State Estimation and Control of a Self-Foraging Rocket

AAUROCKET

Master Thesis
June 2015

AALBORG UNIVERSITY
Control and Automation

AALBORG UNIVERSITY
STUDENT REPORT

Aalborg University
The Faculty of Engineering and Science
School of Information and
Communication Technology
Frederik Bajers Vej 7
9220 Aalborg Ø
Tlf.: 9940 9730
Telefax: 9940 9725
http://www.es.aau.dk

Subject: Control and Automation
Title: State Estimation and Control of a Self-Foraging Rocket
Project period: September 1$^{\text{st}}$ 2014 – June 3$^{\text{rd}}$ 2015
Semester: 10$^{\text{th}}$ Semester (P10)
Project group: 1030
Authors:

Michael Nauheimer

Kasper Hemme

Supervisor: John-Josef Leth and Jesper Abildgaard Larsen
Number of copies: 5

Pages: 162
Appendix pages: 19
Attachment: 1 DVD

Synopsis:

This master thesis describes the design and development of position and attitude determination and control of a rocket. The project is motivated by the general need of cheaper launch opportunities for an ever growing micro satellite market. The goal is to achieve a cheap and efficient solution in the form of the self-foraging rocket concept, inspired by the work of Yemets et al. at Dniepropetrovsk National University.
The overall design of the rocket is done from scratch starting with the initial mechanical concept of the rocket, followed by the sensors and actuators it will employ, and consecutively the mechanical dimensioning of each thruster to be utilized. Additionally a general ascent trajectory is calculated to launch the rocket into orbit, to be tested in a complete simulation environment designed in Simulink. Furthermore an extended Kalman filter is designed to estimate the states of the rocket, while both waypoint and sliding mode is designed for the control of the rocket.
The EKF was shown to be able to sufficiently estimate the states of the system in three distinct nominal tests.
A waypoint tracking controller, for following an optimal ascent trajectory, and a sliding mode controller is implemented as position control systems for launching the rocket into orbit. Despite for showing good performance for controlling the rocket, none of the controllers are able to inject the rocket into orbit with satisfactory results.

Preface

Reading Guide

Sources given in the beginning of a section apply to all subsections. Additional sources may be given in subsections. Citations are given in square brackets indicating the author (first) and the year of publication (last two ciphers), i.e. *Robust Model-Based Fault Diagnosis for Dynamic Systems* from 1999 by J. Chen & R. J. Patton is given as [Chen 99]. Figures, equations and tables are numbered according to chapter and location, e.g. third figure in chapter five is numbered 5.3. Units are given in square brackets in symbolic equations and without brackets otherwise.

Symbols, acronyms, general notation and reference frames used in this report are presented in the nomenclature after this preface. Appendices are placed after the main report, referred to by capital letters, followed by the Bibliography.

Contents

Nomenclature

Acronyms

AAU	Aalborg University
CC	Combustion Chamber
CoM	Center of Mass
CoP	Center of Pressure
CoT	Center of Thrust
DNU	Dnipropetrovs'k National University
ECEF	Earth Centered Earth Fixed
ECI	Earth Centered Inertial
EKF	Extended Kalman Filter
ENU	East North Up
GC	Gasification Chamber
GPS	Global Positioning System
HDPE	High Density Poly Ethylene
IGRF	International Geomagnetic Reference Model
LEO	Low Earth Orbit
LoS	Line of Sight
LSP	Launch Service Provider
MPC	Model Predictive Control
ORF	Orbital Reference Frame
PE	Polyethylene
RAAN	Right Ascension of the Ascending Node

Notation

Vectors and Matrices

A vector, $\mathbf{v}$, is noted

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \quad , \quad \mathbf{v} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \quad \text{or} \quad \mathbf{v} = \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix}^\top \tag{1}$$

A vector, $\mathbf{v}$, that has the subscript, k, is noted

$$\mathbf{v}_k = \begin{bmatrix} v_{k,1} \\ v_{k,2} \\ v_{k,3} \end{bmatrix} \quad \text{or} \quad \mathbf{v}_k = \begin{bmatrix} v_{k,x} \\ v_{k,y} \\ v_{k,z} \end{bmatrix} \tag{2}$$

A vector, $\mathbf{v}$, that has the subscript, k, and an index number, i, is noted

$$\mathbf{v}_{k,i} = \begin{bmatrix} v_{k,i,1} \\ v_{k,i,2} \\ v_{k,i,3} \end{bmatrix} \quad \text{for } i = 1, 2, ..., p \quad \mathbf{v}_k = \left\{ \begin{bmatrix} v_{k,1,1} \\ v_{k,1,2} \\ v_{k,1,3} \end{bmatrix}, \begin{bmatrix} v_{k,2,1} \\ v_{k,2,2} \\ v_{k,2,3} \end{bmatrix}, \dots, \begin{bmatrix} v_{k,p,1} \\ v_{k,p,2} \\ v_{k,p,3} \end{bmatrix} \right\} \tag{3}$$

A matrix, $\underline{\mathbf{M}}$, is noted

$$\underline{\mathbf{M}} = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix} \tag{4}$$

A matrix, $\underline{\mathbf{M}}$, that has the subscript, k, is noted

$$\underline{\mathbf{M}}_k = \begin{bmatrix} M_{k,11} & M_{k,12} & M_{k,13} \\ M_{k,21} & M_{k,22} & M_{k,23} \\ M_{k,31} & M_{k,32} & M_{k,33} \end{bmatrix} \tag{5}$$

A matrix, $\underline{\mathbf{M}}$, that has the subscript, k, and an index number, i, is noted

$$\underline{\mathbf{M}}_{k,i} = \begin{bmatrix} M_{k,i,11} & M_{k,i,12} & M_{k,i,13} \\ M_{k,i,21} & M_{k,i,22} & M_{k,i,23} \\ M_{k,i,31} & M_{k,i,32} & M_{k,i,33} \end{bmatrix} \quad \text{for } i = 1, 2, ..., p \tag{6}$$

$$\underline{\mathbf{M}}_k = \left\{ \begin{bmatrix} M_{k,1,11} & M_{k,1,12} & M_{k,1,13} \\ M_{k,1,21} & M_{k,1,22} & M_{k,1,23} \\ M_{k,1,31} & M_{k,1,32} & M_{k,1,33} \end{bmatrix}, \dots, \begin{bmatrix} M_{k,p,11} & M_{k,p,12} & M_{k,p,13} \\ M_{k,p,21} & M_{k,p,22} & M_{k,p,23} \\ M_{k,p,31} & M_{k,p,32} & M_{k,p,33} \end{bmatrix} \right\} \tag{7}$$

If a part, of more than one value, of a vector, $\mathbf{v}$, or a matrix, $\underline{\mathbf{M}}$, is wanted for use in another equation, the following notation is used

$$\mathbf{v}_{2:3} = \begin{bmatrix} v_2 \\ v_3 \end{bmatrix} \qquad \mathbf{v}_{1;3} = \begin{bmatrix} v_1 \\ v_3 \end{bmatrix} \tag{8}$$

$$\underline{\mathbf{M}}_{1:2;1:3} = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \end{bmatrix} \qquad \underline{\mathbf{M}}_{1,2;1,3} = \begin{bmatrix} M_{11} & M_{13} \\ M_{21} & M_{23} \end{bmatrix} \tag{9}$$

The entry-wise multiplication of two vectors or matrices of the same dimension are noted as

$$\underline{\mathbf{M}}_a \circledast \underline{\mathbf{M}}_b = \begin{bmatrix} M_{a,11}M_{b,11} & M_{a,12}M_{b,12} & \cdots & M_{a,1n}M_{b,1n} \\ M_{a,21}M_{b,21} & M_{a,22}M_{b,22} & \cdots & M_{a,23}M_{b,23} \\ \vdots & \vdots & \ddots & \vdots \\ M_{a,m1}M_{b,m1} & M_{a,m2}M_{b,m2} & \cdots & M_{a,mn}M_{b,mn} \end{bmatrix} \tag{10}$$

Reference Frames and Versors

A vector, $\mathbf{v}$, given in ECI is noted by

$${}^{\mathrm{i}}\mathbf{v} = \begin{bmatrix} {}^{\mathrm{i}}v_1 \\ {}^{\mathrm{i}}v_2 \\ {}^{\mathrm{i}}v_3 \end{bmatrix} \qquad \text{or} \qquad {}^{\mathrm{i}}\mathbf{v} = \begin{bmatrix} {}^{\mathrm{i}}v_x \\ {}^{\mathrm{i}}v_y \\ {}^{\mathrm{i}}v_z \end{bmatrix} \tag{11}$$

A quaternion $\mathbf{q}$ given by

$$\mathbf{q} = q_4 + \mathbf{i}q_1 + \mathbf{j}q_2 + \mathbf{k}q_3 \qquad \text{is noted} \qquad \mathbf{q} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} \tag{12}$$

Given two quaternions

$$\mathbf{q} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} \qquad \text{and} \qquad \mathbf{p} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix} \tag{13}$$

The quaternion multiplication of these is given by

$$\mathbf{q} \otimes \mathbf{p} = [\mathbf{q}]_{\otimes}\mathbf{p} = \begin{bmatrix} q_4 & -q_3 & q_2 & q_1 \\ q_3 & q_4 & -q_1 & q_2 \\ -q_2 & q_1 & q_4 & q_3 \\ -q_1 & -q_2 & -q_3 & q_4 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix} \tag{14}$$

A versor, $\mathbf{q}$, rotating a vector from ECI to CRF, and its complex conjugate, rotating a vector from CRF to ECI, are noted

$${}^{\mathrm{c}}_{\mathrm{i}}\mathbf{q} = \begin{bmatrix} {}^{\mathrm{c}}_{\mathrm{i}}q_1 \\ {}^{\mathrm{c}}_{\mathrm{i}}q_2 \\ {}^{\mathrm{c}}_{\mathrm{i}}q_3 \\ {}^{\mathrm{c}}_{\mathrm{i}}q_4 \end{bmatrix} \qquad \text{and} \qquad {}^{\mathrm{c}}_{\mathrm{i}}\mathbf{q}^* = \begin{bmatrix} -{}^{\mathrm{c}}_{\mathrm{i}}q_1 \\ -{}^{\mathrm{c}}_{\mathrm{i}}q_2 \\ -{}^{\mathrm{c}}_{\mathrm{i}}q_3 \\ {}^{\mathrm{c}}_{\mathrm{i}}q_4 \end{bmatrix} = {}^{\mathrm{i}}_{\mathrm{c}}\mathbf{q} = \begin{bmatrix} {}^{\mathrm{i}}_{\mathrm{c}}q_1 \\ {}^{\mathrm{i}}_{\mathrm{c}}q_2 \\ {}^{\mathrm{i}}_{\mathrm{c}}q_3 \\ {}^{\mathrm{i}}_{\mathrm{c}}q_4 \end{bmatrix} \tag{15}$$

The quaternion representation of a vector, ${}^{\mathrm{i}}\mathbf{v}$, is given by

$$\mathbf{q}({}^{\mathrm{i}}\mathbf{v}) = \begin{bmatrix} {}^{\mathrm{i}}\mathbf{v} \\ 0 \end{bmatrix} \tag{16}$$

Symbols

Symbol	Description	Unit
A	Cross sectional area of object perpendicular to $\|\|\mathbf{v}\|\|$	[m^2]
A_e	Nozzle exit area	[m^2]
A_f	Cross sectional area of fuel shell	[m^2]
A^*	Throat area	[m^2]
${}^{\mathrm{r}}\mathbf{Acc}$	Accelerometer measurement	[m/s^2]
a	Semimajor axis	[m]
${}^{\mathrm{r}}\mathbf{a}$	Dynamical acceleration of rocket	[m/s^2]
b	Semiminor axis	[m]
C_d	Aerodynamic drag coefficient	[$\cdot$]
c	Center of ellipsis	[m]
${}^{\mathrm{r}}\mathbf{d}_{IMU}$	Position of IMU in RBRF	[m]
${}^{\mathrm{r}}\mathbf{d}_k$	Position vector of kth vernier thruster in RBRF	[m]
e	Eccentricity	[$\cdot$]
$\mathbf{f}_{ext}$	Sum of external forces	[N]
g_0	Gravitational acceleration of Earth at sea-level	[m/s^2]
${}^{\mathrm{r}}\mathbf{g}$	Gravitational acceleration seen in RBRF	[m/s^2]
h	Height above sea level	[m]
h_e	Height of the engine	[m]
h_f	Height of fuel shell	[m]
h_p	Height of the payload	[m]
${}^{\mathrm{i}}\mathbf{h}$	Angular momentum	[Nms]
I_{sp}	Specific impulse	[s]
i	Inclination	[°]
J_2	Oblateness	[$\cdot$]
${}^{\mathrm{r}}\underline{\mathbf{J}}$	Moment of inertia	[kg$\cdot$m^2]
$\mathbf{L}$	Angular momentum	[Nms]
m_0	Initial mass of the rocket	[kg]
m_e	Mass of the engine part of AAURocket	[kg]
m_f	Final mass of the rocket	[kg]
m_o	Mass of the oxidizer cylinder of AAURocket	[kg]
m_{PL}	Mass of payload	[kg]
m_P	Propellant mass	[kg]
m_S	Structural mass	[kg]
$\dot{m}$	Mass flow	[kg/s]
$\dot{m}_f$	Fuel mass flow	[kg/s]
$\mathfrak{M}$	Molecular weight	[g/mol]
$\mathfrak{M}_a$	Molecular weight of air	[g/mol]
p_e	Exhaust pressure	[Pa]
p_a	Ambient pressure	[Pa]
p_c	Chamber pressure	[Pa]
R_E	Radius of Earth	[m]
$\mathfrak{R}$	Universal gas constant	$[\frac{\mathrm{J}}{\mathrm{mol\cdot K}}]$
r_a	Distance from center of orbit to apoapsis	[m]
${}^{\mathrm{i}}\mathbf{r}$	General position vector	[m]
r_e	Radius of the engine part of AAURocket	[m]
r_f	Width of the hollow fuel shell of AAURocket	[m]
r_o	Radius of the oxidizer cylinder of AAURocket	[m]
$\|\|{}^{\mathrm{i}}\mathbf{r}_o\|\|$	Orbit height	[m]
r_p	Distance from center of orbit to periapsis	[m]
T_a	Air temperature	[K]
T_c	Chamber temperature	[K]

${}^{\mathrm{r}}\mathbf{T}_k$	Thrust of kth vernier thrust in RBRF	[N]
T_{max}	Max thrust	[N]
$\mathbf{T}$	Thrust	[N]
${}^{\mathrm{i}}\mathbf{u}_r$	Unit vector along ${}^{\mathrm{i}}\mathbf{r}$	[·]
${}^{\mathrm{i}}\mathbf{u}_\perp$	Unit vector along ${}^{\mathrm{i}}\mathbf{u}_r$ along direction of travel	[·]
u_e	True exhaust velocity	[m/s]
v	True anomaly	[°]
v_e	Effective exhaust velocity	[m/s]
$\|\|{}^{\mathrm{i}}\mathbf{v}_e\|\|$	Velocity of mass expelled	[m/s]
$\|\|{}^{\mathrm{i}}\mathbf{v}_o\|\|$	Orbit speed	[m/s]
$\|\|{}^{\mathrm{i}}\mathbf{v}_r\|\|$	Speed component of object along ${}^{\mathrm{i}}\mathbf{u}_r$	[m/s]
$\|\|{}^{\mathrm{i}}\mathbf{v}_\perp\|\|$	Speed component of object along ${}^{\mathrm{i}}\mathbf{u}_\perp$	[m/s]
Δv	Change in velocity	[m/s]
α	Angle between noon line and the ascending node	[°]
$\boldsymbol{\alpha}$	General rocket angular acceleration vector	[rad/s^2]
γ	Reference vector	[·]
γ	Isentropic expansion factor	[·]
ϵ	Structural ratio	[·]
η_n	Nozzle efficiency	[·]
η_r	Reaction efficiency	[·]
η_t	Total engine efficiency	[·]
$\boldsymbol{\theta}$	General rocket angle vector	[rad]
θ_a	Angle between the apse line at periapsis	[°]
λ	Payload ratio	[·]
μ	Standard gravitational parameter	[m^3/s^2]
μ_E	Standard gravitational parameter for Earth	[m^3/s^2]
π_{PL}	Payload fraction	[·]
ρ_f	Density of the fuel (HDPE) for AAURocket	[kg/m^3]
ρ_o	Density of the oxidizer (H_2O_2) for AAURocket	[kg/m^3]
ρ_a	Density of ambient air	[kg/m^3]
τ_e	External torques	[Nm]
τ_t	Thrust torque	[Nm]
τ_v	Vernier thruster torque	[Nm]
Ω	Longtitude of ascending node	[°]
$\|\|\Omega\|\|$	Magnitude of change rate	[rad/s]
ω	Argument of periapsis	[°]
${}^{\mathrm{r}}\boldsymbol{\omega}$	Angular velocity of rocket	[rad/s]

Terminology

apoapsis The point in the path of an orbiting body at which it is farthest to the body that it orbits

Burn-out Main rocket engine shut down

burn-time The time period where the primary thruster is performing forward thrust

Coasting A period of the launch (or spaceflight) where no control and/or actuator action is performed.

Cryogenic Rocket Engine A rocket engine utilizing fuel and/or oxidizer that are cooled to very low temperatures in order to remain in liquid state. A common fuel-oxidizer combination for a cryogenic engine is H_2 and O_2.

Dry weight Dry weight is the weight of a vehicles/vessel excluding any consumables, passengers and cargo.

Endoatmospheric Inside the inner atmosphere of earth. The inner atmosphere extends up to approximately 100 km above the surface of earth.

Exoatmospheric Outside the inner atmosphere of earth. The inner atmosphere extends up to approximately 100 km above the surface of earth.

Fairing An external metal or plastic structure added to increase streamlining on a high-performance vehicle

Periapsis The point in the path of an orbiting body at which it is nearest to the body that it orbits.

Vernier Thruster Small thruster mounted at an angle of the main thrust vector. By enabling the vernier thruster the vehicle in question will rotate

Versor Quaternion with l^2-norm of 1. In this thesis, these are used for describing rotation between reference frames.

List of Figures

List of Tables

Chapter 1

Purpose and Mission of AAURocket

1.1 Introduction

The idea of rockets being used as launch vehicles to travel to space goes back more than a hundred years. However, it was not until russian inventor Konstantin Tsiolkovsky coined the idea of multi-stage rockets in 1929 in his book “The space rocket trains” [Encyclopedia 04] that the theory of chemical rockets reaching space actually seemed plausible. In 1957 this multi-stage concept was successfully utilized with the Sputnik Launch Vehicle when it made the first successful launch of a satellite to space with Sputnik 1 [Zak 13]. This marked the start of the space age, and with it, a rocket industry was born which is infamous for its astronomical costs and virtually zero reusability of its hardware. While launch vehicles like the NASA Space Shuttle has used designs that can reuse jettisoned hardware, the most common scenario is, that anything that is ejected during launch is discarded after use. Through the decades this norm has been a satisfactory solution to the needs of the space industry.
However, during the past 15 years the demand for launches for microsatellites (1kg-100kg) has seen a large growth [Bradford 13], and the tendency in demand is still on the rise. By the year 2020 there is a projected demand of up to 188 launches. A large part of the small satellite market is dominated by the nanosatellites (1kg-10kg) which includes the popular CubeSats. Due to the often limited budget of a CubeSat, the most common way of launching these satellites is to use the way of “piggyback launch” [JAXA 03]. These kind of launch opportunities usually suffer from orbital restrictions set by the main payload, leading to suboptimal orbital parameters for the microsatellite itself.
These microsatellite launch clients all have a common interest in being able to dictate their own orbital parameters while still obtaining a cheap launch opportunity. This will only be possible by creating a cheap and inherently efficient launch system.
This master thesis will attempt to create the basis for such a launch system, utilizing design ideas from the Dnipropetrovs’k National University (DNU). In particular the concept of a self-foraging rocket. This launch vehicle will throughout the report be referred to as AAURocket.

While the focus of the master thesis is to develop estimation and control algorithms, the project is also considered as an initial study to lay the basic groundwork for a possible larger umbrella of projects revolving around the subject of space and rocket theory, for other students interested in these matters. This inherently means that a lot of theoretical and practical assumptions are made throughout the report, sometimes with recommendations to examine certain problems closer if anyone were ever to proceed where this thesis ends.

1.2 Mission

The mission of AAURocket is the core objective which it must achieve in order to be reviewed as a success. The following section will outline the different choices and limitations that has been considered in order to obtain the final mission statement.

1.2.1 Payload

One of the primary factors that defines the mission of a launch vehicle is obviously the payload which it carries. The intended payloads of AAURocket will primarily be CubeSats and/or other microsatellites. The choice of payload stems from the needs of cheap launch vehicles for such satellites as expressed in 1.1.
A restriction all satellites has to comply with is the 2002 UN Inter-Agency Space Debris Mitigation Coordination Committees guideline, of allowing satellites to have a maximum lifetime of 25 years, in the low earth orbit region [Committee 02].
Combined with the fact that a common feature of CubeSats is a lack translational actuation capabilities. This in turn imposes a maximum average orbital injection periapsis of 640km [Leveque 11].

1.2.2 Orbital Parameters

While an upper bound has been established for the orbital periapsis, there are no restrictions for other orbital parameters. These parameters is ideally chosen by the client of AAURocket.
For this master thesis several assumptions are made as to have a general use-case for all of the orbital calculations.
The launch center of AAURocket is chosen to be at Aalborg University, while the target orbit is chosen to be a circular orbit with an altitude of 500km.
Furthermore the orbit is also chosen to be sun-synchronous, as this type of orbit provides sufficient communication capabilities for Danish space missions, with mission control centers located in Denmark.

1.2.3 Mission Statement

Following is the mission statement of AAURocket:

"The mission of AAURocket is to launch a payload [1] of maximum 50kg into a 500km altitude sun synchronous circular orbit."

1.3 Thesis Outline

Following is a short description of what each of the chapters containts.

Chapter 2 - Orbit General reference frames used for respective object orientations are presented, followed by the versors used to calculate the rotations between each reference frame.
The general Keplerian orbit elements are also presented in order to describe the target orbit of the AAURocket. The orbital parameters guarding this orbit is successively calculated using orbital mechanics. These orbital parameters include orbit velocity, period, inclination, and angle of the ascending node.

[1] In this case the payload is referred to as the entire nose of the rocket. That is, including both satellite, fairing, and other structural components launched in to orbit.

Chapter 3 - System Requirements This chapter outlines the general system requirements set for the overall mission of AAURocket, by examining other launch vehicles and their requirements. These mission requirements ultimately lead to specific subsystem requirements for both the position and attitude determination system and the control system.

Chapter 4 - Mechanical Design The basic concept of launch vehicle based on the chemical rocket design is presented in this chapter. Additionally Tsiolkovsky's Rocket Equation is presented in order to evaluate different rocket setup efficiencies, of which the single stage, the multi stage and the infinite stage rockets are compared.
Finally the concept of a self-foraging rocket is presented as imagined by Yemets et al. from Dniepropetrovsk National University. In order to achieve a cheap high efficiency design the concept of self-foraging rockets are adopted as the design strategy for AAURocket.

Chapter 5 - Guidance-Hardware Design In this chapter both general sensors and actuators are presented in order to make qualified choices on what types of hardware to consider for AAURocket. Several different sensor types are chosen embedded on a single inertial measurement unit, while the general actuator choice falls on the vernier thruster method.
Additionally different general thrust profile strategies are presented in order to determine overall constraints for the AAURocket thrust and ascent profile. From these constraints the general engine block is designed. Furthermore, each thruster type used for AAURocket is consecutively dimensioned from the thermodynamic chemical rocket equations presented.

Chapter 6 - Rocket and Disturbance Modeling The general equations of motion and rotation considered in this thesis is presented in this chapter, as well as the common major disturbance forces. Furthermore, analytical expressions for the changing moment of inertia are presented, followed by the thermal model of the gasification process used to gasify and consume the body of the self-foraging rocket.

Chapter 7 - Simulink Simulation Environment In this chapter the simulation environment created in Simulink is presented. The overall structure is outlined and each Simulink subsystem block guarding dynamical equations, versor rotations, rocket parameters, environmental disturbances, sensors, and actuators are described.

Chapter 8 - Ascent Trajectory This chapter presents different ascent profile types, of which one is chosen to design the overall optimization problem to calculate the actual ascent trajectory to orbit. Two optimization calculations are performed: a "minimum propellant to orbit" and a "spare propellant to orbit". Each of these trajectories are compared and the latter is chosen to be the primary optimal ascent trajectory for AAURocket.

Chapter 9 - Position and Attitude Determination This chapter presents the linear and extended Kalman filter, of which an overall design is made of an extended Kalman filter to serve as the primary position and attitude determination algorithm on AAURocket. The extended Kalman filter is tested with three nominal test setups, and is successively evaluated with respect to the system requirements.

Chapter 10 - Position and Attitude Control In this chapter a general waypoint controller is presented, which will control the LV along a predefined trajectory for the early phase of the ascent to orbit. Furthermore a sliding mode controller is presented which will control the rocket in the later phase of the ascent trajectory.

Chapter 11 - Closure This chapter includes the conclusion of this master thesis as well as a section about further development.

Chapter 2

Orbit

This chapter will outline the different general reference frames and elements of orbital mechanics used for this project. Using these definitions the general orbital parameters for the mission of AAURocket will be calculated.

2.1 Reference Frames

Throughout this thesis several reference frames will be utilized. These frames are used to describe the orientation of the rocket during its launch and mission. Furthermore these frames are a convenient way to interpret sensor data relative to the rocket.

2.1.1 Earth Centered Inertial reference frame (ECI)

The Earth Centered Inertial reference frame is the primary inertial reference frame of this thesis. A frame that is inertial implies that its motion is constrained to only be rectilinear. This in turn means that the frame is clean of any fictitious forces.

The Earth Centered Inertial reference frame (ECI) is defined as having its origin in the centre of mass (CoM) of the Earth. The z-axis of the ECI is pointing towards the geometric North Pole and the x-axis is pointing towards the Sun at the vernal equinox which happens around March 20th. The y-axis is the right-hand cross product between the z- and x-axis, making it extend through the equator somewhere in the Indian ocean

This frame in essence does not move rectilinearly, since the Earth rotates around the sun. However, since an orbit around the sun takes a year, the change in direction of the Earth's movement is minuscule. In turn, the movement of this frame is assumed rectilinear due to the short timeframe of the launch and mission of the AAURocket.

Vectors specified in the ECI frame will be denoted as: ${}^{\mathrm{i}}\mathbf{v}$.

2.1.2 Earth Centred Earth Fixed reference frame (ECEF)

The Earth Centred Earth Fixed reference frame (ECEF) also has its origin placed at the centre of mass of the Earth. It shares the same orientation for its z-axis as the ECI, however, the x-axis is crossing the point where the Greenwich meridian and the equator meets. The y-axis is the right-hand cross product of the z- and x-axis. Due to these features the ECEF is exactly equal to the ECI reference frame at vernal equinox.

The ECEF frame is used to describe the position of objects with fixed orientation to the surface of the Earth, e.g. the rocket launch pad.

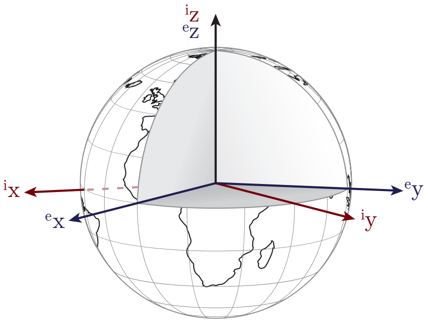


Figure 2.1: ECI and ECEF superimposed on the Earth. The ECI frame is marked in red, and the ECEF frame is marked in blue, with a shared black z-axis.

Vectors specified in the ECEF frame will be denoted as: ${}^{e}\mathbf{v}$.

2.1.3 Rocket Body Reference Frame (RBRF)

The Rocket Body Reference Frame (RBRF) is a rocket body fixed reference frame used to describe the orientation and position of different elements within the rocket. The RBRF axes for AAURocket has been aligned parallel with the rocket frame. The origin of the RBRF is placed in the CoM of the rocket with the x-axis pointing in the direction of the nose. The y- and z-axis are placed with respect to thruster actuators presented in section 5.4 since these are the only defining factors of the rocket around the x-axis.
Vectors specified in the RBR frame will be denoted as: ${}^{r}\mathbf{v}$.

2.1.4 Control Reference Frame

The Control Reference Frame (CRF) is a reference frame placed with its origin in the CoM of the rocket, like the RBRF.
The relation or rotation between RBRF and CRF is stationary, and is determined by the eigenvectors of the inertia matrix given in RBRF The z-axis of the CRF is pointing in the direction of the major axis of the rocket, i.e. the axis with the largest inertia eigenvalue, and the x-axis is pointing in the direction of the minor axis of inertia. The y-axis is the right-hand cross product of the z- and x-axis, and is oriented along the intermediate axis of inertia.
With the three axes pointing in the same direction as the inertial axes, the CRF is body fixed as the inertia of the satellite is fixed. However, had the satellite been able to change shape (i.e. deploy solar panels), then the CRF would change.
Vectors specified in the RBR frame will be denoted as: ${}^{c}\mathbf{v}$.

2.2 Rotation between frames

Sources: [Wertz 78]

In order to represent vectors from one frame in another, reference frame rotations are needed. These are mathematical operations which can be applied to a vector in a specific frame and produce a new set of coordinates representing said vector in a new frame.
There are several different ways to perform rotations including direct cosine matrices, Euler angles, euler eigenaxis rotation, and more. For this project the primary construct used for rotations are quaternions, or more specific; versors.

The primary reason for using versors is their inherent lack of singularities unlike their mathematical rotational counterparts.

2.2.1 Quaternions

A quaternion is a hyper-complex number consisting of a real and an imaginary part, the complex part being a three-dimensional vector of imaginary units i, j, and k. The general quaternion definition can be seen in equation 2.1.

$$\mathbf{q} = iq_1 + jq_2 + kq_3 + q_4 \Leftrightarrow \mathbf{q}^* = -iq_1 - jq_2 - kq_3 + q_4 \tag{2.1}$$

Seeing as i, j, and k are imaginary, these have non-commutative behaviour when being multiplied. The relation of multiplication between the complex part can be seen in equation 2.2

$$i^2 = j^2 = k^2 = ijk = -1 \qquad ij = -ji \qquad jk = -kj \qquad ki = -ik \tag{2.2}$$

2.2.2 Versors

The versor is a subset of the quaternion construct. Basically a versor is a quaternion which is constrained to always have its norm equal to one. An inherent property of this constraint is that $\mathbf{q}^{-1} = \mathbf{q}^*$. Geometrically the unit length constraint means, that quaternions can be interpreted as a four-dimensional sphere, the rotation describing a trajectory on the surface of this sphere.

$$\mathbf{q} = \begin{bmatrix} \hat{\mathbf{e}} \sin \frac{\theta}{2} \\ \cos \frac{\theta}{2} \end{bmatrix} \tag{2.3}$$

In essence the versor components are closely related to the euler eigen-axis rotation. This can be seen in equation 2.3 where the initial three components are comprised of a vector (euler vector $\mathbf{e}$) multiplied by a factor of rotation around said vector. The rotation around the euler vector can always be readily derived from the fourth component.

Versor multiplication is defined as:

$$\mathbf{q} \otimes \mathbf{p} = \underbrace{q_4 p_4 - \mathbf{q}_{1:3} \cdot \mathbf{p}_{1:3}}_{\text{scalar part}} + \underbrace{q_4 \mathbf{p}_{1:3} + p_4 \mathbf{q}_{1:3} + \mathbf{q}_{1:3} \times \mathbf{p}_{1:3}}_{\text{vector part}} = [\mathbf{q}]_{\otimes} \, \mathbf{p} \tag{2.4}$$

with

$$[\mathbf{q}]_{\otimes} = \begin{bmatrix} q_4 & -q_3 & q_2 & q_1 \\ q_3 & q_4 & -q_1 & q_2 \\ -q_2 & q_1 & q_4 & q_3 \\ -q_1 & -q_2 & -q_3 & q_4 \end{bmatrix}$$

Successive rotations can be combined as: ${}^{\mathrm{i}}_{\mathrm{c}}\mathbf{q} = {}^{\mathrm{i}}_{\mathrm{r}}\mathbf{q} \otimes {}^{\mathrm{r}}_{\mathrm{c}}\mathbf{q} \Leftrightarrow {}^{\mathrm{i}}_{\mathrm{r}}\mathbf{q} = {}^{\mathrm{i}}_{\mathrm{c}}\mathbf{q} \otimes {}^{\mathrm{r}}_{\mathrm{c}}\mathbf{q}^*$.
Rotating vectors by versors is written by transforming a vector into a versor with zero real part. The angular velocity of a reference frame r as seen from an inertial reference frame i, denoted ${}^{\mathrm{i}}\boldsymbol{\omega}_s$, can be found in the reference frame e by rotating the vector from i to e as:

$$\begin{bmatrix} {}^{\mathrm{e}}\boldsymbol{\omega}_r \\ 0 \end{bmatrix} = {}^{\mathrm{e}}_{\mathrm{i}}\mathbf{q} \otimes \begin{bmatrix} {}^{\mathrm{i}}\boldsymbol{\omega}_r \\ 0 \end{bmatrix} \otimes {}^{\mathrm{e}}_{\mathrm{i}}\mathbf{q}^* \tag{2.5}$$

Consecutive rotations described by quaternions give a non-singular, bi-linear method to go from one reference frame to another through successive rotations. Rotating from i to r to c is implemented as:

$$\begin{bmatrix} {}^{\mathrm{c}}\boldsymbol{\omega}_r \\ 0 \end{bmatrix} = {}^{\mathrm{c}}_{\mathrm{r}}\mathbf{q} \otimes {}^{\mathrm{r}}_{\mathrm{i}}\mathbf{q} \otimes \begin{bmatrix} {}^{\mathrm{i}}\boldsymbol{\omega}_r \\ 0 \end{bmatrix} \otimes {}^{\mathrm{c}}_{\mathrm{r}}\mathbf{q}^* \otimes {}^{\mathrm{r}}_{\mathrm{i}}\mathbf{q}^* \tag{2.6}$$

For ease of notation the vector rotation by versor will throughout this report be written as:

$${}^{\mathrm{e}}\boldsymbol{\omega}_r = {}^{\mathrm{e}}_{\mathrm{i}}\mathbf{q} \otimes {}^{\mathrm{i}}\boldsymbol{\omega}_r \tag{2.7}$$

The versor $\mathbf{q}$ and the versor $-\mathbf{q}$ represent the same rotation about $\hat{\mathbf{e}}$ in SO(3): the former of an angle θ, and the latter of an angle $2\pi + \theta$.

Given a versor, ${}^{\mathrm{i}}_{\mathrm{r}}\mathbf{q}$, describing the rotation between the RBRF and the ECI frame, and knowing that rocket is rotating within the ECI frame, it is also known that the quaternion, ${}^{\mathrm{i}}_{\mathrm{r}}\mathbf{q}$, changes with time. Knowing the angular velocity, ${}^{\mathrm{r}}\boldsymbol{\omega}$, of the rocket within its own frame, it is possible to determine the time derivative of the quaternion as:

$${}^{\mathrm{i}}_{\mathrm{r}}\dot{\mathbf{q}} = \frac{1}{2}\underline{\boldsymbol{\Omega}}\,{}^{\mathrm{i}}_{\mathrm{r}}\mathbf{q} = \frac{1}{2}\begin{bmatrix} 0 & \omega_3 & -\omega_2 & \omega_1 \\ -\omega_3 & 0 & \omega_1 & \omega_2 \\ \omega_2 & -\omega_1 & 0 & \omega_3 \\ -\omega_1 & -\omega_2 & -\omega_3 & 0 \end{bmatrix} {}^{\mathrm{i}}_{\mathrm{r}}\mathbf{q} \tag{2.8}$$

2.3 Orbital Parameters in Terms of Keplerian Orbit Elements

An orbit described by Keplerian elements is an ideal (reference) orbit defined by two gravitationally interacting spherically symmetric objects, where their relative velocity and position define the orbit plane. The shape of the orbit must be an ellipsis for permanently associated objects. The ellipsis is defined by two focus points, one being the position of the larger of the two objects (when $m_{\text{large}} \gg m_{\text{small}}$; in this case the Earth), and the ellipsis boundary being the trajectory of the satellite.

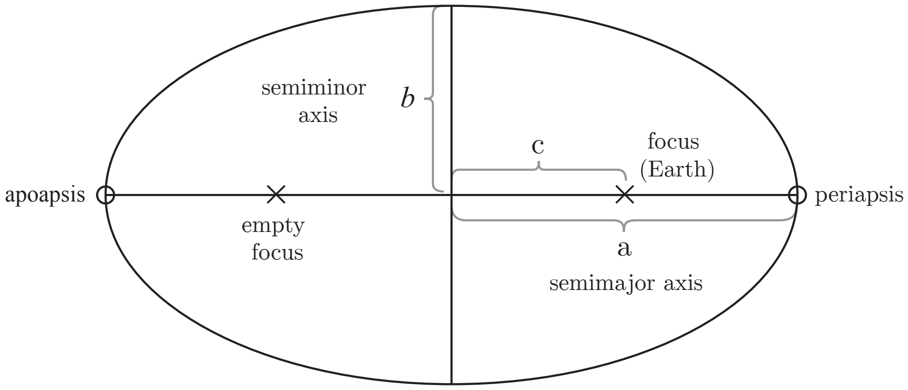


Figure 2.2: Geometric description of an ellipsis, defined by its semimajor and semiminor axes, a and b. The primary (the Earth) is situated in one of the foci of the satellite trajectory ellipsis.

The shape of the orbit ellipsis is uniquely defined by the *eccentricity* parameter, e, described in terms of the *semimajor axis*, a, defining the size of the ellipsis; and the distance from the focus point to the centre of the ellipsis, c; see Figure 2.2. The magnitude of the eccentricity will be

$0 < e < 1$ for an ellipsis, and the smaller the value, the more circular the orbit. The eccentricity of an orbit can be calculated by equation 2.9

$$e = \frac{c}{a} = \frac{\sqrt{a^2 - b^2}}{a} = \frac{r_a - r_p}{r_a + r_p} \tag{2.9}$$

where:

r_a	is the distance from the center of the orbit to the apoapsis	[m]
r_p	is the distance from the center of the orbit to the periapsis	[m]

The orientation of the orbit plane in space is defined by the *inclination* parameter, i, and the east-west orientation Ω. The inclination is the angle from the equatorial plane to the orbit plane. The intersection line between the two planes, called the line of nodes or nodeline, intersects the orbit trajectory in two points: the ascending node, where the satellite crosses the equatorial plane going from south to north; and the descending node, where it crosses the plane going from north to south. The east-west orientation is defined by the *right ascension of the ascending node* (RAAN), which is the angle measured in the equatorial plane eastward from the ECI x-axis (equal to the ECEF x-axis at vernal equinox) to the ascending node, see Figure 2.3.

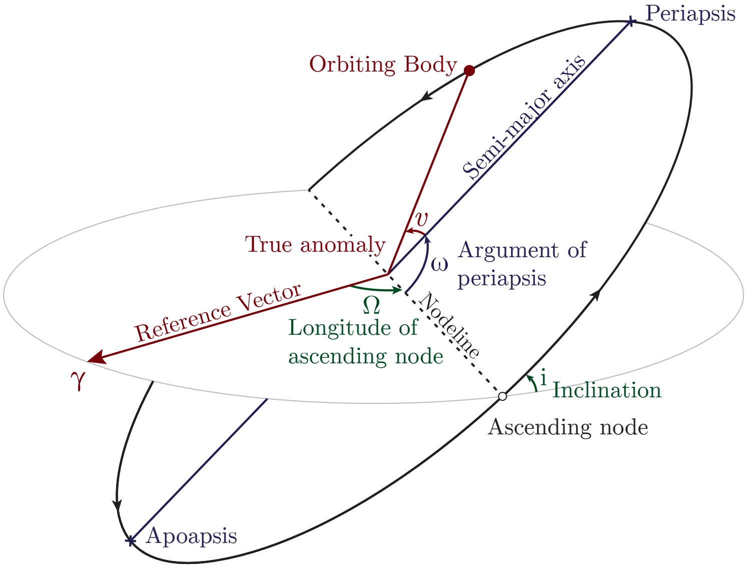


Figure 2.3: Contorted sketch of a satellite orbit, describing the Keplerian orbital elements. The centre of the Earth is in one of the two foci of the orbit ellipse. The two angles Ω (RAAN) and ω (argument of perigee) are measured in the equatorial and the orbital plane, respectively.

The terms periapsis and apoapsis describes the lowest and highest altitudes, respectively, during an orbit of the satellite, measured from the surface of the Earth. The rotation of the orbit ellipsis within the orbital plane is defined by the *argument of periapsis*, ω. This is the angle measured in the orbital plane in the direction of motion of the satellite from the ascending node to periapsis.

The largest disturbance causing deviations from the ideal Keplerian orbit for low altitude orbits, are the predominant atmospheric drag, which tend to decrease the semimajor axis of the orbit.

2.4 Orbital Parameters for the target orbit of AAURocket

This section will outline the characteristic parameters of the target orbit of AAURocket.

2.4.1 Target Orbit Velocity and Period

Source: [Curtis 05]

In order for an object (e.g. a satellite) to stay in orbit around a larger object (e.g. the Earth), an equilibrium between altitude, eccentricity and velocity has to be obtained. The closer the satellite is to the planet which it orbits, the faster it needs to travel, and conversely, the further away it is the slower it has to travel.
In order to determine the orbital speed of a satellite, the orbital equation is needed. Equation 2.10, known as the orbital equation, defines an orbiting objects path around a planet.

$$||{}^{\mathrm{i}}\mathbf{r}|| = \frac{\frac{||{}^{\mathrm{i}}\mathbf{h}||^2}{\mu}}{1 + e\cos(\theta_a)} \tag{2.10}$$

where

${}^{\mathrm{i}}\mathbf{r}$	is the position vector ${}^{\mathrm{i}}\mathbf{r}$ of the element in orbit.
${}^{\mathrm{i}}\mathbf{h}$	is the angular momentum vector of the object in orbit.
μ	is the standard gravitational parameter of the planet which is orbited.
e	is the eccentricity of the orbit.
θ_a	is the angle between the apse line at the periapsis of the orbital ellipse and the vector ${}^{\mathrm{i}}\mathbf{r}$.

As described in section 1.2 the target orbit is required to be circular, or in other words, the eccentricity is required to be 0. By setting $e = 0$ equation 2.11 is obtained from the orbital equation.

$$||{}^{\mathrm{i}}\mathbf{r}|| = \frac{||{}^{\mathrm{i}}\mathbf{h}||^2}{\mu} \tag{2.11}$$

With ${}^{\mathrm{i}}\mathbf{r}$ as the position vector of the orbiting object, then it also follows that ${}^{\mathrm{i}}\dot{\mathbf{r}}$ is the velocity vector of said object. This velocity vector can be split up in to two components, as seen in equation 2.12.

$${}^{\mathrm{i}}\dot{\mathbf{r}} = ||{}^{\mathrm{i}}\mathbf{v}_r||{}^{\mathrm{i}}\mathbf{u}_r + ||{}^{\mathrm{i}}\mathbf{v}_\perp||{}^{\mathrm{i}}\mathbf{u}_\perp \tag{2.12}$$

where

$||{}^{\mathrm{i}}\mathbf{v}_r||$	is the speed component of the object in orbit along the same direction as ${}^{\mathrm{i}}\mathbf{u}_r$
${}^{\mathrm{i}}\mathbf{u}_r$	is the unit vector placed at center of orbit, sharing the direction of ${}^{\mathrm{i}}\mathbf{r}$
$||{}^{\mathrm{i}}\mathbf{v}_\perp||$	is the speed component of the object in orbit along the same direction as ${}^{\mathrm{i}}\mathbf{u}_\perp$
${}^{\mathrm{i}}\mathbf{u}_\perp$	is the the unit vector perpendicular (in the direction of travel) to ${}^{\mathrm{i}}\mathbf{u}_r$

Furthermore the magnitude of the angular momentum is defined in equation 2.13.

$$||{}^{\mathrm{i}}\mathbf{h}|| = ||{}^{\mathrm{i}}\mathbf{r}||||{}^{\mathrm{i}}\mathbf{v}_\perp|| \tag{2.13}$$

Seeing that $||{}^{\mathrm{i}}\mathbf{r}||$ is constant (due to the circularity of the orbit), then $||{}^{\mathrm{i}}\dot{\mathbf{r}}||$=0. This leads to $||{}^{\mathrm{i}}\mathbf{v}|| = ||{}^{\mathrm{i}}\mathbf{v}_\perp||$ which means that the magnitude of the angular momentum for a circular orbit can be written as seen in equation 2.14.

$$||{}^{\mathrm{i}}\mathbf{h}|| = ||{}^{\mathrm{i}}\mathbf{r}||||{}^{\mathrm{i}}\mathbf{v}|| \tag{2.14}$$

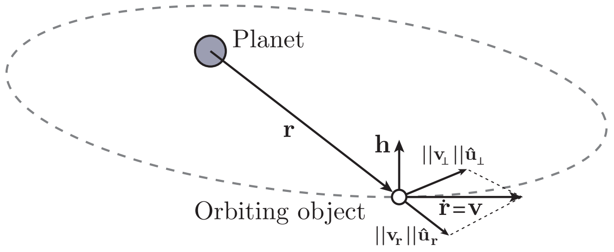

Planet
r
h
$||\mathbf{v}_\perp||\hat{\mathbf{u}}_\perp$
Orbiting object
$\dot{\mathbf{r}}=\mathbf{v}$
$||\mathbf{v}_\mathbf{r}||\hat{\mathbf{u}}_\mathbf{r}$

Figure 2.4: Essentials of sun synchronous orbit.

