to the coast duration for short coast arcs [24], but for long coast arcs it might become significant. There are several ways of implementing variable length coast arcs, such as integrating them in the root-finding problem with an additional shooting node [24] or using a one-dimensional optimizer in an outer loop [26].

Altitude constraint release

The release of the altitude constraint is currently based on the remaining time to insertion. While this is adequate if an accurate insertion is always required, it could lead to large correction manoeuvres in case there is a large jump in the navigation state close to insertion with the altitude constraint still active. While it might be simple to mitigate this by increasing the threshold for the constraint release, this will negatively impact the orbit insertion accuracy for nominal scenarios. Ideally the altitude constraint would only get released early if a jump is detected and a large attitude change would be required to correct for the accumulated error.

Thrust angle

Currently the parameter estimator uses the norm of the sensed acceleration as its measurement. However this can lead to a bias in the measurement if the noise or perturbations are significant in the directions normal to the thrust. A solution for this would be to use the full acceleration vector with the thrust angles either being provided externally or added as new states to the filter. Estimating these thrust angles would also allow the guidance to correct for a thrust misalignment.

Vibration

Another aspect that was largely ignored in the development of the parameter estimation algorithm is the effect of large vibrations. As was previously noted if the sample frequency of the IMU velocity increment is large compared to the lower bound of the vibration spectrum, the effective noise level would make the current implementation of the Kalman filter diverge completely. This could be resolved by first passing the acceleration measurement through a low-pass filter or adding additional states to the Kalman filter.

Propellant level sensing

As was previously noted the primary limiting factor on the accuracy of the predicted margins, is the estimation error for the burn-time. This impacts both the optimality of the solutions found by the QPEG during the second stage and the target selection algorithm, which requires larger minimum margins to ensure orbit insertion. This issue stems from the inability to differentiate between inert or propellant mass using just an acceleration measurement. Furthermore the fact that one propellant might run out before the other is currently also not taken into account. In order to resolve these issues additional sensors are required to provide information on the amount of remaining propellant in the tanks.

Fast pre-selection of feasible orbits

While the target selection algorithm could in theory use a very large list of targets, in practice it is limited by the amount of onboard computational power. The limiting factor is that the decision criterion, i.e. estimated margin, requires a converged solution from the QPEG. It might however be more efficient if a pre-selection of targets is made by using a simpler algorithm to eliminate grossly infeasible targets. This could be achieved using for example a simplified less accurate version of the QPEG or even by assuming an impulsive transfer.

Maximize periapsis

The target selection algorithm presented in this thesis is most suited to maximizing the probability of inserting into one of the predefined target orbits. While this can be useful to ensure fulfillment of complex operational constraints it also implies that the minimum margins need to be set sufficiently high to account for estimation errors. If however insertion into one of these orbits is not strictly required and the minimum margins are decreased, a target change can sometimes be counter-productive. For example if the goal is to maximize the probability of insertion into a stable orbit by maximizing the periapsis altitude, a different decision criterion would have to be developed. The state at the predicted propellant depletion could be obtained from the QPEG, from which the predicted periapsis altitude at depletion can be computed.

Bibliography

[1] Doris C. Chandler and Isaac E. Smith. “Development of the iterative guidance mode with its application to various vehicles and missions.” In: *Journal of Spacecraft and Rockets* 4.7 (July 1967), pp. 898–903. issn: 0022-4650. doi: 10.2514/3.28985.

[2] Timothy J. Brand, Dennis W. Brown, and John P. Higgins. *Space Shuttle GN&C Equation Document No. 24 Unified Powered Flight Guidance*. Tech. rep. NASA, June 1973. url: https://ntrs.nasa.gov/citations/19740004402.

[3] Paul Von Der Porten et al. “Powered Explicit Guidance modifications & enhancements for the Space Launch System Block-1 and Block-1B vehicles”. In: *AAS GNC (Guidance, Navigation, and Control) Conference*. 2018. url: https://ntrs.nasa.gov/citations/20180002035.