Substituting equation 2.14 into equation 2.11 produces equation 2.15.

$$
\begin{aligned}
||{}^{\mathrm{i}}\mathbf{r}|| &= \frac{||{}^{\mathrm{i}}\mathbf{r}||^2||{}^{\mathrm{i}}\mathbf{v}||^2}{\mu} \Leftrightarrow \\
||{}^{\mathrm{i}}\mathbf{v}_{circular}|| &= \sqrt{\frac{\mu}{||{}^{\mathrm{i}}\mathbf{r}||}}
\end{aligned}
\tag{2.15}
$$

Using equation 2.15 it is now possible to calculate any required orbital speed for any given height for circular orbits (see figure 2.5).

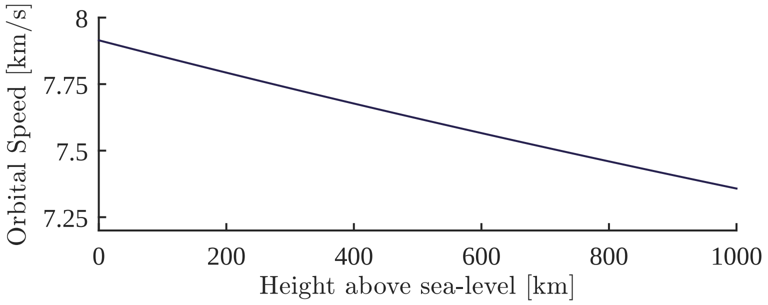


Figure 2.5: Graph showing the relation between orbital speed, $||\,{}^{\mathrm{i}}\mathbf{v}||$, and height above sea-level for circular orbits.

The orbital speed for a satellite orbiting the Earth is then:

$$
||{}^{\mathrm{i}}\mathbf{v}_o|| = \sqrt{\frac{\mu_{\mathrm{E}}}{\mathrm{R_E} + ||{}^{\mathrm{i}}\mathbf{r}_o||}} = 7621.6\,[\mathrm{m/s}]
\tag{2.16}
$$

where

$\lVert{}^{\mathrm{i}}\mathbf{v}_o\rVert$	is the orbital speed for the object in orbit	[m]
μ_{E}	is the standard gravitational parameter of the Earth	[[m^3/s^2]]
$\mathrm{R_E}$	is the radius of the Earth	[m]
$\lVert{}^{\mathrm{i}}\mathbf{r}_o\rVert$	is the height above sea level of the orbit	[m]

Furthermore the period of the target circular orbit of 500km altitude can be calculated as:

$$T_{orbit} = \frac{2\pi||{}^{\mathrm{i}}\mathbf{r}||}{||{}^{\mathrm{i}}\mathbf{v}_o||} = 5668.1\,[\mathrm{s}] \tag{2.17}$$

2.4.2 Orbit Inclination

Source: [Curtis 05]

For an orbit to be Sun synchronous the change of the longitude of the ascending node, Ω, must equal that of the change of Earths position around the Sun. In this way the orbital plane will always have the same orientation with respect to the Sun, and therefore be Sun synchronous.
However, the law of conservation of angular momentum states that when no external torque is applied to a system, no change in angular momentum can occur. This is also true for an object in orbit around a planet. Hence, an object in orbit around the Earth will have a static orbital plane with respect to the ECI frame, ie. a static Ω.
Due to rotation the Earth is not a perfect spheroid, but rather oblate instead. This results in an uneven gravitational pull from the Earth on an orbiting object depending on its position. This oblateness can in turn be used to exert a torque on an orbiting object by choosing an inclination which is different from 0° or 90°.
Equation 2.18 describes the inclination angle of an orbit needed to obtain a certain change rate in Ω.

$$i = \cos^{-1}\left(\frac{\dot{\Omega}}{\left[\frac{3}{2}\frac{\sqrt{\mu}\mathrm{J}_2\mathrm{R}_E^2}{(1-e^2)^2 a^{\frac{7}{2}}}\right]}\right) \tag{2.18}$$

where

$\|\|\dot{\Omega}\|\|$	is the magnitude of the change rate of Ω	[rad/s]
J_2	is the oblateness of the Earth	[·]

As mentioned the change rate in the longitude of the ascending node, Ω, is required to equal that of the Earths orbital period for the satellite orbit to be sun synchronous. This Ω change rate is calculated in equation 2.19.

$$||\dot{\Omega}|| = \frac{2\pi}{365 \cdot 24 \cdot 60 \cdot 60} = 0.19924 \cdot 10^{-6} \tag{2.19}$$

Furthermore, the oblateness of the Earth, J_2 can be found as seen in equation 2.20.

$$\mathrm{J}_2 = \frac{\text{equatorial radius} - \text{polar radius}}{\text{equatorial radius}} = 0.00108263\,[\cdot] \tag{2.20}$$

The orbital parameters for a 500 km sun synchronous circular orbit can then be substituted in to 2.18 which results in:

$$inclination = 1.6996[\mathrm{rad}] = 97.38\,[\mathrm{deg}] \tag{2.21}$$

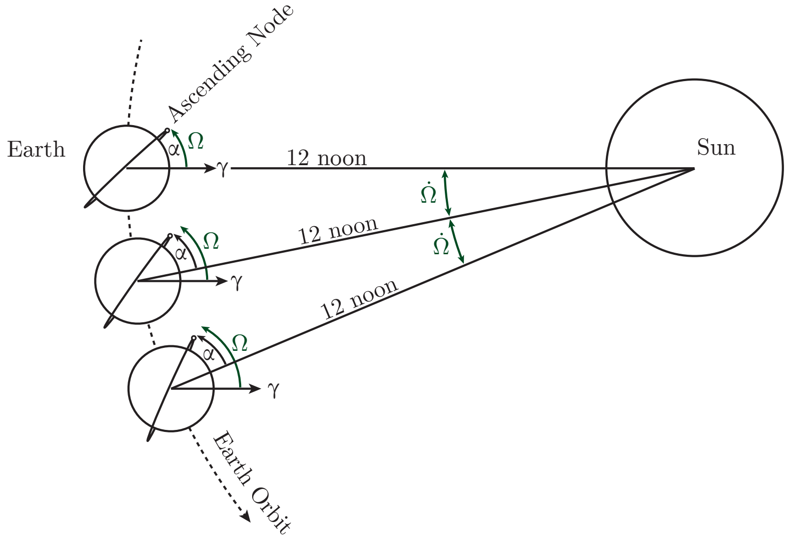


Figure 2.6: Essentials of a sun synchronous orbit, where γ is the keplerian reference line (in this case it could be the ECI x-axis at vernial equinox), α is the angle between the noon-line and the ascending node, Ω is the angle between the reference line and the ascending node, and $\dot{\Omega}$ is the change change rate of Ω

2.4.3 Initial Angle of the Ascending Node

It is clear from the previous section, that the only constraint that applies for an orbit to be sun synchronous is the specific change rate of Ω. The initial angle of the ascending node is in essence a free variable to be determined depending on the mission. There is only one factor influencing what the initial angle of the right ascending node should be, and that is the amount of exposure the orbit gets to the sun.

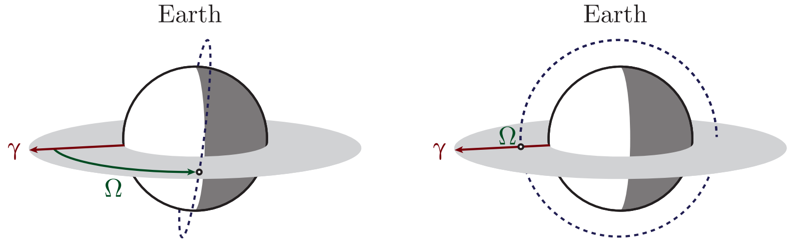


Figure 2.7: Two different sun synchronous orbits with 90° and 0° angle of the right ascending node. The particular time instance pictured is with the sun placed in the direction of the reference vector γ.

On figure 2.4.3 the two extremum cases of sun synchronous orbits are shown. In this case the initial angle to the right ascending node has been pictured in such a fashion that the initial angle of 90° has the orbit plane perpendicular to the sun position vector (which is aligned with the reference vector γ). This results in this particular orbit to always be fully exposed to sunlight. For space missions with high power demands this can be an orbit of interest due to the higher

power input from the sun (via solar cells).
In the other case where the angle of the ascending node is 0° the orbit plane is parallel to the position vector of the sun. Such an orbit can be of interest for space missions that needs to be shielded from the sun, e.g. star observation.

In this project it is assumed that the launch time will be at either dawn or dusk such that the final orbit will be fully exposed to the sun.

2.5 Subconclusion

In this chapter the primary reference frames, the versors to describe the rotation between each frame, and Keplarian orbital elements has been outlined. Furthermore, all the critical parameters for the AAURocket mission has been calculated and chosen. Following is a table showing the combined results from this section:

Parameter	**Value**	**Unit**
$\|\|{}^{\mathrm{i}}\mathbf{v}_o\|\|$	7621.7	[m/s]
T_o	5658.4	[s]
i	97.38	[deg]
$\|\|\dot{\Omega}\|\|$	$0.19924{\cdot}10^{-6}$	[rad/s]

Table 2.1: Orbital parameters essential for the target orbit of the AAURocket.

Chapter 3

System Requirements

In this chapter the general requirements for the AAURocket missions will be outlined, followed by the requirements set for each subsystem in order to ensure these mission requirements are fulfilled.

3.1 Mission Requirements

The mission statement states that AAURocket has to obtain a circular sun synchronous orbit with altitude of 500 km. The orbital parameters for such a target can be seen in 3.1

Orbital Parameter	**Value**	**Unit**
Altitude	500	[km]
Inclination	97.38	[deg]
Eccentricity	0	[·]

Table 3.1: Target orbital parameters.

As AAURocket is supposed to be able to compete with professional launch service providers like Arianespace and SpaceX, the requirements for orbital insertion should be similar to those of their product line e.g. Ariane 5 or Falcon 9.
According to the user manual for Ariane 5 [Arianespace 11a], Arianespace offers a typical standard deviation in orbit eccentricity of $0.35 \cdot 10^{-3}$ for a circular orbit while SpaceX expects their Falcon 9 to ensure an eccentricity of $1.45 \cdot 10^{-3}$ [SpaceX 09]. Furthermore, Ariane 5 ensures perturbations below $0.02°$, while the expected maximum is $0.1°$ for the Falcon 9.
With regards to altitude, the Ariane 5s typical standard deviation is 2.5 km, while SpaceX expects a maximum deviation of ±10 km.

Thereby performance for AAURocket, with regards to altitude, is deemed satisfactory if the target parameters for apoapsis and periapsis are obtained within ±10 km.

Taking into account that Arianespace has far more experience than SpaceX, it is deemed satisfactory that AAURocket should comply with the expected maximum deviation of the Falcon 9. Thus the allowed perturbation for each orbital parameter for AAURocket can be seen in table 3.2.

The requirements listed in table 3.2 are the chosen requirement perturbations for the orbital

Requirement	Orbital Parameter	Value	Maximum Perturbation	Unit
M.R1	Eccentricity	0	$+\ 1.45 \cdot 10^{-3}$	[·]
M.R2	Altitude	500	± 10.0	[km]
M.R3	Inclination	97.38	± 0.100	[deg]

Table 3.2: Positional mission requirements.

parameters. Each of the orbital parameters are affected by both the position and velocity at the time of insertion. When determining if the requirement for eccentricity is fulfilled, equation 2.9 can be used by finding the apoapsis and periapsis of the obtained orbit and using these to calculate the eccentricity. Table 3.3 can be used as a guideline for determining if the system complies with the requirements. The table shows orbital insertions in the range of altitudes at ± 10 km at speed errors in the range of ± 10 m/s, where the entries containing a 1, are the orbital insertions that comply with the requirements for altitude and eccentricity.

		Speed error [m/s]												
		-10	-8	-6	-4	-2	-1	0	1	2	4	6	8	10
Altitude error [km]	-10	0	0	0	0	0	0	0	0	0	0	0	0	0
	-8	0	0	0	0	0	0	0	0	0	1	1	1	0
	-6	0	0	0	0	0	0	0	0	0	1	1	0	0
	-4	0	0	0	0	0	0	0	1	1	1	1	0	0
	-3	0	0	0	0	0	0	1	1	1	1	0	0	0
	-2	0	0	0	0	0	1	1	1	1	1	0	0	0
	-1	0	0	0	0	0	1	1	1	1	0	0	0	0
	0	0	0	0	0	1	1	1	1	1	0	0	0	0
	1	0	0	0	0	1	1	1	1	0	0	0	0	0
	2	0	0	0	1	1	1	1	1	0	0	0	0	0
	3	0	0	0	1	1	1	1	0	0	0	0	0	0
	4	0	0	1	1	1	1	0	0	0	0	0	0	0
	6	0	0	1	1	0	0	0	0	0	0	0	0	0
	8	0	1	1	1	0	0	0	0	0	0	0	0	0
	10	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3.3: Orbital insertions with altitude errors in the range of ± 10 km and speed errors in the range of ± 10 m/s. Each entry contains a logical value that indicates if the insertion properties comply with the requirements for altitude and eccentricity.

By the data shown in table 3.3 it is seen that the orbital insertion have to be done with an accuracy in altitude of ± 2 km and an accuracy in speed of ± 1 m/s. It is also seen that if the altitude is off by 8 km it is still possible to get into acceptable orbit if the error of the speed is - 8 m/s. Even though there is a risk of getting into an orbit that does not comply entirely with the mission requirements, it chosen that the maximum allowed deviation altitude is 2 km and the maximum deviation in speed is 6 m/s.

While different satellite missions have different attitude and spin requirements, a goal of AAURocket will be to ensure a “limited spin” orbital insertion. It is then up to each satellite to utilize their own attitude and control system to obtain their desired spin (if any). There will be no specific requirements for AAURocket with regards to the spin at the time of deployment of the payload. However the spin should be kept to a minimum to ensure the best possible deployment of the payload.

3.2 Subsystem Requirements

In general AAURocket can be split in to two primary subsystems. A position and attitude determination subsystem, and a control subsystem. The requirements for each of these subsystems are determined with respect to the primary mission requirements set forth in 3.1.

Position and Attitude Determination system requirements

The Position and Attitude Determination systems task is, as the name implies, to determine the position and attitude of AAURocket. It has been chosen that the Position and Attitude Determination system can be responsible for a maximum of 35% of the total deviation with respect to the mission requirements for position and velocity.

Requirement	Parameter	Value	Unit
SS1.R1	Position determination accuracy	<0.7	[km]
SS1.R2	Velocity determination accuracy	<2.1	[m/s]

Table 3.4: Subsystem 1 requirements.

Control system requirements

It has been chosen that the control system for AAURocket can be responsible for a maximum of 65% of the total deviation with respect to the mission requirements for position and velocity.

Requirement	Parameter	Value	Unit
SS2.R1	Position control accuracy	<1.3	[km]
SS2.R2	Velocity control accuracy	<3.9	[m/s]

Table 3.5: Subsystem 2 requirements.

Chapter 4

Mechanical Design

4.1 Basic elements of a rocket

Source: [Turner 01]

In this section the basic elements of rockets used for space launches will be outlined.

Typically a rocket launch vehicle (LV) is a complex construction consisting of a lot of different mechanical elements and subsystems. However, the rocket can in general be divided into several major components as follows:

- Payload
- Propellant
- Engine
- Structure

Payload The payload is a general term used for the dead-weight that is to be transported from point a to point b. In military ballistic rockets this is usually an explosive warhead, whereas for space rockets it is commonly satellites or probes.
This part is commonly placed at/in the top/nose of the rocket.

Propellant The propellant of a rocket is the matter that is to be expulsed from the engine in order to gain velocity. In chemical rocket LVs the propellant consists of two primary components: fuel and oxidizer. The main reason for bringing oxidizer is the obvious lack of air (oxygen) as the rocket gains altitude.
This part is commonly placed within the main body of the rocket.

Engine The engine of the rocket is the machine which converts the potential energy of the propellant into kinetic energy. This part is commonly placed at the bottom of the rocket.

Structure The structure of the rocket is the entire mechanical skeleton of rocket. Furthermore, the structure also includes the propellant container tanks inside the shell of the rocket.
The component grouping both the structure and engine is referred to as the dry weight of the rocket. Depending on the author the engine is sometimes made to include the structure of the rocket.

4.1.1 Staging

A core meta concept of the rocket LV design is staging. The idea of staging, as mentioned in section 1.1, is a concept of combining several rockets into a bigger rocket, either in a serial connection (train-cart like setup as coined by Tsiolkovsky), or in parallel.

Serial The most classic of the two configurations is the serial connection which utilizes a larger bottom stage (stage 1) as an initial thrust rocket, which accelerates the remaining stages to some velocity. When this first stage has consumed all its propellant it is ejected, and after a short coasting period the next stage (stage 2) is ignited. This procedure can be repeated for as many stages as the rocket is designed for.

Parallel The more modern approach is the parallel staging (most famous is the NASA Space Shuttle) where a primary rocket utilizes two or more parallel attached boosters for the initial flight of the ascent. Normally parallel staged rockets never exceeds two stages. That is; an initial burn phase with the main rocket and boosters ignited, until booster ejection after (or just preceding) booster burnout. The rest of the ascent will be performed solely by the main rocket.

4.2 Comparison of Rocket Stage Setups

In this section the characteristics of single-, mutli-, and a infinite-staged rockets will be outlined and compared. Furthermore advantages and disadvantages of each setup will be considered. Finally these considerations will be used to determine the setup for AAURocket.

Efficiency is a broad term that can be applied to a wide variety of different elements of a system. In this section the efficiency that is studied is the ability of a rocket to obtain a certain velocity. Engine and/or aerodynamic efficiency parameters are not considered in this section.

4.2.1 Tsiolkovsky's Rocket Equation

In order to study a rockets ability to obtain a certain velocity, it is first needed to study the equations that guard rocket velocity.
At the heart of all rocket theory is the classical Newtonian laws of motion, where the third law is of particular importance. Newtons third law specifically states:

> *When one body exerts a force on a second body, the second body simultaneously exerts a force equal in magnitude and opposite in direction on the first body.*

This is the principle that all existing rockets utilize to obtain velocity. For chemical rockets that means combusting fuel and oxidizer thereby expelling an exhaust stream in one direction and obtaining a reactionary velocity in the opposite direction of expulsion. By combining this principle with Newtons second law, the thrust equation is obtained, which can be seen in equation 4.1

[Turner 01].

$$\mathbf{T} = \dot{m}\mathbf{v}_e \tag{4.1}$$

where

$\mathbf{T}$	is the thrust force applied on the system	[N]
$\dot{m}$	is the rate of which mass is expelled (mass flow)	[kg/s]
$\mathbf{v}_e$	is the velocity of the mass being expelled	[m/s]

From this equation it is clear that a proportionality exists between the mass flow and the velocity at which the mass is expelled. This means that a given thrust can be obtained either by expelling a lot of mass at a relative slow speed, or expelling very little mass at a much greater speed.

By using the thrust equation (directly relatable to Newtons second and third law), Konstantin Tsiolkovsky was in 1903 able to derive the rocket equation, which has since been dubbed Tsiolkovsky's rocket equation (see equation 4.2) [Turner 01].

$$\Delta v = v_e \ln\left(\frac{m_0}{m_f}\right) \tag{4.2}$$

with

$$m_0 = m_S + m_P + m_{PL} \tag{4.3}$$

$$m_f = m_S + m_{PL} \tag{4.4}$$

where

Δv	is the change in velocity	[m/s]
v_e	is the effective exhaust velocity of the propellant	[m/s]
m_0	is the initial total mass of the rocket	[kg]
m_f	is the final total mass of the rocket	[kg]
m_S	is the structural mass of the rocket	[kg]
m_P	is the mass of the propellant	[kg]
m_{PL}	is the mass of the payload	[kg]

The general principle of the rocket is illustrated in the free body diagram shown on figure 4.2.1, where an initial mass m is moving with velocity v at time t. The mass is then accelerated by ejecting a mass Δm with velocity v_e in opposite direction and the remaining mass $m - \Delta m$ is accelerated to the velocity $v + \Delta v$ at time $t + \Delta t$.

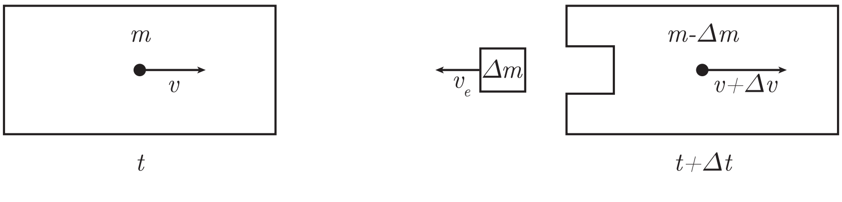


Figure 4.1: Free body diagram of the principle behind the rocket equation [Turner 01].

It is clear that from the rocket equation it is possible to calculate velocity increments in field free space by knowing the effective exhaust velocity of the rocket and the amount of mass being

expelled (difference of m_0 and m_f). Obviously it is also possible to calculate the amount of propellant mass needed to be expelled at some v_e to obtain a certain Δv.
The effective exhaust velocity is an efficiency measure of the individual rocket engine. It is closely related to the velocity component from the thrust equation (4.1), that is the velocity which the propellant is expelled from the engine.

Note that the rocket equation is limited to theoretical impulsive maneuvers and as such whenever the equation is applied for practical use, deviations will arise (seeing as no maneuver can ever be instantaneous).

In order to evaluate the performance of different rocket staging configurations it is convenient to rewrite the rocket equation to:

$$\Delta v = v_e \ln\left(\frac{1+\lambda}{\epsilon+\lambda}\right) \tag{4.5}$$

with

$$\lambda = \frac{m_{PL}}{m_0 - m_{PL}} \tag{4.6}$$

$$\epsilon = \frac{m_S}{m_0 - m_{PL}} \tag{4.7}$$

where

λ	is the payload ratio	[$\cdot$]
ϵ	is the structural ratio	[$\cdot$]

The structural mass of the rocket (m_S) includes both the rocket structure as well as the engine and other "empty" masses. The payload mass only encompasses the payload, however, as mentioned in 1.2.3 the payload mass for this project also includes the structure and fearing for the nose of the rocket.

Specific Impulse and Effective Exhaust Velocity

For convenience the efficiency of a rocket is also described in specific impulse. The relation between effective exhaust velocity and specific impulse can be seen in equation 4.8.

$$v_e = I_{sp} g_0 \tag{4.8}$$

where

I_{sp}	is the specific impulse	[s]
g_0	is the gravitational acceleration of the Earth at sea-level	[m/s^2]

Specific impulse is merely a reformulation of the effective exhaust velocity but has garnered popularity as the space industry has expanded to countries outside of the United States/NASA. The specific impulse thus removes the worry of efficiency measures being defined in units of [feet/s] or [m/s].
An intuitive understanding of the specific impulse is to think of it as the time the different rockets are able to hover above the surface of the Earth at sea level with the same amount of propellant.

4.2.2 Single Stage Rockets

By using Tsiolkovsky's rocket equation it is possible to evaluate the maximum velocity increment a given single staged rocket can obtain. This evaluation will serve as a performance benchmark

to which multi- and infinite staged rockets will be compared. As such some standard parameters will be defined in the following.
The effective exhaust velocity's typical range for chemical rockets is between 2450 to 4400 [Curtis 05], so for this preliminary single stage analysis, a value of $v_e = 3000$ [m/s] is chosen. Furthermore, the structural ratio ϵ is chosen to be 0.1 (which is the current lower limit with todays mechanical technology [Curtis 05]), with a payload ratio λ of 0.05.
Using these values, a theoretical single stage rocket will then (in field free space) be able to obtain a velocity increment of:

$$\Delta v = 3000 \ln \left(\frac{1 - 0.05}{0.1 + 0.05} \right) = 5538 \,[\mathrm{m/s}] \tag{4.9}$$

From figure 2.5 it is seen that the required orbital speed at the surface of the Earth (in field free space) is 7910 [m/s]. This means that the above single stage rocket is not able to make a successful orbital injection. In fact for this single stage rocket to be capable of reaching the required orbital speed, the effective exhaust velocity has to be improved to:

$$v_e = \frac{7910}{\ln \left(\frac{1-0.05}{0.1+0.05} \right)} = 4285 \,[\mathrm{m/s}] \tag{4.10}$$

As mentioned the current most efficient chemical rocket engines are only able to obtain a maximum effective exhaust velocity of 4400 [m/s], which means that it is theoretically possible to launch a single stage rocket to orbit, however, such a project has never been undertaken. This is mostly due to practicality and economic cost, since it is easier to use less efficient engines and combine them in a multi stage setup [Curtis 05].

4.2.3 Multi Staged Rockets

For multi stage rockets each stage will successively perform some velocity increment Δv_k, up to a final velocity of $\sum_{k=1}^{N} \Delta v_k$, where each Δv_k is calculated using the rocket equation. It is clear the first stage of such a rocket will have to propel the entire mass of all successive stages to some velocity, at which point the next stage will take over. This sequence will be repeated N times until all the stages has been spent and the payload has reached the final velocity.

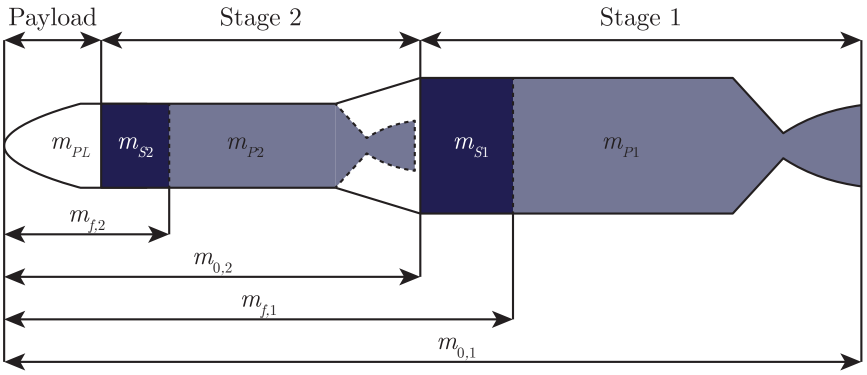


Figure 4.2: Mass distribution and sections of a multi, in this particular case two, staged rocket.

On figure 4.2 a theoretical two stage rocket is pictured. In order to make a performance evaluation on such a rocket the masses has to be organized correctly for each stage.

In principle the top most part of this rocket (Stage 2 + Payload) can be studied as the previous single stage rocket, since it comprises the same basic mass elements: m_{PL}, m_{S2}, and m_{P2} (see figure 4.2.
The first stage of this multi stage rocket can also roughly be considered in the same way. However, the payload of this stage is instead the entire second stage. This means that if the rocket equation is to be applied to this stage, then m_{PL} of Stage 1 is now equal to $m_{0,2} = m_{PL} + m_{S2} + m_{P2}$. This mass organization is repeated for each additional stage that is utilized for the multi stage rocket.
It is clear that the velocity calculations for a multi stage rocket will become tedious if each stage utilizes different values for λ, ϵ, and v_e. However, if each stage is assumed to be identical (in ratio) then it is possible to reformulate the rocket equation as follows [Curtis 05]:

$$\Delta v_N = v_e \ln\left(\frac{1}{\pi_{PL}^{\frac{1}{N}}(1-\epsilon)+\epsilon}\right) \tag{4.11}$$

with

$$\pi_{PL} = \frac{m_{PL}}{m_0} \tag{4.12}$$

where

π_{PL} is the payload fraction [·]

The benchmark values for λ, ϵ, and v_e defined in subsection 4.2.2 will be used for each stage of the multi stage rocket in order to construct a benchmark performance graph of a multi stage rocket setup. The values $\lambda = 0.05$ and $\epsilon = 0.1$ will in turn result in $\pi_{PL} = 0.0476$. The graph on figure 4.3 was constructed by applying these parameters to equation 4.11 and using a vector space of N={1,2,3,...,10}.

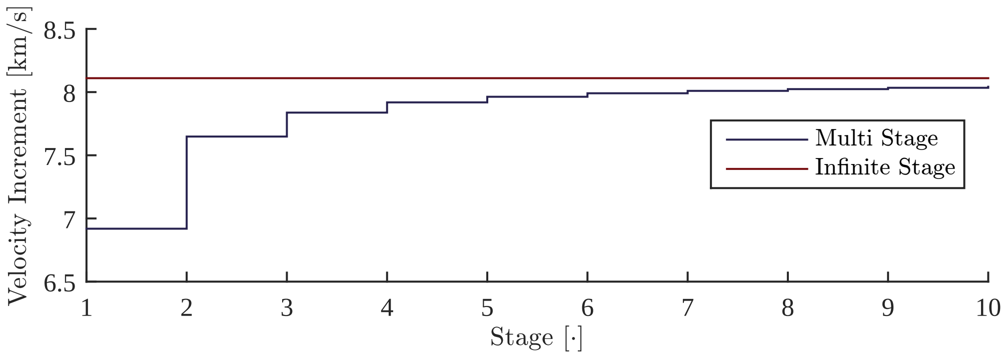


Figure 4.3: Velocity increments for each stage in a multi- and infinite stage rockets.

It is clear that a lot of extra performance is gained by using a multi stage setup, however, each additional stage suffers greatly from diminishing returns. As such, when combined with practicality and complexness of construction, rockets rarely utilize more than three stages [Curtis 05].

4.2.4 Infinite Staged Rockets

The infinite staged rocket is in principle a specific case of a multi staged rocket. For a classical chemical rocket it is a theoretical case of attaching an infinite amount of stages on top of each other and then calculating the maximum velocity increment this LV is able to obtain. The equation related to this calculation is derived from equation 4.11 by letting the amount of stages tend to infinity, which will result in [Curtis 05]:

$$\Delta v_N = v_e \left(1 - \epsilon\right) \ln \frac{1}{\pi_{PL}} \tag{4.13}$$

Applying the same values of $\epsilon = 0.1$ and $\pi_{PL} = 0.0476$ a maximum Δv can be found for the infite staged version of the multi staged rocket of subsection 4.2.3:

$$\Delta v_N = 3000 \left(1 - 0.1\right) \ln \frac{1}{0.0476} = 8220[\mathrm{m/s}] \tag{4.14}$$

This result is also shown on the graph on figure 4.3.
As mentioned the infinite staged rocket is mostly a theoretical case, however, there are concepts being developed which allow for a rocket to be considered infinitely staged. One such concept is the Dniepropetrovsk National University (DNU) self-foraging rocket which is described in further detail in section 4.3

4.3 Proposed Concept of the Self-Foraging rocket

In this section the concept of a self-foraging rocket will be outlined. Furthermore, the initial work done at DNU and their ideas will be outlined.

4.3.1 The DNU Self-Foraging rocket

This section is based on information obtained from DNU and the papers [Yemets 08a] [Yemets 08b].

Yemets et al. at DNU has proposed a new concept for LVs in an effort to realize a smaller and cheaper alternative to the several hundred million dollar LVs that exists today. The concept is that of a self-foraging rocket which consists of three primary elements:

- Payload
- Propellant
- Engine

A basic schematic of the concept can be seen on figure 4.4. When compared to the basic rocket elements in section 4.1, it is clear that the major difference between the classical and the self-foraging rocket is the lack of the structure element. While structure is of course still present on the self-foraging rocket, it is vastly reduced. This is a result of utilizing a solid fuel source which doubles as the main structure of the rocket body and functions as an oxidizer container.

During flight the rocket will consume the shell by heating and gasifying the material and mixing it with the oxidizer in the combustion process thereby expelling the propellant and gaining forward thrust.
It is clear that for this process to work the engine of the rocket has to move its way up towards the payload to consume the shell and oxidizer. In the original concept from DNU it is proposed that this is done passively by exposing the rocket to a constant acceleration throughout the ascent, which in turn will apply an opposite force on the body driving it into the engine.

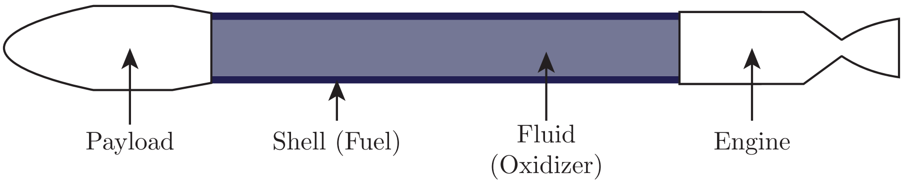


Figure 4.4: The self-foraging rocket concept as proposed by Yemets et al. at DNU. The middle section (shell and fluid) is the part of the rocket that is gradually consumed during flight. In the case where 100% of the propellant is consumed, only the Payload attached to the Engine will remain [Yemets 08b].

Fuel and Oxidizer mix

Source: [Yemets 08a] [Yemets 08b]

One of the most vital aspects of the self-foraging rocket is the fuel, seeing as this material has to be strong enough to serve as the main body and structure of the rocket, while also have the ability to be consumed and mixed with a suitable oxidizer to produce a chemical reaction and provide thrust.
For this purpose Yemets et al. suggests using High Density PolyEthylene (HDPE) which is a general purpose thermoplastic. They estimate that the strength of HDPE should be sufficient as shell material for a self-foraging rocket.
Furthermore, during their initial combustion tests of their self-foraging rocket engine, Hydrogen Peroxide (HPO) was used as suitable oxidizer.

4.4 Subconclusion

In this chapter the basic elements of standard chemical rockets has been outlined. Furthermore three different performance evaluations were made on the different type of staging setups: single-, multi-, and infinite staged.
Finally the self-foraging rocket concept (an infinite stage rocket) from DNU was presented, along with their design ideas and choices.

Mechanical Setup of AAURocket

From the performance evaluations of rocket staging it is clear that the single stage solution present significant challenges in designing a state of the art rocket engine with regards to efficiency. As such this is not considered a feasible staging setup for AAURocket.
While the multi stage setup is the classical and most well tested approach to rocket design, the focus of this project will instead be on utilizing some of the conceptual ideas from DNU and attempt to design a self-foraging rocket (which in turn will be considered infinitely staged), thereby creating a cheap and efficient alternative to modern LVs.
In particular the AAURocket will be designed to use HDPE as the fuel with HPO as suitable oxidizer. Furthermore, the HDPE will be used to construct the shell that will contain the oxidizer, and act as the main body of the rocket.
However, AAURocket will not be foraging itself by the passive pressure applied from the acceleration during the ascent to orbit. Instead an active method is desired which can control the shortening rate (and in turn the fuel consumption) of the rocket.

Chapter 5

Guidance-Hardware Design

In this chapter common sensor and actuator types will be presented, and a choice made in each category on what types will be utilized for AAURocket. Additionally the general mechanical design will be of the AAURocket engine block will be outlined, along with the constraints it imposes. Furthermore, different acceleration profiles are presented to outline the advantages and disadvantages of different thrust strategies.
Finally all the thrusters utilized on AAURocket will be mechanically dimensioned.

5.1 Position and Attitude Determination System

Sources: [Email correspondence with Copenhagen Suborbitals, Appendix A]

When a rocket is launched one of the most important issues is to have accurate knowledge of the position, velocity, attitude, and angular rate. Knowledge of the attitude is the only thing that ensures that actuation of the rocket happens in the wanted direction. The most common sensor system used when estimating rocket attitude and velocity are so-called Inertial Measurement Units (IMUs). An IMU is a sensor system consisting of accelerometers and gyroscopes, and in some cases also magnetometers. This section aims to analyze which sensors would be advantageous in order to estimate the states of AAURocket.

5.1.1 Sensor considerations

There exist several sensors which can be used in the matter of determining attitude, velocity, and position of a rocket. In general these sensors can be split in to two primary categories: reference sensors and inertial sensors.
Reference sensors utilizes external reference points in order to give valid measurements. In the case of a horizon sensor the reference point is the horizon itself and the measurement (in some cases) is a vector pointing towards the center of the Earth. The inherent drawback of reference sensors arises once the external reference is not observable, e.g. the horizon is no longer visible for the horizon sensor.
Inertial sensors measure changes in physical states of the system with respect to an inertial reference frame. An example is the gyroscope which measures the angular acceleration of the system it measures. Due to the inertial property of the sensor, it will never rely on external conditions in order to function and give a measurement. Position and/or attitude change can be obtained from these sensors via integration, but an initial conditions or known fix points has to be determined in order to obtain actual position and/or attitude.
While it is in theory possible to launch a rocket into an orbit using only inertial sensors, it is more reliable to combine these sensors with reference sensors to obtain a better estimation system

overall. Common candidate sensors for rocket position and attitude determination systems are as follows (where [R] denotes reference sensor and [I] denotes inertial sensor):

[R] Global Positioning System (GPS)

[R] Magnetometer

[R] Pressure sensor

[R] Horizon sensor

[R] Earth sensor

[R] Sun sensor

[R] Star Tracker

[R] Ground based observation

[I] Accelerometer

[I] Gyroscope

Several of the above sensors can be combined into a single IMU. The usual sensors used to comprise an IMU are accelerometers and Gyroscopes, however, depending on application more sensors can be included.
These IMUs can either be constructed by combining several individual bought sensors, or bought as pre-combined package. The IMU is not constrained to use any particular production method, and as such any IMU can be made using Micro-Electronic-Mechanical-System (MEMS)-, mechanical-, optical-, and/or purely electrical-components. Advantages of MEMS IMUs are reduced price compared to their mechanical or optical counterparts, however, this inherently means lower performance (including accuracy) as well.
Due to pricing concerns it is expected that the primary IMU type of interest is the MEMS type. Only in the case of unacceptable performance will sensors of higher precision (and price) be considered.

5.1.2 Sensor configurations on other rockets

To determine what kind and how many sensors to use for the AAURocket, two existing sensor configurations on operational rockets are considered. The specifications for these rockets have been obtained via correspondence with Copenhagen Sub-orbitals (CS).
CS launched their first Sapphire rocket on June 23 2013. The data from this launch, sampled from their chosen IMU (ADIS16367) has been provided by CS. Table 5.1 shows the two sensor configurations for their Sapphire and NEXØ rockets.
[Maestro 13]

From table 5.1 it can be deducted that the CS Sapphire rocket purely utilizes inertial sensors to determine its position and attitude. CS also notes that no sensor fusion was used between the accelerometer and gyroscope during their first flight, which was sufficient seeing as the burn-time on this rocket only lasted 11 seconds. CS are, however, concerned about accuracy during longer flights since the precision of the gyroscope decreases over time due to drift.
The CS NEXØ rocket is the successor of the CS Sapphire, which is also reflected in the increased burn time of 82 seconds. Due to the aforementioned concerns of gyroscopic drift the IMU for this rocket has been changed along with development of a sensor fusion method that they intend to use for this flight.

Rocket	IMU Model	Sensors
CS Sapphire	ADIS16367	Accelerometer Gyroscope
CS NEXØ	ADIS16448	Accelerometer Gyroscope Magnetometer Pressure sensor

Table 5.1: Sensor configuration for the CS Sapphire and CS NEXØ rockets. Listed is the type of sensor along with the serial number of each component.

5.2 AAURocket Sensor Choice and Characteristics

Since the mission of AAURocket is to reach a sustainable orbit, the requirements for the sensors are in turn higher. The ADIS16448 is chosen to be utilized on the AAURocket position and attitude determination system due to the following considerations:

- Availability of flight data for the ADIS16xxx series (provided by CS)
- Higher precision than the ADIS16367
- Combined reference and inertial sensors (inclusion of magnetometer in particular)

In table 5.2 the measurement type of each of the sensors included in ADIS16446 is listed. As can be seen all the reference sensors are not able to fully determine either position nor attitude on their own. Furthermore, in the case of the pressure sensor its effectiveness only lasts as long as the atmosphere is still sufficiently dense, which reduces the effectiveness of this sensor the closer the rocket gets to its destination. In fact for most of the ascent to orbit this sensor will not be able to measure anything useful with respect to state estimation. However, it is a useful sensor in the case of estimating the thrust of the LV based on input signals.

Sensor	Position [R]	Position [I]	Attitude [R]	Attitude [I]
Pressure sensor	×[1]			
Accelerometer		×		×[3]
Magnetometer			×[2]	
Gyroscope				×

Table 5.2: The correlation between the sensors of the ADIS16446 and the respective measurement types. [1]: Reduced position. Only height above sea-level is measured. [2]: Reduced attitude. Can only determine 2 out of 3 degrees of freedom. [3]: Only possible when the sensor is displaced from the center of mass.

From these features there is a concern that the position determination can drift over time, seeing as the only reliable sensor functioning for the full duration of the ascent to orbit is the accelerometer.

As such in the case where the performance of the position and attitude determination algorithm is not sufficient, an additional sensor can be considered.

GPS Considerations

A sensor of particular interest would be a GPS, which is a reference sensor able to determine the exact position of the rocket. However, strict constraints are applied to GPS units by the manufacture. This is part of the Coordinating Committee for Multilateral Export Controls (CoCom) by the United States of America foreign policy that limits all exportable GPS products to a maximum measurable altitude and velocity of 18km and 514.4 m/s [Ravtrack 15]. These policies primary function is to ensure that exported GPS units are not used for foreign military purposes (in particular for ballistic missile use).
It is possible though, to get special permission to buy GPS units which has these limitations bypassed if significant evidence can be provided for peaceful use of the unit. Such permission falls under the rules of the United States' International Traffic in Arms Regulations (ITAR) [of State 15].

Due to these limitations the initial sensor setup will include a token GPS unit, since it is assumed that such an unlocked GPS can be acquired.

Inertial Measurement Unit: ADIS16448

Table 5.3 shows the general characteristics of each sensor included in the ADIS16448.

ADIS16448

Sensor	**Property**	
Gyroscope	Type	Triaxial digital angular rate gyroscope
	Range	±1000 °/s
	Sensitivity	0.04 °/s
	Output noise	0.27 °/s RMS
	Linear acceleration effect on bias	0.015 °/s/g
Accelerometer	Type	Triaxial digital accelerometer
	Range	±18 g
	Sensitivity	0.833 mg
	Output noise	5.1 mg RMS
Magnetometer	Type	Triaxial digital magnetometer
	Range	±1.9 G
	Sensitivity	142.9 µG
	Output noise	2400 µG RMS
Barometer	Type	Digital barometer
	Range	1-120 kPa
	Sensitivity	2 Pa
	Output noise	8 Pa RMS
	Maximum Refresh Rate	1100 Hz

Table 5.3: The characteristic values of each sensor included in the ADIS16448 [Devices 14].

The general sensor parameters of the token GPS can be seen in table 5.4.

Sensor	**Frequency**	**Resolution**	**Variance**	**Range (Height)**
GPS	1100 Hz	0.005 m	10 m	1000 km

Table 5.4: Token GPS sensor parameters.

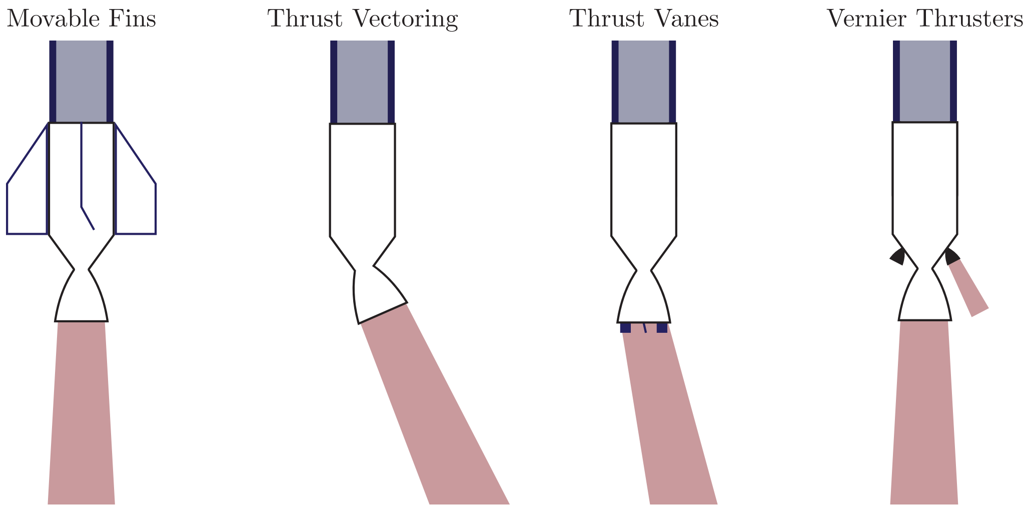


Figure 5.1: Different examples of rocket directional actuators.

5.3 Actuators

Source: [Sutton 10] [NASA 14]

For classical rockets the matter of actuation is generally two-fold. Primarily a main propulsion engine to drive the translational motion of the rocket, and secondly a smaller system to guide the direction of flight.
Seeing as the method of primary propulsion has already been established to be a chemical rocket engine (as seen in chapter 4), the following section focus on the secondary directional actuators.

5.3.1 Directional Actuators

A set of general common directional actuators for classical chemical rockets, that include: Movable Fins, Gimbaled Thrust, Thrust Vanes, and Vernier Thrusters, can be seen in figure 5.1, and will be discussed in the following.

Movable Fins The movable fins are the simplest of the aforementioned actuation methods. It utilizes the atmosphere to actuate the rocket by changing the drag-profile of the system when moving its fins. The movable fins are one of the simplest and easiest methods to implement on a rocket, seeing as it is merely consisting of a number of aerodynamic control surfaces of which a part of them can be tilted in either direction. Of the different methods pictured on figure 5.1 they are also the only actuators to passively provide an amount of stability for the rocket during flight.
One of the major drawbacks of this actuator is its inherent dependability of an atmosphere. It is obvious that while this method can be a good solution for a military ballistic missile, its use on a LV is of limited use. Furthermore, though the movable fins can provide passive stability it comes at the price of overall increased aerodynamic drag.

Gimbaled Thrust The most mechanically advanced of the listed actuation methods is the gimbaled thrust. In principle the entire main chemical engine has been mounted on a movable gimbal system, thus enabling the rocket to change flight direction by changing the thrust vector

via gimbaling. This is also the reason for this system being listed as the most advanced. The entire mechanism of the primary propulsion engine being mounted on a gimbaling system can be a mechanically very complex task. The advantage of this setup, however, is that the magnitude of the thrust vector always remains the same, regardless of control action.
The primary drawback of gimabled thrusting is it only works when the main chemical engine is turned on. As soon as the primary engine is shut down, all controllability is lost from this method. Furthermore, it is the only technology of the listed methods that does not have the ability to actuate in the roll axis of the rocket. This can only be achieved if several main engines are used in parallel.

Thrust Vanes The thrust vanes attempt to manipulate the primary exhaust stream giving similar thrust vectoring results as the gimbaled thrust. However, the way of manipulation is different in the sense that the engine is locked, but instead has movable vanes inside the exhaust stream. These flaps can then deflect the exhaust stream in the desired direction by a combination of different vane tilt positions.
The main drawback of the thrust vanes approach lies in their inherent positioning. The temperature of rocket exhaust streams are upwards of several thousand degrees celsius. This means that the vanes usually have very limited life times since they rapidly degrade during the ascent. Furthermore, the thrust vanes also acts as a retarding force on the primary thrust of the rocket, since they are placed within the exhaust stream. Mechanically the vanes are easy to implement as they can be implemented externally on the main rocket engine.

Vernier Thrusters The vernier thrusters are like the movable fins completely decoupled from the main chemical rocket engine. In principle they are small propulsion engines displaced from the center of mass and at an angle on the rocket. By turning a combination of them on, the total thrust vector of the rocket is in turn affected.
While the vernier thrusters are mechanically easy to implement, they pose greater impact on the control scheme. This is due to the fact that whenever the vernier thrusters are active they provide both translational and rotational force to the rocket. The impact of the translational force can often be neglected since the main rocket engine provides forces of magnitudes many times greater than the vernier thrusters. However, if the main engine is significantly small (in the case of throttling or later staging) then the vernier thrusters will in turn have a bigger impact whenever control action is needed.

Choice of Directional Actuator

The choice of directional actuator for AAURocket has fallen on the vernier thrusters. The vernier thrusters are slightly more complex to implement compared to the movable fins, but on the contrary they offer the possibility to control the orientation when the rocket leaves the inner atmosphere of earth. The reason for choosing the vernier thrusters over the thrust vanes and the gimbaled thrust is the possibility to control the rotation about the roll axis of the rocket. This has advantages, both in the sense of being able to choose to spin stabilize the rocket, and also in the case that it is necessary to limit or stop the rotation about the roll axis.
Furthermore, it is deemed advantageous to be able to perform control when the main chemical engine is turned off.

5.4 Engine Design

In this section different general thrust profile approaches will be outlined. In turn these general thrust profiles will be used to determine the best possible thrust profile option for AAURocket, through some general constraints.

Additionally the primary mechanical structure of the AAURocket engine block will be designed on the basis of the general throttle and actuation needs of the ascent profile of AAURocket. Furthermore the general thermodynamic equations of chemical rocket engine will be presented and finally used to dimension all of the thruster of AAURocket.

5.4.1 Thrust Profile

In order to dimension a rocket engine the first and primary factor to determine is how much thrust it should be able to produce. To do this a general thrust strategy for the ascent to orbit has to be chosen. The entire thrust scheme used during the ascent to orbit is referred to as the thrust profile of the rocket.
A thrust profile for a given LV is generally tailored very specifically to said LV, and can employ varying levels of throttle and/or coasting periods. These different thrust profiles can in theory be bounded between two extremities:

- Constant Thrust
- Constant Acceleration

Using static thrust

If an engine is designed with constant thrust in mind, the result will be an LV which will undergo faster and faster acceleration. This is caused by the fact that the more fuel the rocket spends, the lighter it gets, and the more acceleration the rocket will experience from the constant thrust.
This can pose a danger to a rocket launched from the surface of the earth into orbit using a single engine, since the mass expulsion needed for this manoeuvre is significant, thus requiring a high thrust at sea level. In turn this scenario could result in the rocket experiencing a couple of g's acceleration at take-off, while just before burnout the g-force could easily be upwards of 20 times as high. With g-forces at those levels there is a high risk that the rocket will be destroyed by the thrust it imposes on itself.
Usually this thrust profile scenario is overcome by rockets utilizing staging, and in turn different rocket engines for each stage. This means that for low altitudes a large thrust engine can be used (i.e. while the rocket is heavy) which can then be discarded when a certain velocity increment has been obtained, and afterwards a new and smaller thrust engine will be utilized.
An example of a (mostly) static thrust multi stage rocket is the Ariane 5 rocket [Arianespace 11b]. It has three stages during its flight to orbit: two solid rocket boosters, a main cryogenic stage and an upper stage. In figure 5.2 the staging of the Ariane 5 rocket can be seen during its flight to orbit.

A corresponding g-profile of the Ariane 5 rocket can be seen in figure 5.3. As can be seen from the graph the acceleration of the Ariane 5 rocket grows exponentially as the fuel gets spent and the mass of the rocket reduces.

Constant thrust in a chemical rocket engine will, however, remove the need for the ability to throttle the engine. This in turn can lead to simpler design of the engine [Turner 01].

Using static acceleration

In order to have an LV that can employ constant acceleration, the engine will need to have the ability to throttle. This throttle is utilized to gradually decrease the level of thrust of the LV to match the weight loss of the system, which in turn results in a constant acceleration. While constant acceleration is a plausible thrust profile, it is seldomly used single handedly throughout

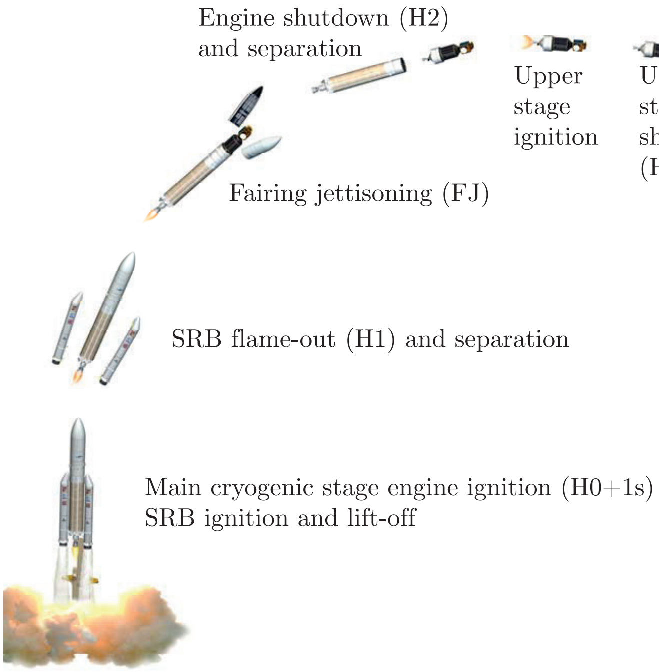

Engine shutdown (H2)
and separation
Upper stage ignition
Upper stage shutdown (H3)
Fairing jettisoning (FJ)
SRB flame-out (H1) and separation
Main cryogenic stage engine ignition (H0+1s)
SRB ignition and lift-off

Figure 5.2: The staging sequence of the Ariane 5 rocket during flight to orbit [Arianespace 11b].

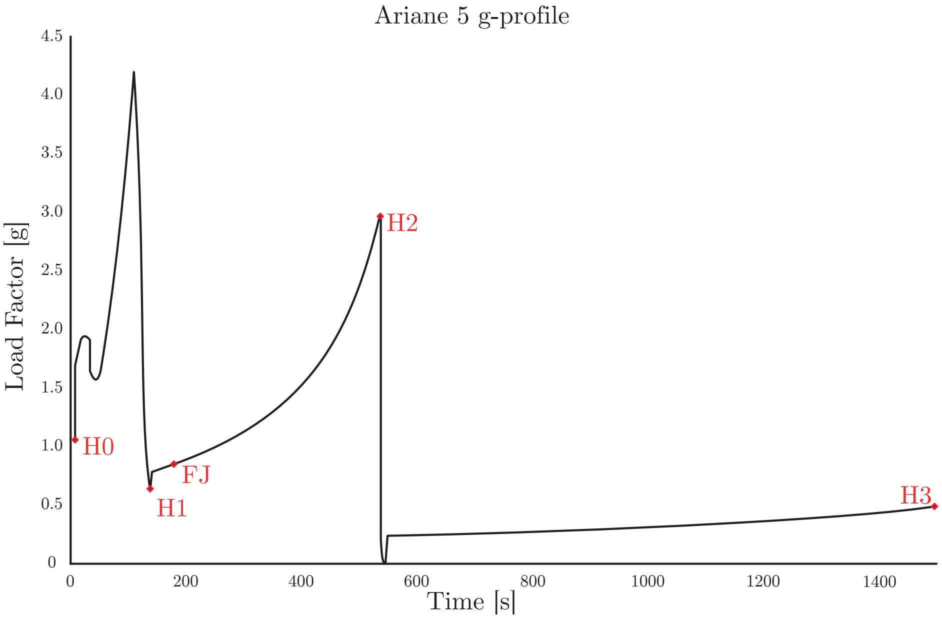


Figure 5.3: The g profile of the Ariane 5 rocket. Each of the peaks represents a staging in the rocket. At the first peak the two solid booster rockets are ejected. At the second peak the primary rocket engine is ejected and only the orbiter remains [Arianespace 11b].

the ascent. The usual application is to limit the maximum g-load the the LV experiences to an upper bound, which can occur after a longer period of constant thrust.

Throttle While the concept of throttle is simple, the execution of throttle in a chemical rocket engine has inherent disadvantages due to some of its static parameters, eg. combustion chamber volume. As such chemical rocket engines rarely employ full throttle schemes of 0-100%. An example is the space shuttles main engine which had a throttle range of 65-109% of the nominal range [Lethbridge 98].
Currently the RL-10 engine (which can be considered state of the art) is able to employ throttling capabilities of 8-100% [NASA 09].

Thrust Profile Engine Constraints of AAURocket

Due to the nature of the self-foraging rocket design of AAURocket several characteristic constraints are posed for the thrust profile.
Firstly, it is deemed necessary for the LV to be able to throttle its engines during its ascent phase as the mass will undergo significant changes from launch to orbital injection.
Secondly, during the ascent phase the thrust is constrained to never equal zero, as this will imply the engine has been turned off. Seeing as the gasification process of the fuel shell relies on heat, it is expected that there is a significant start up process related to turning the engine on, due to a needed build up in thermal momentum.

These engine constraints are used in section 8.2 to calculate the thrust profile used for the ascent to orbit of AAURocket.

5.4.2 Structure of the Engine Block

It is, at best, optimistic to consider throttle capabilities as good as the RL-10 rocket engine, however, a reduced throttle profile may indeed be possible. Hence a throttle capability of [50%;100%] is assumed feasible for any single independent engine designed for AAURocket.
This limitation combined with the throttle range needed of [2.5%;11.5%] $\bigcup$ [28%;100%] (as seen in section 8.3.2), means that AAURocket will not be able to utilize a single main chemical engine setup. Instead by using five independent chemical engines in conjunction a wider thrust profile can be employed. On figure 5.4 the physical design of such a setup is shown.

In this design four larger engines (which will be referred to as boosters) are symmetrically placed around a smaller engine (which will be referred to as the orbital thruster) in the middle. Each thruster is able to utilize a throttle range of [50%;100%]. Furthermore, in this setup it is possible to turn off some of the engines completely while the rest remains on through the remainder of the ascent.
The setup has the orbital thruster performing in the range of [2.5%;5%] of the maximum thrust, T_{max}, and each of the four larger boosters will be performing in the range of [11.875;23.75%] of T_{max}.
As such, when all five engines are turned on the thrust will result in full T_{max}. Conversely, with five engines turned on, but each at 50% throttle, a total thrust of $0.5T_{max}$ will be produced.
If less thrust is required two oppositely positioned boosters can be turned off completely. The remaining two boosters combined with the orbit engine now has a thrust range of $0.2625T_{max}$ to $0.525T_{max}$.
Finally all four boosters can be turned off which results in the orbital thruster being able to perform $0.025T_{max}$ to $0.05T_{max}$.
This means that AAURocket will have an effective throttle range of [2.5%;5%] $\bigcup$ [26.25%;100%].
An illustration of these ranges can be seen on figure 5.5

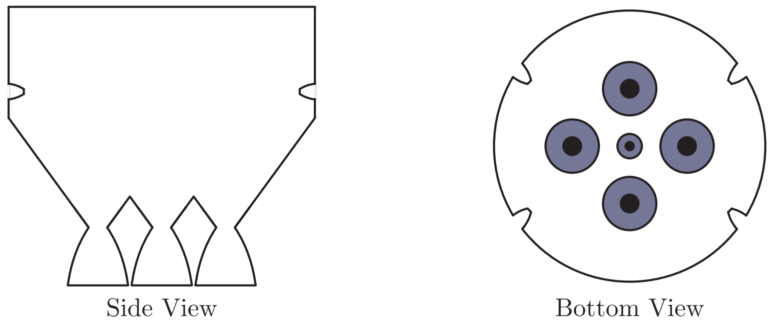


Figure 5.4: Physical structural design of the engine block of AAURocket. The engine consists of five primary thrusters with no gimbal capabilities. Each larger thruster (referred to as "booster") will perform in the range [11.875;23.75%] of T_{max} and the smaller center thruster (referred to as "orbital" thruster) will perform in the range [2.5%;5%] of T_{max}. The indents observed in the top of the "Side View" and on the outer circle on the "Bottom View" are vernier thruster nozzles.

It is clear that this does not fully meet the throttle profile calculated in section 8.3.2. However, seeing as the range of [2.5%;11.5%] is only needed at the point of orbital injection, it is expected that this manoeuvre can be overcome by a longer burn at 5%, hence, the presented engine setup is deemed satisfactory.

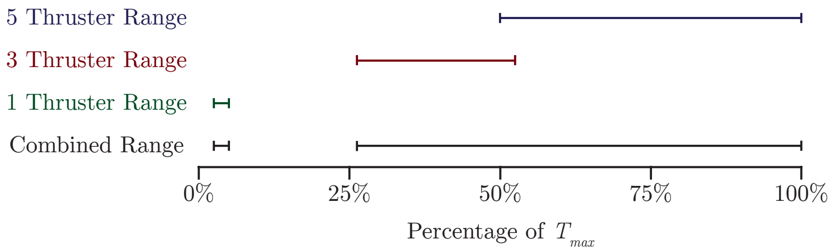


Figure 5.5: Ranges of the engine block in each of the three different configurations; with five thrusters turned on, three thrusters turned on, and one thruster turned on. Furthermore, the fully combined range of the three different configurations are shown at the bottom.

Vernier Thruster Setup

The vernier thrusters of AAURocket will be placed perpendicular to the LV body x-axis. Each vernier thruster will point in a ±45° and 180° ± 45° direction with respect to the y-axis of the LV body frame. The placement of each vernier thruster will be slightly displaced in the y/z-plane with angles of ±35° and 180° ± 35°. An illustration of this setup can be seen on figure 5.6.

The small asymmetrical placement of each vernier thruster extends the attitude control domain from the $^{\mathrm{r}}\mathbf{1}_{\mathrm{Y}}$ and $^{\mathrm{r}}\mathbf{1}_{\mathrm{Z}}$ to also include the $^{\mathrm{r}}\mathbf{1}_{\mathrm{X}}$ (i.e. roll manoeuvres).

With this setup the control of rotation of each primary body axis is done by turning on two corresponding vernier thrusters. An example would be to turn on the two vernier thruster represented by the vectors $T_{v,1}$ and $T_{v,4}$, on figure 5.6. This action will result in a positive rotation about the $^{\mathrm{r}}\mathbf{1}_{\mathrm{Z}}$ vector (pitch), seeing as the vernier thrusters are displaced in the -$^{\mathrm{r}}\mathbf{1}_{\mathrm{X}}$ direction (location of the engine block).

Another example would be turning on the vernier thrusters represented by $T_{v,1}$ and $T_{v,3}$. This action will result in a negative rotation about the ${}^{\mathrm{r}}\mathbf{1}_{\mathrm{X}}$ vector (roll).

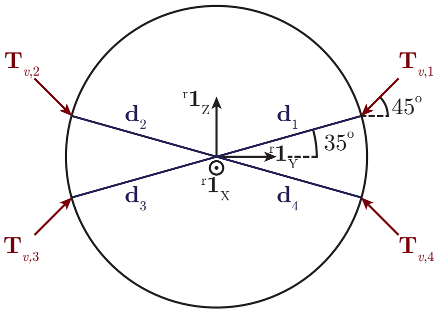


Figure 5.6: A diagram showing the AAURocket body frame vectors, with ${}^{\mathrm{r}}\mathbf{1}_X$ pointing outward (top view). All of the vectors displayed are displaced in the -${}^{\mathrm{r}}\mathbf{1}_X$ direction, as the vernier thrusters are located on the engine block. Each vernier placement vector $\mathbf{d}$ is angled at $\pm 35^{\circ}$ and $180^{\circ} \pm 35^{\circ}$ with respect to the y-axis. Each thrust vector $\mathbf{T}$ of each vernier engine is angled at $\pm 45^{\circ}$ and $180^{\circ} \pm 45^{\circ}$ with respect to the y-axis.

Shell Consumption Actuators

Additionally, within the engine block there are placed several DC-motors mounted with cogs. These mesh into teeth on the fuel shell to function as a rack and pinion. This enables the hardware on board AAURocket to actively control the consumption rate of the fuel shell.

Finally, the diameter of the main engine block is set to 1 m with a total weight of 100 kg.

5.5 The Chemical Rocket Engine

Source: [Turner 01]

As mentioned in section 4.1 the basic objective of a rocket engine is to convert potential energy of a propellant into kinetic energy. For a chemical rocket engine this process can basically be divided into three consecutive steps: mixture, combustion, expansion.

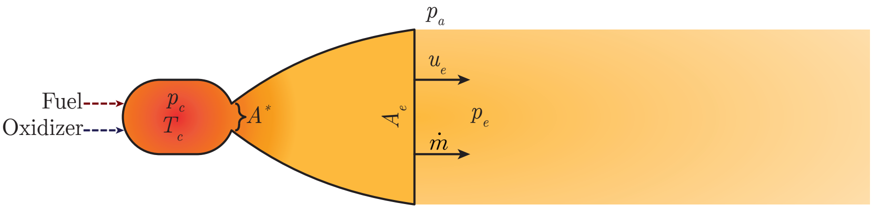


Figure 5.7: A diagram of the chemical rocket engine, with fuel and oxidizer injected into the combustion chamber at pressure p_c and temperature T_c, followed by the narrow throat area, A^*, and finally the nozzle to the right. The pressure of the ambient environment is denoted p_a, the pressure of the exhaust stream is denoted p_e, and the area of the nozzle is denoted A_e.

The process of these three steps starts with the mixture of the fuel and oxidizer (in bipropellant engines) in the combustion chamber of the chemical rocket engine. Both fuel and oxidizer is then ignited in the combustion chamber at high temperature and pressure. The resulting propellant of the chemical reaction will then convert its heat and pressure energy into kinetic energy as it passes through the throat of the engine and down the nozzle. The propellant will expand down through the nozzle, and the expansion acting on the walls of the nozzle will result in the rocket being propelled forward. In figure 5.7 a diagram of an arbitrary chemical rocket engine can be seen.

The thrust of the chemical rocket engine can be described by the equation:

$$T = \dot{m}u_e + (p_e - p_a)A_e \tag{5.1}$$

where

u_e	is the true exhaust velocity	[m/s]
p_e	is the pressure of the exhaust pressure at the exit of the nozzle	[Pa]
p_a	is the pressure of the ambient environment	[Pa]
A_e	is the nozzle exit area	[m^2]

While equation 5.1 is similar to the thrust equation (4.1) there are two differences; namely the addition of the true exhaust velocity and a pressure force, called the pressure thrust.
The true- and effective exhaust velocities are closely related, however, their distinction originates from the pressure thrust. The pressure thrust originates from the ambient pressures effect on the ability of the engine to harvest the energy of the propellants expansion. In general three primary scenarios arise from this last term: when $p_a > p_e$, when $p_a < p_e$, and when $p_a = p_e$.

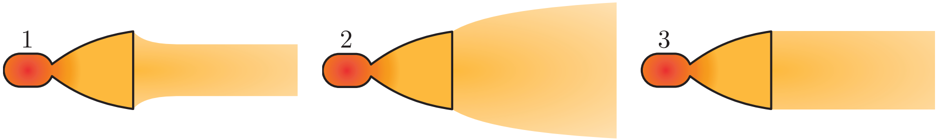


Figure 5.8: Three different scenarios of the chemical rocket engines exhaust pressures. When the exhaust gas is over expanded ($p_a > p_e$)(1), when the gas is under expanded ($p_a < p_e$)(2), and when the gas is optimally expanded ($p_a = p_e$)(3).

These effects are illustrated on figure 5.8. The first scenario ($p_a > p_e$) has the ambient pressure apply more pressure to the front of the rocket than to the back (the nozzle exit) which in turn has a negative contribution to the thrust. The second scenario ($p_a < p_e$) will in principle have a positive contribution to the amount of thrust, however, the gas being under expanded means that the rocket nozzle has not utilized all of the energy for forward thrust (not all of the expansion energy has been utilized). The third and final scenario ($p_a = p_e$) is the optimal design seeing as the two pressures cancels each other out, thereby leading to $u_e = v_e$, and the chemical engine thrust equation in 5.1 is reduced to the same as the original thrust equation 4.1.

5.5.1 True- and Effective Exhaust Velocity

Source: [Turner 01] By substitution of equation (5.1) into (4.1), it is possible to show the relation between v_e and u_e as seen in (5.2).

$$v_e = u_e + \left(\frac{(p_e - p_a)A_e}{\dot{m}}\right) \tag{5.2}$$

From this relation it is seen that the effective exhaust velocity includes both the true exhaust velocity and the pressure thrust caused by the exterior environment in which the chemical engine operates. As such the effective exhaust velocity is a performance measure of the chemical engine in some ambient environment.
The true exhaust velocity on the other hand is in fact a performance measure of the engine itself. That is, the theoretical speed at which the engine is able to expel its propellant. As such it is the chemical conditions and the mechanical design of the chemical rocket engine that defines its magnitude. The true exhaust velocity is defined as seen in equation 5.3.

$$u_e = \sqrt{\frac{2\gamma}{\gamma - 1}\frac{\mathfrak{R}T_c}{\mathfrak{M}}\left[1 - \left(\frac{p_e}{p_c}\right)^{\frac{\gamma-1}{\gamma}}\right]} \tag{5.3}$$

where

$\mathfrak{R}$	is the universal gas constant	[J/(molK)]
$\mathfrak{M}$	is the molecular weight of the exhaust gas	[g/mol]
T_c	is the combustion chamber temperature	[K]
p_c	is the combustion chamber pressure	[Pa]
γ	is the isentropic expansion factor	[·]

It is clear that the true exhaust velocity is determined by both the chemical properties of the propellant ($\mathfrak{M}, T_c$, and γ) as well as the pressures (p_e and p_c) under which the engine is operated. While it is possible to manipulate the pressures by design, it is seen from the equation that they merely act as a retarding force. So even if the best possible pressure parameters were obtained ($p_e = 0$ for instance), the exhaust velocity will end up being entirely governed by the chemical process of the fuel and oxidizer, and the temperature at which they are combusted.

5.5.2 Nozzle Throat Area

Source: [Turner 01]

The throat of the chemical rocket is an important parameter in chemical rocket design. The throat area is directly proportional to both mass flow and chamber pressure, seeing as the area has to contain the propellant inside the combustion chamber long enough for the chemical reaction to take place, before ejection. The throat area of the chemical rocket engine is governed by the equation:

$$A^* = \frac{\dot{m}}{p_c\sqrt{\gamma\left(\frac{2}{\gamma+1}\right)^{\frac{\gamma+1}{\gamma-1}}\frac{\mathfrak{M}}{\mathfrak{R}T_c}}} \tag{5.4}$$

where

A^*	is the throat area of the chemical rocket engine	[m^2]

The throat areas dependency on both p_c and $\dot{m}$ is the primary issue behind chemical engine throttling. Since the thrust of a rocket is dependant on the mass flow, the only way to throttle is to reduce the mass flow. From equation 5.4 it is clear that a decrease in $\dot{m}$ will either mean a decrease in chamber pressure or a decrease in throat area. Since the chemical rocket engine is usually built of metal, the throat area will always be a static parameter. Hence, the only option is for the chamber pressure to decrease. By purely considering equation 5.4 this effect does not impact the performance, however, equation 5.3 states that a drop in chamber pressure will reduce the true exhaust velocity. If the pressures are changed to extreme relative levels (p_c » p_e) this can result in the propellant being only partially combusted inside the engine before ejection, which leads to unstable performance. Furthermore, if the pressure becomes to low the thermal-energy-to-droplet-evaporation ratio becomes too low, in turn resulting in insufficient combustion.

5.5.3 Nozzle Exit Area

Source: [Turner 01]

The nozzle exit area is the interface between the nozzle and the environment in which the engine is functioning. The primary function of the area is to ensure that the propellant has reached sufficient expansion inside the nozzle in order to perform translational work on the rocket. As such the nozzle exit area is determined primarily by the optimum expansion factor of the propellant in combination with the size of the throat area. The equation governing the nozzle exit area is as follows:

$$A_e = A^* \sqrt{\frac{\left(\frac{\gamma-1}{2}\right)\left(\frac{2}{\gamma+1}\right)^{\frac{\gamma+1}{\gamma-1}}}{\frac{p_e}{p_c}^{\frac{2}{\gamma}}\left(1-\frac{p_e}{p_c}^{\frac{\gamma-1}{\gamma}}\right)}} \tag{5.5}$$

As seen the primary parameters dictating the nozzle exit area is A^* and $\frac{p_e}{p_c}$. With A^* and p_c pertaining to the combustion process, the only parameter left to vary is p_e. The choice of p_e is thus completely up to the designer and how the desired performance of the engine should be in at given ambient conditions.

From the equation it is also obvious that the chemical rocket engine can not function if $p_e > p_c$ as this will introduce imaginary solutions to equation 5.5. Furthermore, p_c can never be 0 either, as this will make the equation unsolvable. Intuitively this makes sense as rocket engines with p_e $> p_c$ or $p_c = 0$, will result in negative expansion in the nozzle, and/or no expansion to harvest.

5.5.4 Engine Efficiency

While the effective exhaust velocity is an efficiency parameter, it has up until this point been calculated from theoretical optimum values. Several practical factors will in reality reduce the effective exhaust velocity further than what the equations show. As such, an additional efficiency parameter is defined to be applied on the effective exhaust velocity:

$$v_e = \eta_t \, v_{e,theoretical} \tag{5.6}$$

with

$$\eta_t = \eta_r \eta_n \tag{5.7}$$

where

η_t	is the total efficiency of the chemical engine	[$\cdot$]
η_r	is the reaction efficiency of the chemical engine	[$\cdot$]
η_n	is the nozzle efficiency of the chemical engine	[$\cdot$]

Reaction efficiency is the engines effective ability to utilize the chemical potential energy of the propellant. This parameter is not necessarily constant and can, among others, vary with respect to mass flow, chamber pressure, and/or throttle level.
Nozzle efficiency is the ability of the nozzle to effectively harvest the expansion energy of the propellant and converting it to kinetic energy. This is primarily a mechanical efficiency parameter which is defined by the shape of the nozzle, the roughness of the inner surface of the nozzle, as well as precision of the construction process of the nozzle.

As both η_r and η_n can be quite difficult to accurately calculate pre-emptively, they will be assumed as two constant efficiency parameters in this project. All efficiency parameters will be conservatively assumed to be:

$$\eta_t = \eta_r \eta_n = 0.9385 \tag{5.8}$$

$$\eta_r = \eta_n = 0.9688 \tag{5.9}$$

5.6 Thruster Dimensioning

In this section the three different types of thrusters used on AAURocket will be dimensioned. These include: the four identical booster thrusters, the orbital thruster, and the four identical attitude control vernier thrusters.

5.6.1 Booster Thruster Dimensioning

All engine parameters connected to the booster thrusters will be subscripted with a b.
The booster thrusters are designed as bipropellant engines utilizing both the fuel and oxidizer carried by AAURocket. Furthermore, the boosters are designed to give optimal thrust at sea level, seeing as they will function from this height and out to space. This will result in optimum expansion at sea-level with some under-expansion occurring in vacuum. This design goal implies that $p_{e,b} = 101.325$ [kPa] (1 atmosphere).
In order to dimension the thrusters, the properties of the propellant has to be determined.

Bipropellant

Sources: [Yemets 08a] [Ponomarenko 15]

All bipropellant parameters connected to the HDPE and HPO fuel/oxidizer mixture will be subscripted with $f|o$.
In order to obtain chemical parameters of the bipropellant, a program called "Rocket Propulsion Analysis (RPA)" is used. This program takes fuel, oxidizer, ingredient temperature, chamber-, and exhaust pressure as input and in turn outputs a large variety of theoretical parameters, including: combustion temperature, molecular weight, and isentropic expansion factor.
As inputs, HDPE and HPO(98%) was used as ingredients with a mixing ratio of 1:6.66 at an ingredient temperature of 750 K [Yemets 08a], at a chamber pressure of 6.08 MPa (60 atmospheres). These values have given the following chemical parameters:

The chamber pressure of 60 atmospheres has been chosen in order to ensure the performance of the chemical engine stays within a feasible performance range. Using a chamber pressure this high will, however, require the engine to utilize a turbo pump of sufficient size.

Bipropellant chemical parameters			
Molecular weight	$\mathfrak{M}_{\mathrm{f}\vert\mathrm{o}}$	21.9	[g/mol]
Isentropic expansion factor	$\gamma_{\mathrm{f}\vert\mathrm{o}}$	1.1319	[·]
Combustion chamber temperature	$T_{c,\,\mathrm{f}\vert\mathrm{o}}$	2943	[K]

Table 5.5: Chemical parameters for the bipropellant comprising the ingredients of polyethylene and hydrogen peroxide (98%).

From the values in table 5.5 it is possible to calculate the true exhaust velocity by using equation 5.3:

$$u_{e,b} = \sqrt{\frac{2\gamma_{\mathrm{f}|\mathrm{o}}}{\gamma_{\mathrm{f}|\mathrm{o}} - 1} \frac{\mathfrak{R}T_{c,\,\mathrm{f}|\mathrm{o}}}{\mathfrak{M}_{\mathrm{f}|\mathrm{o}}} \left[1 - \left(\frac{p_{e,b}}{p_{c,b}}\right)^{\frac{\gamma_{\mathrm{f}|\mathrm{o}}-1}{\gamma_{\mathrm{f}|\mathrm{o}}}}\right]} = 2697\,[\mathrm{m/s}] \tag{5.10}$$

Booster Maximum Thrust and Resulting Mass Flow

In section 8.2.2 the total maximum needed thrust of AAURocket was calculated to be $T_{max} =$ 101.231 kN. Using the characteristic velocity of the optimization problem at sea level it is possible to calculate the needed nominal maximum mass flow: [1]

$$\dot{m}_{max} = \frac{T_{max}}{v_e(sea)} = 39.99\,[\mathrm{kg/s}] \tag{5.11}$$

where

$v_e(sea)$ is the effective exhaust velocity of the engine at sea-level [m/s]

In turn the maximum needed thrust and mass flow for a single booster thruster can be calculated by using the level from section 5.4.2:

$$T_{max,b} = 0.2375\,T_{max} = 24.04\,[\mathrm{kN}] \tag{5.12}$$

$$\dot{m}_{max,b} = 0.2375\,\dot{m}_{max} = 9.498\,[\mathrm{kg/s}] \tag{5.13}$$

With a maximum total mass flow of 40 kg/s it is strongly advised to do a physical verification test of whether this exceeds the gasification rate limit of the polyethylene.

Booster Throat and Nozzle Area

The design goal of optimum expansion at sea level, combined with the properties of the bipropellant, and the conditions under which it will be combusted, makes the dimensioning of the throat area possible. The throat area of each booster is calculated to:

$$A_b^* = \frac{\dot{m}_{max,b}}{p_{c,b}\sqrt{\gamma_{\,\mathrm{f}|\mathrm{o}} \left(\frac{2}{\gamma_{\,\mathrm{f}|\mathrm{o}}+1}\right)^{\frac{\gamma_{\,\mathrm{f}|\mathrm{o}}+1}{\gamma_{\,\mathrm{f}|\mathrm{o}}-1}} \frac{\mathfrak{M}_{\,\mathrm{f}|\mathrm{o}}}{\mathfrak{R}T_{c,\,\mathrm{f}|\mathrm{o}}}}} = 26\,[\mathrm{cm}^2] \tag{5.14}$$

[1] Note that the values of effective exhaust velocity at sea level and vacuum, along with the maximum thrust needed was calculated simultaneously in an iterative process between the thruster dimensioning and the trajectory optimization.

Furhtermore, this throat area results in a nozzle exit area of:

$$A_{e,b} = A_b^* \sqrt{\frac{\left(\frac{\gamma_{\mathrm{f|o}}-1}{2}\right)\left(\frac{2}{\gamma_{\mathrm{f|o}}+1}\right)^{\frac{\gamma_{\mathrm{f|o}}+1}{\gamma_{\mathrm{f|o}}-1}}}{\frac{p_{e,b}}{p_{c,b}}^{\frac{2}{\gamma_{\mathrm{f|o}}}}\left(1-\frac{p_{e,b}}{p_{c,b}}^{\frac{\gamma_{\mathrm{f|o}}-1}{\gamma_{\mathrm{f|o}}}}\right)}} = 241\,[\mathrm{cm}^2] \tag{5.15}$$

Booster Effective Exhaust Velocities

The nominal effective exhaust velocity can now be calculated at both optimum expansion as well as in vacuum while applying the efficiency parameters of section 5.5.4:

$$v_{e,b}(sea) = \eta_t \left[u_{e,b} + \left(\frac{(p_{e,b} - p_{a,sea})A_{e,b}}{\dot{m}_{max,b}}\right)\right] = 2531\,[\mathrm{m/s}] \tag{5.16}$$

$$v_{e,b}(vac) = \eta_t \left[u_{e,b} + \left(\frac{p_{e,b}A_{e,b}}{\dot{m}_{max,b}}\right)\right] = 2773\,[\mathrm{m/s}] \tag{5.17}$$

Note that the nominal effective velocity in vacuum is how the thruster will perform at maximum mass flow and matching chamber pressure in vacuum. Hence $v_{e,b}(vac)$ will be different in practice, however, the nominal effective exhaust velocity will be used as an approximate value for later calculations.

5.6.2 Orbital Thruster Dimensioning

All engine parameters connected to the orbital thruster will be subscripted with an o.
The orbital thruster is designed to utilize the same bipropellant as the booster thrusters. Furthermore, it is also designed to have optimum thrust performance at sea level, which implies that $p_{e,o} = 101.325$ [Pa] (1 atmosphere). These design parameters cause the true exhaust velocity of the orbital thruster to equal that of the booster thrusters:

$$u_{e,o} = u_{e,b} = 2697\,[\mathrm{m/s}] \tag{5.18}$$

Orbital Thruster Maximum Thrust and Resulting Mass Flow

By using the orbital thruster range from section 5.4.2, the maximum thrust and mass flow for the orbital engine is calculated to be:

$$T_{max,o} = 0.05\,T_{max} = 5.062\,[\mathrm{kN}] \tag{5.19}$$

$$\dot{m}_{max,o} = 0.05\,\dot{m}_{max} = 1.9995\,[\mathrm{kg/s}] \tag{5.20}$$

Orbital Thruster Throat and Nozzle Area

The throat area of the orbital thruster is calculated as:

$$A_o^* = \frac{\dot{m}_{max,o}}{p_{c,o}\sqrt{\gamma_{\mathrm{f|o}}\left(\frac{2}{\gamma_{\mathrm{f|o}}+1}\right)^{\frac{\gamma_{\mathrm{f|o}}+1}{\gamma_{\mathrm{f|o}}-1}}\frac{\mathfrak{M}_{\mathrm{f|o}}}{\Re T_{c,\,\mathrm{f|o}}}}} = 5.475\,[\mathrm{cm}^2] \tag{5.21}$$

Furthermore, this throat area results in a nozzle exit area of:

$$A_{e,o} = A_o^* \sqrt{\frac{\left(\frac{\gamma_{\mathrm{f|o}}-1}{2}\right)\left(\frac{2}{\gamma_{\mathrm{f|o}}+1}\right)^{\frac{\gamma_{\mathrm{f|o}}+1}{\gamma_{\mathrm{f|o}}-1}}}{\frac{p_{e,o}}{p_{c,o}}^{\frac{2}{\gamma_{\mathrm{f|o}}}}\left(1-\frac{p_{e,o}}{p_{c,o}}^{\frac{\gamma_{\mathrm{f|o}}-1}{\gamma_{\mathrm{f|o}}}}\right)}} = 51\,[\mathrm{cm}^2] \tag{5.22}$$

Orbital Thruster Effective Exhaust Velocities

The nominal effective exhaust velocity can now be calculated at both optimum expansion as well as in vacuum while applying the efficiency parameters of section 5.5.4:

$$v_{e,o}(sea) = \eta_t \left[u_{e,o} + \left(\frac{(p_{e,o} - p_{a,sea})A_{e,o}}{\dot{m}_{max,o}}\right)\right] = 2531\,[\mathrm{m/s}] \tag{5.23}$$

$$v_{e,o}(vac) = \eta_t \left[u_{e,o} + \left(\frac{p_{e,o}A_{e,o}}{\dot{m}_{max,o}}\right)\right] = 2773\,[\mathrm{m/s}] \tag{5.24}$$

Note that the effective exhaust velocities of the orbital thruster equals that of the booster thrusters. This is an inherent effect of utilizing the same bipropellant and pressures as the boosters, as well as a proportional down scaling of A_e and $\dot{m}_{max}$.

5.6.3 Vernier Thruster Dimensioning

[Turner 01]

All engine parameters connected to the vernier thrusters will be subscripted with a v.
The task of the vernier thrusters differs from that of the main thrusters, in the sense that they are used for attitude corrections throughout the ascent. As such, there is a need for the vernier thrusters to be able to turn on and off. To overcome this problem the vernier thrusters is designed as monopropellant rocket engines. Monopropellant rocket engines use propellants that are either able to react with itself or react through a catalyst. The oxidizer HPO is able to be used as a monopropellant through the use of a proper catalyst. This reaction will cause the HPO to become hot and produce steam and oxygen. With HPO going from a liquid state to steam and oxygen means there is chemical expansion that can be harvested for kinetic energy.
Seeing as the HPO will need to pass through a catalyst, the chamber pressure of the vernier thruster is reduced to 1.013 MPa (10 atmospheres). This is done to ensure that the catalyst is not being destroyed by the HPO as it passes through the catalyst.

Monopropellant

Sources: [Ponomarenko 15]

All monopropellant parameters connected to the HPO will be subscripted with an h.
The theoretical parameters for the vernier thruster is passed to the program RPA to obtain the accompanying chemical parameters for the monopropellant. As inputs HPO(98%) was used as monopropellant, at a chamber pressure of 1.013 MPa. These values has resulted in the following chemical parameters:

Bipropellant chemical parameters			
Molecular weight	$\mathfrak{M}_{\mathrm{h}}$	22.56	[g/mol]
Isentropic expansion factor	γ_{h}	1.2511	[·]
Combustion chamber temperature	$T_{c,\,\mathrm{h}}$	1223	[K]

Table 5.6: Chemical parameters for the monopropellant of hydrogen peroxide (98%).

From the values in table 5.5 it is possible to calculate the true exhaust velocity for the vernier thruster by using equation 5.3:

$$u_{e,v} = \sqrt{\frac{2\gamma_{\mathrm{h}}}{\gamma_{\mathrm{h}}-1}\frac{\Re T_{c,\,\mathrm{h}}}{\mathfrak{M}_{\mathrm{h}}}\left[1-\left(\frac{p_{e,v}}{p_{c,v}}\right)^{\frac{\gamma_{\mathrm{h}}-1}{\gamma_{\mathrm{h}}}}\right]} = 1290\,[\mathrm{m/s}] \tag{5.25}$$

Vernier Thruster Maximum Thrust and Resulting Mass Flow

In order to determine a plausible thrust level of the vernier thrusters, the input sequence from the ascent trajectory calculated in section 8.3.2. Specifically the change rate of the input parameters θ and ϕ which are the two spherical components used to represent the orientation of the LV during the ascent. By taking the double derivative of each angle a maximum needed acceleration at each time instant of the ascent can be found. The combined magnitude of these accelerations are calculated as:

$$\alpha_\theta(t) = \frac{\mathrm{d}^2\theta(t)}{\mathrm{d}t^2} \tag{5.26}$$

$$\alpha_\phi(t) = \frac{\mathrm{d}^2\phi(t)}{\mathrm{d}t^2} \tag{5.27}$$

$$\alpha_{max}(t) = \sqrt{\alpha_\theta(t)^2 + \alpha_\phi(t)^2} \tag{5.28}$$

The magnitude of the combined acceleration from 5.28 can then be multiplied with the moment of inertia of the LVs body x-axis to get the maximum needed torque at each time instant:

$$\tau_{max} = \alpha_{max}(t)^{\mathrm{r}}I_{y|z}(t) \tag{5.29}$$

Due to the placement of the vernier thrusters on the engine block, it is apparent that their torque capability will be reduced over time, as the engine block approaches the CoM of the LV. Equation 5.30 shows the relation between the four vernier thrusters and the resulting torque vector they produce.

$$\boldsymbol{\tau}_v(t) = {}^{\mathrm{r}}\mathbf{d}_1(t) \times {}^{\mathrm{r}}\mathbf{T}_{v,1} + {}^{\mathrm{r}}\mathbf{d}_2(t) \times {}^{\mathrm{r}}\mathbf{T}_{v,2} + {}^{\mathrm{r}}\mathbf{d}_3(t) \times {}^{\mathrm{r}}\mathbf{T}_{v,3} + {}^{\mathrm{r}}\mathbf{d}_4(t) \times {}^{\mathrm{r}}\mathbf{T}_{v,4} \tag{5.30}$$

where

${}^{\mathrm{r}}\boldsymbol{\tau}_v(t)$	is the torque produced by the vernier thrusters	[Nm]
${}^{\mathrm{r}}\mathbf{d}_k(t)$	is the placement of the k^{th} vernier thruster	[m]
${}^{\mathrm{r}}\mathbf{T}_k(t)$	is the thrust of the k^{th} vernier thruster	[N]

From the vernier thruster alignment shown in section 5.4.2 it is clear that only two vernier thrusters are needed to rotate the LV about either ${}^{\mathrm{r}}\mathbf{1}_Y$ or ${}^{\mathrm{r}}\mathbf{1}_Z$. Furthermore, since all vernier thrusters are

placed symmetrically around the LV, it is only necessary to study a single torque case, in order to determine the maximum vernier thrust force needed.
Equation 5.31 shows the decomposed vector components of equation 5.30. Note that the time indices of the placement and torque vectors has been omitted.

$$\begin{bmatrix} \tau_{v,x} \\ \tau_{v,y} \\ \tau_{v,z} \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} -d_{1,y}T_{v,1} - d_{2,y}T_{v,2} - d_{3,y}T_{v,3} + d_{4,y}T_4 + d_{1,z}T_{v,1} - d_{2,z}T_{v,2} + d_{3,z}T_{v,3} - d_{4,z}T_{v,4} \\ -d_{1,x}T_{v,1} - d_{2,x}T_{v,2} + d_{3,x}T_{v,3} + d_{1,x}T_{v,4} \\ d_{1,x}T_{v,1} - d_{2,x}T_{v,2} - d_{3,x}T_{v,3} + d_{4,x}T_{v,4} \end{bmatrix} \tag{5.31}$$

with

$$\begin{aligned} d_{1,x} &= d_{2,x} = d_{3,x} = d_{4,x} \\ d_{1,y} &= d_{2,y} = d_{3,y} = d_{4,y} \\ d_{1,z} &= d_{2,z} = d_{3,z} = d_{4,z} \\ T_{v,1} &= T_{v,2} = T_{v,3} = T_{v,4} \end{aligned}$$

By using a simple case of only turning on vernier thruster three and four (which equals $T_{v,3}$ and $T_{v,4}$), it is possible to find an expression for the maximum thrust force needed, as seen in equation 5.32.

$$\begin{bmatrix} \tau_{v,x} \\ \tau_{v,y} \\ \tau_{v,z} \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 0 \\ d_{3,x}T_{v,3} + d_{4,x}T_{v,4} \\ 0 \end{bmatrix} \tag{5.32}$$

Since $d_{3,x} = d_{4,x}$ and $T_{v,3} = T_{v,4} = T_{max,v}$ the expression can be reduced to:

$$\begin{aligned} \tau_{v,y} &= \sqrt{2} d_x T_{max,v} \Leftrightarrow \\ T_{max,v} &= \frac{\tau_{v,y}}{\sqrt{2} d_x} \end{aligned} \tag{5.33}$$

Finally $\tau_{v,y}$ in equation 5.33 can be substituted with τ_{max} from equation 5.29 which in turn gives:

$$T_{max,v}(t) = \frac{\tau_{max}(t)}{\sqrt{2} d_x(t)} \tag{5.34}$$

The result of equation 5.34 applied on the solution of the optimization problem calculated in section 8.3.2 can be seen on figure 5.9.