[4] Nick Stein. “Modern development of an Ascent Guidance Program for a Multistage Orbital Launch Vehicle”. MA thesis. 2021.

[5] Ping Lu. *What is guidance?* 2021. doi: 10.2514/1.G006191.

[6] R. Jaggers. “An explicit solution to the exoatmospheric powered flight guidance and trajectory optimization problem for rocket propelled vehicles”. In: *Guidance and Control Conference*. Reston, Virigina: American Institute of Aeronautics and Astronautics, Aug. 1977. doi: 10.2514/6.1977-1051.

[7] R. L. McHenry et al. “Space Shuttle Ascent Guidance, Navigation, and Control”. In: *J Astronaut Sci* 27.1 (1979). issn: 00219142.

[8] Frank M Perkins. *Derivation of linear-tangent steering laws*. Tech. rep. Aerospace Corporation, Nov. 1966. url: https://apps.dtic.mil/sti/citations/AD0643209.

[9] Naeem Ahmad et al. “Closed loop guidance trade study for Space Launch System Block-1B vehicle”. In: *AAS/AIAA Astrodynamics Specialist Conference*. Aug. 2018. url: https://ntrs.nasa.gov/citations/20180006370.

[10] Matt Hawkins and Naeem Ahmad. “Guidance Modifications and Enhancements for Space Launch System Block-1 in Support of Artemis I and Beyond”. In: *AAS/AIAA Astrodynamics Specialist Conference*. 2020. url: https://ntrs.nasa.gov/citations/20205004411.

[11] A S Craig et al. “Space Launch System Engine Out Capabilities”. In: *AAS/AIAA Astrodynamics Specialist Conference*. 2020. url: https://ntrs.nasa.gov/citations/20205004525.

[12] Binfeng Pan and Ping Lu. “Improvements to optimal launch ascent guidance”. In: *AIAA Guidance, Navigation, and Control Conference*. 2010. isbn: 9781600869624. doi: 10.2514/6.2010-8174.

[13] Greg A Dukeman. “Closed-Loop Nominal and Abort Atmospheric Ascent Guidance for Rocket-Powered Launch Vehicles”. PhD thesis. Georgia Institute of Technology, May 2005.

[14] Ping Lu. “Entry guidance: A unified Method”. In: *Journal of Guidance, Control, and Dynamics* 37.3 (2014), pp. 713–728. issn: 15333884. doi: 10.2514/1.62605.

[15] Xinfu Liu, Zuojun Shen, and Ping Lu. “Entry Trajectory Optimization by Second-Order Cone Programming”. In: *Journal of Guidance, Control, and Dynamics* 39.2 (2016), pp. 227–241. issn: 15333884. doi: `10.2514/1.G001210`.

[16] Morgan C. Baldwin and Ping Lu. “Optimal deorbit guidance”. In: *Journal of Guidance, Control, and Dynamics* 35.1 (2012), pp. 93–103. issn: 15333884. doi: `10.2514/1.53937`.

[17] Behçet Açikmeşe and Scott R. Ploen. “Convex programming approach to powered descent guidance for mars landing”. In: *Journal of Guidance, Control, and Dynamics* 30.5 (Sept. 2007), pp. 1353–1366. issn: 15333884. doi: `10.2514/1.27553`.

[18] Binfeng Pan, Ping Lu, and Zheng Chen. “Coast arcs in optimal multiburn orbital transfers”. In: *Journal of Guidance, Control, and Dynamics* 35.2 (2012), pp. 451–461. issn: 15333884. doi: `10.2514/1.54655`.

[19] Stephen W. Thrasher and Thomas J. Fill. “Orion’s exoatmospheric burn guidance architecture and algorithm”. In: *AIAA Guidance, Navigation, and Control Conference 2011*. 2011. isbn: 9781600869525. doi: `10.2514/6.2011-6262`.