It is clear that a lower bound of 110 N for each vernier thruster will fulfill the requirement set by the optimal ascent trajectory. However, the maximum thrust for each vernier thruster will be designed significantly higher in order to ensure that they can overcome any unforeseen disturbance factors, or other sudden needs of attitude change. Therefore, the maximum thrust and resulting mass flow for each vernier thruster is designed to be:

$$T_{max,v} = 500\,[\text{N}] \tag{5.35}$$

$$\dot{m}_{max,v} = \dot{m}_{max} = 0.3878\,[\text{kg/s}] \tag{5.36}$$

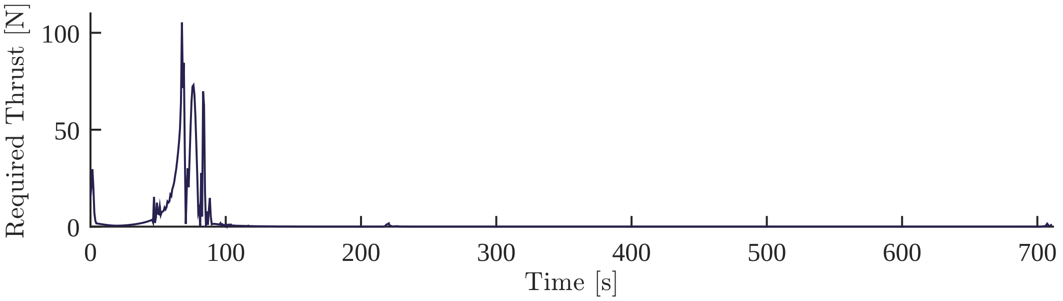


Figure 5.9: The maximum required vernier thrust force needed throughout the trajectory calculated in section 8.3.2.

Vernier Thruster Throat and Nozzle Area

The throat area of the vernier thruster is calculated as:

$$A_v^* = \frac{\dot{m}_{max,v}}{p_{c,v}\sqrt{\gamma_{\mathrm{h}}\left(\frac{2}{\gamma_{\mathrm{h}}+1}\right)^{\frac{\gamma_{\mathrm{h}}+1}{\gamma_{\mathrm{h}}-1}}\frac{\mathfrak{M}_{\mathrm{h}}}{\mathfrak{R}T_{c,\,\mathrm{h}}}}} = 3.904\,[\mathrm{cm}^2] \tag{5.37}$$

Furthermore, this throat area results in a nozzle exit area of:

$$A_{e,v} = A_v^*\sqrt{\frac{\left(\frac{\gamma_{\mathrm{h}}-1}{2}\right)\left(\frac{2}{\gamma_{\mathrm{h}}+1}\right)^{\frac{\gamma_{\mathrm{h}}+1}{\gamma_{\mathrm{h}}-1}}}{\frac{p_{e,v}}{p_{c,v}}^{\frac{2}{\gamma_{\mathrm{h}}}}\left(1-\frac{p_{e,v}}{p_{c,v}}^{\frac{\gamma_{\mathrm{h}}-1}{\gamma_{\mathrm{h}}}}\right)}} = 8.429\,[\mathrm{cm}^2] \tag{5.38}$$

Vernier Thruster Effective Exhaust Velocities

The effective exhaust velocity can now be calculated at both optimum expansion as well as in vacuum while applying the efficiency parameters of section 5.5.4:

$$v_{e,v}(sea) = \eta_t\left[u_{e,v} + \left(\frac{(p_{e,v}-p_{a,sea})A_{e,v}}{\dot{m}_{max,v}}\right)\right] = 1210.1\,[\mathrm{m/s}] \tag{5.39}$$

$$v_{e,v}(vac) = \eta_t\left[u_{e,v} + \left(\frac{p_{e,v}A_{e,v}}{\dot{m}_{max,v}}\right)\right] = 1416.8\,[\mathrm{m/s}] \tag{5.40}$$

Note that the vernier thrusters are expected to always apply full thrust, and only vary in time period of each actuation sequence. As such the vernier thrusters will indeed have an effectivity of $v_{e,v}(vac)$ in vacuum.

5.7 Subconclusion

In this chapter several different kinds of inertial and reference sensors were presented. A study into existing sensor setups on some of the CS rockets were performed and based on their experience the IMU ADIS16448 was chosen to be the primary sensor platform of AAURocket.
Additionally, this chapter introduced different common types of attitude actuators for rocket LVs,

among which, the vernier thrusters were chosen to be used as the attitude actuators for AAU-Rocket.

Furthermore, the general thrust profile characteristics for ascent trajectories was presented, followed by the constraints imposed on the ascent trajectory of AAURocket. These constraints included the need for the ability to throttle the thrusters as well as the necessity for the main thrust to never be turned off during the ascent.

Parameter	**Booster**	**Orbital**	**Vernier**	**Unit**
T	24.04	5.062	0.5	[kN]
$\dot{m}$	9.498	1.9995	0.3878	[kg/s]
p_c	6.08	6.08	1.013	[MPa]
p_e	0.1013	0.1013	0.1013	[MPa]
u_e	2697	2697	1290	[m/s]
A^*	26	5.475	3.904	[cm^2]
A_e	241	51	8.429	[cm^2]
A_e/A^*	9.265	9.265	2.159	[·]
$v_e(sea)$	2531	2531	1210	[m/s]
$v_e(vac)$	2773	2773	1417	[m/s]
Throttle Range	[50%;100%]	[50%;100%]	100%	[·]

Table 5.7: Design parameters for for each of the three types of chemical rocket engines on board AAURocket, including: booster thrusters, orbital thruster, and vernier thruster.

The general mechanical structure for the engine block was also presented, where a five thruster setup was chosen to overcome the limitations of a wide throttle range coupled with the attainable low single thruster throttle range of [50%;100%].

Finally the thermodynamic design equations for chemical rocket engines was presented follow by the dimensioning of each of the three types of thrusters to be utilized on AAURocket. These included the four booster thrusters, the orbital thruster, and the four vernier thrusters. The design parameters for each of these thruster types can be found in table 5.7.

Chapter 6

Rocket and Disturbance Modeling

The previous chapters have described the mechanical design of the rocket together with the dimensioning and design of the main thrusters and vernier thrusters used for guiding AAURocket into space. This chapter will present the modelling of the translational and rotational dynamics of the rocket including modelling of control inputs and disturbances.

6.1 Modeling of Translational Dynamics

The translational dynamics of the rocket are affected by thrust, gravity, and aerodynamic friction. In the following section each of these forces will be described and modelled.
In general, a complete force equation describing the full system can be written as shown in equation 6.1

$$\mathbf{f}_a = \dot{m}v_e + \mathbf{f}_{ext} \tag{6.1}$$

where $\mathbf{f}_{ext}$ is the sum of all the external disturbance forces described in this section.

6.1.1 Thrust

The only controllable input force to the system is the thrust applied by the main thrusters and the vernier thrusters of the rocket. The main thrusters produce the main thrust in the x-axis of the RBRF. The four vernier thrusters produce minor positive and negative thrusts in the y- and z-axis of the RBRF. In general the thrust force produced by the main thrusters, $T_{b,o}$, can be described as a thrust vector as shown in equation 6.2

$${}^{\mathrm{r}}\mathbf{T} = \begin{bmatrix} {}^{\mathrm{r}}T_x \\ {}^{\mathrm{r}}T_y \\ {}^{\mathrm{r}}T_z \end{bmatrix} = \begin{bmatrix} T_{b,o} \\ 0 \\ 0 \end{bmatrix} \tag{6.2}$$

As described in section 5.4, the vernier thrusters are placed in an angle such that the thrust produced by the vernier thrusters are as given in equation 6.3

$${}^{\mathrm{r}}\mathbf{T}_v = \begin{bmatrix} {}^{\mathrm{r}}T_{v,x} \\ {}^{\mathrm{r}}T_{v,y} \\ {}^{\mathrm{r}}T_{v,z} \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 0 & 0 & 0 & 0 \\ -1 & 1 & 1 & -1 \\ -1 & -1 & 1 & 1 \end{bmatrix} \begin{bmatrix} T_{v,1} \\ T_{v,2} \\ T_{v,3} \\ T_{v,4} \end{bmatrix} \tag{6.3}$$

where:

${}^{\mathrm{r}}\mathbf{T}_v$	is the thrust force produced by the vernier thrusters	[N]
$T_{v,k}$	is the thrust force produced by vernier thruster k	[N]

6.1.2 Disturbances

Gravity

In general the gravitational acceleration of an object towards the center of Earth can be described as a function of the position of the object, with respect to the center of earth, as shown in equation 6.4

$$ {}^{\mathrm{i}}\mathbf{g}({}^{\mathrm{i}}\mathbf{r}) = \frac{\mu_{\mathrm{E}}}{||{}^{\mathrm{i}}\mathbf{r}||^3}{}^{\mathrm{i}}\mathbf{r} \tag{6.4} $$

where:

$\mathbf{g}({}^{\mathrm{i}}\mathbf{r})$	is the gravitational acceleration at position (${}^{\mathrm{i}}\mathbf{r}$)	[m/s^2]

And then the gravitational force, f_g, imposed on the rocket can be determined by equation 6.5

$$ \mathbf{f}_g = m{}^{\mathrm{i}}\mathbf{g}({}^{\mathrm{i}}\mathbf{r}) \tag{6.5} $$

Friction due to motion

When an object is moving inside the atmosphere it is subjected to friction with the air. The friction drag is a disturbance force mainly dependent on the speed of the object, v, and it can be determined by equation 6.6.

$$ f_D = \frac{1}{2}C_D\rho_a v|v|A \tag{6.6} $$

where:

C_D	is the drag coefficient of the object	[·]
ρ_a	is the density of the ambient air	[kg/m^3]
v	is the speed of the object	[m/s]
A	is the cross sectional area of the object perpendicular to v	[m^2]

To determine the drag in all directions, the drag coefficient for each face of the object is needed. Each drag coefficient can then be used to determine the effect of the friction in each direction. Thereby the drag force vector, ${}^{\mathrm{r}}\mathbf{f}_D$, for the rocket can be determined as shown in equation 6.7

$$ {}^{\mathrm{r}}\mathbf{f}_D = \begin{bmatrix} {}^{\mathrm{r}}f_{D,x} \\ {}^{\mathrm{r}}f_{D,y} \\ {}^{\mathrm{r}}f_{D,z} \end{bmatrix} = \frac{1}{2}\rho_a \begin{bmatrix} C_{D,x}A_x \\ C_{D,y}A_y \\ C_{D,z}A_z \end{bmatrix} \circledast \begin{bmatrix} {}^{\mathrm{r}}v_x \\ {}^{\mathrm{r}}v_y \\ {}^{\mathrm{r}}v_z \end{bmatrix} \circledast \begin{bmatrix} |{}^{\mathrm{r}}v_x| \\ |{}^{\mathrm{r}}v_y| \\ |{}^{\mathrm{r}}v_z| \end{bmatrix} \tag{6.7} $$

The drag coefficient of an object is normally determined through testing in a laboratory. As AAURocket has not yet been constructed, it is not possible to determine the exact drag coefficients. Various sources describes different drag coefficients for each type of shape. In general it has been chosen that AAURocket, seen from the side, is a cylinder, and seen from the top, is a cone. Conservatively the drag coefficients has been chosen as following:

$$C_{D,x} = 0.295 \tag{6.8}$$
$$C_{D,y} = 1.17 \tag{6.9}$$
$$C_{D,z} = 1.17 \tag{6.10}$$

6.2 Modelling of Rotational Dynamics

The rotational dynamics of the rocket are affected by the same forces as for the translational dynamics. The rotational dynamics are described within this chapter.

6.2.1 Rotating reference frames

When an object, in this case the rocket, rotates within an inertial reference frame, so does the frame of the object. To help describe this rotation Euler's second law of motion is used. Euler's second law of motion states that the time derivative of the angular momentum, $\mathbf{L}$ around a fixed point in an inertial reference frame is equal to the sum of external torques, $\boldsymbol{\tau}_{ext}$, about this point. This can be described as shown in equation 6.11

FiXme Note: Der skal tilføjes en kilde!

$${}^{\mathrm{i}}\dot{\mathbf{L}} = {}^{\mathrm{i}}\boldsymbol{\tau}_{ext} \tag{6.11}$$

To describe the angular momentum of the rocket in the RBRF, it is necessary to apply rotation as shown in equation 6.12

$${}^{\mathrm{r}}\mathbf{L} = {}^{\mathrm{r}}_{\mathrm{i}}\underline{\mathbf{R}}\,{}^{\mathrm{i}}\mathbf{L} \tag{6.12}$$

where:

${}^{\mathrm{r}}_{\mathrm{i}}\underline{\mathbf{R}}$ is the rotation matrix describing the rotation from ECI to RBRF [·]

Then by differentiating the angular momentum given in the RBRF, the time derivative of the angular momentum, in the RBRF, is obtained as shown in equation 6.13

$${}^{\mathrm{r}}\dot{\mathbf{L}} = {}^{\mathrm{r}}_{\mathrm{i}}\dot{\underline{\mathbf{R}}}\,{}^{\mathrm{i}}\mathbf{L} + {}^{\mathrm{r}}_{\mathrm{i}}\underline{\mathbf{R}}\,{}^{\mathrm{i}}\dot{\mathbf{L}} \tag{6.13}$$

The derivative of the rotation between the RBRF and the inertial frame can be determined by equation 6.14

$${}^{\mathrm{r}}_{\mathrm{i}}\dot{\underline{\mathbf{R}}} = -[{}^{\mathrm{r}}\boldsymbol{\omega}]_{\times}\,{}^{\mathrm{r}}_{\mathrm{i}}\underline{\mathbf{R}} \tag{6.14}$$

Thereby by substituting equation 6.11 and 6.14 into equation 6.13 the expression for the rate of change of the angular momentum in the RBRF is obtained as shown in equation 6.15

$${}^{\mathrm{r}}\dot{\mathbf{L}} = -[{}^{\mathrm{r}}\boldsymbol{\omega}]_{\times}\,{}^{\mathrm{r}}_{\mathrm{i}}\underline{\mathbf{R}}\,{}^{\mathrm{i}}\mathbf{L} + {}^{\mathrm{r}}_{\mathrm{i}}\underline{\mathbf{R}}\,{}^{\mathrm{i}}\boldsymbol{\tau}_{ext} = -[{}^{\mathrm{r}}\boldsymbol{\omega}]_{\times}\,{}^{\mathrm{r}}\mathbf{L} + {}^{\mathrm{r}}\boldsymbol{\tau}_{ext} \tag{6.15}$$

Furthermore it is also known that the angular momentum of a rigid body can be determined by equation 6.16.

$${}^{\mathrm{r}}\mathbf{L} = {}^{\mathrm{r}}\underline{\mathbf{J}}\,{}^{\mathrm{r}}\boldsymbol{\omega} \tag{6.16}$$

where:

${}^{\mathrm{r}}\underline{\mathbf{J}}$ is the moment of inertia of the rigid body in the RBRF [kgm^2]

As the final result, equation 6.15 and the derivative of equation 6.16 can be combined into equation 6.17, describing the full rotational dynamics of the rocket given in the RBRF.

$${}^{\mathrm{r}}\underline{\mathbf{J}}\,{}^{\mathrm{r}}\dot{\boldsymbol{\omega}} = {}^{\mathrm{r}}\boldsymbol{\tau}_{ext} - [{}^{\mathrm{r}}\boldsymbol{\omega}]_{\times}\,{}^{\mathrm{r}}\mathbf{L} = {}^{\mathrm{r}}\boldsymbol{\tau}_{ext} - [{}^{\mathrm{r}}\boldsymbol{\omega}]_{\times}\,{}^{\mathrm{r}}\underline{\mathbf{J}}\,{}^{\mathrm{r}}\boldsymbol{\omega} \tag{6.17}$$

6.2.2 Thrust

When applying a thrust force, ${}^{\mathrm{r}}\mathbf{T}$, to a point on a rocket located at a distance, ${}^{\mathrm{r}}\mathbf{d}$, away from the center of mass of the rocket, it results in a torque, ${}^{\mathrm{r}}\boldsymbol{\tau}_v$, given by equation 6.18.

$${}^{\mathrm{r}}\boldsymbol{\tau}_v = [{}^{\mathrm{r}}\mathbf{d}]_\times {}^{\mathrm{r}}\mathbf{T} \tag{6.18}$$

When neglecting the possibility of disturbances in the force produced by the boosters, only the vernier thrusters will be able to produce a torque. Furthermore each vernier thruster produces a torque in two axes. The torque produced by the vernier thrusters, ${}^{\mathrm{r}}\boldsymbol{\tau}_{vt}$, can be determined by equation 6.19.

$${}^{\mathrm{r}}\boldsymbol{\tau}_v = \sum_{i=1}^{4} [{}^{\mathrm{r}}\mathbf{d}_i]_\times {}^{\mathrm{r}}\mathbf{T}_{v,i} \tag{6.19}$$

As seen in figure 5.8 in section 5.4, the vernier thrusters are placed on the engine part of the rocket. Thereby the distance from the rocket CoM towards the vernier thrusters changes along with the rocket becoming shorter as it uses the fuel. This also means that the displacement vectors, ${}^{\mathrm{r}}\mathbf{d}_i$, are determined as a function of the height of fuel shell.

6.3 Dynamic equations

$$m(t){}^{\mathrm{r}}\dot{\mathbf{v}}(t) = {}^{\mathrm{r}}\mathbf{T}(t) - {}^{\mathrm{r}}\mathbf{f}_g(t) - {}^{\mathrm{r}}\mathbf{f}_D(t) \tag{6.20}$$

$$= {}^{\mathrm{r}}\mathbf{T}(t) - {}^{\mathrm{r}}_{\mathrm{i}}\mathbf{q}(t) \otimes \frac{\mu_{\mathrm{E}}}{||{}^{\mathrm{i}}\mathbf{r}(t)||^3}{}^{\mathrm{i}}\mathbf{r}(t) - \frac{1}{2}\rho_a {}^{\mathrm{r}}\mathbf{A}(t) \circledast {}^{\mathrm{r}}\mathbf{C}_D(t) \circledast {}^{\mathrm{r}}\mathbf{v}(t) \circledast |{}^{\mathrm{r}}\mathbf{v}(t)| \tag{6.21}$$

$${}^{\mathrm{r}}\mathbf{J}(t){}^{\mathrm{r}}\dot{\boldsymbol{\omega}}(t) = \sum_{k=1}^{4} [{}^{\mathrm{r}}\mathbf{d}_k(t)]_\times {}^{\mathrm{r}}\mathbf{T}_{v,k}(t) - {}^{\mathrm{r}}\boldsymbol{\omega}(t) \times {}^{\mathrm{r}}\underline{\mathbf{J}}(t){}^{\mathrm{r}}\boldsymbol{\omega}(t) \tag{6.22}$$

6.4 Moment of Inertia

The inertia matrix for AAURocket is modeled as a diagonal matrix with the diagonal entries being as shown in the equations below. The derivations for the equations are shown in appendix B.

$${}^{\mathrm{r}}J_x = \frac{3}{10}m_p r_p^2 + \frac{1}{2}\rho_o \pi r_o^4 h_f + \frac{1}{2}m_o r_o^2 + \frac{1}{2}m_f(r_f^2 + r_o^2) + \frac{1}{2}m_e r_e^2 \tag{6.23}$$

$${}^{\mathrm{r}}J_y = \frac{1}{20}m_p(5h_f^2 + 5h_f h_p + 2h_p^2 + 3r_p^2) + \frac{1}{12}m_o(3r_o^2 + h_f^2) + ... \tag{6.24}$$

$$\frac{1}{12}m_f(3(r_f^2 + r_o^2) + h_f^2) + \frac{1}{12}m_e(3h_f^2 + 6h_f h_e + 4h_e^2 + 3r_e^2)$$

$${}^{\mathrm{r}}J_z = \frac{1}{20}m_p(5h_f^2 + 5h_f h_p + 2h_p^2 + 3r_p^2) + \frac{1}{12}m_o(3r_o^2 + h_f^2) + ... \tag{6.25}$$

$$\frac{1}{12}m_f(3(r_f^2 + r_o^2) + h_f^2) + \frac{1}{12}m_e(3h_f^2 + 6h_f h_e + 4h_e^2 + 3r_e^2)$$

6.5 Thermal Modeling of the Gasification Process

Sources: [Yemets 08a]

This section will present some of the challenges introduced by the need for gasification of the HDPE. It is not fair to assume that the thrust of the rocket can be changed instantly and thereby it is necessary to take the PE gasification process into account. On the contrary, the focus of this thesis is not to control the thermal process, but to control attitude, position and velocity of the rocket during a launch to an altitude of 500 km. Hence the thermal modelling is kept to a minimum, but some simplified thermal dynamics are assumed to make sure that the dynamics of the gasification process is taken into account.

DNU has made some experimental estimations of the possible gasification rate of a HDPE rod. According to [Yemets 08a] the work at DNU, with PE as rocket fuel, has shown a Gasification Chamber (GC) with the possibility of consuming up to 0.01 m/s of a HDPE rod. With this knowledge, it is possible to calculate a theoretical fuel mass flow by using equation 6.26.

$$\dot{m}_f = -\dot{h}_f A_f \rho_f \tag{6.26}$$

where:

$\dot{m}_f$	is the fuel mass flow	[kg/s]
h_f	is the height of the fuel shell	[m]
A_f	is the cross-sectional area of the fuel shell	[m^2]

When the HDPE has been gasified in the GC, it will be led into the combustion chamber (CC) together with the oxidizer. The excess heat from the combustion is led to the GC walls to gasify the HDPE. As mentioned, it is not the focus of this thesis to describe a precise thermal model, hence a simple dynamic model, will be assumed. The gasification process will be modeled as a first order system with time constant, τ_{ga}. It is assumed that when the engine has been ignited, the CC always produces enough heat for the GC to be able gasify the fuel at a sufficient rate. The block diagram in figure 6.1 shows how the gasification process is modeled.

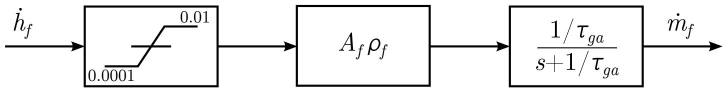


Figure 6.1: Shows the model assumed for the gasification process

As seen in figure 6.1, the input to the system is given as the shortening rate, $\dot{h}_f$, of the fuel shell. This has been bounded by by a maximum value of 0.01 m/s, to stay within the maximum rate determined by DNU. Furthermore it has also been bounded by an arbitrarily chosen minimum value of 0.0001 m/s to ensure that there is always some flow. This is done due to the assumption that the CC always produces enough heat to keep gasifying the fuel. This would not be the case if the the flow of fuel to the CC stays zero for a period of time. However, this is not expected to be of significance due to the fuel mass flow being highest in the beginning of the launch, and thereby the only risk of the engine turning off happens when it is close to burnout.

As well as it is the case for the gasification process, the combustion of the fuel does not happen instantly. This process is also simplified by assuming a first order model with a time constant, τ_{co}. Figure 6.2 shows the model assumed for the combustion inside the CC. With the help of a catalyst, the H_2O_2 is converted to O_2 and H_2O and enters the CC together with the gasified PE. After the combustion the total mass, $\dot{m}$, is exhausted through the expansion nozzle.

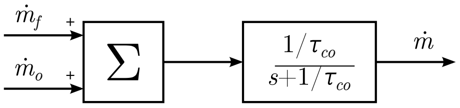


Figure 6.2: Shows the model assumed for the combustion process

As it is not entirely known how fast it is possible change the gasification- and combustion rate, the exact values of τ_{ga} and τ_{co} have not been determined. It is assumed that none of the time constants will be above 1 second, and thereby this value is chosen for both for the evaluation of the of the model. The model is evaluated with a step function on $\dot{h}_f$, to see to see how it performs for a sudden change, and with a falling ramp, to see how it performs as the fuel need is slowly decreasing during a launch. The result of the evaluation can be seen in figure 6.3.

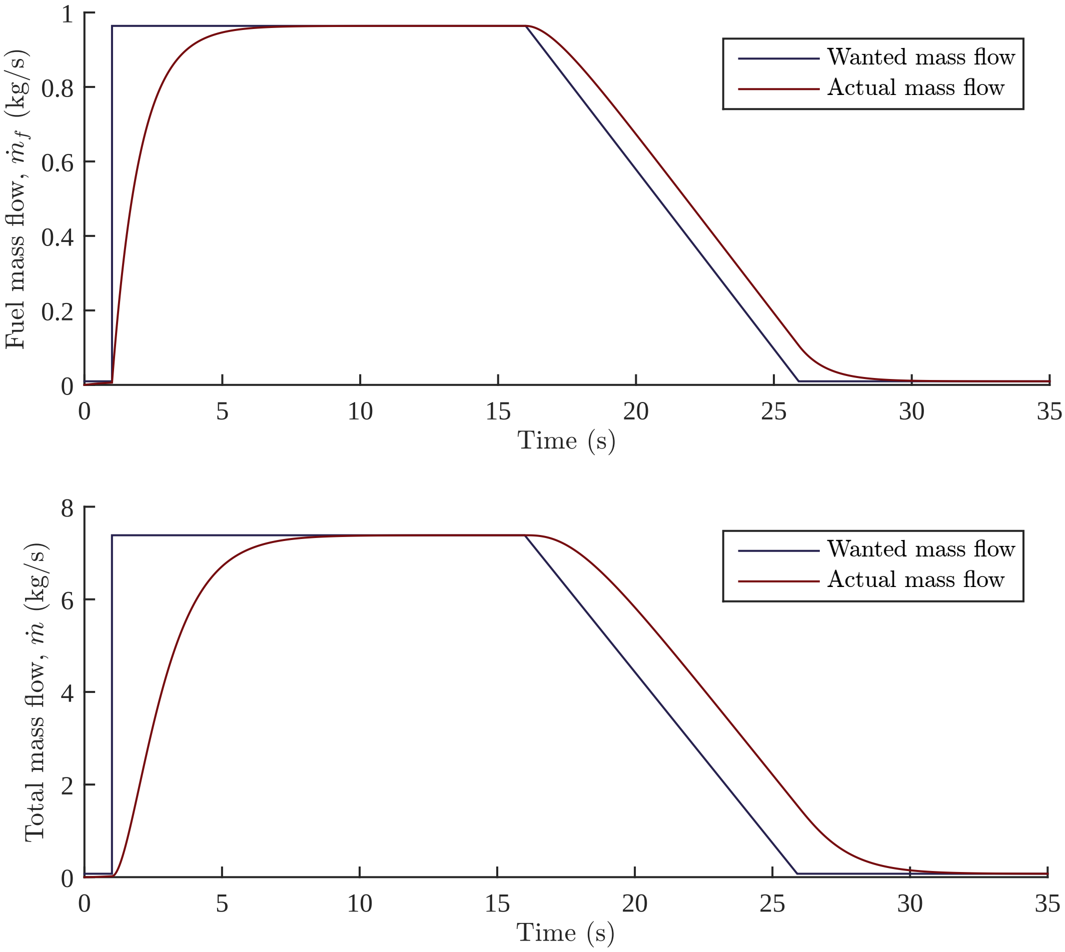


Figure 6.3: Shows the evaluation of the mass flows given by a step response and a decreasing ramp response.

On the step it is seen that the fuel mass flow takes about 4 seconds to settle. As the system is evaluated at maximum $\dot{h}_f$, a controller would not be able to make the system settle faster. For the total mass flow the settling time is about 6 seconds, and the same applies here for the controller. As the thrust, and thereby the mass flow, is expected to be highest in the start of the launch, a possibility to overcome this problem would be to increase the thrust before releasing the rocket for launch.

When looking at the responses of the decreasing ramp, it is seen that the system has a hard time following its reference. It is not expected to be necessary to design a controller to control the fuel- and total mass flow, since the controller designed for controlling the thrust will take care of this.

6.6 Subconclusion

In this chapter the general dynamic equations and disturbance forces for translational and rotational motion was presented. The full set of dynamical equations was derived to be:

$$\begin{aligned}
m(t)^{\mathrm{r}}\dot{\mathbf{v}}(t) =& ^{\mathrm{r}}\mathbf{T}(t) - {}^{\mathrm{r}}\mathbf{f}_g(t) - {}^{\mathrm{r}}\mathbf{f}_D(t) \\
=& ^{\mathrm{r}}\mathbf{T}(t) - {}^{\mathrm{r}}_{\mathrm{i}}\mathbf{q}(t) \otimes \frac{\mathrm{\mu}_{\mathrm{E}}}{||^{\mathrm{i}}\mathbf{r}(t)||^3}{}^{\mathrm{i}}\mathbf{r}(t) - \frac{1}{2}\rho_a{}^{\mathrm{r}}\mathbf{A}(t) \circledast {}^{\mathrm{r}}\mathbf{C}_D(t) \circledast {}^{\mathrm{r}}\mathbf{v}(t) \circledast |^{\mathrm{r}}\mathbf{v}(t)|
\end{aligned}$$

$$^{\mathrm{r}}\mathbf{J}(t)^{\mathrm{r}}\dot{\boldsymbol{\omega}}(t) = \sum_{k=1}^{4}[^{\mathrm{r}}\mathbf{d}_k(t)]_{\times}{}^{\mathrm{r}}\mathbf{T}_{v,k}(t) - {}^{\mathrm{r}}\boldsymbol{\omega}(t) \times {}^{\mathrm{r}}\underline{\mathbf{J}}(t)^{\mathrm{r}}\boldsymbol{\omega}(t)$$

Furthermore, the analytical expressions describing the evolving moment of inertia was introduced:

$$\begin{aligned}
^{\mathrm{r}}J_x =& \frac{3}{10}m_p r_p^2 + \frac{1}{2}\rho_o\pi r_o^4 h_f + \frac{1}{2}m_o r_o^2 + \frac{1}{2}m_f(r_f^2 + r_o^2) + \frac{1}{2}m_e r_e^2 \\
^{\mathrm{r}}J_y =& \frac{1}{20}m_p(5h_f^2 + 5h_f h_p + 2h_p^2 + 3r_p^2) + \frac{1}{12}m_o(3r_o^2 + h_f^2) + ... \\
& \frac{1}{12}m_f(3(r_f^2 + r_o^2) + h_f^2) + \frac{1}{12}m_e(3h_f^2 + 6h_f h_e + 4h_e^2 + 3r_e^2) \\
^{\mathrm{r}}J_z =& \frac{1}{20}m_p(5h_f^2 + 5h_f h_p + 2h_p^2 + 3r_p^2) + \frac{1}{12}m_o(3r_o^2 + h_f^2) + ... \\
& \frac{1}{12}m_f(3(r_f^2 + r_o^2) + h_f^2) + \frac{1}{12}m_e(3h_f^2 + 6h_f h_e + 4h_e^2 + 3r_e^2)
\end{aligned}$$

Finally a general approximation for the gasification process was modeled as:

$$\dot{m}_f = -\dot{h}_f A_f \rho_f$$

Chapter 7

Simulink Simulation Environment

In order to test the estimation and control methods and algorithms for AAURocket, a general simulation environment has been created. This environment has been implemented in Simulink. This is done primarily for ease of use, and to provide other people quick and easy access to different sub-systems within the simulation if they so desire.

While a lot of different dynamical components and environmental disturbances can be modeled and simulated in order to obtain the best possible truth model, the actual implementation has been limited to the dynamical equations and disturbances discussed in chapter 6.

In this chapter each major part of the simulation environment implemented in Simulink will be outlined. First the general dynamics, physical properties, environment, and quaternions will be presented, followed by the implementation of the sensor and actuator models.

Additionally a block containing the algorithms run on the AAURockets onboard computer is present, but not discussed in this section, as all the algorithms and calculations this block contains are presented in chapter 9 and 10.

7.1 AAURocket Dynamics and Environment

Figure 7.1 shows the general blocks concerning the dynamics and environment implemented in Simulink. The block named "AAURocket Dynamics" contains the dynamical equations of AAURocket as presented in section 6, and as such will not be discussed further in this section. The block named "Quaternions" calculates each of the different reference frame orientations with respect to each other. This subsystem block will not be discussed further in this section neither. The block named "AAURocket Mass, Moment of Inertia, and Height" calculates, as the name implies the mass, moment of inertia, and height of AAURocket. Finally, the block named "Environment" calculates each of the environmental parameters considered at the position of AAURocket.

7.1.1 AAURocket Mass, Moment of Inertia, and Height

This block uses oxidizer and fuel change rates as inputs in order to calculate the change in initial mass, and resulting change in height of the AAURocket. This is done by using "limited integrators" which has the initial masses as upper limits, and zero as the lower limit. In turn the integrated change rates are outputted as the total-, oxidizer-, and fuel mass of the rocket.

Additionally the developing moment of inertia is calculated throughout the simulation. The calculations involving the moment of inertia can be found in section 6.4.

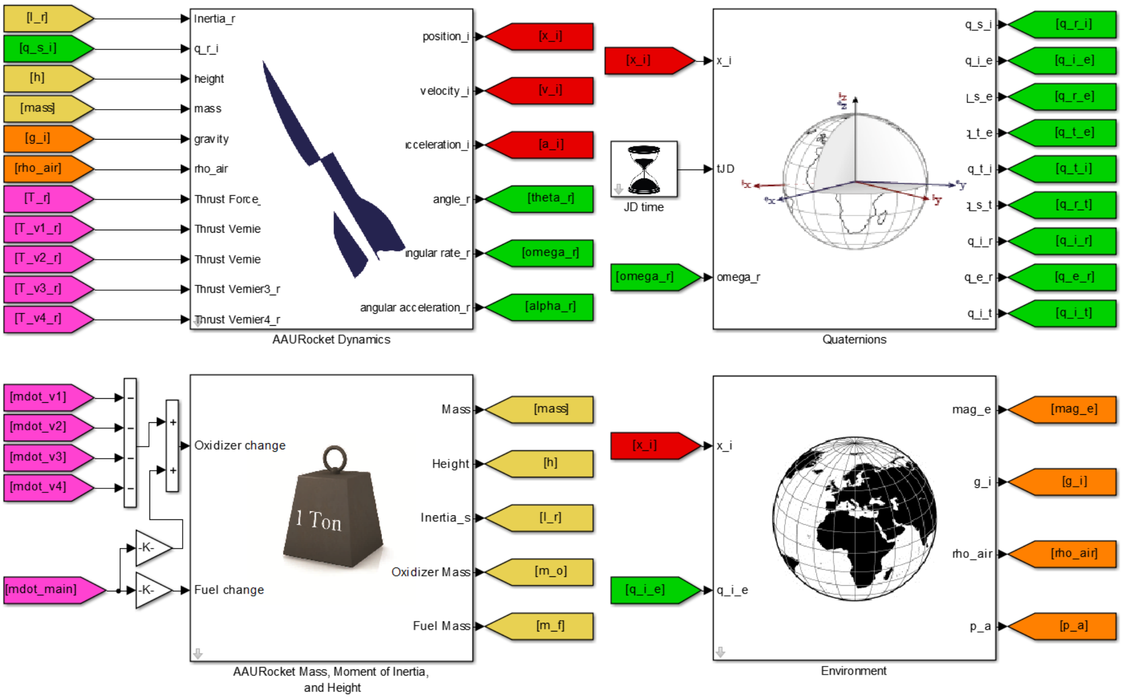


Figure 7.1: AAURocket dynamics and environment Simulink blocks. Each label is colored as identification. Red is positional states, green is rotational states, yellow is other physical non-state parameters regarding AAURocket, magenta is inputs and/or control signals, and orange is environmental parameters.

7.1.2 Environment

The "Environment" block contains the three primary environmental factors that is considered in this project: The gravitational acceleration of the Earth, the Earths magnetic field (calculated for the sake of the magnetometer), and the atmosphere of the Earth.

Gravitational Acceleration

The gravitational acceleration of the Earth has been implemented by the simple analytical model as given in chapter 6:

$$^{i}\mathbf{g} = \frac{\mu_E}{||^{i}\mathbf{r}||^3}\,^{i}\mathbf{r}$$

Magnetic Field Model

Source: [NOAA 15]

In general the Earths magnetic field can be considered as a giant dipole magnet within the center of the Earth, which is oriented with its magnetic south pole close to the Earths geographical north pole, and the magnetic north pole close to the Earths geographical south pole. However, the reality has the magnetic field strength varying across the entire surface of the Earth, as well as out into space. An example of the intensity of the Earths magnetic field at sea level can be seen on figure 7.2.

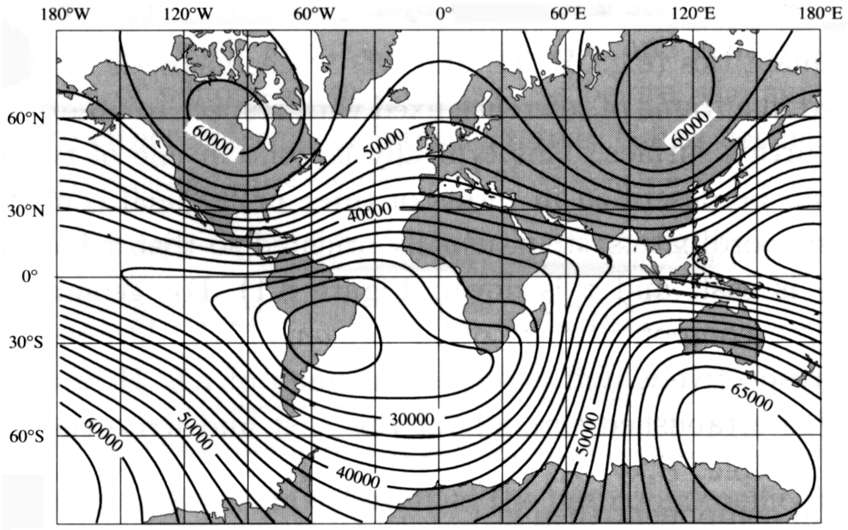


Figure 7.2: Geomagnetic field intensity based on an IGRF model [Ahem 09].

The magnetic field has been modeled using the International Geomagnetic Reference Field (IGRF), which is a mathematical description of the Earths magnetic field at a given spherical position. Due to the fact that the Earths magnetic field is changing over time, the IGRF model is only valid in five year periods. As such the current valid IGRF model is of the 11th generation and is inherently called IGRF11. This version is considered valid in the period 2010-2015.
Though the IGRF model accurately calculates the magnetic field strength at any position on and above the Earth, there are some magnetic disturbances it does not take into account as these are short time period events. One such disturbance is the currents in the ionosphere (60-600km) caused by solar winds which may induce variations in the magnetic field as high as 10%. However, these extra uncertainties are not considered within the scope of this project.

7.1.3 The Atmosphere of the Earth

Source: [NASA 76]

The atmospheric model of the Earth consists of two parts; a model for the atmospheric density and a model of the atmospheric pressure.

Atmospheric Density

The implemented atmospheric density model is split into two parts. For altitudes between 0 km and 50 km a continuous analytical model is utilized. This analytical expression is as follows:

$$1.277\mathrm{e}^{-0.1194h} \tag{7.1}$$

where

h is the height above sea-level [m]

This continuous model is a fitted exponential function that have been determined based on data samples from [NASA 76]. For altitudes above 50 km, the atmospheric density is determined by looking up values in a table. Between 50 km and 100 km the table has data points with a resolution of 1 km. From 100 km to 200 km the table has data points with a resolution of 5 km. Finally from 200 km and higher the table has a resolution of 20 km. Table 7.1 below shows how the atmospheric density is determined at various heights.

Height (km)	Calculation	Density (kg/m^3)
4.8	$1.277\mathrm{e}^{-0.1194\cdot 4.8}$	0.71993
63.7	$\rho_t(63)$	$0.210 \cdot 10^{-3}$
148.3	$\rho_t(145)$	$2.70 \cdot 10^{-9}$
431.9	$\rho_t(420)$	$2.00 \cdot 10^{-12}$

Table 7.1: A few examples of how the atmospheric density is determined at various heights, where the function ρ_t is a lookup table.

Atmospheric pressure

The atmospheric pressure model is implemented as an analytical model by using the ideal gas law which states:

$$p_a = \rho_a \frac{\mathfrak{R}}{\mathfrak{M}_a} T_a \tag{7.2}$$

where

$\mathfrak{M}_a$	is the molecular weight of air	[g/mol]
T_a	is the temperature of the air	[K]

Additionally the temperature of the air can be calculated with respect to height by the following equation:

$$T_a = T_0 - Lh \tag{7.3}$$

where

T_0	is initial temperature of the air at sea-level	[K]
L	is the temperature lapse rate	[K/km]

By substituting equation 7.3 into 7.2 a general expression for the pressure of the atmosphere is obtained:

$$p_a = \rho_a \frac{\mathfrak{R}}{\mathfrak{M}_a} (T_0 - Lh) \tag{7.4}$$

7.2 Sensor and Actuator Models

The sensor and actuator Simulink blocks are the interfaces between the onboard computer and the simulated truth model of the dynamics and environment of AAURocket. Figure 7.3 shows the sensor and actuator blocks as implemented in Simulink.

7.2.1 Sensors

The sensors are all modeled from their datasheet values presented in section 5.2, and as such all noise characteristics are assumed white Gaussian. It is a possibility that actual sensors would have other color variants than white, but in order to determine the actual color, noise estimation tests are needed.

In total there are four actual datasheet sensors which are simulated which include: a magnetometer, an accelerometer, a gyroscope, and a barometer. Two token sensors are also simulated which include: a GPS and a tachometer. Finally all mass flows into each combustion chamber of each engine are assumed measured as well, however, these flow sensors are not modelled with stochastic

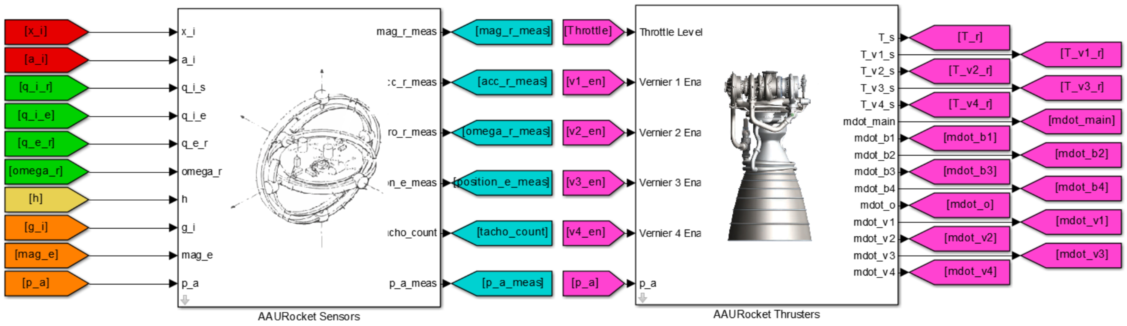


Figure 7.3: AAURocket sensors and actuator Simulink blocks. The new color code of cyan represents measurements and/or estimates.

uncertainty.

Magnetometer, Gyroscope, Barometer, and GPS

The general Simulink model for the magnetometer can be seen on figure 7.4.

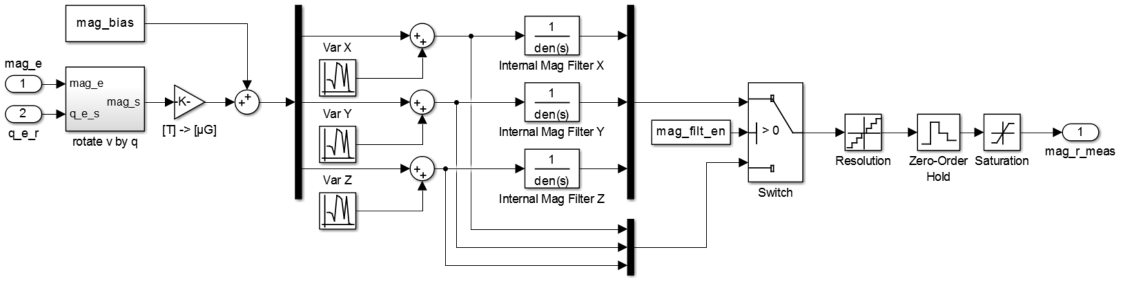


Figure 7.4: Structure of the modeled magnetometer sensor.

The magnetometer utilizes two primary inputs from the truth model: the magnetic field as calculated by the IGRF in the ECEF frame, and the quaternion ${}^{\mathrm{r}}_{\mathrm{e}}\mathbf{q}$. The magnetic field vector calculated in the ECEF frame is rotated into the RBRF and converted through a gain block to μG (which is the output unit of the sensor). The magnetic field vector is then added to a bias value set by the user, after which white Gaussian noise, in the magnitude specified by the datasheet, is added to each vector component. At this point the vector is either passed through, or bypasses the internal filter present on the ADIS16446. Whether or not the filter is used, is specified by the user, and throughout this project none of the internal filters have been utilized. The vector is finally passed through a "quantizer", a "zero-order hold", and a "saturation" block in order to represent the magnetometers resolution, discrete sampling, and absolute measurement range respectively.
In turn the true magnetic field and orientation of the LV enters the block, and an observed biased, noisy, and discrete measurement vector is returned.

The gyroscope,barometer, and GPS are all modelled in a similar fashion as the magnetometer.

Accelerometer

The general Simulink model for the magnetometer can be seen on figure 7.5.

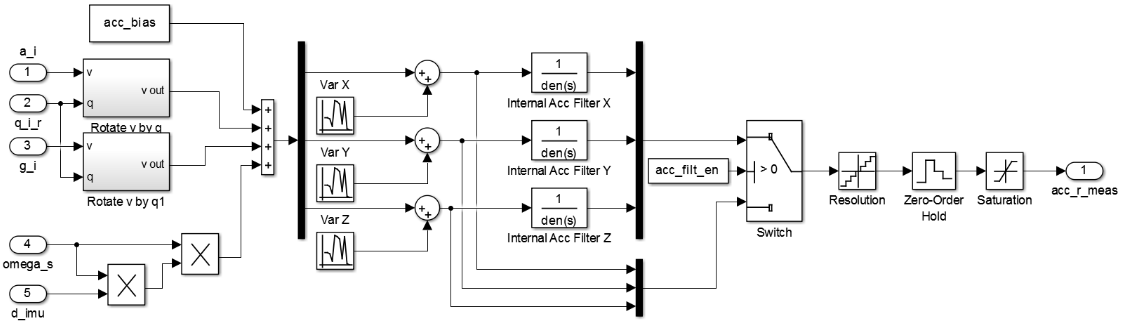


Figure 7.5: Structure of the modeled accelerometer sensor.

While the general structure of the accelerometer is similar to that of the magnetometer, it utilizes several additional inputs. As the name implies the accelerometer measures accelerations. As such, it not only experiences accelerations caused by translational accelerations, but also from rotations. The accelerations caused by rotation is only true for accelerometers not placed in the CoM of the system it measures. However, seeing as the CoM is almost always within the fuel canister, these accelerations has to be modeled. As such the nominal noiseless acceleration experienced by the accelerometer is modeled as:

$$^{\mathrm{r}}\mathbf{Acc} = {}^{\mathrm{r}}\mathbf{a} + {}^{\mathrm{r}}\mathbf{g} + {}^{\mathrm{r}}\boldsymbol{\omega} \times ({}^{\mathrm{r}}\boldsymbol{\omega} \times {}^{\mathrm{r}}\mathbf{d}_{IMU}) \tag{7.5}$$

where

$^{\mathrm{r}}\mathbf{Acc}$	is the accelerometer measurement	[m/s^2]
$^{\mathrm{r}}\mathbf{a}$	is the dynamical acceleration of AAURocket	[m/s^2]
$^{\mathrm{r}}\mathbf{g}$	is the gravitational acceleration imposed on AAURocket	[m/s^2]
$^{\mathrm{r}}\boldsymbol{\omega}$	is the angular velocity of AAURocket	[rad/s]
$^{\mathrm{r}}\mathbf{d}_{IMU}$	is the position of the IMU in the RBRF	[m]

The $^{\mathrm{r}}\boldsymbol{\omega} \times ({}^{\mathrm{r}}\boldsymbol{\omega} \times {}^{\mathrm{r}}\mathbf{d}_{IMU})$ term includes the acceleration experienced by the accelerometer due to displaced rotation. It is clear that the experienced acceleration is proportional to both angular velocity as well as displacement. As such, the magnitude of this effect will reduce as $^{\mathrm{r}}\mathbf{d}_{IMU}$ reduces during the ascent.
Furthermore, the reason for the addition of the gravitational acceleration, is that the gravity is already present as a negative contribution in $^{\mathrm{r}}\mathbf{a}$. Without this addition the accelerometer would experience acceleration towards the center of earth when it is in free fall, where normal performance will have it measuring zero.

The rest of the accelerometer model is identical to that of the magnetometer.

Tachometer

As mentioned in section 5.4.2 DC motors are mounted inside the engine block to control the rate of consumption of the fuel shell. It is assumed that these DC motors are equipped with tachometers in order to determine the amount of fuel shell that has been consumed. By knowing the amount of fuel shell consumed, it is in turn possible to calculate the remaining height of the rocket. As such the tachometer takes the height of the truth model as an input and outputs a height measurement. The truth height is "quantized" inside the tachometer Simulink block to make the signal a discrete stair signal. Following is an "edge detector" which detects each falling edge of the stair signal within the resolution of the tachometer. Finally these edges are passed to a "counter" block which has a range of 255 counts. As such tachometer outputs a value between 1-255 depending on how much of the shell has been consumed. Note that the range of 255 in this case means that the counter will wrap around to 1 after 255 has been reached. Hence, it is up to the onboard computer to keep track of the actual remaining height of AAURocket.

7.2.2 Actuators

The actuators considered in the simulation environment are all contained within the "AAURocket Thrusters" block, and the general contents of this block can be seen on figure 7.6.

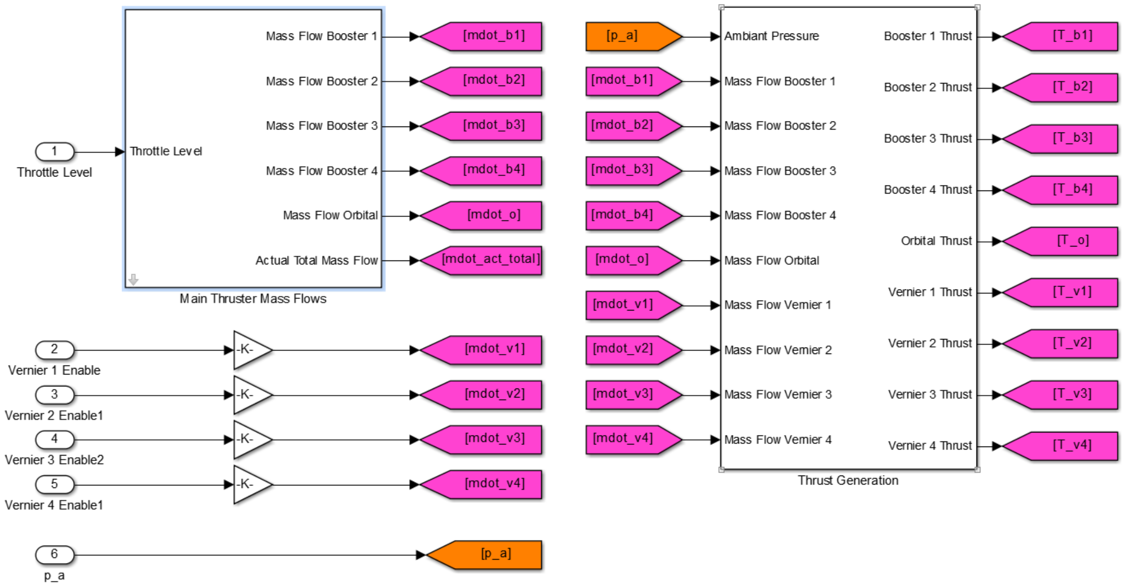


Figure 7.6: Structure of the modeled actuators on AAURocket.

The block takes six total inputs: a throttle level, four vernier thruster enable signals, and finally the pressure of the surrounding atmosphere. The primary signal flow of the actuator block is to first calculate the mass flow going into each combustion chamber, followed by the calculation of the thrust produced by each thruster.

As mentioned in section 5.6.3 it is assumed that all vernier thrusters have no throttle capability. As such they all function as either fully on or fully off, hence the relatively simple calculation with respect to mass flow, where the logical enable operator is simply multiplied with the vernier thruster nominal mass flow.

A more complex mass flow scheme is implemented for the remaining four boosters and one orbital thruster, inside the "Main Thruster Mass Flows" block. An illustration of the contents of this block can be seen on figure 7.7

The primary function of the block is to utilize the correct thrusters for a given throttle level as per the diagram on figure 5.5. This is done by analyzing the throttle level input and determine what range the current value lies within (i.e. five thrusters, three thrusters, or one thruster). If the throttle is above 0.5 the top section of the Simulink environment shown on figure 7.7 is active. Between 0.2625 and 0.5 and the middle section is the active one. Finally when the throttle is below 0.2625 the bottom section is active.

The two top most sections are identical in structure and the only real difference is whether five or

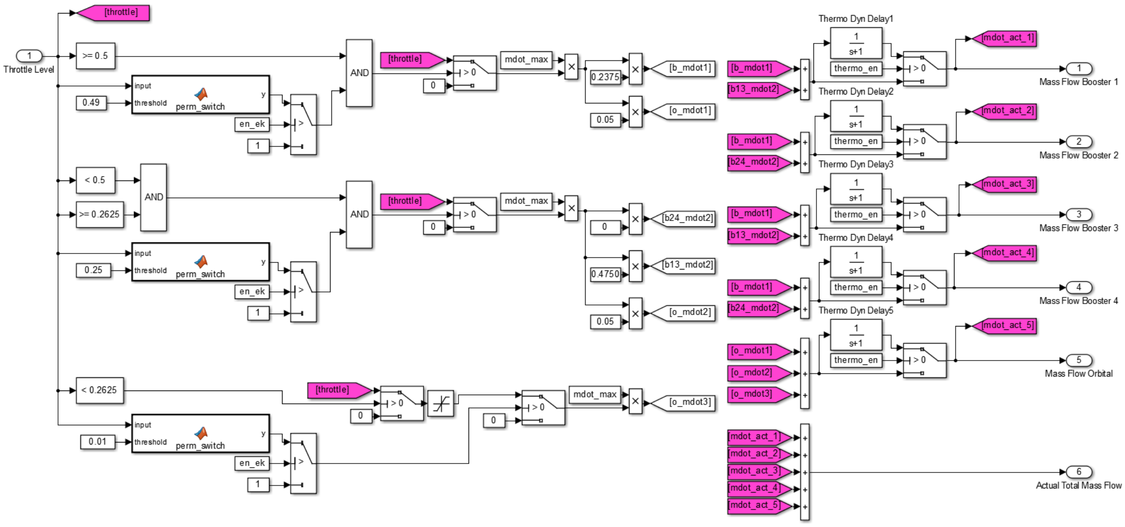


Figure 7.7: Structure of the primary mass flows calculator block.

three boosters are active. The mass flow is passed through a first order filter in the end in order to simulate the thermal delay from change in throttle input to actual change in shell consumption in the gasification process as per 6.5.

The bottom most section only has the orbital engine enabled. The section becomes active as soon as the throttle level goes below 0.2625, however, the mass flow output is limited to equal that of throttle level range 0.025 to 0.05. This means that when the throttle lies in the range between 0.05 and 0.2625, the actual thrust production of the thruster system will only equal 0.05. This is in accordance with the dead zone observed on figure 5.5.

Finally the two top most range sections are permanently shut off as soon as a new section is utilized. This means that if the throttle goes from the five thruster setup range into the three thruster setup range, it is no longer possible to turn on the two killed thrusters. This is also in accordance with the thruster setup presented in section 5.4.1 where it is delimited that no primary thruster is able to restart.

Finally the block outputs the mass flow of each individual thruster (booster 1 to 4 and the orbital thruster). Furthermore, a summarized mass flow is also outputted of the block.

It is possible for the user to enable/disable the thermal delay as well as the permanent kill option of the thrusters, if it is desired.

Each mass flow for each thruster is passed to the next block calculating the actual thrust output of each thruster. This is a two step process consisting of first calculating the chamber pressure with respect to the given mass flow, and then calculating the thrust based on this chamber pressure in conjunction with the ambient pressure.

The chamber pressure is calculated as [Turner 01]:

$$p_c = \frac{\dot{m}}{A^* \sqrt{\gamma \left(\frac{2}{\gamma+1}\right)^{\frac{\gamma+1}{\gamma-1}} \frac{\mathfrak{M}}{\Re T_c}}} \tag{7.6}$$

The actual thrust force produced by a single thruster is then calculated as [Turner 01]:

$$T = p_c A^* \sqrt{\frac{2\gamma^2}{\gamma-1} \left(\frac{2}{\gamma+1}\right)^{\frac{\gamma+1}{\gamma-1}} \left[1 - \frac{p_e}{p_c}^{\frac{\gamma-1}{\gamma}}\right]} + p_e A_e - p_a A_e \tag{7.7}$$

From equation 7.7 there is a requirement for $\frac{p_e}{p_c}$ to never exceed the maximum value of one as this would introduce a negative sign under the squareroot and in turn result in an imaginary thrust. To ensure this p_c must never be smaller than p_e. Furthermore p_c must never equal 0 as this will return "NaN". As such additional logic operations has been implemented to ensure that whenever $p_c < p_e$ or $p_c = 0$ then a thrust of 0 will be passed instead of the imaginary/NaN results.

Chapter 8

Ascent Trajectory

In this chapter the target ascent trajectory of AAURocket will be calculated.
Firstly different general ascent profiles are presented, of which one is chosen as the ascent guideline for AAURocket. Additionally two optimization problem statements will be presented: a "minimum propellant to orbit" and a "spare propellant to orbit". The first of which is described in detail throughout the chapter ("minimum propellant to orbit"), and the second of which is almost identical, and as such the differences are only outlined in the result section of "spare propellant to orbit".

8.1 Ascent Profiles

Sources: [Curtis 05] [Cornelisse 79]

When determining the ascent trajectory of a rocket the first thing to determine is what kind of ascent profile is to be utilized. While all ascent trajectories are different they can generally be identified and grouped in two different categories: The Direct Ascent (DA) and The Hohmann Transfer Ascent (HTA).
In the following section the two different ascent profiles will be presented. Furthermore, a special ascent strategy called the gravity turn will also be presented. The choice of ascent profile for the AAURocket will be outlined at the end of this section.

8.1.1 Direct Ascent

The direct ascent profile is the most intuitive of the two different profiles. The purpose of the profile is to launch a rocket from an initial point and through actuation and control be injected directly into the required orbit. As such it is the most direct route from launch pad to orbit, which is also apparent from the name.
An apparent advantage of the DA-profile is possible line-of-sight (LOS) from launch pad to LV throughout parts or the entire ascent phase. This LOS capability means that the launch operation can be carried out from a single ground station.
Furthermore, with a faster ascent profile also comes a shorter ground track. This means if a launch is aborted during the ascent, the possible crash area is limited in to a triangular area downrange of the launch pad. Obviously this triangular area is time dependant, as the further into the ascent the launch is aborted, the greater the possible crash area.
An inherent disadvantage of the DA-profile is its lower propellant efficiency when compared to the HTA-profile. This reduced efficiency is an effect of how quick the rocket is injected into the orbit from time of launch. If the ascent has to be very quick, then a lot of vertical moment has

to be built up in the rocket, which in turns leads to greater gravity losses.

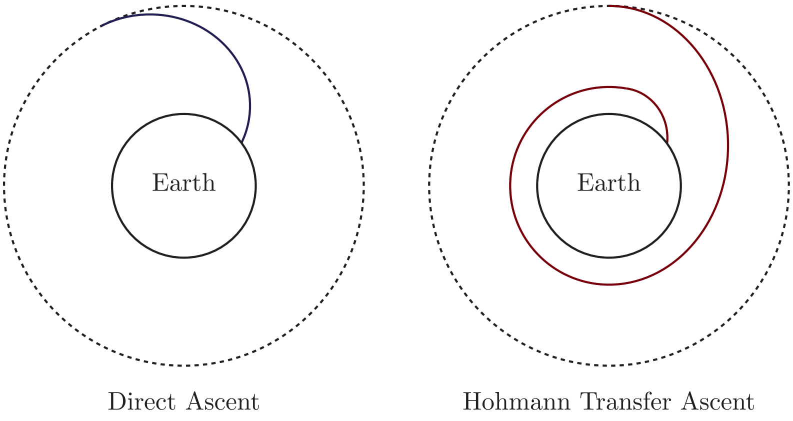


Figure 8.1: The two different ascent profiles.

8.1.2 Hohmann Transfer Ascent

The Hohmann transfer ascent profile is less intuitive than the DA-profile (see figure 8.1). As the name implies this ascent profile utilizes the general method of a Hohmann transfer to achieve orbital injection.

In short the Hohmann transfer is the most propellant efficient transfer maneuver an impulsive action spacecraft can make in order to go from one orbit to another. This transfer maneuver consists of two primary engine burns. As an example consider a spacecraft transfering from a lower orbit into a higher orbit. The first burn is then made at the periapsis of the initial orbit to raise the apoapsis to intersect the destination orbit, and the second burn is made at the new apoapsis to raise the periapsis to intersect that of the destination orbit. The trajectory between the two burns (connecting the two orbits) is called the Hohmann transfer trajectory.

The HTA profile consists of three primary action periods as well as one or two coasting periods. The first action period is an ascent from the surface of the Earth to an initial "parking" orbit of about 200 km (this initial ascent is in essence a DA). Depending on the destination orbit, a coasting period can take place from parking orbit injection until the Hohmann transfer burn, which is the second action. This burn will increase the apoapsis of the parking orbit to intersect that of the destination orbit. From here the LV will coast while travelling on the Hohmann transfer trajectory, and finally make the second burn to raise the periapsis of the parking orbit to intersect the destination orbit.

The advantage of the HTA profile is, the optimum propellant consumption is always guaranteed for chemical rockets on destination orbits higher than 300 km [Cornelisse 79]. However, the HTA profile requires the LV to be able to turn its primary engine completely off during the coasting periods. While this is not necessarily a disadvantage for every LV, it is considered a disadvantage for a LV in the format considered in this project, due to the melting of the propellant shell which requires thermodynamic momentum to be built up. Another aspect of the HTA profile is its inherently long downrange ground track. If radio link is required during the ascent, several ground stations are needed in order to maintain constant LOS.

8.1.3 The Gravity Turn

The gravity turn is an ascent (or descent) strategy which can be applied to any trajectory belonging to the DA category. The core concept of the gravity turn is to utilize the gravity of the

celestial body that a spacecraft is ascending from (or descending to) to minimize the amount of attitude control action throughout the trajectory.
As an example consider an LV with initial position on the surface of the Earth with initial attitude normal to its surface. The LV will start its ascent with a short period (seconds) of completely vertical ascent, followed by a control action turning the LV into an initial tipped angle (angle different from vertical). From here no further attitude control action should be needed, as the gravity will assist the LV in making a passive turn into orbit injection. As such the LV will retain zero (or close to zero) angle of attack throughout the ascent.
Two inherent advantages arise from using the gravity turn strategy: low aerodynamic stress, as a result of zero angle of attack and minimum propellant expenditure on attitude control actions.
However, a trade off evaluation has to be made of when to do the initial pitch over maneuver, in order to reduce the combined gravity and aerodynamic drag losses. Gravity losses are time dependant in the sense that the quicker the LV gets to orbit, and as such a fast downrange ascent is desired to reduce gravity losses. Conversely aerodynamic drag is only present at low altitudes and is velocity dependent, and therefore a slower vertical ascent is desired to reduce aerodynamic drag losses.

8.1.4 Choice of Ascent Profile

The ascent profile chosen for AAURocket is the DA approach. The primary reason for choosing DA over HTA is that the primary engine of AAURocket is not able to restart. Furthermore, it is deemed advantageous that only a single ground station is needed during the ascent for the DA profile.
The gravity turn strategy is not actively chased in the calculation of the ascent trajectory, however, it is expected that the optimized trajectory will resemble the gravity turn, at least during the exo-atmospheric stage of the ascent.

8.2 Optimum Propellant Consumption Ascent Trajectory

This section will outline the general problem formulation for "minimum propellant to orbit". The "spare propellant to orbit" formulation can be found in section 8.3.2, however, only the differences between the two problems will be listed.

8.2.1 Division of the problem

When constructing the optimization problem formulation for a LV ascent trajectory, it is possible to divide the ascent in to different phases and solve each phase separately. Such a division is advantageous in case of multistaged LVs where each stage will have different dynamics, and the changes to these dynamic changes are instantaneous. As such, a problem statement can be posed for each individual stage of the rocket.
An example of phase division is to divide the ascent with respect to the primary disturbance forces. First an endo-atmospheric phase where two primary disturbances are present in the dynamic equations, namely aerodynamic drag and gravity. This phase ends when the aerodynamic drag disturbance diminishes to relative marginal magnitude in compared to the gravity disturbance, at which point the exo-atmospheric phase begins. This phase lasts the remainder of the ascent.

While the endo/exo-atmospheric division could be used, there is no decisive target for states and inputs at the transition between the two phases.
Seeing as the AAURocket will utilize a continuous burn in an infinite stage setup, the stage phase division will not be utilized either. Instead the optimization problem will be posed as a single

phase from t_0 at launch pad to some free t_f at orbital injection.

8.2.2 Problem Formulation

The goal of the optimization problem is to calculate the optimal trajectory to orbit for the LV based on a primary objective function, while also calculating the accompanying optimal maximum thrust and thrust profile.

The primary objective is to determine the least amount of propellant needed to get into orbit. As such, the objective function is formulated as:.

$$minimize \quad m(t_0) \tag{8.1}$$

Or in other words, the objective is to minimize the initial mass of the entire system, which in turn will result in the lightest possible LV that can obtain the required orbit, which inherently is the LV with the optimal propellant consumption.
The objective function is subject to several different constraints, which is grouped as follows: dynamics of the LV, path constraints, and boundary constraints. Each of these constraint groups will be discussed in the following sections.

8.2.3 Dynamics

The dynamical equations of the optimization problem is given in the ECI frame, and include the two primary disturbances considered in the trajectory calculations (gravity and aerodynamic drag). The dynamical equations are formulated as follows:

$$\begin{aligned} \dot{\mathbf{r}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= \frac{\eta T_{max}}{m}\mathbf{1}_{\mathrm{A}} - \mathbf{g}(\mathbf{r}) - \frac{\mathrm{D}_A(||\mathbf{r}||, \mathbf{v}_r)}{m}\mathbf{1}_{\mathrm{A}} - \frac{\mathrm{D}_N(||\mathbf{r}||, \mathbf{v}_r)}{m}\mathbf{1}_{\mathrm{N}} \\ \dot{m} &= -\frac{\eta T_{max}}{\mathrm{g}_0 I_{sp}(||\mathbf{r}||)} \end{aligned} \tag{8.2}$$

with

$$\begin{aligned} \mathbf{v}_r &= \mathbf{v} - \omega_{\mathrm{E}} \times \mathbf{r} \\ T_{max} &= m(t_0)\mathrm{g}_0 p \end{aligned}$$

where

$\mathbf{r} =$	${}^{\mathrm{i}}\mathbf{r}(t)$	is the position vector of the LV [m]
$\mathbf{v} =$	${}^{\mathrm{i}}\mathbf{v}(t)$	is the velocity vector of the LV [m/s]
$m =$	$m(t)$	is the mass of the LV [kg]
$\mathbf{1}_{\mathrm{A}} =$	${}^{\mathrm{i}}\mathbf{1}_{\mathrm{A}}(t)$	is the LV axial unit vector (control vector) [·]
$\mathbf{1}_{\mathrm{N}} =$	${}^{\mathrm{i}}\mathbf{1}_{\mathrm{N}}(t)$	is the LV normal unit vector [·]
$\eta =$	$\eta(t)$	is the throttle of the LV [·]
$\mathbf{v}_r$		is the air relative velocity vector of the LV [m/s]
T_{max}		is the maximum thrust that the LV is able to produce [kgm/s^2]
p		is the ratio between initial weight of the LV and acceleration in g [·]
$\mathbf{g}(\mathbf{r})$		is the gravity loss [m/s^2]
$\mathrm{D}_{\mathrm{A}}(\|\|\mathbf{r}\|\|, \mathbf{v}_r)$		is the axial aerodynamic drag loss component [kgm/s^2]
$\mathrm{D}_{\mathrm{N}}(\|\|\mathbf{r}\|\|, \mathbf{v}_r)$		is the normal aerodynamic drag loss component [kgm/s^2]
$I_{sp}(\|\|\mathbf{r}\|\|)$		is the specific impulse of the primary thrusters with respect to height [s]

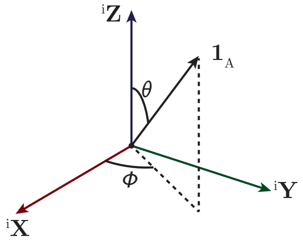
Of these variables $\mathbf{r}, \mathbf{v}$, and m are states, θ,ϕ (which inherently means $\mathbf{1}_{\mathrm{A}}$), and η are inputs to be optimized, and p is a constant parameter to be optimized. Furthermore, the final time t_f is a free variable.

In the following sections each of the different constraint functions and variables will be presented, starting with input vectors, followed by disturbance forces, path constraints, and finally boundary constraints.

LV orientation vectors

The LV in the optimization problem is represented by two unit vectors: $\mathbf{1}_{\mathrm{A}}$ and $\mathbf{1}_{\mathrm{N}}$, which are the axial and normal unit vectors for the LV respectively.

The axial unit vector is calculated as (see figure 8.2:



$^{\mathrm{i}}\mathbf{Z}$
$\mathbf{1}_{\mathrm{A}}$
θ
ϕ
$^{\mathrm{i}}\mathbf{Y}$
$^{\mathrm{i}}\mathbf{X}$

Figure 8.2: The input variables θ and ϕ shown in the ECI frame, defining the $\mathbf{1}_A$ vector.

$$\mathbf{1}_{\mathrm{A}} = \begin{bmatrix} \sin\theta(t)\cos\phi(t) \\ \sin\theta(t)\sin\phi(t) \\ \cos\theta(t) \end{bmatrix} \tag{8.3}$$

where

θ is the angle between the ECI z-axis and $\mathbf{1}_{\mathrm{A}}$ [rad]
ϕ is the angle between the ECI x-axis and $\mathbf{1}_{\mathrm{A}}$ [rad]

So while $\mathbf{1}_{\mathrm{A}}$ is the control vector of the problem, it is θ and ϕ that are in reality the inputs to be optimized.

The LV normal unit vector is in turn calculated as:

$$\mathbf{1}_{\mathrm{N}} = \mathbf{1}_{\mathrm{A}} \times (\mathbf{1}_{\mathrm{A}} \times \mathbf{1}_{\mathbf{v}_r}) \tag{8.4}$$

Figure 8.3 shows the relation between the two body vectors,$\mathbf{1}_{\mathrm{A}}$ and $\mathbf{1}_{\mathrm{N}}$, and the relative velocity $\mathbf{1}_{\mathbf{v}_r}$.

From equation 8.4 and figure 8.3 it is clear that $\mathbf{1}_{\mathrm{N}}$ will always lies in the plane spanned by $\mathbf{1}_{\mathrm{A}}$ and $\mathbf{1}_{\mathbf{v_r}}$.

Gravity

One of the two primary disturbance forces considered in the trajectory calculation is the gravity. Of the two disturbances, gravity is the only one present throughout the entirety of the trajectory. The gravity formulation used is as follows:

$$\mathbf{g}(\mathbf{r}) = \frac{\mu_{\mathrm{E}}}{||\mathbf{r}||^3}\mathbf{r} \tag{8.5}$$

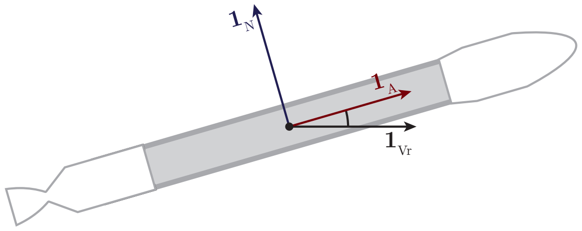


Figure 8.3: The body vectors of the LV ($\mathbf{1}_{\text{A}}$ and $\mathbf{1}_{\text{N}}$), with respect to the relative velocity of the LV $\mathbf{1}_{\mathbf{v}_r}$. The angle α between $\mathbf{1}_{\text{A}}$ and $\mathbf{1}_{\mathbf{v}_r}$ is the angle of attack of the LV.

The energy loss that the gravity imposes is proportional with the time spent getting into orbit. As such, the gravity by itself dictates that the LV needs to perform its ascent trajectory in the shortest possible time frame. Furthermore, to reduce gravity losses it is also preferable to start downrange acceleration as soon as possible after t_0.

Aerodynamic Drag

The second primary disturbance is the aerodynamic drag force. This disturbance is derived from the standard aerodynamic drag formula introduced in section 6.1.2, and restated in equation 8.6.

$$\text{D}(||\mathbf{r}||, \mathbf{v}) = \frac{\text{C}_\text{D}\text{A}||\mathbf{v}^2||\rho_{air}(||\mathbf{r}||)}{2} \tag{8.6}$$

As mentioned in section 6.1.2 the term $\text{C}_\text{D}\text{A}$ only holds true when the object described is a symmetrically shaped object described in an orthogonal basis. Since the LV is not symmetrical, it is needed to expand the term $\text{C}_\text{D}\text{A}$ in order to accurately describe the aerodynamic drag experienced by the LV.

The expansion of the term will result in two components since the LV is symmetrical about one axis, and so two of the three aerodynamic drag coefficients are equal. Consequently the aerodynamic drag is modelled as an axial and normal drag component. The expanded aerodynamic axial and normal drag equations can be seen in equation 8.7 and 8.8 respectively.

$$\text{D}_\text{A}(||\mathbf{r}||, \mathbf{v}_r) = \frac{\text{C}_\text{DA}\text{A}_\text{A}}{2}||\mathbf{v}_r||^2\rho_{air}(||\mathbf{r}||)(\mathbf{1}_\text{A}\cdot\mathbf{1}_{\mathbf{v}_r}) \tag{8.7}$$

$$\text{D}_\text{N}(||\mathbf{r}||, \mathbf{v}_r) = \frac{\text{C}_\text{DN}\text{A}_\text{N}m}{2}||\mathbf{v}_r||^2\rho_{air}(||\mathbf{r}||)(\mathbf{1}_\text{N}\cdot\mathbf{1}_{\mathbf{v}_r}) \tag{8.8}$$

where

C_DA is the axial aerodynamic drag coefficient (cone) of the rocket $[\cdot]$
C_DN is the normal aerodynamic drag coefficient (cylinder) of the rocket $[\cdot]$
A_A is the axial area of the rocket $[\text{m}^2]$
A_N is the normal area-to-kg factor of the rocket $[\text{m}^2/\text{kg}]$

It is clear from equation 8.7 and 8.8 that if the LV is travelling with an angle of attack of 0 radians then the term $\mathbf{1}_\text{N}\cdot\mathbf{1}_{\mathbf{v}_r}$ becomes 0 and the axial component is the only term present in the equation (which includes the head on collision of air onto a cone shaped nose). And conversely the same is true for the normal component when the angle of attack is equal to $\frac{\pi}{2}$ radians (which includes the head on collision of air onto a cylindrical body).

Note that the normal drag term is dependant on the mass, which reduces the normal drag as the LVs shell is consumed.

The energy loss that the aerodynamic drag imposes is proportional to both the time spent in the atmosphere as well as the squared velocity travelled through it. Due to the limited height of the atmosphere, the aerodynamic drag by itself dictates that the LV needs to fly relatively slow in a completely vertical ascent, until the atmosphere is negligible. So even though the drag loss is time dependant, the squared velocity dependability results in slower ascent times.

8.2.4 Path Constraints

Path constraints are expressions that must not be violated at any point during the calculations of the ascent trajectory. These path constraints include the state and input bounds as well as mathematical expressions ensuring the structural stability of the LV.

The state constraints are as follows:

$$\mathrm{R_E} \leq ||\mathbf{r}|| \tag{8.9}$$

$$150 \leq m \tag{8.10}$$

Equation 8.9 eliminates all solutions where the LV uses the gravity to boost itself through the center of the Earth and out the other side, and equation 8.10 ensures that no mass is ejected from the payload and engine masses (i.e. only propellant is expelled from the system).

The input constraints are stated as follows:

$$0 \leq \theta \leq \pi \tag{8.11}$$

$$0 \leq \phi < 2\pi \tag{8.12}$$

$$0.025 \leq \eta \leq 1 \tag{8.13}$$

Equation 8.11 and 8.12 ensures that there is only one unique representation of the $\mathbf{1}_\mathrm{A}$ vector. Furthermore equation 8.13 is merely a definition of the throttle, seeing as this is a percentage of the maximum thrust. The lower bound of 2.5% is in accordance with the design goal of the main engine never shutting off (see section 5.4.1).

There are two major mathematical path constraints which are posed in order to ensure structural survival of the LV during the ascent. These are: maximum allowable acceleration of the LV, and maximum allowable dynamic pressure of the LV.

Maximum Allowable Acceleration

The maximum allowable acceleration constraint is necessary in order to ensure that the thrust force does not rip the LV apart during ascent. This is a particularly important aspect when considering the self foraging setup where the thrusters stay the same throughout the ascent, compared to staged LVs where the thrusters are changed (and down scaled) for each successive stage.
By using the Ariane 5 rocket as a case reference, an initial upper bound can be set on the maximum allowable g-force on the rocket (see figure 5.3). As seen at no point does the Ariane 5 exceed a total g-force of 5 g, which will be the guideline maximum allowable g-force set for the AAURocket.

The maximum allowable acceleration constraint is calculated as:

$$a = \frac{\eta T_{max}}{m} \leq a_{\max} = 5 \; [\mathrm{g}_0] \tag{8.14}$$

Maximum Allowable Dynamic Pressure

The dynamic pressure is the pressure that the atmosphere exerts on the LV during its flight through the atmosphere. The dynamic pressure constraint is calculated as:

$$q = \frac{1}{2}\mathbf{v}^2 \rho_{air}(||\mathbf{r}||) \leq q_{\max} = 33 \; [\mathrm{kPa}] \tag{8.15}$$

The value of q_{max} has been chosen to match the values of both the Ariane 5 rocket and the Space Shuttle, which has q_{max} values of 35 kPa and 33 kPa respectively [Blau 14] [Jackson 01].
By simple analysis of equation 8.15 it is clear that at t_0 the dynamic pressure q is 0, seeing as the velocity is 0. Furthermore, at atmospheric exit the density of the air becomes 0, and in turn the dynamic pressure q will once again become 0. As such, a maximum point exists in the ascent through the atmosphere where the velocity and the density will result in the highest q value, called $q_{\max}$. This is a critical point of every LV and is often considered a milestone in the ascent. For some LVs active measure is taken to overcome the point of $q_{\max}$ by reducing the thrust level while passing through this critical phase. This will also be the case for AAURocket of the optimization hits the upper bound set by this constraint, as this will force the thrust level to be reduced.

8.2.5 Boundary Constraints

The boundary constraints of the optimization will ensure that the ascent trajectory starts at the initial point of the launch pads, and ends with the correct orbital injection. While the initial conditions of the problem are well known (with the exception of the initial mass), the terminal conditions are not and are instead governed by equations ensuring that orbital injection has been obtained.
The initial conditions for the problem are given as:

$$\begin{aligned}
\mathbf{r}(t_0) &= \mathbf{r}_{\mathrm{AAU}} \\
\mathbf{v}(t_0) &= \mathbf{1}_{\mathbf{r}_{\mathrm{AAU}}} + \mathbf{v}_{\mathrm{coriolis}} \\
\mathbf{1}_{\mathrm{A}}(t_0) &= \mathbf{1}_{\mathbf{r}_{\mathrm{AAU}}} \\
m(t_0) &= m(t_0)
\end{aligned}$$

These conditions state that the LV is positioned at AAU, has an initial vertical velocity of 1 m/s combined with a sideways velocity matching the coriolis velocity at AAU, and is facing straight up. The last parameter of $m(t_0)$ is the state to be optimized and is consequently free.

The final conditions ensuring orbital injection at t_f are stated as:

$$\text{(Mass)} \qquad m(t_f) = 150 \tag{8.16}$$

$$\text{(Orbit Height)} \qquad ||\mathbf{r}(t_f)|| = \mathrm{R_E} + ||\mathbf{r}_o|| \tag{8.17}$$

$$\text{(Orbit Velocity)} \qquad ||\mathbf{v}(t_f)|| = ||\mathbf{v}_o|| \tag{8.18}$$

$$\text{(Eccentricity)} \qquad \mathbf{r}(t_f) \cdot \mathbf{v}(t_f) = 0 \tag{8.19}$$

$$\text{(Inclination)} \qquad \mathbf{1}_{{}^{i}\mathbf{Z}} \cdot \mathbf{h}(t_f) = (\mathrm{R_E} + ||\mathbf{r}_o||)\, ||\mathbf{v}_o||\, \cos(i) \tag{8.20}$$

$$\text{(Ascent Direction)} \qquad \mathbf{r}_x(t_f) \leq \mathbf{r}_x(t_0) \tag{8.21}$$

with t_f as a free variable.

Equation 8.16, 8.17, and 8.18 are state magnitude constraints ensuring fixed final mass, correct orbital height, and correct orbital speed.
Equation 8.19 ensures circularity of the orbit by utilizing the dot product to maintain that the position and velocity vector are perpendicular at the time t_f, which is the case for circular orbits as mentioned in section 2.4.1.
Equation 8.20 is also a utilization of the dot product. Here the angle between the ECIs Z axis vector and the angular momentum of the orbit is calculated and constrained to match that of the target orbits inclination angle. However, this equation coupled with the initial start point of the launch results in two possible trajectories. Both starting at AAU but one travelling in a North-Eastern direction and the other in a South-Eastern direction. To ensure that each successive solution of the problem has the same unique boundary equation 8.21 is posed.

8.2.6 Method

Source: [Falugi 10]

The optimization problem has been implemented into the Imperial College London Optimal Control Software (ICLOCS) which is a Matlab interface that has been set to utilize the COIN-OR Interior Point Optimizer (IPOPT), the Suite of Nonlinear and Differential/Algebraic equation Solvers (SUNDIALS) CVODES-solver, and the Harwell Subroutine Library (HSL) ma57 linear solver; to solve complex non-linear control problems.

The ICLOCS Matlab code allows for user specified optimal control problem formulations on the following form [Falugi 10]:

$$\min \int_{t_0}^{t_f} L(x(t), u(t), p, t)dt + E(x_0, x_f, u_0, u_f, p, t_f) \tag{8.22}$$

subject to

$$\dot{x} = f(x(t), u(t), p, t), x(t_0) = x_0 \qquad \forall t \in [t_0, t_f] \tag{8.23}$$

$$g_L \leq g(x(t), u(t), p, t) \leq g_U \qquad \forall t \in [t_0, t_f] \tag{8.24}$$

$$b_L \leq b(x_0, x_f, u_0, u_f, p, t_f) \leq b_U \qquad \forall t \in [t_0, t_f] \tag{8.25}$$

$$x_L \leq x(t) \leq x_U \qquad \forall t \in [t_0, t_f] \tag{8.26}$$

$$u_L \leq u(t) \leq u_U \qquad \forall t \in [t_0, t_f] \tag{8.27}$$

$$p_L \leq p \leq p_U \tag{8.28}$$

where

Equation 8.22	is the objective function stated as a Bolza problem
Equation 8.23	is the dynamical equation constraints
Equation 8.24	is the path constraints
Equation 8.26	is the state constraints
Equation 8.27	is the input constraints
Equation 8.28	is the parameter constraints

It is clear that the optimization problem utilizes almost the full range of the capabilities of ICLOCS. While the equations specify an objective function on Bolza form, it is possible to also pass an object function on Lagrangian or Mayer Form. Seeing as the objective function of equation 8.1 is only dependent on the initial mass state, the problem is passed as a Mayer problem to ICLOCS.
Apart from the objective function, all other ICLOCS constraint functions are utilized to solve the optimization problem.
For complete code implementation of the optimization problem into ICLOCS, see appendix C.

The implementation of the posed problem has been modified by several different approximations, substitutions, and normalizations in order to ease the load of calculation on the computer. The most major of the modifications are mentioned in the following sections.

Normalizers

The optimization includes states and control vectors which are several orders of magnitudes in difference when compared to each other. As an example the position vector exists in the magnitude range of 6.3-6.8 million while the magnitude of the throttle input is in the range 0-1. In order to reduce the conditioning of the problem a nondimensionalization is performed, and all physical states are normalized around the values of $\mathrm{R_E}$ and $\mathrm{g_0}$. The normalizers for each physical unit are as follows:

$$\begin{aligned} \mathrm{t}_n &= \sqrt{\frac{\mathrm{R_E}}{\mathrm{g_0}}} \\ \mathrm{r}_n &= \mathrm{R_E} \\ \mathrm{v}_n &= \sqrt{\mathrm{R_E g_0}} \\ \mathrm{a}_n &= \mathrm{g_0} \\ \mathrm{m}_n &= 1000 \end{aligned}$$

where

t_n	is the time normalizer [s]
r_n	is the position normalizer [m]
v_n	is the velocity normalizer [m/s]
a_n	is the acceleration normalizer [m/s^2]
m_n	is the mass normalizer [kg]

After each solution the states are again redimensionalized with the same normalizers.

Exclusion of Native Matlab functions

Several native matlab functions has been excluded from the implementation, some because they were simply not able to work inside the optimization loop, and others because their inherent structure made the problem unsolvable in the different searches for a minimum.

A function that proved to slow the calculations significantly as well as causing irregular "infeasibility" solution returns is the `sqrt()` function. This function is excluded by utilizing the equivalent mathematical operation of taking the power to a half ($x^{\frac{1}{2}}$).

An example of a function that is excluded is the `norm()` function, which is only able to calculate the norm of a vector/matrix at a single time instance. Seeing as the optimal control problem is posed over the entire time series the norm of each vector is always calculated as $(x_1^2 + x_2^2 + x_3^2)^{\frac{1}{2}}$.

Approximations

One approximation used is the spherical Earth. In reality the Earth is not a perfect sphere but an oblate spheroid. However, in these calculations the Earth has been approximated to a sphere with radius of 6375.750 km (which is the radius of the real Earth at AAU).

8.3 Results

Two primary optimization calculation runs has been made: one where the objective is to minimize the initial mass (minimum propellant to orbit), and one where the objective is to maximize the final mass (excess propellant to orbit). The "minimum propellant to orbit" problem formulation and equations are the ones stated earlier in the previous section (section 8.2.2).
The problem formulation and accompanying equations of the "spare propellant to orbit" problem are almost identical and any deviations of the original formulation are outlined in section 8.3.2.

8.3.1 Minimum Propellant to Orbit

Objective Function Result

The objective function of the minimum propellant to orbit optimization was calculated to be 4710.6 kg which in turn means that the minimum amount of propellant needed for orbital injection is 4560.6 kg. By using the rocket equation it is possible to calculate an approximate matching velocity increment, Δv, for such a mass expulsion. However, since the optimization uses a height dependent specific impulse, a lower and an upper bound can be calculated for the final Δv:

$$\Delta v_{lower} = \mathrm{g}_0 I_{sp}(\text{sea}) \ln\left(\frac{4710.6}{150}\right) = 8.726\,[\text{km/s}] \tag{8.29}$$

$$\Delta v_{upper} = \mathrm{g}_0 I_{sp}(\text{vac}) \ln\left(\frac{4710.6}{150}\right) = 9.560\,[\text{km/s}] \tag{8.30}$$

Using these these results it is possible to calculate the total disturbance loss in the form of lost Δv for both the lower and upper bound case:

$$\Delta v_{lower,loss} = \Delta v_{lower} - (||\mathbf{v}(t_f)|| - ||\mathbf{v}(t_0)||) = 1.243\,[\text{km/s}] \tag{8.31}$$

$$\Delta v_{upper,loss} = \Delta v_{upper} - (||\mathbf{v}(t_f)|| - ||\mathbf{v}(t_0)||) = 2.077\,[\text{km/s}] \tag{8.32}$$

These losses are primarily attributed to the gravity and drag disturbances, as well as the constraints of dynamic pressure, maximum acceleration, and lower bounded throttle level of 2.5%. The lower bounded throttle level will force the LV into a scenario where it will not be able to coast at any point in the ascent, and hence some of the energy put into the climb to orbit will have to be taken out by performing a "breaking" manoeuvre in the end. That is; reducing the vertical climb velocity to 0 at the time t_f.

Height and Speed Development

On figure 8.4 and 8.5 the height and speed development of the LV during the ascent to orbit is shown.

It is clear from the speed development that several characterisitic phases are present within the ascent to orbit. These phases will be discussed in the following sections.

Dynamic Pressure and Acceleration Developments

On figure 8.6 the dynamic pressure of the LV during the ascent is shown. As seen from the plot the dynamic pressure increases rapidly in the beginning of the ascent and hits the upper bound

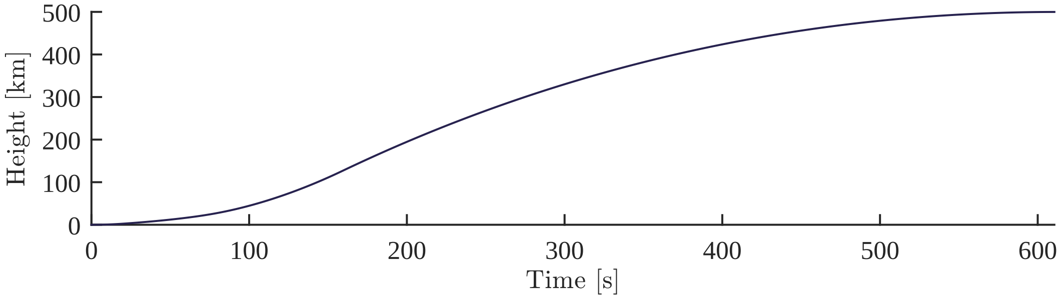


Figure 8.4: Height of the LV above the surface of the Earth with respect to time during the ascent.

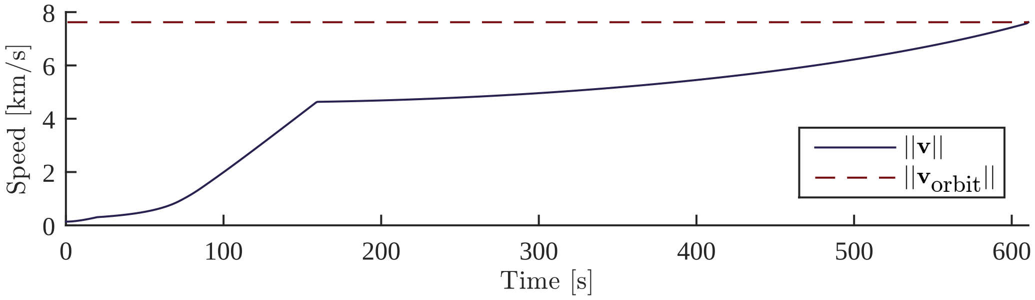


Figure 8.5: Speed of the LV with respect to time during the ascent.

at 20 seconds, which corresponds to approximately 2.5 km. Incidently the velocity of the LV at this point is just below mach 1, so the inherent reduction in acceleration is considered preferable in further reducing mechanical stress as the sound barrier is passed.

The LV proceeds to travel at the q_{max} bound for 52 seconds at which point the density of the atmosphere has reduced significantly. The dynamic pressure proceeds to decline as the LV exits the atmosphere.

The amount of time spent at q_{max} can give some rise to concern as the point of q_{max} is usually a critical point in an ascent trajectory and is often considered a milestone along the different ascent objectives.