[20] Sara K Scarritt, Thomas Fill, and Shane Robinson. “Advances in Orion’s on-orbit guidance and targeting system architecture”. In: *AAS Guidance Navigation and Control Conference*. 2015. url: `https://ntrs.nasa.gov/citations/20150001961`.

[21] M. Delporte and F. Sauvinet. “Explicit guidance law for manned spacecraft”. In: *Aerospace Design Conference*. Reston, Virigina: American Institute of Aeronautics and Astronautics, Feb. 1992. doi: `10.2514/6.1992-1145`.

[22] Greg Dukeman. “Atmospheric Ascent Guidance for Rocket-Powered Launch Vehicles”. In: *AIAA Guidance, Navigation, and Control Conference and Exhibit*. Reston, Virigina: American Institute of Aeronautics and Astronautics, Aug. 2002. isbn: 978-1-62410-108-3. doi: `10.2514/6.2002-4559`.

[23] Greg A. Dukeman and Anthony J. Calise. “Enhancements to an atmospheric ascent guidance algorithm”. In: *AIAA Guidance, Navigation, and Control Conference and Exhibit*. American Institute of Aeronautics and Astronautics Inc., 2003. isbn: 9781563479786. doi: `10.2514/6.2003-5638`.

[24] Ping Lu et al. “Rapid Optimal Multiburn Ascent Planning and Guidance”. In: *Journal of Guidance, Control, and Dynamics* 31.6 (Nov. 2008), pp. 1656–1664. issn: 0731-5090. doi: `10.2514/1.36084`.

[25] Binfeng Pan and Ping Lu. “Rapid optimization of multiburn rocket trajectories revisited”. In: *AIAA Guidance, Navigation, and Control Conference and Exhibit*. American Institute of Aeronautics and Astronautics Inc., 2009. isbn: 9781563479786. doi: `10.2514/6.2009-6105`.

[26] Ping Lu, Stephen Forbes, and Morgan Baldwin. “A versatile powered guidance algorithm”. In: *AIAA Guidance, Navigation, and Control Conference 2012*. American Institute of Aeronautics and Astronautics Inc., 2012. isbn: 9781600869389. doi: `10.2514/6.2012-4843`.

[27] Binfeng Pan et al. “Reduced transversality conditions in optimal space trajectories”. In: *Journal of Guidance, Control, and Dynamics* 36.5 (2013), pp. 1289–1300. issn: 15333884. doi: `10.2514/1.60181`.

[28] Erwin Mooij. *The motion of a vehicle in a planetary atmosphere*. Tech. rep. Delft University of Technology, Faculty of Aerospace Engineering, 1994. url: `http://resolver.tudelft.nl/uuid:e5fce5a0-7bce-4d8e-8249-e23293edbb55`.

[29] Astos Solutions GmbH. *ASTOS 9 Model Reference*. 2019.

[30] Karel F Wakker. *AE4874 Fundamentals of Astrodynamics*. 2015. isbn: 9789461864192. url: `http://resolver.tudelft.nl/uuid:3fc91471-8e47-4215-af43-718740e6694e`.

[31] James Richard Wertz. *Mission Geometry; Orbit and Constellation Design and Management*. 1st ed. Microcosm Press, 2001. isbn: 978-0-7923-7148-9.

[32] Shane B Robinson. *Spacecraft Guidance Techniques for Maximizing Mission Success*. Tech. rep. 2014. url: `https://digitalcommons.usu.edu/etd/2175`.

[33] Arthur E. Bryson and Yu-Chi Ho. *Applied Optimal Control*. Taylor & Francis, 1975. isbn: 978-0-89116-228-5.

[34] John T. Betts. “Survey of Numerical Methods for Trajectory Optimization”. In: *Journal of Guidance, Control, and Dynamics* 21.2 (Mar. 1998), pp. 193–207. issn: 0731-5090. doi: `10.2514/2.4231`.

[35] Anil V Rao. “A survey of numerical methods for optimal control”. In: *Advances in the Astronautical Sciences*. 2010.