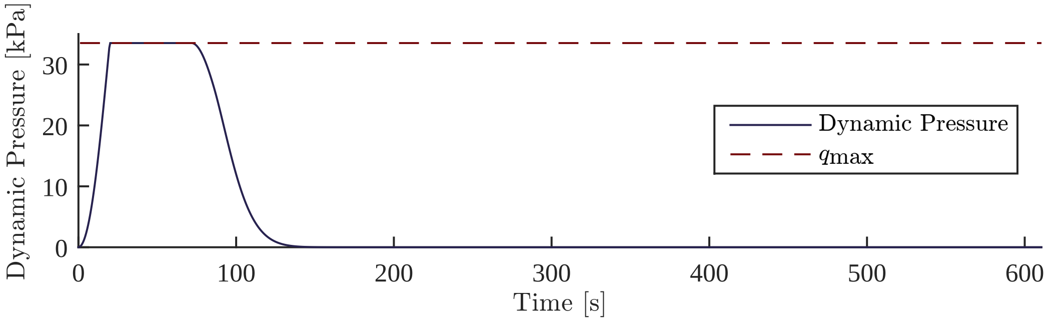


Figure 8.6: Dynamic pressure of the LV with respect to time during the ascent.

Figure 8.7 shows the acceleration development of the LV during the ascent. The development clearly shows the reduction in acceleration experienced by the LV when maximum dynamic pressure is reached at the 20 second mark. During the period of maximum dynamic pressure the acceleration of the LV is, however, still increasing do to the loss of mass. Maximum acceleration load is reached at 86 seconds, and the LV spends approximately 73 seconds at maximum acceleration.

After this phase the LV enters a longer dormant phase where minimum thrust is applied. This is the point where the LV would prefer to coast. Finally at t_f the LV performs a last impulsive thrust manoeuvre to ensure orbital injection.

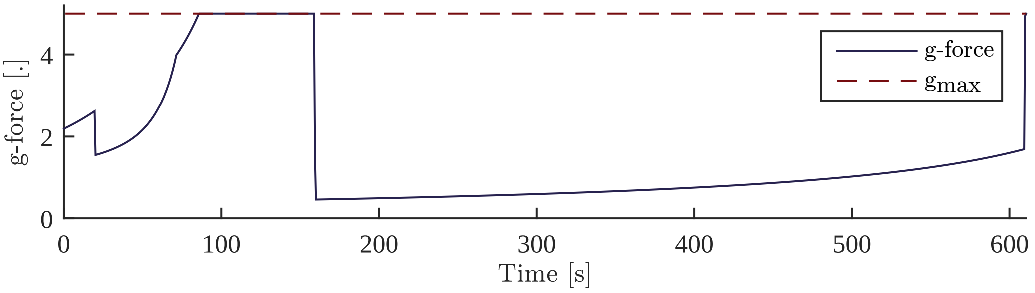


Figure 8.7: Acceleration of the LV with respect to time during the ascent.

Maximum Thrust and Throttle Profile

One of the secondary objectives of the optimization problem was to determine the maximum thrust needed at sea level. The maximum thrust needed was found to be:

$$T_{max} = m(t_0)g_0 p = 101.231\,[\text{kN}] \tag{8.33}$$

Another secondary objective was to calculate the accompanying optimum throttle profile. This profile can be seen on figure 8.8. As seen two primary regions has been marked as blue and red respectively. These regions are the phases of the ascent where the LV is constrained by its path constraints.

As such the first region, the blue, is the phase of the ascent where the LV has reached the maximum allowable dynamic pressure, q_{max}, which inherently results in a reduction in throttle level. This reduction in throttle slowly ramps up to 100% again as the vehicle gains altitude and the atmosphere becomes less dense.

The second region, the red, is the phase of the ascent where the LV has reached the maximum allowable acceleration load. This regions decline in throttle level is characterised by the accompanying reduction in mass.

From the red region forward the minimum thrust level is applied in order for the LV to attempt coasting, followed be the final impulsive thrust manoeuvre to orbital injection.

8.3.2 Spare Propellant to Orbit

In the "spare propellant to orbit" scenario the optimization will calculate how much excess propellant is carried into orbit if a given $m(t_0)$ is chosen. As such the objective function is changed to:

$$maximize\ m(t_f) \quad = \quad minimize\ -m(t_f) \tag{8.34}$$

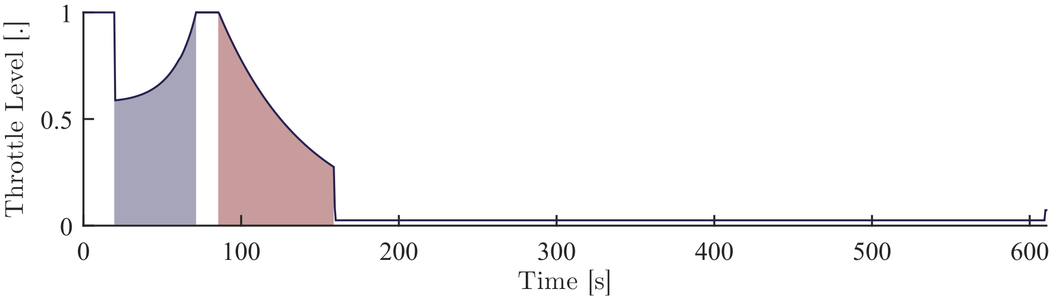


Figure 8.8: LV throttle profile during the ascent. The blue region is the phase where the LV has reached q_{max}, and the region marked with red is the phase where the LV has reached g_{max}.

with boundary conditions

$$m(t_0) = 8000\,[\text{kg}] \tag{8.35}$$

$$m(t_f) = m(t_f) \tag{8.36}$$

Here the initial mass has been set to 8000 kg, compared to the optimal 4710.6 kg from the previous calculations. This gives an overhead of over 3000 kg to be used for disturbance regulation and attitude control action (seeing as attitude control action is free in these problem formulations).

Another change in this problem formulation is that the maximum thrust T_{max} is set to be equal to the previously calculated T_{max} of 101.231 [kN]. This will cause the LV to have slower initial acceleration, and will in turn mean more propellant expenditure. However, this will also cause the point of q_{max} to be delayed and the duration of time spent at q_{max} to be reduced. As such the propellant loss accompanied with reduced thrust is deemed necessary to ensure a better safety margin with respect to dynamic pressure.

Objective Function Result

The objective function of the "spare propellant to orbit" optimization was calculated to be 243.3 kg which in turn means that the amount of spare propellant carried into orbit is 93.3 kg. It should be noted that this number is the amount of propellant carried all the way into orbit, if the optimum path was followed, while attitude control action is free. Furthermore, while 93.3 kg is the excess propellant at t_f available for control action, the amount of propellant available for attitude control action throughout the ascent will be higher as the final "payload" mass will in turn be lighter.
The resulting trajectory from the solution of the problem can be seen on figure 8.9.

Using the rocket equation it is again possible to calculate approximate matching upper and lower bound Δv's for such a mass expulsion:

$$\Delta v_{lower} = \text{g}_0 I_{sp}(\text{sea}) \ln\left(\frac{8000}{243.3}\right) = 8.842\,[\text{km/s}] \tag{8.37}$$

$$\Delta v_{upper} = \text{g}_0 I_{sp}(\text{vac}) \ln\left(\frac{8000}{243.3}\right) = 9.687\,[\text{km/s}] \tag{8.38}$$

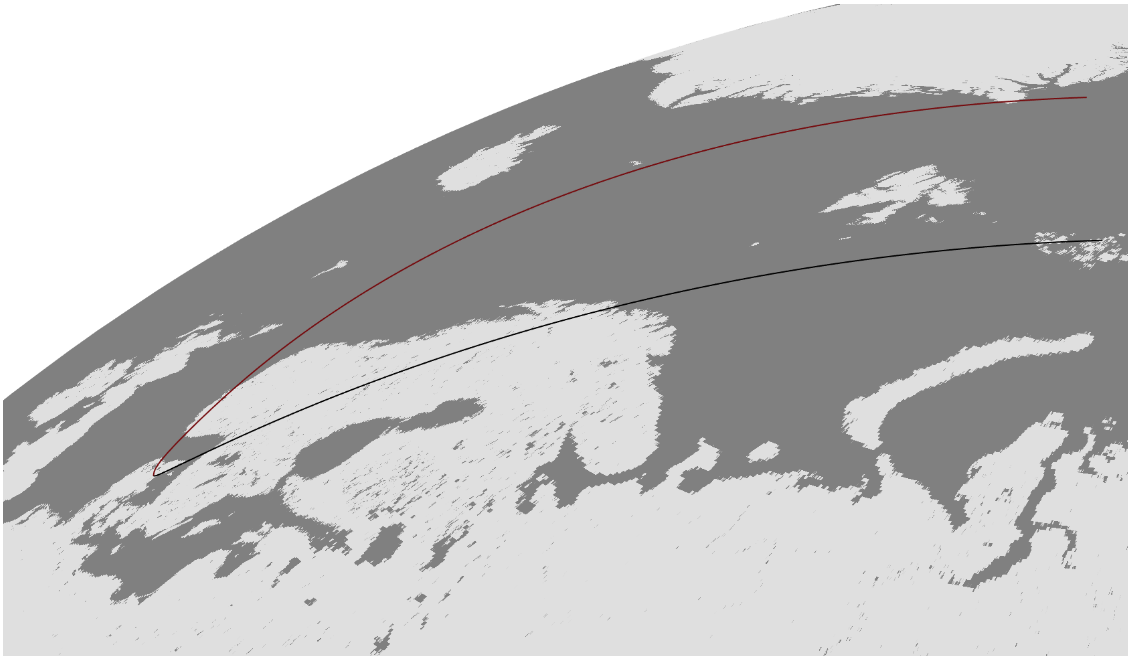


Figure 8.9: LV trajectory in the ECI frame, where the red line is the trajectory and the black line is the ground track.

Using these these results it is possible to calculate the total disturbance loss in the form of lost Δv for both the lower and upper bound case:

$$\Delta v_{lower,loss} = \Delta v_{lower} - (||\mathbf{v}(t_f)|| - ||\mathbf{v}(t_0)||) = 1.357\,[\text{km/s}] \tag{8.39}$$

$$\Delta v_{upper,loss} = \Delta v_{upper} - (||\mathbf{v}(t_f)|| - ||\mathbf{v}(t_0)||) = 2.203\,[\text{km/s}] \tag{8.40}$$

Comparing these losses to the ones calculated in "minimum propellant to orbit" in section 8.3.1 it is seen that the "spare propellant to orbit" run only has an increased loss of between 0.1142 to 0.1254 km/s more. This increased loss is primarily attributed to the previously mentioned static T_{max}, increased aerodynamic normal surface area (seeing as the vehicle is heavier which means it is also higher), and increased gravity loss due to increased ascent time (from a previous t_f of 610 seconds to a new t_f of 712 seconds).

Height and Speed Development

On figure 8.10 and 8.11 the respective height and speed developments for the "spare propellant to orbit" calculations can be seen.

By comparing the speed development of the "minimum propellant to orbit" run on figure 8.11 with that of the "spare propellant to orbit" on figure 8.11, it is seen that the latter has a more smooth speed development in the earlier phase of the ascent, which is attributed to the reduced T_{max} capability of the LV in the latter run.

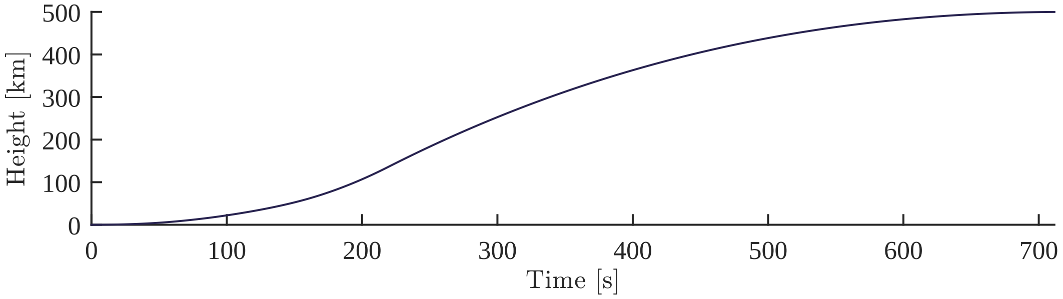


Figure 8.10: Height of the LV above the surface of the Earth with respect to time during the ascent.

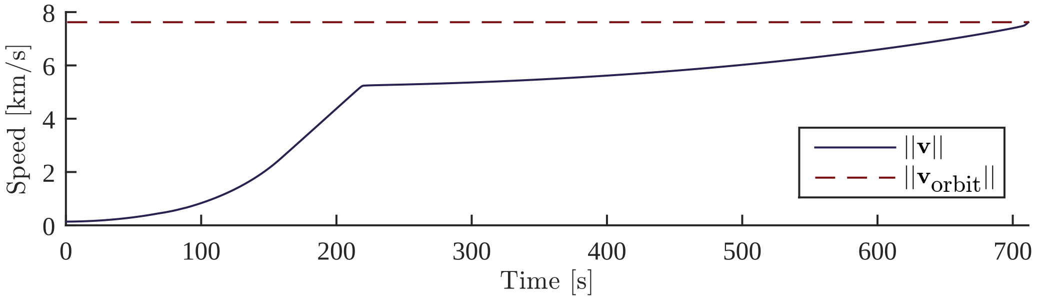


Figure 8.11: Speed of the LV with respect to time during the ascent.

Dynamic Pressure and Acceleration Developments

On figure 8.12 the dynamic pressure of the LV during the ascent is shown. As seen the from the plot the dynamic pressure has a slower ramp up compared to that of the "minimum propellant to orbit" run. In this iteration the time where q_{max} is reached is at approximately 68 seconds. Furthermore, the time spent at q_{max} is also significantly reduced (to a duration of 16 seconds) which is deemed preferable in compared to the other result.

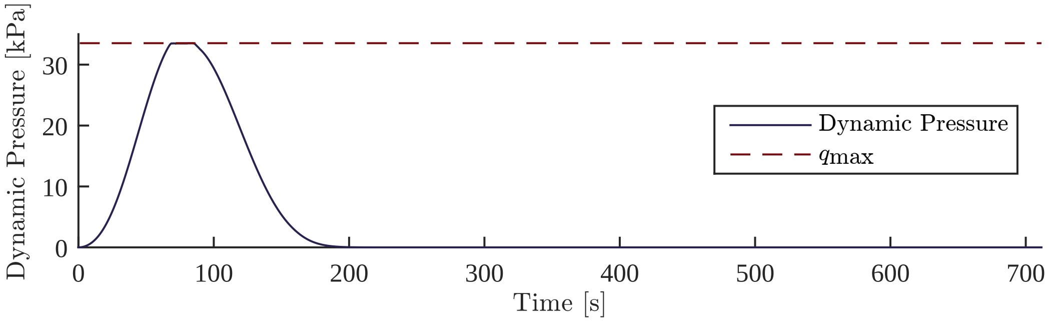


Figure 8.12: Dynamic pressure of the LV with respect to time during the ascent.

Figure 8.7 shows the acceleration development of the LV during the ascent. Compared to the "minimum propellant to orbit" acceleration profile on figure 8.7 the overall development has not changed significantly, other than a reduced dent at the maximum dynamic pressure phase. Only a reduced time spent at g_{max} can be seen. A period which lasts approximately 50 seconds.

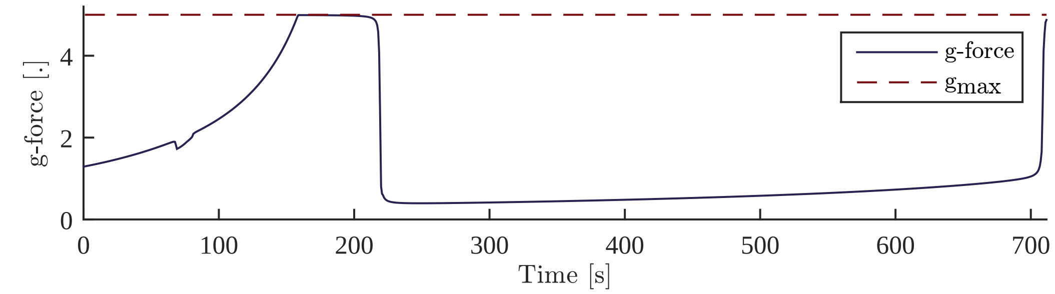


Figure 8.13: Acceleration of the LV with respect to time during the ascent.

Throttle Profile

The accompanying throttle profile for the "spare propellant to orbit" calculations can be seen on figure 8.8. Due to the changed dynamic pressure and acceleration load developments, the throttle profile has inherently changed. However, the overall behaviour remains roughly equal to that of figure 8.8, with a reduced period of decreased throttle at the phase of maximum dynamic pressure.

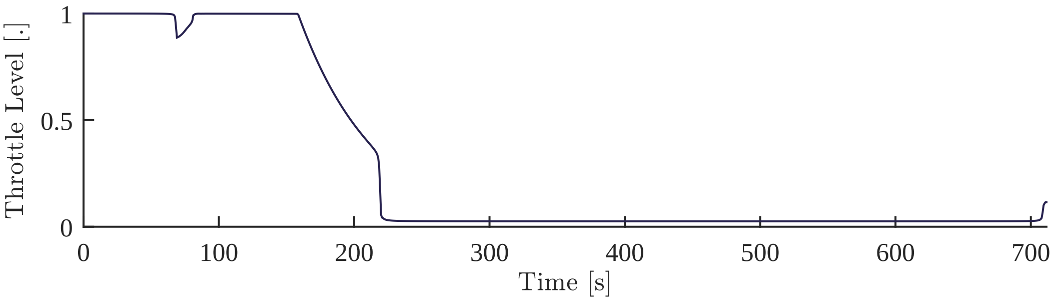


Figure 8.14: LV throttle profile during the ascent.

While the throttle profile in theory utilizes the entire span of 2.5% to 100%, there is in fact an almost instantaneous drop from 28% to 2.5% at the 218 second mark. Including the final control move at 710 second this results in two effective throttle ranges of [2.5%;11.5%] $\bigcup$ [28%;100%].

8.4 Subconclusion

In this chapter two different ascent profiles were presented; the Direct Ascent profile and the Hohmann Transfer Ascent profile. Of these two ascent profiles the DA approach was deemed the most suitable for the mission of AAURocket.
Furthermore, two optimization problem formulations was presented.
The "Minimum propellant to orbit":

$$
\begin{aligned}
&\min\ m(t_0)\\
&\text{subject to}\\
&\qquad \dot{\mathbf{r}} = \mathbf{v}\\
&\qquad \dot{\mathbf{v}} = \frac{\eta\,\mathrm{T}_{\max}}{m}\mathbf{1}_{\mathrm{A}} - \mathbf{g}(\mathbf{r}) - \frac{\mathrm{D}_A(||\mathbf{r}||,\mathbf{v}_r)}{m}\mathbf{1}_{\mathrm{A}} - \frac{\mathrm{D}_N(||\mathbf{r}||,\mathbf{v}_r)}{m}\mathbf{1}_{\mathrm{N}}\\
&\qquad \dot{m} = -\frac{\eta\,T_{max}}{\mathrm{g}_0 I_{sp}(||\mathbf{r}||)}\\
&\quad \text{(path:)}\\
&\qquad \mathrm{R}_{\mathrm{E}} \leq ||\mathbf{r}||\\
&\qquad 150 \leq m\\
&\qquad 0 \leq \theta \leq \pi\\
&\qquad 0 \leq \phi < 2\pi\\
&\qquad 0.025 \leq \eta \leq 1\\
&\qquad a = \frac{\eta\,T_{max}}{m} \leq a_{\max} = 5\ [\mathrm{g}_0]\\
&\qquad q = \frac{1}{2}\mathbf{v}^2\rho_{air}(||\mathbf{r}||) \leq q_{\max} = 33\ [\mathrm{kPa}]\\
&\text{(boundary:)}\\
&\qquad \mathbf{r}(t_0) = \mathbf{r}_{\mathrm{AAU}}\\
&\qquad \mathbf{v}(t_0) = \mathbf{1}_{\mathbf{r}_{\mathrm{AAU}}} + \mathbf{v}_{\mathrm{coriolis}}\\
&\qquad \mathbf{1}_{\mathrm{A}}(t_0) = \mathbf{1}_{\mathbf{r}_{\mathrm{AAU}}}\\
&\qquad m(t_0) = m(t_0)\\
&\qquad m(t_f) = 150\\
&\qquad ||\mathbf{r}(t_f)|| = \mathrm{R}_{\mathrm{E}} + ||\mathbf{r}_{\mathrm{orbit}}||\\
&\qquad ||\mathbf{v}(t_f)|| = ||\mathbf{v}_{\mathrm{orbit}}||\\
&\qquad \mathbf{r}(t_f)\cdot\mathbf{v}(t_f) = 0\\
&\qquad \mathbf{1}_{^i\mathbf{Z}}\cdot\mathbf{h}(t_f) = (\mathrm{R}_{\mathrm{E}} + ||\mathbf{r}_{\mathrm{orbit}}||)\,||\mathbf{v}_{\mathrm{orbit}}||\,\cos(i)\\
&\qquad \mathbf{r}_x(t_f) \leq \mathbf{r}_x(t_0)
\end{aligned}
$$

And the "Spare propellant to orbit" (with only the differences listed):

$$
\begin{aligned}
&\max\ m(t_f) \quad = \quad \min\ -m(t_f)\\
&\text{subject to}\\
&\qquad m(t_0) = 8000\,[\mathrm{kg}]\\
&\qquad m(t_f) = m(t_f)
\end{aligned}
$$

One to estimate the optimal trajectory of an LV of the nature of AAURocket using a minimum amount of propellant to reach orbital injection, and another to estimate the optimal trajectory of the same LV bringing excess propellant into orbit.
Table 8.1 presents a quick overview of several key data points of each calculated solution.
Of the two ascent trajectories the latter, "spare propellant to orbit", will be the primary ascent profile considered for use on AAURocket.

Parameter	**Minimum Propellant to Orbit**	**Spare Propellant to Orbit**	**Unit**
Ascent Time	611	712	[s]
$m(t_0)$	4710.7	8000	[kg]
$m(t_f)$	150	243.3	[kg]
T_{max}	101.231	101.231	[kN]
Δv	[8.726 ; 9.560]	[8.850 ; 9.687]	[km/s]
Δv_{loss}	[1.243 ; 2.077]	[1.357 ; 2.203]	[km/s]
q_{max} start	20	68	[s]
Time at q_{max}	52	16	[s]
g_{max} start	86	159	[s]
Time at g_{max}	73	50	[s]

Table 8.1: Key data points comparison of the two optimization runs; "Minimum Propellant to Orbit" and "Spare Propellant to Orbit".

Chapter 9

Position and Attitude Determination

In this chapter the general linear Kalman filter will be presented, followed by the closely related extended Kalman filter. Additionally an extended Kalman filter will be developed and tuned, and finally nominally tested in three distinct tests. Finally the PAD will be evaluated with respect to the system requirements.

9.1 Position and Attitude Determination Algorithm

The purpose of the Position and Attitude Determination (PAD) system is to determine the system states of the LV. The primary states considered for estimation are:

$$\mathbf{x} = \begin{bmatrix} {}^{\mathrm{i}}\mathbf{r}^T & {}^{\mathrm{i}}\mathbf{v}^T & {}^{\mathrm{i}}_{\mathrm{c}}\mathbf{q}^T & {}^{\mathrm{c}}\boldsymbol{\omega}^T \end{bmatrix}^T \tag{9.1}$$

Furthermore, several physical parameters are estimated alongside the states which include: mass of the oxidizer, mass of the fuel, height of the LV, moment of inertia, and thrust magnitude of each thruster.
In order to achieve accurate estimates the sensors chosen in section 5.2 are utilized in conjunction with a Kalman filter which in turn will reduce noise magnitude and increase sensor accuracy through sensor fusion.

9.1.1 Kalman Filter

Source: [Grewal 01]

The Kalman filter is a stochastic recursive filter which utilizes known system dynamics and sensor measurements to calculate a better estimate of the states of the system than any sensor could on its own. These calculations can be split into a two step process:

Predict step In the predict step the previous a posteriori state estimate $\hat{\mathbf{x}}^+_{k-1}$ is propagated through the dynamical equations of the Kalman filter to estimate the current system a priori states $\hat{\mathbf{x}}^-_k$.

Update step The update step calculates the current a posteriori state estimate $\hat{\mathbf{x}}^+_k$ by correcting the calculated $\hat{\mathbf{x}}^-_k$ with a weighted correction factor of the measurements.

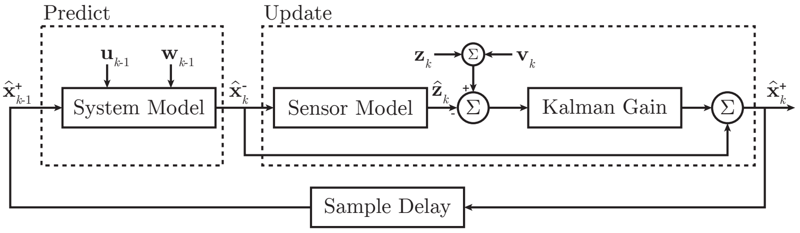


Figure 9.1: A diagram showing the general flow of a kalman filter.

This two step process is also illustrated on figure 9.1, where first the previous a posteriori $\hat{\mathbf{x}}^+_{k-1}$ is propagated through a system model, where a known input $\mathbf{u}_{k-1}$, and an unknown process noise $\mathbf{w}_{k-1}$ is affecting the system. This propagation results in the current a priori states $\hat{\mathbf{x}}^-_k$. These a priori states are used to calculate the current sensor estimate $\hat{\mathbf{z}}_k$ which is then subtracted from the actual measurements $\mathbf{z}_k$ which are affected by measurement noise. The difference between the two measurements are finally used to calculate the a posteriori states $\hat{\mathbf{x}}^+_{k-1}$ by correcting the a priori states $\hat{\mathbf{x}}^-_k$ through the Kalman gain.

For the propagation of the states and measurements, the Kalman filter uses linear system matrices. Since the dynamic system equations for AAURocket are non-linear, the closely related extended version of the Kalman filter is utilized instead.

9.1.2 Extended Kalman Filter

The Extended Kalman Filter (EKF) is, as the name implies, an extension of the original linear Kalman filter. The extension of the filter is in essence the utilization of first order linearization through Jacobians of the system and measurement matrices. The extended Kalman filter considers the non-linear dynamical system equations as:

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) + \mathbf{w}_{k-1} \tag{9.2}$$

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k \tag{9.3}$$

$$\tag{9.4}$$

where

$\mathbf{x}$	is the state vector
$\mathbf{u}$	is the input vector
$\mathbf{w}$	is the process noise vector
$\mathbf{z}$	is the measurement vector
$\mathbf{v}$	is the measurement noise vector

Both noise vectors $\mathbf{w}_{k-1}$ and $\mathbf{v}_k$ are assumed white Gaussian independent processes, with $\underline{\mathbf{Q}}_{k-1}$ and $\underline{\mathbf{R}}_k$ as the respective process and measurement covariance matrices.

Predict Step

The initial propagation in the EKF is performed by using the non-linear system equation on the previous a posteriori state vector:

$$\hat{\mathbf{x}}^-_k = f(\mathbf{x}^+_{k-1}, \mathbf{u}_{k-1}) \tag{9.5}$$

Following the state propagation is the linearization of the system equations at the current point of the a priori state estimate:

$$\underline{\boldsymbol{\Phi}}_k = \frac{\partial f(\mathbf{x}, \mathbf{u})}{\partial \hat{\mathbf{x}}_k^-} \tag{9.6}$$

where

$\underline{\mathbf{P}}_k^-$ is the a priori error covariance matrix
$\underline{\mathbf{P}}_{k-1}^+$ is the previous a posteriori error covariance matrix

It is clear that if the system is very non-linear then this linearization may not be sufficient, and a larger error will be observed through the prediction iterations.
An a priori error covariance matrix is calculated from the linearized system matrix, the a posteriori error covariance matrix, and the process covariance matrix:

$$\underline{\mathbf{P}}_k^- = \underline{\boldsymbol{\Phi}}_k \underline{\mathbf{P}}_{k-1}^+ \underline{\boldsymbol{\Phi}}_k^T + \underline{\mathbf{Q}}_{k-1} \tag{9.7}$$

where

$\underline{\boldsymbol{\Phi}}_k$ is the linearized system matrix at the point $\hat{\mathbf{x}}_k^-$

Update Step

In the update step a sensor estimate is calculated by propagating the a priori state estimate through the non-linear measurement relations of the system:

$$\hat{\mathbf{z}}_k = h(\mathbf{x}_k^-) \tag{9.8}$$

Following the measurement estimation is the linearization of the measurement relations:

$$\underline{\mathbf{H}}_k = \frac{\partial h(\mathbf{x})}{\partial \hat{\mathbf{x}}_k^-} \tag{9.9}$$

where

$\underline{\mathbf{H}}_k$ is the linearized measurement relation matrix at the point $\hat{\mathbf{x}}_k^-$

As with the linearization of the system equations, there is a possibility that the linearization is insufficient if the measurement relations are too non-linear.
Following the linearization, the Kalman gain is calculated using the a priori error covariance matrix, the linearized measurement relation matrix, and the measurement covariance matrix:

$$\underline{\mathbf{K}}_k = \underline{\mathbf{P}}_k^- \underline{\mathbf{H}}_k^T (\underline{\mathbf{H}}_k \underline{\mathbf{P}}_k^- \underline{\mathbf{H}}_k^T)^{-1} \tag{9.10}$$

where

$\underline{\mathbf{K}}_k$ is the Kalman gain matrix

The Kalman gain is then used to correct the a priori state estimate based on the measurement prediction error, as follows:

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \underline{\mathbf{K}}_k (\mathbf{z}_k - \hat{\mathbf{z}}_k) \tag{9.11}$$

Finally the a posteriori error covariance matrix is calculated based on the current Kalman gain matrix, the linearized measurement relations matrix, and the a priori error covariance matrix:

$$\underline{\mathbf{P}}_k^+ = (\underline{\mathbf{I}} - \underline{\mathbf{K}}_k \underline{\mathbf{H}}_k)\underline{\mathbf{P}}_k^- \tag{9.12}$$

From the equations it is clear that the matrices $\underline{\mathbf{Q}}$ and $\underline{\mathbf{R}}$ are tuning matrices. They are generally diagonalized unless there are sensor or state cross couplings, and the initial diagonal values for $\underline{\mathbf{R}}$ is normally taken to be each sensors noise variance. While both matrices are denoted as time dependent they can also be set to static values used across the entire sampling period.

9.2 Implementation of the Position and Attitude Determination System

The PAD system has been implemented into the Simulink simulation environment in the onboard computer block. The structure of the PAD system is divided into two primary subsystems: an miscellaneous estimation block and an EKF block.

9.2.1 Miscellaneous Estimation Block

The purpose of the miscellaneous estimation block is to isolate and estimate parameters outside of the extended Kalman filter algorithm. While a lot of these parameters could be estimated inside the EKF block, they have been isolated in order to reduce the overall complexity of the EKF algorithm.

As mentioned in section 9.1 the external estimates include: height measurement conversion, GPS measurement rotation, magnetic field estimation, oxidizer mass estimation, fuel mass estimation, moment of inertia estimation, and thrust vectors estimation.

Height Measurement

The height measurement algorithm functions as a simple unbounded counter which keeps track of the tachometer counts and converts the number of counts into a height measurement in meters. Currently the height measurement is only a single counter, however, in reality where several DC motors are probably utilized this algorithm could utilize several tachometer counts and average the values to get a more precise estimate, in case of missed tachometer counts.

GPS measurement rotation

The GPS measurement block outputs the position of the LV in the ECEF frame. Seeing as the position state to be estimated is given in the ECI frame, the GPS measurement is rotated from ECEF to the ECI frame with the estimated quaternion ${}^{\mathrm{e}}_{\mathrm{i}}\mathbf{q}$. This quaternion is estimated by using the current time in the Julian Date format and calculating the time since vernial equinox. As such the number of days (full Earth rotations) since the last alignment of ECI and ECEF is known, and the current misalignment can be calculated. The GPS measurement rotation is then calculated as:

$${}^{\mathrm{i}}\mathbf{x}_{\mathrm{meas}} = {}^{\mathrm{e}}_{\mathrm{i}}\mathbf{q}^* \otimes {}^{\mathrm{e}}\mathbf{x}_{\mathrm{meas}} = {}^{\mathrm{i}}_{\mathrm{e}}\mathbf{q} \otimes {}^{\mathrm{e}}\mathbf{x}_{\mathrm{meas}} \tag{9.13}$$

Magnetic Field Estimation

The estimation of the magnetic field is calculated in the same way as the truth model calculates the magnetic field strength at a given position. However, the estimation of the magnetic field utilizes the position estimate from the EKF ${}^{\mathrm{i}}\mathbf{x}$, and the previously estimated quaternion ${}^{\mathrm{i}}_{\mathrm{e}}\mathbf{q}$. As such the estimated magnetic field will be dependent on the accuracy of the EKF algorithm in order to minimize estimation error.

Oxidizer and Fuel Mass Estimation

The oxidizer and fuel mass estimations are calculated in the same way as the truth model, as described in section 7.1.1. This means two limited integrators takes the mass flow inputs and returns the amount of remaining fuel and oxidizer.

Moment of Inertia Estimation

The moment of inertia estimation is calculated as described in section 6.4.

Thrust Vectors Estimation

The overall calculation method of the thrust vectors are identical to those of actual actuator block described in section 7.2.2. However, this block takes the known inputs and the atmospheric pressure measurement as inputs. As such the resulting magnitude of each thrust vector will be dependent on the accuracy of the barometer.

9.2.2 Extended Kalman Filter Block

The EKF block utilizes inputs both sensors as well as the previous estimation block. In total 14 inputs are used in the EKF which include:

- GPS measurement
- Accelerometer measurement
- Magnetometer measurement
- Gyroscope measurement
- Magnetic field estimate
- LV height estimate
- Moment of inertia estimate
- Oxidizer mass estimate
- Fuel mass estimate
- Forward thrust input vector
- 4 vernier thrust input vectors

Though the EKF is implemented in Simulink, it is mostly a collection of Matlab m files that are interfaced to the environment through a "matlab-fcn" block.
When the EKF is started for the first time, an initial run procedure is executed. This procedure sets various persistent constants to be used inside the estimation algorithm. Furthermore, the initial states of AAURocket are passed to the $\mathbf{x}^{+}_{k-1}$ vector. There are no mathematical procedure to estimate the initial states of the system, as such it is assumed that all initial state parameters

for a given launch is known.
In this report an initial state vector used for a launch is as follows:

$$\mathbf{x}_0 = \begin{bmatrix}\mathbf{r}_{\text{AAU}}^T & \mathbf{v}_{\text{coriolis}}^T & {}^{\text{i}}_{\text{r}}\mathbf{q}_0^T & \mathbf{0}_{3\times 1}\end{bmatrix} \tag{9.14}$$

After the initial run the algorithm will enter the recursive loop of the EKF.

Predict Step

From the dynamical system presented in chapter 6 the following system equations has been utilized in the predict step of the EKF:

$$\begin{aligned}
{}^{\text{i}}\dot{\mathbf{r}}_k &= {}^{\text{i}}\mathbf{v}_{k-1} \qquad (9.15)\\
{}^{\text{i}}\dot{\mathbf{v}}_k &= {}^{\text{i}}_{\text{c}}\mathbf{q}_{k-1} \otimes \left[\frac{{}^{\text{c}}\mathbf{T}_{k-1}}{m_{k-1}} - \frac{(\frac{1}{2}\mathbf{C}_D \circledast \mathbf{A}_{k-1} \circledast |{}^{\text{c}}\mathbf{v}_{k-1}| \circledast {}^{\text{c}}\mathbf{v}_{k-1}\rho_{air}(||{}^{\text{i}}\mathbf{r}_{k-1}||)}{m_{k-1}}\right] - \frac{\mu_E}{||{}^{\text{i}}\mathbf{r}_{k-1}||^3}{}^{\text{i}}\mathbf{r}_{k-1}\\
{}^{\text{i}}_{\text{c}}\dot{\mathbf{q}}_k &= \frac{1}{2}[{}^{\text{c}}\boldsymbol{\omega}_{k-1}]_{\otimes}{}^{\text{i}}_{\text{c}}\mathbf{q}_{k-1}\\
{}^{\text{c}}\dot{\boldsymbol{\omega}}_k &= {}^{\text{c}}\underline{\mathbf{J}}_{k-1}^{-1}\left({}^{\text{c}}\boldsymbol{\omega}_{k-1} \times {}^{\text{c}}\underline{\mathbf{J}}_{k-1}{}^{\text{c}}\boldsymbol{\omega}_{k-1}\right)
\end{aligned}$$

with

$${}^{\text{c}}\mathbf{v}_{k-1} = {}^{\text{i}}_{\text{c}}\mathbf{q}_{k-1}^* \otimes {}^{\text{i}}\mathbf{v}_{k-1} = {}^{\text{c}}_{\text{i}}\mathbf{q}_{k-1} \otimes {}^{\text{i}}\mathbf{v}_{k-1} \tag{9.16}$$

A 4^{th} order Runge-Kutta numerical integration midpoint method is used when propagating the previous a posteriori states through the non-linear system equations, in order to obtain better accuracy than in compared to the forward Euler numerical integration method. The 4^{th} order Runge-Kutta method utilized is calculated as follows:

$$\begin{aligned}
&\text{for } n = 1:4 \qquad (9.17)\\
&\quad k_1 = f(\mathbf{x}_{k-1}^+, \mathbf{u}_{k-1})\\
&\quad k_2 = f(\mathbf{x}_{k-1}^+ + \frac{T_s}{8}k_1, \mathbf{u}_{k-1})\\
&\quad k_3 = f(\mathbf{x}_{k-1}^+ + \frac{T_s}{8}k_2, \mathbf{u}_{k-1})\\
&\quad k_4 = f(\mathbf{x}_{k-1}^+ + \frac{T_s}{4}k_3, \mathbf{u}_{k-1})\\
&\quad \mathbf{x}_{k-1}^+ = \mathbf{x}_{k-1}^+ + \frac{T_s}{24}(k_1 + k_2 + k_3 + k_4)
\end{aligned}$$

It should be noted that for each increment of n, the quaternion state is renormalized, as the numerical precision of Matlab is deteriorating the quaternion magnitude.

From the non-linear system equations, an analytical Jacobian matrix $\underline{\boldsymbol{\Phi}}_k$ can be calculated as:

$$\underline{\boldsymbol{\Phi}}_k = \underline{\mathbf{1}}_{13\times 13} + T_s \begin{bmatrix}
\underline{\mathbf{0}}_{3\times 3} & \underline{\mathbf{1}}_{3\times 3} & \underline{\mathbf{0}}_{3\times 4} & \underline{\mathbf{0}}_{3\times 3}\\
\frac{\partial\,{}^{\text{i}}\mathbf{v}_k}{\partial\,{}^{\text{i}}\mathbf{r}} & \frac{\partial\,{}^{\text{i}}\mathbf{v}_k}{\partial\,{}^{\text{i}}\mathbf{v}} & \underline{\mathbf{0}}_{3\times 4} & \underline{\mathbf{0}}_{3\times 3}\\
\underline{\mathbf{0}}_{4\times 3} & \underline{\mathbf{0}}_{4\times 3} & \frac{\partial\,{}^{\text{i}}_{\text{c}}\mathbf{q}_k}{\partial\,{}^{\text{i}}_{\text{c}}\mathbf{q}} & \frac{\partial\,{}^{\text{i}}_{\text{c}}\mathbf{q}_k}{\partial\,{}^{\text{i}}\boldsymbol{\omega}}\\
\underline{\mathbf{0}}_{3\times 3} & \underline{\mathbf{0}}_{3\times 3} & \underline{\mathbf{0}}_{3\times 4} & \frac{\partial\,{}^{\text{c}}\boldsymbol{\omega}_k}{\partial\,{}^{\text{c}}\boldsymbol{\omega}}
\end{bmatrix} \tag{9.18}$$

Note that even though the quaternion state is present in the ${}^{\mathrm{i}}\mathbf{v}_k$ equation, the derivative $\frac{\partial {}^{\mathrm{i}}\mathbf{v}_k}{\partial {}^{\mathrm{i}}_{\mathrm{c}}\mathbf{q}}$ is deliberately set to 0. This is due to the quaternion estimation not having any influence on the velocity equations, and are only used to rotate the dynamical equations. If the derivative is included, the quaternion is no longer able to be estimated correctly.

Update Step

As mentioned the following sensors are employed for use in the EKF algorithm: GPS, accelerometer, magnetometer, and gyroscope. The measurement relations for each of these sensors are as follows:

$$\hat{\mathbf{z}}_k = \begin{bmatrix} {}^{\mathrm{i}}\mathbf{GPS}_{est} \\ {}^{\mathrm{c}}\mathbf{Acc}_{est} \\ {}^{\mathrm{c}}\mathbf{Mag}_{est} \\ {}^{\mathrm{c}}\mathbf{Gyro}_{est} \end{bmatrix} = \begin{bmatrix} {}^{\mathrm{i}}\mathbf{r}_k^- \\ \frac{{}^{\mathrm{c}}\mathbf{T}_k^-}{m_k} - \frac{(\frac{1}{2}\,\mathbf{C}_D \circledast \mathbf{A}_k \circledast |{}^{\mathrm{c}}\mathbf{v}_k^-| \circledast {}^{\mathrm{c}}\mathbf{v}_k^- \rho_{air}(||{}^{\mathrm{i}}\mathbf{r}_k^-||)}{m_k} + {}^{\mathrm{c}}\boldsymbol{\omega}_k^- \times ({}^{\mathrm{c}}\boldsymbol{\omega}_k^- \times \mathbf{d}_{imu,k}) \\ {}^{\mathrm{i}}_{\mathrm{c}}\mathbf{q}_k^{-*} \otimes {}^{\mathrm{i}}\mathbf{B}_{est,k} \\ {}^{\mathrm{c}}\boldsymbol{\omega}_k^- \end{bmatrix} \tag{9.19}$$

where

${}^{\mathrm{c}}\boldsymbol{\omega}_k^- \times ({}^{\mathrm{c}}\boldsymbol{\omega}_k^- \times \mathbf{d}_{imu,k})$	is the accelerometer displacement acceleration caused by rotations [m/s^2]
${}^{\mathrm{i}}\mathbf{B}_{est,k}$	is the estimated magnetic field vector [T]

The GPS relation is a direct state sensor, and as such does not include any extra terms to its relation.

The accelerometer is modelled as the general system acceleration equation with an extra acceleration term in the form of rotational contribution in conjunction with displacement. Furthermore, the gravity term is removed.

The magnetometer is used to determine the orientation of the LV through the measurement relation involving the quaternion ${}^{\mathrm{i}}_{\mathrm{c}}\mathbf{q}$. By using the estimated magnetic field in the ECI frame and rotating that into the CRF, it is possible to match that to the actual measured magnetic field. Any discrepancy between the two vectors is then corrected through the quaternion, and as such the attitude of the LV is estimated. It should be noted that this attitude estimation is of reduced order, as a single vector can only determine two out of the three attitude degrees of freedom. The rotation about the axis of the vector itself is free, and normally an extra attitude sensor is utilized to obtain full observability. For this EKF the gyroscope is instead used to handle the last degree of freedom. Using a perfectly ideal gyroscope this will in theory be sufficient to estimate the attitude at any point. However, most gyroscopes do not perform well at lower angular velocities, and as such a drift is expected to occur as time progresses forward away from t_0. Even with this concern in mind, due to the small time horizon of the ascent it is deemed plausible to fly with only a single attitude reference sensor and a gyroscope.

Finally the gyroscope sensor relation is a direct state sensor and therefore no extra terms are included in its relation.

In turn the Jacobian of the measurement relations are calculated as:

$$\underline{\mathbf{H}}_k = \begin{bmatrix} \underline{\mathbf{1}}_{3\times3} & \underline{\mathbf{0}}_{3\times3} & \underline{\mathbf{0}}_{3\times4} & \underline{\mathbf{0}}_{3\times3} \\ \frac{\partial {}^{\mathrm{c}}\mathbf{Mag}_{est}}{\partial {}^{\mathrm{c}}\mathbf{r}} & \frac{\partial {}^{\mathrm{c}}\mathbf{Mag}_{est}}{\partial {}^{\mathrm{c}}\mathbf{v}} & \underline{\mathbf{0}}_{3\times4} & \underline{\mathbf{0}}_{3\times3} \\ \underline{\mathbf{0}}_{4\times3} & \underline{\mathbf{0}}_{3\times4} & \frac{\partial {}^{\mathrm{c}}\mathbf{Gyro}_{est}}{\partial {}^{\mathrm{i}}_{\mathrm{c}}\mathbf{q}} & \underline{\mathbf{0}}_{3\times3} \\ \underline{\mathbf{0}}_{3\times3} & \underline{\mathbf{0}}_{3\times3} & \underline{\mathbf{0}}_{3\times4} & \underline{\mathbf{1}}_{3\times3} \end{bmatrix} \tag{9.20}$$

The measurement relation derivatives $\frac{\partial {}^{\mathrm{c}}\mathbf{Mag}_{est}}{\partial {}^{\mathrm{i}}_{\mathrm{c}}\mathbf{q}}$ and $\frac{\partial {}^{\mathrm{c}}\mathbf{Mag}_{est}}{\partial {}^{\mathrm{c}}\boldsymbol{\omega}}$ are both set equal to zero, since both these derivatives will corrupt the estimation of both the attitude states.

Tuning of the EKF

The tuning of the EKF refers to the initial values set for the covariance matrices $\underline{\mathbf{Q}}$,$\underline{\mathbf{R}}$, and $\underline{\mathbf{P}}$. While both the state and sensor covariance matrices can be time dependent, they are kept constant.
Each matrix has been tuned through iterative testing and the resulting values are (where diag() is the matrix with the vector components as its diagonal):

$$\underline{\mathbf{Q}} = diag([(1\ 1\ 1)10^{-5} \quad (1\ 1\ 1)10^{-5} \quad (1\ 1\ 1\ 1)10^{-12} \quad (1\ 1\ 1)10^{-8}]) \tag{9.21}$$

$$\underline{\mathbf{R}} = diag([(1\ 1\ 1)10^{-1} \quad (2.6\ 2.6\ 2.6)10^{-3} \quad (5.8\ 5.8\ 5.8)10^{-3} \quad (1.97\ 1.97\ 1.97)10^{-5}]) \tag{9.22}$$

$$\underline{\mathbf{P}}_0 = diag([0\ 0\ 0 \quad 0\ 0\ 0 \quad 0\ 0\ 0\ 0 \quad 0\ 0\ 0]) \tag{9.23}$$

During the tuning the value for $\underline{\mathbf{R}}$ is held constant at the variances of the sensors as presented in section 5.2, after which $\underline{\mathbf{Q}}$ is varied until the most satisfactory result is obtained.
The initial value for $\underline{\mathbf{P}}$ is set to zero as it is assumed that all initial states are known, and as such the estimates at the first time instance are completely correct.

9.3 Verification Test of the Position and Attitude Determination System

In order to verify the PAD system three different case studies are considered: an orbit test, an atmospheric re-entry and crash test, and a launch test from the surface of the Earth.
The purpose of the orbit test is to test performance of the PAD void of atmospheric disturbances, which are considered the most severe and non-linear of the system equations.
The purpose of the re-entry test is to stress test the PAD systems ability to properly estimate states during a sudden rise in disturbance forces, caused by the increasing density of the atmosphere as the LV approaches the surface of the Earth.
Finally, the purpose of the ascent test is to assess the PAD systems performance during nominal operations.
All tests are performed with a time horizon of 800 seconds, which is longer than the estimated 710 seconds ascent time calculated in section 8.3.2.

All tests are subject to the estimation subsystem requirements from section 3. The final acceptance evaluation of the PAD system will be made in section 9.3.4.

9.3.1 Estimation in Orbit

The orbit verification test will be performed by initiating the LV with the following initial conditions:

$$^{\mathrm{i}}\mathbf{r} = [||^{\mathrm{i}}\mathbf{r}_o|| \quad 0 \quad 0]^T \tag{9.24}$$

$$^{\mathrm{i}}\mathbf{v} = [0 \quad ||^{\mathrm{i}}\mathbf{v}_o|| \quad 0]^T \tag{9.25}$$

$$^{\mathrm{i}}_{\mathrm{r}}\mathbf{q} = [0 \quad 0 \quad 0 \quad 1]^T \tag{9.26}$$

$$^{\mathrm{r}}\boldsymbol{\omega} = [0 \quad 0 \quad 0]^T \tag{9.27}$$

With these initial conditions the LV is in a 500 km circular orbit above the equator of the Earth. The angular velocity has been set equal to 0 in order to test the drift of the quaternion state. As no rotation is affecting the gyroscope this is considered the worst case scenario for correct attitude

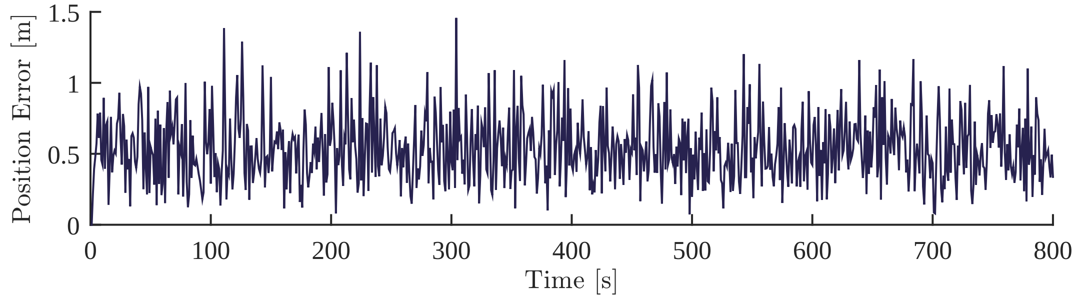


Figure 9.2: Positional estimation error magnitude ($||^{\mathrm{i}}\mathbf{r} - ^{\mathrm{i}}\hat{\mathbf{r}}||$) for the orbit estimation test.

estimation.

Figure 9.2 shows the magnitude of the positional error ($||^{\mathrm{i}}\mathbf{r} - ^{\mathrm{i}}\hat{\mathbf{r}}||$) during the orbit test. The largest peak in the position error signal was calculated to be 1.458 m.

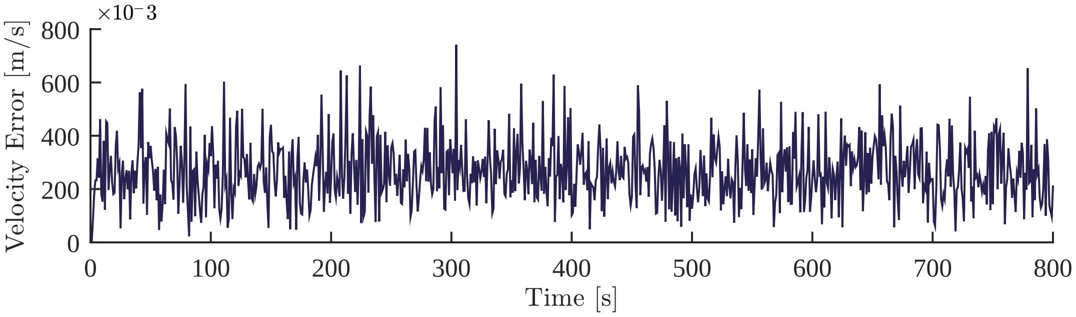


Figure 9.3: Velocity estimation error magnitude ($||^{\mathrm{i}}\mathbf{v} - ^{\mathrm{i}}\hat{\mathbf{v}}||$) for the orbit estimation test.

Figure 9.3 shows the magnitude of the velocity error during the orbit test. The largest peak in the velocity error signal was calculated to be 0.7415 m/s.

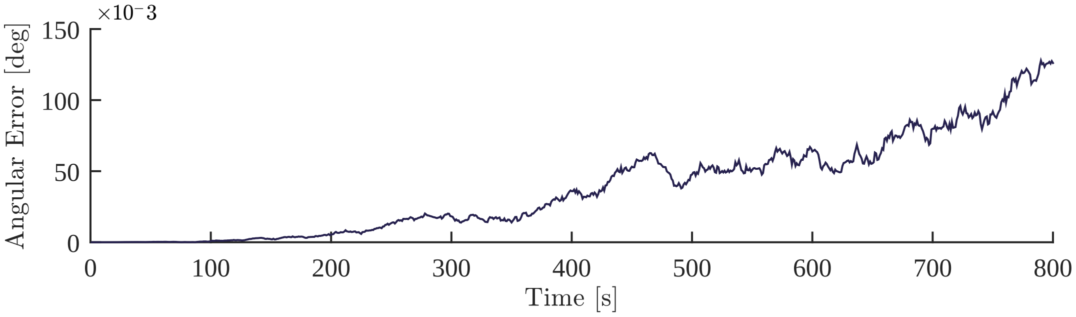


Figure 9.4: Angular estimation error magnitude for the orbit estimation test.

Figure 9.4 shows the angular error during the orbit test. The largest peak in the error signal was calculated to be 0.128°.
Note that one component of the quaternion was estimated to much greater precision than the rest. This is expected to be the magnetometer that enables the EKF to estimate one axis correctly while the two other perpendicular axes, have higher errors as these are only supported by the derivative measurement of the gyroscope.

9.3.2 Estimation during Re-entry

The re-entry verification test will be performed by initiating the LV with the following initial conditions:

$$^{\mathrm{i}}\mathbf{r} = [\mathrm{R_E} + 50 \quad 0 \quad 0]^T \tag{9.28}$$

$$^{\mathrm{i}}\mathbf{v} = [0 \quad ||^{\mathrm{i}}\mathbf{v}_o|| \quad 0]^T \tag{9.29}$$

$$^{\mathrm{i}}_{\mathrm{r}}\mathbf{q} = [0 \quad 0 \quad 0 \quad 1]^T \tag{9.30}$$

$$^{\mathrm{r}}\boldsymbol{\omega} = [0 \quad 0 \quad 0]^T \tag{9.31}$$

These initial condition has the LV only 50 km above the Earth, with a relatively high velocity. While this velocity is enough to have the LV burn up in reality, this will not be the case in the simulation, and the result will instead be an rapidly increasing retarding force induced by the atmospheric drag force.

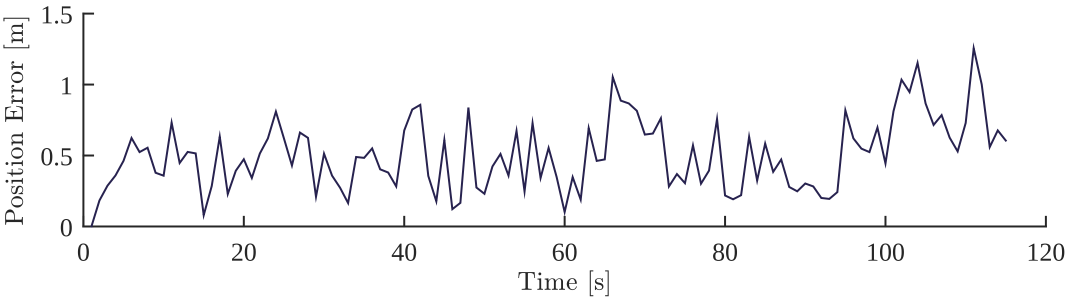


Figure 9.5: Positional estimation error magnitude ($||^{\mathrm{i}}\mathbf{r} - {}^{\mathrm{i}}\hat{\mathbf{r}}||$) for the re-entry estimation test..

Figure 9.5 shows the magnitude of the positional error ($||^{\mathrm{i}}\mathbf{r} - {}^{\mathrm{i}}\hat{\mathbf{r}}||$) during the orbit test. The largest peak in the position error signal was calculated to be 1.256 m.
While the initial simulation time was set to 800 seconds, it did in fact only take 116 seconds for the LV to crash into the Earth.

Figure 9.6 shows the magnitude of the velocity error ($||^{\mathrm{i}}\mathbf{v} - {}^{\mathrm{i}}\hat{\mathbf{v}}||$) during the orbit test. The largest peak in the velocity error signal was calculated to be 0.924 m/s.

Figure 9.7 shows the angular error during the orbit test. The largest peak in the error signal was calculated to be 0.0013 °.
Again it is noted that one component of the quaternion was estimated to much greater precision than the rest.

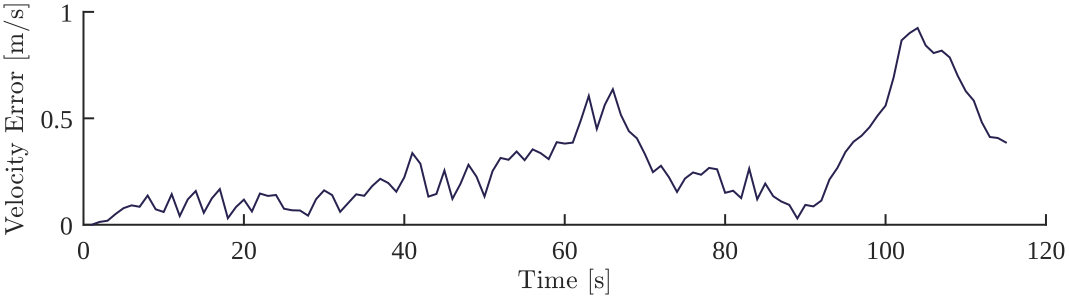


Figure 9.6: Velocity estimation error magnitude ($||{}^{\mathrm{i}}\mathbf{v} - {}^{\mathrm{i}}\hat{\mathbf{v}}||$) for the re-entry estimation test.

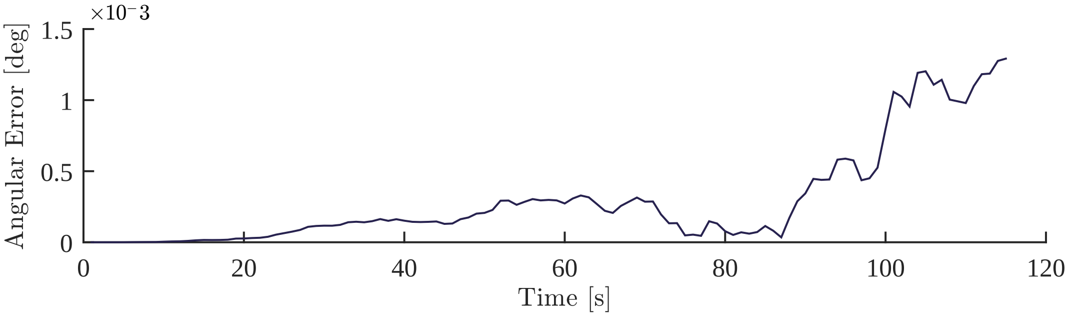


Figure 9.7: Angular estimation error magnitude for the re-entry estimation test.

9.3.3 Estimation during Ascent

The ascent verification test will be performed by initiating the LV with the following initial conditions:

$$
\begin{align}
{}^{\mathrm{i}}\mathbf{r} &= {}^{\mathrm{i}}\mathbf{r}_{\mathrm{AAU}}^{T} \tag{9.32}\\
{}^{\mathrm{i}}\mathbf{v} &= [0 \quad 0 \quad 0]^{T} \tag{9.33}\\
{}^{\mathrm{i}}_{\mathrm{r}}\mathbf{q} &= [-0.876 \quad -0.0767 \quad -0.474 \quad -0.0415]^{T} \tag{9.34}\\
{}^{\mathrm{r}}\boldsymbol{\omega} &= [0 \quad 0 \quad 0]^{T} \tag{9.35}
\end{align}
$$

These initial condition has the LV starting at AAU, pointing straight up and at rest. The throttle input to the thrusters was set as a function decreasing in proportion to the mass of the rocket. As such the LV reached a maximum altitude of 1600 km and a velocity of 5 km/s.

Figure 9.8 shows the magnitude of the positional error ($||{}^{\mathrm{i}}\mathbf{r} - {}^{\mathrm{i}}\hat{\mathbf{r}}||$) during the ascent test. The largest peak in the position error signal was calculated to be 1.074 m.

Figure 9.9 shows the magnitude of the velocity error ($||{}^{\mathrm{i}}\mathbf{v} - {}^{\mathrm{i}}\hat{\mathbf{v}}||$) during the ascent test. The largest peak in the velocity error signal was calculated to be 0.222 m/s.

Figure 9.10 shows the angular error during the orbit test. The largest peak in the error signal was calculated to be 0.0013 °.

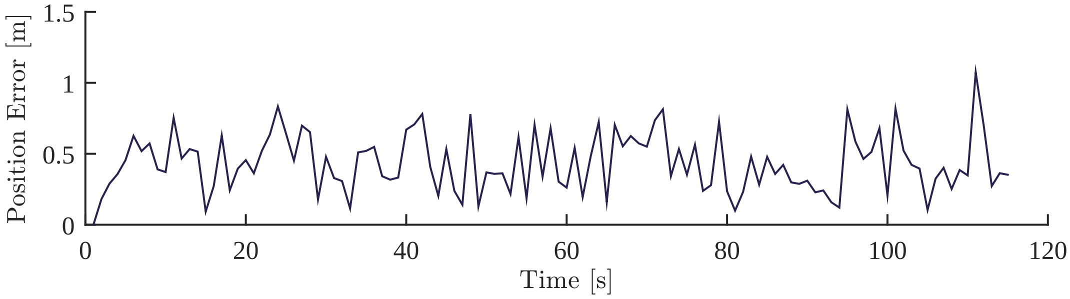


Figure 9.8: Positional estimation error magnitude ($||^{\mathrm{i}}\mathbf{r}-^{\mathrm{i}}\hat{\mathbf{r}}||$) for the ascent estimation test..

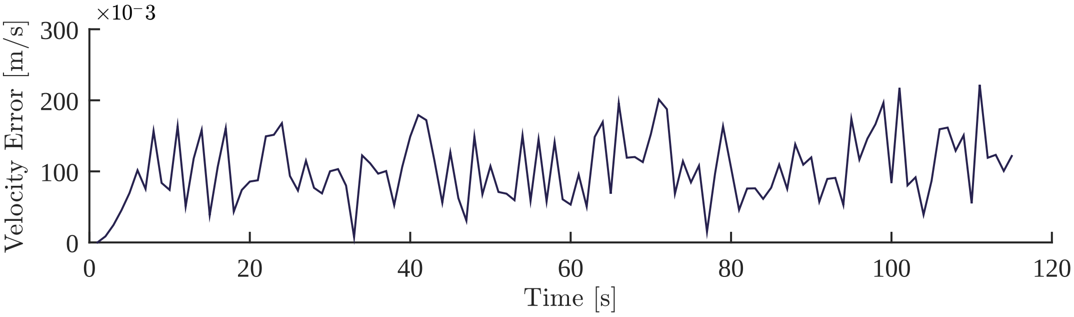


Figure 9.9: Velocity estimation error magnitude ($||^{\mathrm{i}}\mathbf{v}-^{\mathrm{i}}\hat{\mathbf{v}}||$) for the ascent estimation test.

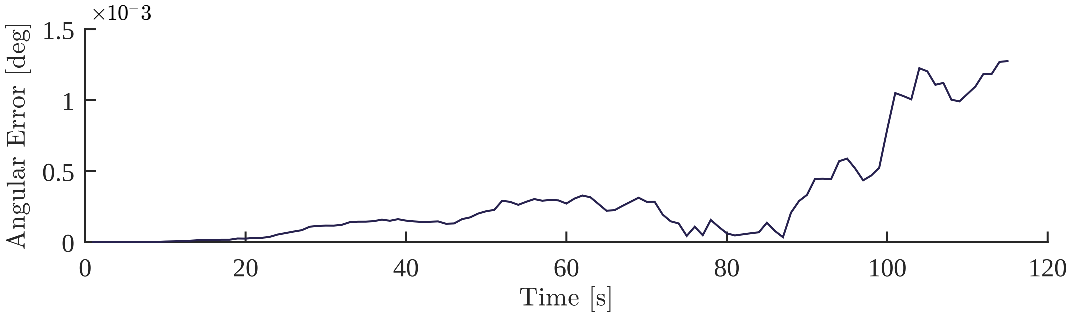


Figure 9.10: Angular estimation error magnitude for the ascent estimation test.

Of the three nominal estimation tests only the ascent test had a thrust input enabled.

9.3.4 Acceptance Verification

The PAD system is one of the primary subsystems designed for the onboard computer as part of the guidance system that shall ensure that AAURocket reaches a sustainable orbit as per the mission requirements in chapter 3.
Furthermore, specific requirements were put forth for the PAD system to achieve. These requirements, as specified in section 3.2, are restated in table 9.1.

Requirement	Parameter	Value	Unit
SS1.R1	Position determination accuracy	<0.7	[km]
SS1.R2	Velocity determination accuracy	<2.1	[m/s]

Table 9.1: PAD Subsystem requirements.

Conversely the worst case errors for each nominal test are listed in table 9.2.

Parameter	Orbit Test	Re-entry Test	Ascent Test	
Position Error	1.458	1.256	1.074	[m]
Velocity Error	0.7415	0.924	0.222	[m/s]
Angular Error	0.128	0.0013	0.0013	[deg]

Table 9.2: The worst state error observed in each nominal test setup.

It is clear that the PAD subsystem is fulfilling both requirement SS1.R1 and SS1.R2, with a significant margin in both cases. However, it is noted that this is the nominal PAD system performance tested in three distinct cases. As such the nominal performance of the PAD system is deemed satisfactory.
A goal for further development is to subject the PAD system to rigorous perturbation testing through Monte-Carlo simulations in order to fully verify the PAD system of AAURocket.

9.4 Subconclusion

In this chapter the general linear Kalman filter and the extended Kalman filter was presented. Of these two algorithms the EKF was chosen as the estimation algorithm of choice for AAURocket. The general design equations and parameters utilized for the PAD EKF was calculated and designed as follows:

$$
\begin{aligned}
{}^{\mathrm{i}}\dot{\mathbf{r}}_k &= {}^{\mathrm{i}}\mathbf{v}_{k-1} \\
{}^{\mathrm{i}}\dot{\mathbf{v}}_k &= {}^{\mathrm{i}}_{\mathrm{c}}\mathbf{q}_{k-1} \otimes \left[\frac{{}^{\mathrm{c}}\mathbf{T}_{k-1}}{m_{k-1}} - \frac{(\frac{1}{2}\mathbf{C}_D \circledast \mathbf{A}_{k-1} \circledast |{}^{\mathrm{c}}\mathbf{v}_{k-1}| \circledast {}^{\mathrm{c}}\mathbf{v}_{k-1}\rho_{air}(||{}^{\mathrm{i}}\mathbf{r}_{k-1}||)}{m_{k-1}} \right] - \frac{\mu_E}{||{}^{\mathrm{i}}\mathbf{r}_{k-1}||^3}{}^{\mathrm{i}}\mathbf{r}_{k-1} \\
{}^{\mathrm{i}}_{\mathrm{c}}\dot{\mathbf{q}}_k &= \frac{1}{2}[{}^{\mathrm{c}}\boldsymbol{\omega}_{k-1}]_{\otimes}{}^{\mathrm{i}}_{\mathrm{c}}\mathbf{q}_{k-1} \\
{}^{\mathrm{c}}\dot{\boldsymbol{\omega}}_k &= {}^{\mathrm{c}}\underline{\mathbf{J}}_{k-1}^{-1} \left({}^{\mathrm{c}}\boldsymbol{\omega}_{k-1} \times {}^{\mathrm{c}}\underline{\mathbf{J}}_{k-1}{}^{\mathrm{c}}\boldsymbol{\omega}_{k-1} \right)
\end{aligned}
$$

$$\underline{\mathbf{\Phi}}_k = \underline{\mathbf{1}}_{13\times 13} + T_s \begin{bmatrix} \underline{\mathbf{0}}_{3\times 3} & \underline{\mathbf{1}}_{3\times 3} & \underline{\mathbf{0}}_{3\times 4} & \underline{\mathbf{0}}_{3\times 3} \\ \frac{\partial^{\mathrm{i}}\mathbf{v}_k}{\partial^{\mathrm{i}}\mathbf{r}} & \frac{\partial^{\mathrm{i}}\mathbf{v}_k}{\partial^{\mathrm{i}}\mathbf{v}} & \underline{\mathbf{0}}_{3\times 4} & \underline{\mathbf{0}}_{3\times 3} \\ \underline{\mathbf{0}}_{4\times 3} & \underline{\mathbf{0}}_{4\times 3} & \frac{\partial^{\mathrm{i}}_{\mathrm{c}}\mathbf{q}_k}{\partial^{\mathrm{i}}_{\mathrm{c}}\mathbf{q}} & \frac{\partial^{\mathrm{i}}_{\mathrm{c}}\mathbf{q}_k}{\partial^{\mathrm{i}}\boldsymbol{\omega}} \\ \underline{\mathbf{0}}_{3\times 3} & \underline{\mathbf{0}}_{3\times 3} & \underline{\mathbf{0}}_{3\times 4} & \frac{\partial^{\mathrm{c}}\boldsymbol{\omega}_k}{\partial^{\mathrm{c}}\boldsymbol{\omega}} \end{bmatrix}$$

$$\hat{\mathbf{z}}_k = \begin{bmatrix} {}^{\mathrm{i}}\mathbf{GPS}_{est} \\ {}^{\mathrm{c}}\mathbf{Acc}_{est} \\ {}^{\mathrm{c}}\mathbf{Mag}_{est} \\ {}^{\mathrm{c}}\mathbf{Gyro}_{est} \end{bmatrix} = \begin{bmatrix} {}^{\mathrm{i}}\mathbf{r}_k^- \\ \frac{{}^{\mathrm{c}}\mathbf{T}_k^-}{m_k} - \frac{(\frac{1}{2}\,\mathbf{C}_D \circledast \mathbf{A}_k \circledast |{}^{\mathrm{c}}\mathbf{v}_k^-| \circledast {}^{\mathrm{c}}\mathbf{v}_k^-\,\rho_{air}(||{}^{\mathrm{i}}\mathbf{r}_k^-||)}{m_k} + {}^{\mathrm{c}}\boldsymbol{\omega}_k^- \times ({}^{\mathrm{c}}\boldsymbol{\omega}_k^- \times \mathbf{d}_{imu,k}) \\ {}^{\mathrm{i}}_{\mathrm{c}}\mathbf{q}_k^{-*} \otimes {}^{\mathrm{i}}\mathbf{B}_{est,k} \\ {}^{\mathrm{c}}\boldsymbol{\omega}_k^- \end{bmatrix}$$

$$\underline{\mathbf{H}}_k = \begin{bmatrix} \underline{\mathbf{1}}_{3\times 3} & \underline{\mathbf{0}}_{3\times 3} & \underline{\mathbf{0}}_{3\times 4} & \underline{\mathbf{0}}_{3\times 3} \\ \frac{\partial^{\mathrm{c}}\mathbf{Mag}_{est}}{\partial^{\mathrm{c}}\mathbf{r}} & \frac{\partial^{\mathrm{c}}\mathbf{Mag}_{est}}{\partial^{\mathrm{c}}\mathbf{v}} & \underline{\mathbf{0}}_{3\times 4} & \underline{\mathbf{0}}_{3\times 3} \\ \underline{\mathbf{0}}_{4\times 3} & \underline{\mathbf{0}}_{3\times 4} & \frac{\partial^{\mathrm{c}}\mathbf{Gyro}_{est}}{\partial^{\mathrm{i}}_{\mathrm{c}}\mathbf{q}} & \underline{\mathbf{0}}_{3\times 3} \\ \underline{\mathbf{0}}_{3\times 3} & \underline{\mathbf{0}}_{3\times 3} & \underline{\mathbf{0}}_{3\times 4} & \underline{\mathbf{1}}_{3\times 3} \end{bmatrix}$$

with the tuning matrices $\underline{\mathbf{Q}}$,$\underline{\mathbf{R}}$, and $\underline{\mathbf{P}}$:

$$\begin{aligned} \underline{\mathbf{Q}} &= diag([(1\ 1\ 1)10^{-5} \quad (1\ 1\ 1)10^{-5} \quad (1\ 1\ 1\ 1)10^{-12} \quad (1\ 1\ 1)10^{-8}]) \\ \underline{\mathbf{R}} &= diag([(1\ 1\ 1)10^{-1} \quad (2.6\ 2.6\ 2.6)10^{-3} \quad (5.8\ 5.8\ 5.8)10^{-3} \quad (1.97\ 1.97\ 1.97)10^{-5}]) \\ \underline{\mathbf{P}}_0 &= diag([0\ 0\ 0 \quad 0\ 0\ 0 \quad 0\ 0\ 0\ 0 \quad 0\ 0\ 0]) \end{aligned}$$

Finally the nominal PAD system performance was deemed satisfactory according to the system requirements, while it is recommended that the PAD EKF be further tested, with Monte-Carlo simulations.

Chapter 10

Position and Attitude Control

10.1 Control Strategy

The overall control of AAURocket will be divided into two main phases: an endo-atmospheric phase and an exo-atmospheric phase. The endo-atmospheric phase will stretch from 0 to a height of 100 km after which the exo-atmospheric phase starts, as seen on figure 10.1.
The reason for the division of the trajectory is that the atmosphere and accompanying disturbance can be really heavy for advanced controllers to handle "online". As such it can be advantageous to attempt to follow a predetermined path or control sequence during the atmospheric phase of the ascent. Furthermore, a more general (and possibly computationally lighter) control algorithm can be utilized in the exo-atmospheric phase.

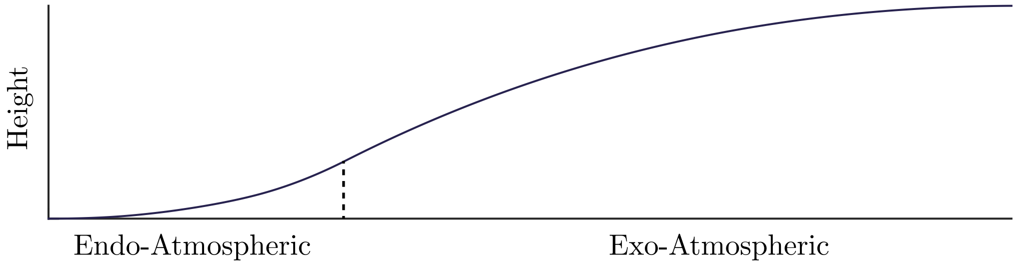


Figure 10.1: The two major phases considered for the control of AAURocket, the endo-atmospheric phase and the exo-atmospheric phase shown on a height plot.

10.2 Waypoint tracking

The waypoint tracking algorithm is designed in an attempt to follow a predetermined trajectory for the endo-atmospheric phase of the ascent. In order to maintain optimum fuel consumption the trajectory calculated in section 8.3.2 is used as the target trajectory to track. This section will present a method for following a 3-dimensional trajectory through waypoint tracking.

10.2.1 Guidance

There exists several methods for waypoint tracking. The simplest idea is by designing a controller that guides the rocket to each waypoint, wp_1, and then switching the reference to the next waypoint, wp_2, whenever the rocket gets within a certain acceptance region of the waypoint wp_1. Another method is called Line of Sight (LoS) guidance. The main idea of using LoS guidance is that, instead of guiding the rocket directly towards the next waypoint, it guides the rocket towards the path between two waypoints. The principle of the LoS guidance is illustrated in figure 10.2. The principle of LoS guidance is solely used to calculate a reference for the direction of movement. In practice this means that the LoS guidance law calculates the direction in which rocket is supposed to travel.

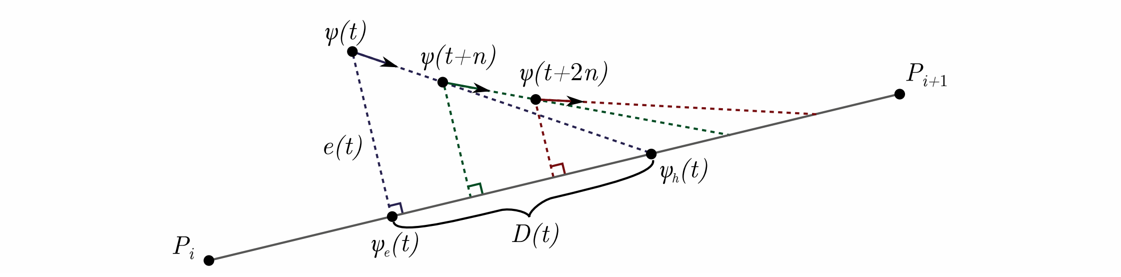


Figure 10.2: Shows the principle of LoS guidance. The rocket aims to change its position, $\psi(t)$, such that it follows the path going from P_i to P_{i+1}. The desired travelling direction of the rocket is towards the point $\psi_h(t)$.

In figure 10.2 there are presented different variables used for calculating the desired movement direction. The variable $e(t)$ is called the cross-track error and is defined as the vector going from the position of the rocket, $\psi(t)$, and being perpendicular to the line going from P_i to P_{i+1}. The lookahead distance, $D(t)$, is a tuning variable that defines how far ahead on the path it is desired to aim. As the cross track error goes towards zero, the direction of movement of the rocket becomes parallel to line between the points P_i and P_{i+1}.
In the case of AAURocket a slightly modified version of the LoS guidance law will be implemented. The incentive for doing this, is that the waypoints for the optimal trajectory does not only contain positions, but also information about the velocities needed for a given point along the trajectory. The waypoint tracking controller for AAURocket will be implemented in such a way that one part of the controller acts to decrease the cross-track error, while the other part of the controller acts to control the velocity to comply with the reference given by the waypoints. Figure 10.3 illustrates how the two different part of the control signals are going to act on the rocket to control the velocity and to decrease the cross-track error.

10.2.2 Waypoint Switching

The switching between waypoints is important for the performance of the waypoint tracking controller. Getting too close to a waypoint before switching to next might cause the cross-error to increase immediately after the switching, resulting in the controller trying to decrease the velocity to remove the error. Furthermore, the region of acceptance for switching waypoint should be designed such that if the rocket misses a waypoint, it will not try to turn around to get within the region of acceptance. Thereby it is chosen to design the waypoint switching such that it switches when the point $\psi_e(t)$ (seen in both figure 10.3 and 10.2) enters the region of acceptance.

10.2.3 The Waypoint Tracking Controller

As mentioned, the waypoint tracking controller is split into two; controlling the velocity and the cross-track. Both controllers have been chosen as linear controllers; a proportional controller for

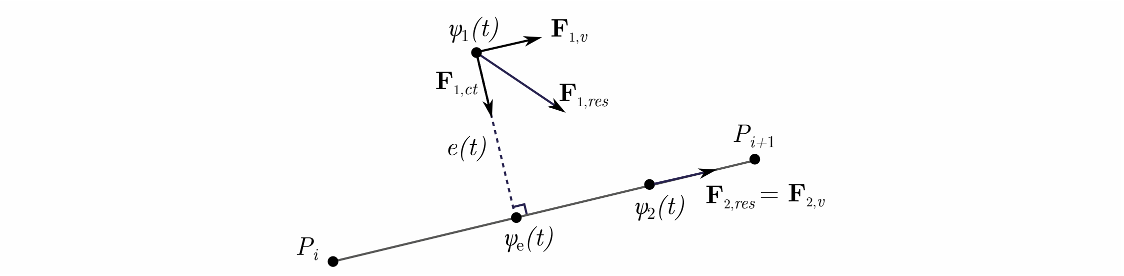


Figure 10.3: Shows the principle for guidance and waypoint tracking for AAURocket. $\mathbf{F}_{k,ct}$ is a force input determined by a control law for making the cross-track error converge towards zero. $\mathbf{F}_{k,v}$ is a force input determined by a control law for making the velocity error converge towards zero. Adding the two force inputs results in a force, $\mathbf{F}_{k,res}$, that is designed to make the rocket follow the specified waypoints and velocities.

the cross-track and proportional-integral controller for the velocity. The block diagram in figure 10.4 shows how the waypoint tracking controller is designed.

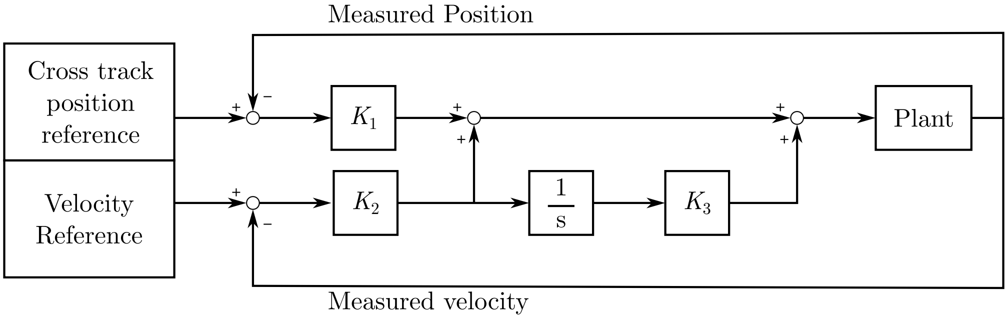


Figure 10.4: Shows the design of the waypoint tracking controller. The cross-track error and the velocity error is calculated and the proportional gains K_1 and K_2 are multiplied on to the errors to calculate the control input. These are added, together with a integration and gain of the

10.2.4 Test of Waypoint Tracking Controller

While the nominal performance range of the waypoint tracking controller is from height 0 to 100 km, the following test will attempt a full launch profile from 0 to 500 km in order to evaluate the controller more thoroughly.

The test of the waypoint tracking controller is performed with state feedback. The test is performed without controlling the attitude of the rocket, and thereby only taking the translational motion into account. The values of K_1, K_2 and K_3 have all been tuned for a rocket with a constant mass of 8000 kg. A dynamic proportional downscaling on the control gains is performed to match the decreasing mass, to ensure the same general performance for the controller. This is done by dividing the control input with the start mass of 8000 kg and multiplying it with the current mass. The control values, K_1, K_2 and K_3, have been tuned as shown in table 10.1. The waypoints have been tuned according to the trajectory calculated in section 8.3.2.

The system used for the test is implemented in a way such that even if the rocket runs out of propellant, it is still able to perform control action, but without going below a mass of 150 kg. This is done to be able to demonstrate the performance of the controller, even though the rocket

Parameter	Value
Initial position	$[3434202.222\ 605537.877\ 5337580.635]^\top$
Initial velocity	$[-23.5466\ 136.6901\ 0.8372]^\top$
K_1	90
K_2	900
K_3	0.9

Table 10.1: Initial states and control gains chosen for the evaluation of the waypoint tracking controller.

should consume all the propellant. Figure 10.5 shows a plot of the altitude of the rocket during the ascent to orbit. After 720 seconds the rocket reaches the target altitude of 500 km.

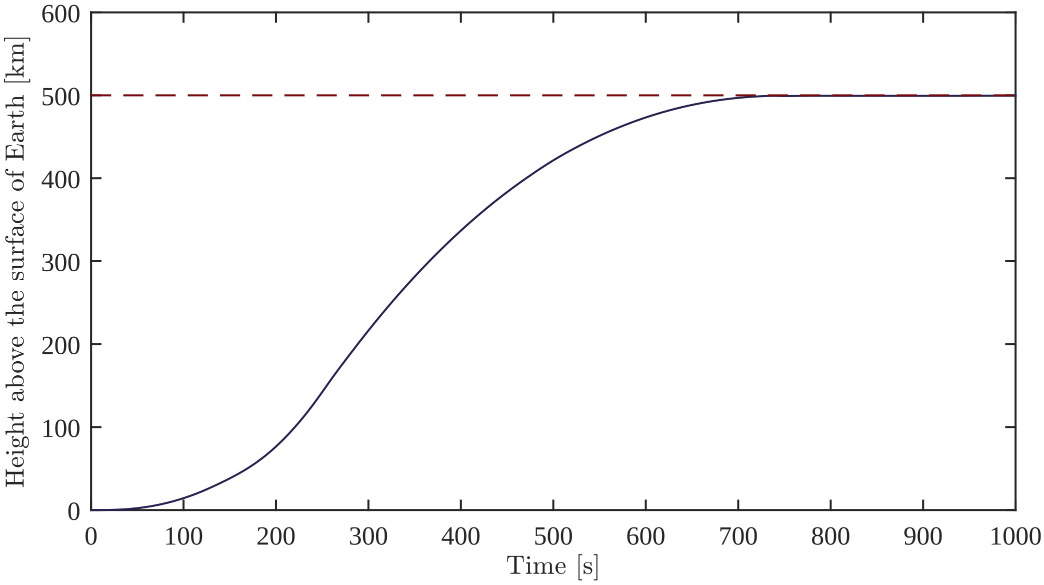


Figure 10.5: Shows the altitude above the surface of earth for the rocket during the ascent. The red dotted line shows the target altitude.

In figure 10.6 a plot of the speed during the ascent is shown. After 740 seconds the rocket obtains the target speed of 7.621 km/s. It should be noticed that the target speed is reached later than the target altitude is reached, which means that some fuel might be wasted on having to increase the orbit speed after the target altitude is obtained.

Figure 10.7 shows the remaining mass of the rocket during the ascent. Here it should be noticed that the mass of the rocket reaches 150 kg after 690 seconds. Looking at the plots of the altitude and the speed, it is seen that after 690 seconds the rocket have obtained an altitude of 495 km at a speed of 7.07 km/s.

10.2.5 Acceptance verification

For the waypoint tracking subsystem to be verified, it has to comply with the requirements given in section 3. These requirements are presented in table 10.2. The test of the waypoint tracking controller shows that with a starting mass of 8000 kg it is not possible to get into the desired orbit. When the rocket runs out of fuel, the rocket is still 5 km from reaching its target altitude and more than 500 m/s away from reaching its target speed. It would be possible to try and increase

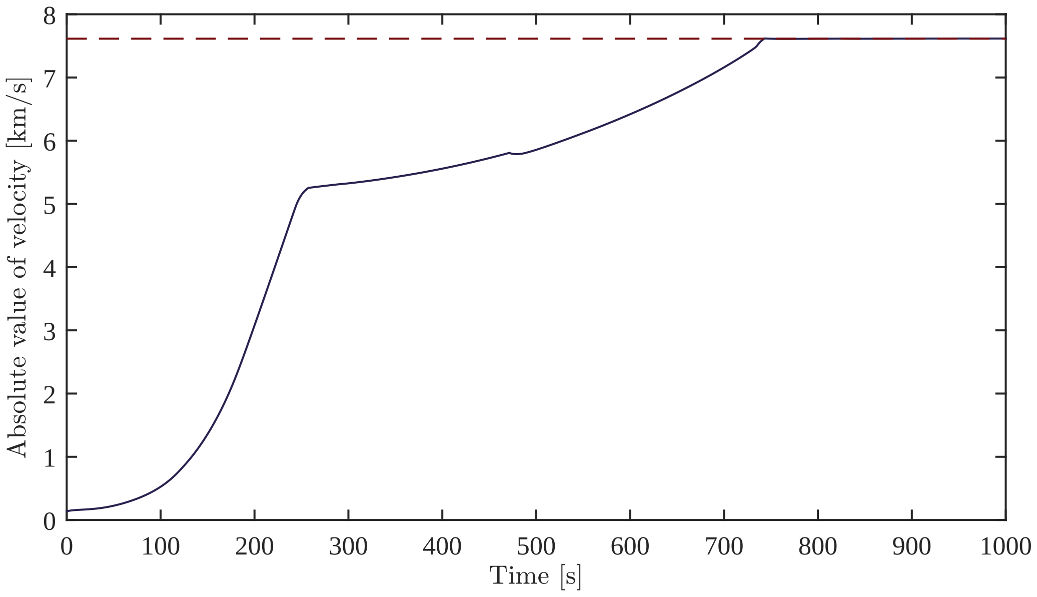


Figure 10.6: Shows the absolute value of the velocity during the ascent. The red dotted shows the target speed.

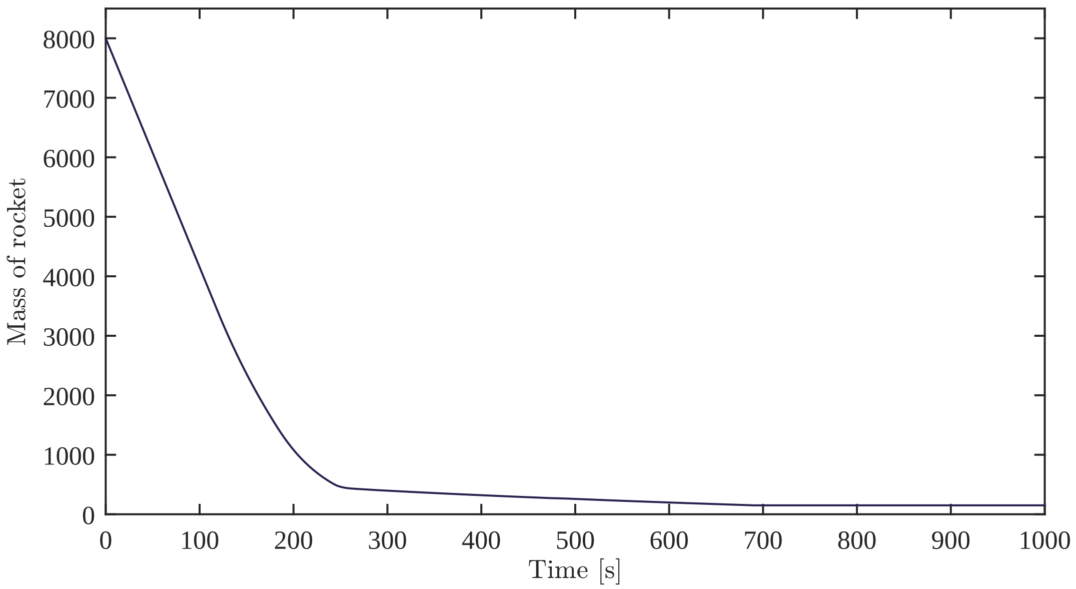


Figure 10.7: Shows the mass of the rocket during the ascent. As seen, the rocket runs out of fuel after 690 seconds.

Requirement	Parameter	Value	Unit
SS2.R1	Position control accuracy	<1.3	[km]
SS2.R2	Velocity control accuracy	<3.9	[m/s]

Table 10.2: Position and velocity control requirements

the initial amount of fuel on the rocket, but according to the solution of the optimization problem for the ascent trajectory, the initial mass of 8000 kg should result in having approximately 90 kg of propellant after getting into orbit. However, if the controller had the desired propellant, the results of the test of the waypoint tracking controller shows that after 740 seconds, the rocket would have reached an altitude of 499.4 km at a speed of 7.615 km/s.
This discrepancy can be caused by a number of factors. One possibility is that the differences between the optimization algorithm and the Simulink environment are greater than the margin set by the 8000 kg initial mass calculation. Another possibility is that the controller is too aggressive and may spend too much excess propellant on control manoeuvres involving "breaking" (thrusting in the opposite direction).