[36] B.K. Walker and E.T. Baumgartner. “Parameter Estimation by Nonlinear Smoothing for Fault Monitoring on Rocket Engines”. In: *IFAC Proceedings Volumes* 24.6 (Sept. 1991), pp. 191–198. issn: 14746670. doi: `10.1016/s1474-6670(17)51140-4`.

[37] John P. Butas et al. “Rocket engine health monitoring using a model-based approach”. In: *37th Joint Propulsion Conference and Exhibit*. American Institute of Aeronautics and Astronautics Inc., 2001. doi: `10.2514/6.2001-3764`.

[38] R. E. Kalman. “A new approach to linear filtering and prediction problems”. In: *Journal of Fluids Engineering, Transactions of the ASME* 82.1 (Mar. 1960), pp. 35–45. issn: 1528901X. doi: `10.1115/1.3662552`.

[39] R. E. Kalman and R. S. Bucy. “New results in linear filtering and prediction theory”. In: *Journal of Fluids Engineering, Transactions of the ASME* 83.1 (Mar. 1961), pp. 95–108. issn: 1528901X. doi: `10.1115/1.3658902`.

[40] K. K. Biswas and A. K. Mahalanabis. “An Approach to Fixed-Point Smoothing Problems”. In: *IEEE Transactions on Aerospace and Electronic Systems* AES-8.5 (1972), pp. 676–682. issn: 00189251. doi: `10.1109/TAES.1972.309584`.

[41] John B. Moore. “Discrete-time fixed-lag smoothing algorithms”. In: *Automatica* 9.2 (Mar. 1973), pp. 163–173. issn: 00051098. doi: `10.1016/0005-1098(73)90071-X`.

[42] John L. Goodman. “Roland jaggers and the development of space shuttle powered explicit guidance (Peg)”. In: *AIAA Scitech 2021 Forum*. American Institute of Aeronautics and Astronautics Inc, AIAA, 2021, pp. 1–31. isbn: 9781624106095. doi: `10.2514/6.2021-2021`.

[43] Peter F. Gath and Anthony J. Calise. “Optimization of launch vehicle ascent trajectories with path constraints and coast arcs”. In: *Journal of Guidance Control and Dynamics* 24.2 (2001), pp. 296–304. issn: 07315090. doi: `10.2514/2.4712`.

[44] Edward M. Henderson and Tri X. Nguyen. “Space shuttle abort evolution”. In: *AIAA SPACE Conference and Exposition 2011*. 2011. isbn: 9781600869532. doi: `10.2514/6.2011-7245`.

[45] Robert D Legler and Floyd V Bennett. *Space Shuttle Missions Summary*. Tech. rep. 2011. url: `https://ntrs.nasa.gov/citations/20110001406`.

[46] M. Jones and U. Mortensen. *Guide to Software Verification and Validation*. Tech. rep. ESA, 1994. url: `http://microelectronics.esa.int/vhdl/pss/PSS-05-10.pdf`.

[47] Yaakov Bar-Shalom, X.-Rong Li, and Thiagalingam Kirubarajan. *Estimation with Applications to Tracking and Navigation*. New York, USA: John Wiley & Sons, Inc., 2001. isbn: 047141655X. doi: `10.1002/0471221279`.

[48] J M Hanson, B B Beard, and George C Marshall. *Applying Monte Carlo Simulation to Launch Vehicle Design and Requirements Analysis*. Tech. rep. NASA, 2010. url: `https://ntrs.nasa.gov/citations/20100038453`.

[49] W B Stein et al. “The Implementation of Maximum Likelihood Estimation in Space Launch System Vehicle Design”. In: *AAS/AIAA Space Flight Mechanics Meeting*. 2019. url: `https://ntrs.nasa.gov/citations/20190000722`.

[50] Omar Ugolini. “Design and Implementation Of A Rocket Launcher Hybrid Navigation”. MA thesis. 2023.