10.3 Sliding Mode Control

10.3.1 Design of sliding mode controller for AAURocket

Sources: [Khalil 02]

The sliding mode controller is meant to control AAURocket such that it gets correctly into a specified circular orbit. The primary region of operation intended for the sliding mode controller is in the exo-atmospheric phase, i.e. from an altitude of 100 km and upward.
The dynamic equations are derived as shown in chapter 6, and to refresh, the model derived translational model is restated in equation 10.1

$$m\mathbf{a} = \mathbf{T} + \mathbf{T}_v - \mathbf{f}_g - \mathbf{f}_D \tag{10.1}$$

For the sliding mode controller, only the translational model is considered. It is proposed to have another controller taking care of controlling the orientation, such that the thrust vector $\mathbf{T}$ is oriented in the right direction of thrust. The mass of the rocket, m, decreases as the fuel of the rocket is used. Hence it is chosen to design the sliding mode controller with the input being the acceleration, $\frac{\mathbf{T}}{m}$, created by the thrust. The PAD system described in section 9.1.2 estimates the mass of the rocket with such precision that it is possible to determine the precise thrust needed to create the acceleration input for the controller. Thereby it is possible to write the translational dynamic equation as shown in equation 10.2

$$\mathbf{f}(\mathbf{x}) = \dot{\mathbf{x}} = \dot{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix}} = \begin{bmatrix} x_4 \\ x_5 \\ x_6 \\ u_1 - \mu_e \frac{x_1}{||x_{1:3}||^3} - \frac{1}{2}\frac{1}{m}\rho_a C_{D,x} A_x x_4 |x_4| \\ u_2 - \mu_e \frac{x_2}{||x_{1:3}||^3} - \frac{1}{2}\frac{1}{m}\rho_a C_{D,y} A_y x_5 |x_5| \\ u_3 - \mu_e \frac{x_3}{||x_{1:3}||^3} - \frac{1}{2}\frac{1}{m}\rho_a C_{D,z} A_z x_6 |x_6| \end{bmatrix} \tag{10.2}$$

where

$\mathbf{x}_{1:3}$	is the position of the rocket given in the ECI frame
$\mathbf{x}_{4:6}$	is the velocity of the rocket
$\mathbf{u}_{1:3}$	is the input to the system, i.e. the acceleration created by the thrusters

As the sliding mode controller is meant to control the rocket in the ECI frame, the actual value of $C_{D,x:z}$ and $A_{x:z}$ may vary significantly during the ascent, depending on the angle of attack of the drag on the rocket. The value of $C_{D,x:z}$ is expected to vary in the interval [0.295; 1.17] while the value of $A_{x:z}$ is roughly expected to vary in the interval [0.79; 12]. The possibility of experiencing maximum values is very unlikely since this would be caused by an angle of attack of

90°. To design a sliding mode controller, it is necessary to put the system into regular form, i.e. a system on the form as shown in equation 10.3

$$
\begin{aligned}
\dot{\boldsymbol{\eta}} &= \mathbf{f}_a(\boldsymbol{\eta}, \boldsymbol{\xi}) \\
\dot{\boldsymbol{\xi}} &= \mathbf{f}_b(\boldsymbol{\eta}, \boldsymbol{\xi}) + \underline{\mathbf{G}}(\mathbf{x})\underline{\mathbf{E}}(\mathbf{x})\mathbf{u} + \boldsymbol{\delta}(t, \mathbf{x}, \mathbf{u})
\end{aligned} \tag{10.3}
$$

As it is seen from equation 10.2, the system is already in regular form, and for convenience it can be rewritten into terms of $\boldsymbol{\eta}$ and $\boldsymbol{\xi}$ in equation 10.4

$$
\begin{aligned}
\dot{\boldsymbol{\eta}} &= \boldsymbol{\xi} \\
\dot{\boldsymbol{\xi}} &= \mathbf{u} - \underline{\mathbf{a}}_1\boldsymbol{\eta} - \underline{\mathbf{a}}_2\boldsymbol{\xi} \circledast |\boldsymbol{\xi}|
\end{aligned} \tag{10.4}
$$

where

$$
\underline{\mathbf{a}}_1 = \frac{\mu_e}{||\mathbf{x}_{1:3}||^3} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{10.5}
$$

$$
\underline{\mathbf{a}}_2 = \frac{1}{2}\frac{1}{m}\rho_a \begin{bmatrix} C_{D,x}A_x & 0 & 0 \\ 0 & C_{D,y}A_y & 0 \\ 0 & 0 & C_{D,z}A_z \end{bmatrix} \tag{10.6}
$$

$$
\tag{10.7}
$$

To be able to take the parameter variations into account, the system can be split into a nominal system and a system with perturbations, which will be included in the $\boldsymbol{\delta}$ term. Equation 10.8 how the system can be split up, with nominal parameters $\underline{\mathbf{a}}_{1,h}$ and $\underline{\mathbf{a}}_{2,h}$.

$$
\begin{aligned}
\dot{\boldsymbol{\eta}} &= \underbrace{\boldsymbol{\xi}}_{\mathbf{f}_a(\boldsymbol{\eta},\boldsymbol{\xi})} \\
\dot{\boldsymbol{\xi}} &= \underbrace{\mathbf{u} - \underline{\mathbf{a}}_{1,h}\boldsymbol{\eta} - \underline{\mathbf{a}}_{2,h}\boldsymbol{\xi} \circledast |\boldsymbol{\xi}|}_{\mathbf{f}_b(\boldsymbol{\eta},\boldsymbol{\xi})} \underbrace{- (\underline{\mathbf{a}}_1 - \underline{\mathbf{a}}_{1,h})\boldsymbol{\eta} - (\underline{\mathbf{a}}_2 - \underline{\mathbf{a}}_{2,h})\boldsymbol{\xi} \circledast |\boldsymbol{\xi}|}_{\boldsymbol{\delta}(t,\boldsymbol{\eta},\boldsymbol{\xi})}
\end{aligned} \tag{10.8}
$$

With the system described in regular form it is desired to design a control law that restricts the movement of the rocket to a certain surface $\mathbf{s} = \boldsymbol{\xi} - \boldsymbol{\phi}(\boldsymbol{\eta}) = 0$. It is desired to design the sliding surface in such a way, that when the system enters the sliding surface, it is not likely to leave it again. In this case the sliding surface is chosen to be an orbit of 500 km altitude above the surface of earth, because when the rocket is properly injected into the orbit plane, it will not leave it again.

$$
\mathbf{s} = 0 \quad \Rightarrow \quad \boldsymbol{\xi} = \boldsymbol{\phi}(\boldsymbol{\eta}) \tag{10.9}
$$

$$
\dot{\mathbf{s}} = \dot{\boldsymbol{\xi}} - \frac{\partial \boldsymbol{\phi}(\boldsymbol{\eta})}{\partial \boldsymbol{\eta}}\dot{\boldsymbol{\eta}} = \mathbf{u} - \underline{\mathbf{a}}_{1,h}\boldsymbol{\eta} - \underline{\mathbf{a}}_{2,h}\boldsymbol{\xi} \circledast |\boldsymbol{\xi}| - \frac{\partial \boldsymbol{\phi}(\boldsymbol{\eta})}{\partial \boldsymbol{\eta}}\boldsymbol{\xi} + \boldsymbol{\delta}(t, \boldsymbol{\eta}, \boldsymbol{\xi}) \tag{10.10}
$$

The value of $\mathbf{u}$ can be chosen such that it cancels the nominal dynamics of the system. Thereby the control input can be designed as shown in equation 10.11.

$$\mathbf{u} = \underline{\mathbf{a}}_{1,h}\boldsymbol{\eta} + \underline{\mathbf{a}}_{2,h}\boldsymbol{\xi} \circledast |\boldsymbol{\xi}| + \frac{\partial \boldsymbol{\phi}(\boldsymbol{\eta})}{\partial \boldsymbol{\eta}}\boldsymbol{\xi} + \mathbf{v} \tag{10.11}$$

This results in the control law shown in equation 10.12

$$\dot{\mathbf{s}} = \mathbf{v} + \boldsymbol{\delta}(t, \boldsymbol{\eta}, \boldsymbol{\xi}) \tag{10.12}$$

By choosing a continuous function, $\rho(\mathbf{x}) \geq 0$, for which the inequality 10.13 holds

$$|\boldsymbol{\delta}(t, \boldsymbol{\eta}, \boldsymbol{\xi})| \leq \rho(\mathbf{x}) + \kappa_0 ||\mathbf{v}||_\infty \tag{10.13}$$

where $\kappa_0 \,\epsilon\, [0;1)$. Then $\mathbf{v}$ can be designed to make s go towards the sliding surface by choosing it as shown in equation 10.14

$$\mathbf{v} \triangleq -\beta(\mathbf{x})\mathrm{sat}\left(\frac{\mathbf{s}}{\epsilon}\right) \tag{10.14}$$

where

$$\beta(\mathbf{x}) \geq \frac{\rho(\mathbf{x})}{1-\kappa_0} + \beta_0 \quad , \quad \beta_0 > 0 \tag{10.15}$$

The value of $\boldsymbol{\phi}(\boldsymbol{\eta})$ is chosen to be a rotation and scaling matrix, $\underline{\mathbf{k}}$, that when multiplied to $\boldsymbol{\eta}$ gives a velocity vector corresponding to the needed target vector for a circular orbit.

$$\boldsymbol{\xi} = \underline{\mathbf{k}}\boldsymbol{\eta} \tag{10.16}$$

For simplicity, but without losing generality, an orbit with movement in only two axes is chosen as sliding surface for this evaluation. The surface between the two vectors $\mathbf{o}_1$ and $\mathbf{o}_2$, seen below, is chosen as the orbit plane. The height of the orbit, h_o, is chosen to be 500 km, to comply with the orbit given in the requirements in section 3.

$$\mathbf{o}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad , \quad \mathbf{o}_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{10.17}$$

With the above chosen orbit plane, $\underline{\mathbf{k}}$ can be determined as shown in equation 10.18

$$\underline{\mathbf{k}} = \frac{\sqrt{\frac{\mu_e}{h_o}}}{h_o} \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} \tag{10.18}$$

This value of $\underline{\mathbf{k}}$ ensures that when $\mathbf{s} = 0$, the velocity vector, $\boldsymbol{\xi}$, is perpendicular to the position vector, $\boldsymbol{\eta}$, and that the magnitude of the velocity vector is equal to the speed required for a circular orbit at an altitude of h_o, as given in equation 2.15. The challenge with choosing $\boldsymbol{\phi}(\boldsymbol{\eta})$ as such, is that even though the velocity converges towards the right value, it will inevitably happen in an altitude different from h_o and in a plane different from the orbit plane. This means that it is necessary to design $\mathbf{s}$ to take care of the altitude error and cross-track error. By projecting the position vector $\boldsymbol{\eta}$ onto the two vectors, $\mathbf{o}_1$ and $\mathbf{o}_2$, it is possible to determine the projection of the position vector onto the surface, $\boldsymbol{\eta}_S$ as shown in equation 10.19

$$\boldsymbol{\eta}_S = (\boldsymbol{\eta}^\top \mathbf{o}_1)\mathbf{o}_1 + (\boldsymbol{\eta}^\top \mathbf{o}_2)\mathbf{o}_2 \tag{10.19}$$

By using this projection it is possible to determine the cross track error, $\boldsymbol{\eta}_{ct}$, i.e. the distance to the surface, and the height error, $\boldsymbol{\eta}_h$, see figure 10.8 and equation 10.20 and 10.21. When adding the cross track error and height error the total position error can be determined.

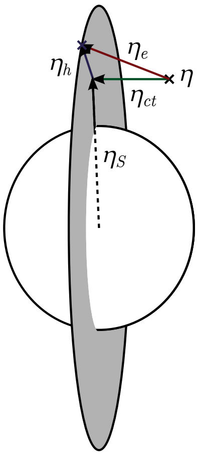


Figure 10.8: Shows how the cross track error and height error is given in relation to the orbit plane. The object is placed in the point $\boldsymbol{\eta}$.

$$\boldsymbol{\eta}_{ct} = \boldsymbol{\eta} - \boldsymbol{\eta}_S \tag{10.20}$$

$$\boldsymbol{\eta}_h = \frac{\boldsymbol{\eta}_S}{||\boldsymbol{\eta}_S||} h_o \tag{10.21}$$

$$\boldsymbol{\eta}_e = \boldsymbol{\eta}_{ct} + \boldsymbol{\eta}_h \tag{10.22}$$

So by choosing $\boldsymbol{\phi}(\boldsymbol{\eta})$ to contain both $\underline{\mathbf{k}}\boldsymbol{\eta}_e$ as shown in equation 10.18 and adding $k_0\boldsymbol{\eta}_e$, $\mathbf{s}$ will be as shown in equation 10.23. The value of k_0 should be chosen with care. Choosing it such that $k_0\boldsymbol{\eta}_e$ becomes much greater than $\underline{\mathbf{k}}\boldsymbol{\eta}_e$, will result in a low rise time, when looking at altitude, but at the same time it might also cause an overshoot. Choosing k_0 very small might cause a very high rise time in altitude, because $k_0\boldsymbol{\eta}_e$ becomes insignificant compared to $\underline{\mathbf{k}}\boldsymbol{\eta}_e$. The value of k_0 should be chosen somewhere in the interval $[0.5 \cdot 10^{-3};10 \cdot 10^{-3}]$ for the nominal values of $k_0\boldsymbol{\eta}_e$ to fit the nominal values $\underline{\mathbf{k}}\boldsymbol{\eta}_e$.

$$\mathbf{s} = \underline{\mathbf{k}}\boldsymbol{\eta} + k_0\boldsymbol{\eta}_e \tag{10.23}$$

Then the input, $\mathbf{u}$, for the control law can be chosen as shown in equation 10.24.

$$\mathbf{u} = \underline{\mathbf{a}}_{1,h}\boldsymbol{\eta} + \underline{\mathbf{a}}_{2,h}\boldsymbol{\xi} \circledast |\boldsymbol{\xi}| + \underline{\mathbf{k}}\boldsymbol{\xi} + \mathbf{v} \tag{10.24}$$

$$\dot{\mathbf{s}} = \mathbf{v} + \underbrace{k_0 \frac{\partial \boldsymbol{\eta}_e}{\partial \boldsymbol{\eta}} \boldsymbol{\xi} + \boldsymbol{\delta}(t, \boldsymbol{\eta}, \boldsymbol{\xi})}_{\boldsymbol{\delta}_1(t,\boldsymbol{\eta},\boldsymbol{\xi})} \tag{10.25}$$

where

$\boldsymbol{\delta}_1(t, \boldsymbol{\eta}, \boldsymbol{\xi})$ is a perturbation term

Then the final step is to design $\beta(\mathbf{x})$ to comply with equation 10.13 to 10.15. The simplest choice of $\rho(\mathbf{x})$ is a constant which will result in $\beta(\mathbf{x})$ being a constant as well.

10.3.2 Test of the Sliding Mode Controller

In the previous section, the general design of the sliding mode controller for AAURocket was shown. This section seeks to evaluate the performance of the designed controller with regards to injection time, eccentricity, control input etc. The control law to be tested is as given in equation 10.26

$$\mathbf{u} = \underline{\mathbf{a}}_{1,h}\boldsymbol{\eta} + \underline{\mathbf{a}}_{2,h}\boldsymbol{\xi} \circledast |\boldsymbol{\xi}| + \underline{\mathbf{k}}\boldsymbol{\xi} - \beta(\mathbf{x})\mathrm{sat}\left(\frac{\mathbf{s}}{\epsilon}\right) \tag{10.26}$$

As mentioned, the value of $\beta(\mathbf{x})$ has been chosen to be constant. However, the constant value of $\beta(\mathbf{x})$ still has to be chosen larger than the maximum value of the perturbations given in $\boldsymbol{\delta}_1$. The value of $\underline{\mathbf{a}}_{1,h}$ is chosen at an altitude of 500 km, this means that the largest perturbation in $(\underline{\mathbf{a}}_{1,h} - \underline{\mathbf{a}}_1)\boldsymbol{\eta}$ is found to be approximately 1.4 m/s^2, i.e. the difference in the gravitational acceleration on Earth compared to at 500 km altitude. As the sliding mode controller is only supposed to work in the exo-atmospheric phase of the launch, the perturbations with regards to atmospheric drag are also quite small. Making a conservative estimate, assuming that the velocity is 3500 m/s in an altitude of 100 km, choosing the worst case angle of attack of 90° and choosing the lowest mass of the rocket of 150 kg, a maximum drag acceleration of approximately 0.1 m/s^2 is obtained. Dependent of the choice of k_0, the perturbation caused by $\frac{\partial \boldsymbol{\eta}_e}{\partial \boldsymbol{\eta}}\boldsymbol{\xi} = \frac{d\boldsymbol{\eta}_e}{dt}$. Choosing k_0 high, i.e. a value of $10 \cdot 10^{-3}$, and conservatively choosing the value of $\frac{d\boldsymbol{\eta}_e}{dt}$ to be 2000 m/s, it will result in a value of 20 m/s^2 for $k_0\frac{d\boldsymbol{\eta}_e}{dt}$. Adding up the conservatively estimated maximum perturbations, it should be safe to evaluate the sliding mode controller with a choice of $\beta(\mathbf{x})$ of a constant value of 40.

With the chosen value for $\beta(\mathbf{x})$, the sliding mode controller can be evaluated. Table 10.3 shows the parameters used for the evaluation of the sliding mode controller.

Orbital Parameter	Value
Initial position	$[0\ 0\ 6478000]^T$
Initial velocity	$[2000\ 0\ 1300]^T$
$\beta(\mathbf{x})$	40
k_0	$5 \cdot 10^-3$
ϵ	300

Table 10.3: Parameters chosen for the evaluation of the sliding mode controller.

The test of the sliding mode controller have been setup in such a way that when the norm of $\mathbf{s}$ gets below a certain value together with the altitude error getting below a certain value, the controller is turned off and a successful orbital insertion has been achieved. For this test case the threshold value for the norm of $\mathbf{s}$ was chosen to 1 and the threshold for the altitude error was chosen to 200 m. Figure 10.9 shows the altitude of the rocket above the surface of earth. The orbital insertion happens after roughly 1500 seconds where the altitude reaches 499.8 km.

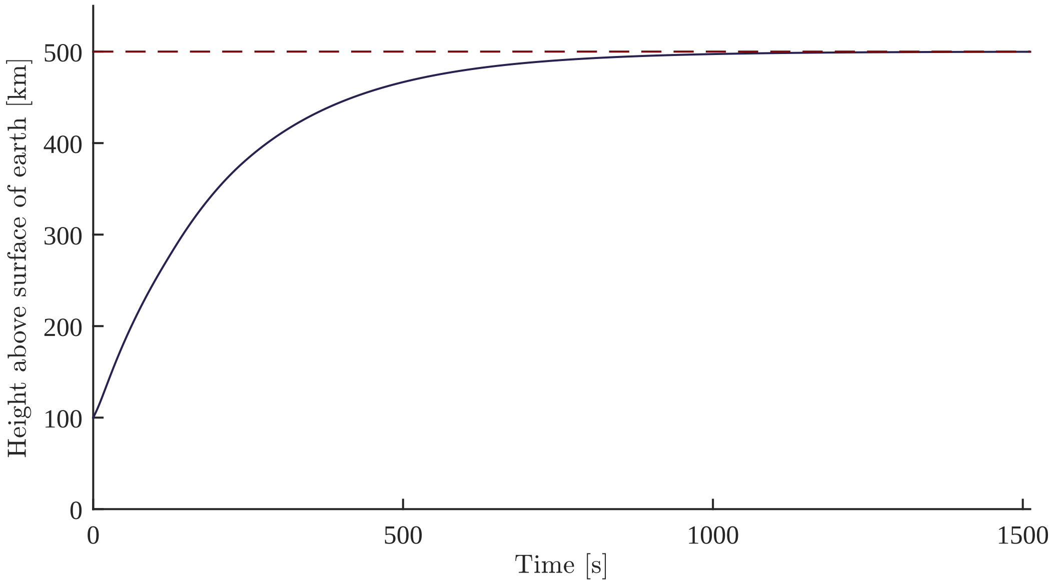


Figure 10.9: Shows a graph of the altitude of the rocket above the surface of earth. The red dotted line indicates the target height of 500 km.

In figure 10.10 the absolute value of the velocity is plotted. As it can be seen from the plot, the rocket reaches its target velocity of 7.621 m/s after approximately 1000 seconds. This also means that the rocket uses around 500 seconds on trying to raise the altitude of the orbit to the required 499.8 km where the orbital insertion is deemed successful. As it can be seen from figure 10.12 the controller uses a lot of control action during this time. It can actually be seen that after around 200 seconds the system reaches the sliding surface and there is a lot of chattering in the control signal. This is most likely due to the very conservative choice of the value of $\beta(\mathbf{x})$.

10.3.3 Acceptance verification

For the sliding mode controller to be verified, it has to comply with the requirements given in section 3. These requirements are presented in table 10.4. The controller is able to perform an orbital insertion within the specified requirements. However, for this test, the mass of the rocket has not been taken into consideration and the control input sequence shown in figure 10.12 indicates that the sliding mode controller would not be able to get to orbit if it was restricted on mass consumption. Furthermore, the same input sequence was not lower bounded as would be the case on the actual system. Hence it cannot be verified that the sliding mode controller is able to comply with the requirements given in table 10.4.

Requirement	Parameter	Value	Unit
SS2.R1	Position control accuracy	<1.3	[km]
SS2.R2	Velocity control accuracy	<3.9	[m/s]

Table 10.4: Position and velocity control requirements

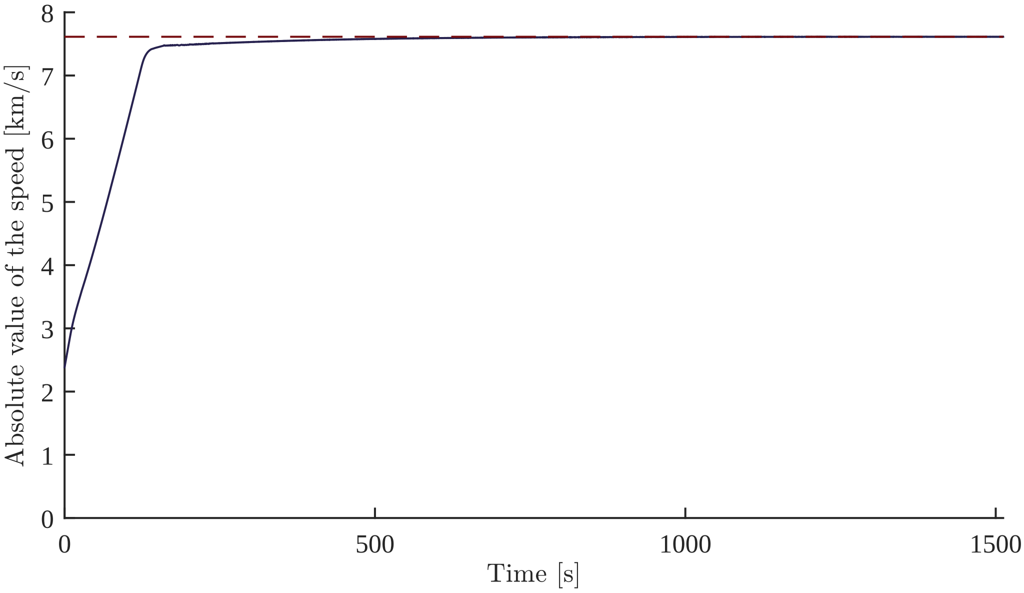


Figure 10.10: Shows a graph of the speed of the rocket. The red dotted line indicates the target speed of 7.613 km/s.

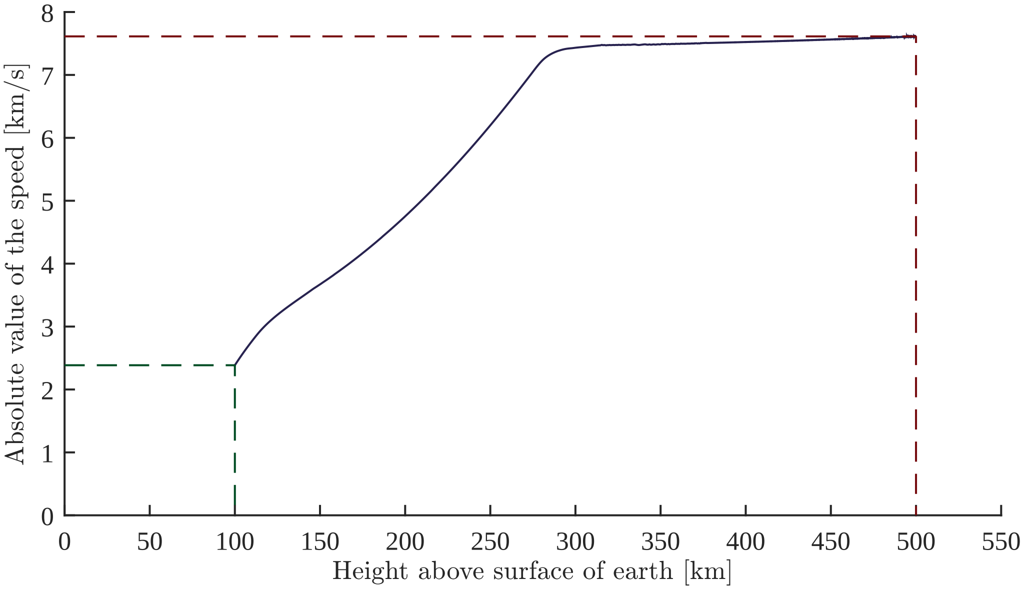


Figure 10.11: Shows a graph of the absolute speed plotted as a function of the height above the surface of earth. The green dotted line indicates the initial conditions and the red dotted line indicates the terminal conditions.

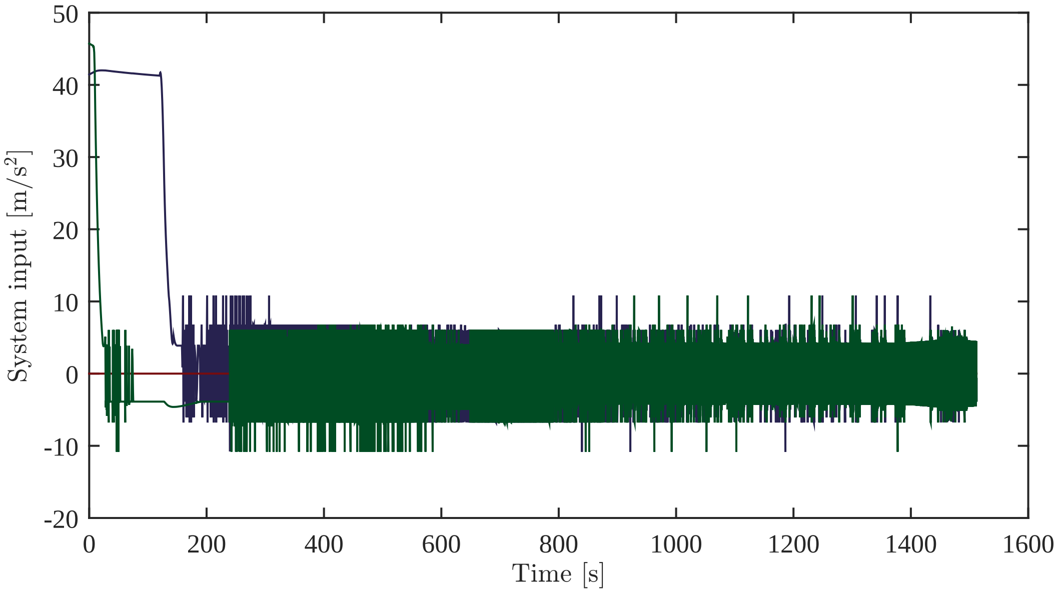


Figure 10.12: Shows a plot of the input sequence generated by the controller.

10.3.4 Subconclusion

This chapter has described two different methods for control of AAURocket. The first control method is a waypoint tracking controller, meant for controlling the rocket during the endo-atmospheric part of the ascent. The other controller is a sliding mode controller meant for controlling the rocket during the exo-atmospheric phase and providing a good orbital insertion. Both controllers have been evaluated on their ability to control the rocket all the way to orbital insertion. Both controllers have shown good performance of controlling the rocket under nominal conditions. However, when the model of consumption of propellant is included, it must be concluded that non of the controllers are able to optimize the consumption such that it is possible to perform an orbital insertion where the rocket stays in orbit. This also leads to the conclusion that it is not deemed relevant to perform an acceptance test for the mission requirements put forward in section 3. The implemented control laws are as follows for the waypoint tracking controller and the sliding mode controller, respectively:

$$\begin{aligned}\text{Waypoint tracking controller :}\mathbf{u} &= K_1 p_{cte} + K_2 v_e + \frac{K_2 K_3}{s} v_e \\ \text{Sliding mode controller :}\mathbf{u} &= \underline{\mathbf{a}}_{1,h}\boldsymbol{\eta} + \underline{\mathbf{a}}_{2,h}\boldsymbol{\xi} \circledast |\boldsymbol{\xi}| + \underline{\mathbf{k}}\boldsymbol{\xi} - \beta(\mathbf{x})\text{sat}\left(\frac{\mathbf{s}}{\epsilon}\right)\end{aligned} \tag{10.27}$$

Chapter 11

Closure

The goal of this thesis has been twofold: to design a position and attitude determination and control system for a self-foraging rocket, as well as lay the groundwork for an eventual AAURocket project, where hopefully other interested students will pitch in with ideas and research.
Following is the conclusion regarding the scope and system requirements of this master thesis.

11.1 Conclusion

In this master thesis a general use case of launching a rocket into a 500 km altitude polar circular orbit has been studied. Different classical rocket designs has been investigated, of which the DNU Self-foraging rocket concept was chosen as a specific design platform for AAURocket. The advantages of such a system would be cheap material, and possibly manufacturing, cost combined with higher efficiency due to the infinite stage nature of the rocket.

Different sensors were considered, while ultimately choosing a sensor setup range including: a barometer, a GPS, an accelerometer, a magnetometer, a gyroscope, a tachometer, and propellant flow sensors.
Furthermore, different classical actuators for chemical rockets were considered with the vernier thruster being the method of choice.
The primary propulsive thruster setup was chosen to be in a setup of four major boosters with a smaller orbital thruster in the middle. Each engine was successively dimensioned from the parameters estimated in the optimization section.

General dynamical equations for both the translational and rotational motion was introduced, as well as some of the major disturbance forces acting on the rocket during the ascent. Additionally analytical functions for the changing moment of inertia was derived for fast computational performance.
And finally a simple thermodynamic model was presented to represent the gasification process.

A general Simulink simulation environment was built. Simulink was chosen as environment in the event that other students want to expand on the work already implemented, as it is an intuitive platform. Models of the general dynamical motion and rotation of the rocket was implemented, along with a set of generally used versors to rotate between frames of interest. Furthermore, a model of the fuel shell was implemented as well as the general environmental forces acting on the rocket. Sensors, actuators, and an onboard computer was also implemented in Simulink to test and verify estimation and control algorithms.

Two different optimal ascent trajectories was calculated, in order to determine both the minimum amount of fuel needed to get into orbit, the magnitude of the maximum thrust needed, as well as the overall throttle profile of the rocket. This trajectory can in turn be used for reference tracking for an endo-atmospheric ascent phase.

An extended Kalman filter was utilized as the general estimation algorithm of the system. The EKF was tested against the system requirements in three distinct nominal test cases where it performed satisfactory within the time horizons specified of 800 seconds.

A waypoint tracking controller and a sliding mode controller was designed for controlling the rocket during the ascent to orbit. While both controllers showed decent performance in controlling the rocket, none of them were able to launch the rocket into orbit with the provided amount of propellant. It is expected that both controllers have the problem of utilizing too much control action to travel along their trajectories, and thereby not alse having enough propellant for raising the altitude and velocity during the ascent.

11.2 Further Development

As has been mentioned this master thesis has been viewed as an initial study into the plausibility of self-foraging rockets as a viable launch vehicle from an estimation and control perspective. As such there is a lot of missing work still to be performed.

One task to further develop is the inclusion of more accurate disturbance forces. While both gravity and some aerodynamic drag has been implemented, several rotational disturbances has been omitted from this master thesis. These disturbances were modeled but never considered beyond that scope. As such further development would be to implement these additional disturbance forces in both Simulink, the estimation algorithms, and the control algorithms.

Another task is to delve deeper into the mechanical part of the project. While definitely not the focus, the mechanical part of this master thesis has often been a lot of unknown variables needed for the design of a lot of the algorithms. This task would probably best be suited mechanical engineering students.

A particular concern arose when the maximum needed mass flow of 40 kg/s arose. Further research into whether this mass flow rate is even plausible or not. If this mass flow rate is too unrealistic a recommendation would be to utilize the self-foraging rocket as an upper stage of a parallel staged launch vehicle. In this way a much smaller engine can be utilized and in turn a lower mass flow rate. Furthermore, this would also significantly reduce the required thrust range of the rocket, with the initial boosters being detachable.

The EKF algorithm needs a bit more robust testing in order to verify whether or not it will fully comply to perturbations or not. Additionally it is recommended that the single attitude sensor setup be expanded to two or more, as this would most likely reduce the attitude drift of the EKF algorithm.

The optimisation algorithm is very slow in its current iteration and some of this calculation time could probably be reduced by further normalizing. As it stands now the task of calculating a trajectory is a tedious and difficult task which require multiple tries as the problem has some local infeasibility regions.

Appendix

Appendix A

Email from Copenhagen Suborbitals

Hej Kasper og Michael,

Tak for jeres intersse.

Vi bruger accelerometre og gyroer, kombineret i en samlet komponent, en såkaldt IMU. Gyroer er kort sagt delt op i to kategorier, de billige bruger MEMS teknologi, og så er der et stort spring i pris og performance op til FOG (fiber optik). Det har vi ikke råd til, så vi bruger generelt de bedste MEMS gyroer vi kan finde. Sapphire fløj med en Analog Devices ADIS16367. I mellemtiden har Analog lavet en lidt bedre (til vores formål) version der hedder ADIS16448, som vi påtænker at flyve NEXØ med. Det helt afgørende for positions nøjagtigheden er, hvor meget gyroernes bias driver efter liftoff, og jo længere tid i skal flyve, jo bedre gyroer skal i bruge. SAPPHIRE kunne kun styre i de 11-12 sek motoren brændte, og til det formål klarede ADIS16367'en sig rigtigt pænt. NEXØ skal brænde ca. 4 gange så længe, så selvom den nye IMU er mere nøjagtig, kan vi nok ikke opretholde helt samme nøjagtighed ved MECO.

Andre kandidater til attitude sensorer er:

Magnetometre, horisont detektorer, solvinkel detektorer. Den store fordel er, at disse ikke lider af gyroernes afdrift. Ulempen er, at de alt andet lige nemmere kan fejle, end en IMU som er uafhængig af input udefra. Der findes nogle fine metoder, f.eks. Kalman filtre, til at kombinere data fra gyroer med data fra sensorer som disse, for at kombinere fordelene. I CS har vi ikke gjort dette endnu, men på NEXØ kommer vi til at flyve et magnetometer. Planen er, at det ikke skal indgå i styringen, men at vi gemmer data til efterfølgende analyse, og evaluering af deres egnethed til fremtidig real time anvendelse.

Appendix B

Moment of inertia

This appendix contains the derivations of the expressions for calculating moment of inertia for AAURocket. The derived expressions are also

B.1 Center of Mass

The rocket consists of mainly two parts, the propulsion system and the payload/guidance system. The propulsion system also consists of two parts, the solid fuel with oxidizer and the engine. The fuel with oxidizer, which acts as the main structure of the rocket, is a hollow cylinder of polyethylene with an oxidizer of H_2O_2 inside. Thereby it is fair to assume the center of mass of the fuel with oxidizer has the center of mass in the geometric center of the of the cylinder. The mission description indicates top part, including the payload/guidance system has a weight of 50 kg, and this is assumed to be a cone with a uniformly distributed mass. The bottom part consisting of the engine is assumed to be a cylinder with a uniformly distributed mass of 100 kg. The figure B.1 shows how the three parts are assumed to be.

With the assumptions made it is fair to assume that the top and bottom part of the rocket will not affect the center of mass particularly, and therefore the center of mass of the entire structure is assumed to be in the geometric center of the fuel part of the rocket.

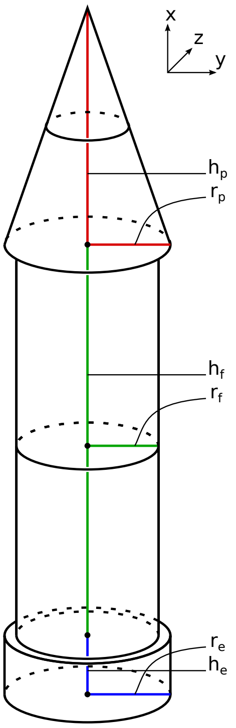


Figure B.1: Shows how the rocket is modeled

B.2 Moment of Inertia

For the calculations of the moment of inertia of the rocket, the same model as shown in figure B.1 is used. The moment of inertia for each of the parts are calculated and summed together. In general the moment of inertia, J, of a point mass, m, located a length, l away from its axis of rotation is calculated as shown in equation B.1

$$J = ml^2 \tag{B.1}$$

Any shape can be described by an infinite amount of small point masses located within the shape. In this case, three shapes with uniformly distributed masses are considered. This also means that the mass of any object can be described by equation B.2

$$m = \rho V \tag{B.2}$$

where:

ρ	is the density of the material of the object	[kg/m^3]
V	is the volume of the object	[m^3]

The general idea is then to take each point mass within an object, calculate the moment of inertia around the objects principal axes and sum all of these moments of inertia together. This is done by taking density of the object, multiplying it by the length towards an arbitrary point within the object and integrating this over the entire volume of the object. This is done as shown in equation B.3

$$J = \int_V \rho l^2 dV \tag{B.3}$$

where:

l is the length towards an arbitrary point within the shape [kg/m^3]

With this general method of determining the moment of inertia of an object, it is possible to determine the moment of inertia for the three parts of the rocket and sum these together to determine the moment of inertia for the entire rocket.

B.2.1 Payload/Guidance system

To determine the moment of inertia of the rocket top, the method described in equation B.3 is used. The limits for the volume integral has to be determined to describe a cone.
For the z-axis the limits is determined by the height of the cone. The bottom of the cone is placed in the height $\frac{h_f}{2}$ and the top of the cone in $\frac{h_f}{2} + h_p$. The y- and x-axis are evaluated over the diameter of the cone, which varies with the height. A function for the radius, $R(z)$, is defined in equation to describe the radius as a function of the height, z. The height, z, denotes the distance from the rocket CoM to the center of the circle plane evaluated.

$$R(z) = r_p \frac{h_p - (z - \frac{hf}{2})}{h_p} \tag{B.4}$$

And thereby all of the limits for the volume integral is determined. The moments of inertia for the Payload/Guidance system can then be determined as shown in equation B.5, B.6 and, B.7.

$$J_x = \rho_p \int_{\frac{h_f}{2}}^{\frac{h_f}{2}+h_p} \int_{-R(z)}^{R(z)} \int_{-\sqrt{R(z)^2-y^2}}^{\sqrt{R(z)^2-y^2}} y^2 + z^2 \ dzdydx = \frac{3}{10} m_p r_p^2 \tag{B.5}$$

$$J_y = \rho_p \int_{\frac{h_f}{2}}^{\frac{h_f}{2}+h_p} \int_{-R(z)}^{R(z)} \int_{-\sqrt{R(z)^2-y^2}}^{\sqrt{R(z)^2-y^2}} x^2 + z^2 \ dzdydx = \frac{1}{20} m_p (5h_f^2 + 5h_f h_p + 2h_p^2 + 3r_p^2) \tag{B.6}$$

$$J_z = \rho_p \int_{\frac{h_f}{2}}^{\frac{h_f}{2}+h_p} \int_{-R(z)}^{R(z)} \int_{-\sqrt{R(z)^2-y^2}}^{\sqrt{R(z)^2-y^2}} x^2 + y^2 \ dzdydx = \frac{1}{20} m_p (5h_f^2 + 5h_f h_p + 2h_p^2 + 3r_p^2) \tag{B.7}$$

B.2.2 Fuel with oxidizer

When determining the moment of inertia of the fuel with oxidizer it is possible to split it in to calculating the moment of inertia for a hollow cylinder with the density of the fuel and a "solid" cylinder with the density of the oxidizer. The moments of inertia for the "solid" cylinder of oxidizer around the three principal axes can be determined by equation B.8, B.9, and B.10

$$J_x = \rho_o \int_{-\frac{h_f}{2}}^{\frac{h_f}{2}} \int_{-r_o}^{r_o} \int_{-\sqrt{r_o^2-y^2}}^{\sqrt{r_o^2-y^2}} y^2 + z^2 \ dzdydx = \frac{1}{2} \rho_o \pi r_o^4 h_f = \frac{1}{2} m_o r_o^2 \tag{B.8}$$

$$J_y = \rho_o \int_{-\frac{h_f}{2}}^{\frac{h_f}{2}} \int_{-r_o}^{r_o} \int_{-\sqrt{r_o^2-y^2}}^{\sqrt{r_o^2-y^2}} x^2 + z^2 \ dzdydx = \frac{1}{12} \rho_o \pi r_o^2 h_f (3r_o^2 + h_f^2) = \frac{1}{12} m_o (3r_o^2 + h_f^2) \tag{B.9}$$

$$J_z = \rho_o \int_{-\frac{h_f}{2}}^{\frac{h_f}{2}} \int_{-r_o}^{r_o} \int_{-\sqrt{r_o^2-y^2}}^{\sqrt{r_o^2-y^2}} x^2 + y^2 \ dzdydx = \frac{1}{12} \rho_o \pi r_o^2 h_f (3r_o^2 + h_f^2) = \frac{1}{12} m_o (3r_o^2 + h_f^2) \tag{B.10}$$

For the hollow cylinder of the fuel the procedure is the same, the only change is that the volume of the cylinder of oxidizer is subtracted. Thereby the moments of inertia for the hollow cylinder of fuel around the threee principal axes is determined by equation B.11, B.12, and B.13

$$
\begin{aligned}
J_x &= \rho_f(\int_{-\frac{h_f}{2}}^{\frac{h_f}{2}}\int_{-r_f}^{r_f}\int_{-\sqrt{r_f^2-y^2}}^{\sqrt{r_f^2-y^2}} y^2+z^2 dxdydz - \int_{-\frac{h_f}{2}}^{\frac{h_f}{2}}\int_{-r_o}^{r_o}\int_{-\sqrt{r_o^2-y^2}}^{\sqrt{r_o^2-y^2}} x^2+y^2\ dzdydx) \qquad \text{(B.11)}\\
&= \frac{1}{2}\rho_f\pi h_f(r_f^4-r_o^4) = \frac{1}{2}\rho_f\pi h_f(r_f^2-r_o^2)(r_f^2+r_o^2)\\
&= \frac{1}{2}m_f(r_f^2+r_o^2)\\
J_y &= \rho_f(\int_{-\frac{h_f}{2}}^{\frac{h_f}{2}}\int_{-r_f}^{r_f}\int_{-\sqrt{r_f^2-y^2}}^{\sqrt{r_f^2-y^2}} x^2+z^2 dxdydz - \int_{-\frac{h_f}{2}}^{\frac{h_f}{2}}\int_{-r_o}^{r_o}\int_{-\sqrt{r_o^2-y^2}}^{\sqrt{r_o^2-y^2}} x^2+z^2\ dzdydx) \qquad \text{(B.12)}\\
&= \frac{1}{12}\rho_f(\pi r_f^2 h_f(3r_f^2+h_f^2) - \pi r_o^2 h_f(3r_o^2+h_f^2)) = \frac{1}{12}\rho_f\pi h_f(r_f^2-r_o^2)(3(r_f^2+r_o^2)+h_f^2)\\
&= \frac{1}{12}m_f(3(r_f^2+r_o^2)+h_f^2)
\end{aligned}
$$

$$
\begin{aligned}
J_z &= \rho_f(\int_{-\frac{h_f}{2}}^{\frac{h_f}{2}}\int_{-r_f}^{r_f}\int_{-\sqrt{r_f^2-y^2}}^{\sqrt{r_f^2-y^2}} x^2+y^2 dxdydz - \int_{-\frac{h_f}{2}}^{\frac{h_f}{2}}\int_{-r_o}^{r_o}\int_{-\sqrt{r_o^2-y^2}}^{\sqrt{r_o^2-y^2}} y^2+z^2\ dzdydx) \qquad \text{(B.13)}\\
&= \frac{1}{12}\rho_f(\pi r_f^2 h_f(3r_f^2+h_f^2) - \pi r_o^2 h_f(3r_o^2+h_f^2)) = \frac{1}{12}\rho_f\pi h_f(r_f^2-r_o^2)(3(r_f^2+r_o^2)+h_f^2)\\
&= \frac{1}{12}m_f(3(r_f^2+r_o^2)+h_f^2)
\end{aligned}
$$

B.2.3 Engine

The engine is also modeled as a cylinder as shown in figure B.1. The moments of inertia for the engine, modeled as a cylinder, around the three principal axes can be determined by equation B.14, B.15, and B.16.

$$
\begin{aligned}
J_x &= \rho_e\int_{-(\frac{h_f}{2}+h_e)}^{-\frac{h_f}{2}}\int_{-r_e}^{r_e}\int_{-\sqrt{r_e^2-y^2}}^{\sqrt{r_e^2-y^2}} y^2+z^2\ dzdydx = \frac{1}{2}\rho_e\pi r_e^4 h_e \qquad \text{(B.14)}\\
&= \frac{1}{2}m_e r_e^2\\
J_y &= \rho_e\int_{-(\frac{h_f}{2}+h_e)}^{-\frac{h_f}{2}}\int_{-r_e}^{r_e}\int_{-\sqrt{r_e^2-y^2}}^{\sqrt{r_e^2-y^2}} x^2+z^2\ dzdydx = \frac{1}{12}\rho_e\pi r_e^2 h_e(3h_f^2+6h_f h_e+4h_e^2+3r_e^2) \qquad \text{(B.15)}\\
&= \frac{1}{12}m_e(3h_f^2+6h_f h_e+4h_e^2+3r_e^2)
\end{aligned}
$$

$$
\begin{aligned}
J_z &= \rho_e\int_{-(\frac{h_f}{2}+h_e)}^{-\frac{h_f}{2}}\int_{-r_e}^{r_e}\int_{-\sqrt{r_e^2-y^2}}^{\sqrt{r_e^2-y^2}} x^2+y^2\ dzdydx = \frac{1}{12}\rho_e\pi r_e^2 h_e(3h_f^2+6h_f h_e+4h_e^2+3r_e^2) \qquad \text{(B.16)}\\
&= \frac{1}{12}m_e(3h_f^2+6h_f h_e+4h_e^2+3r_e^2)
\end{aligned}
$$

Appendix C

Optimization ICLOCS Implementation Code

C.1 main.m

```
% MAIN - Main script to solve the Optimal Control Problem
%
% Copyright (C) 2010 Paola Falugi, Eric Kerrigan
% and Eugene van Wyk. All Rights Reserved.
% This code is published under the BSD License.
% Department of Electrical and Electronic Engineering,
% Imperial College London London  England, UK
% ICLOCS (Imperial College London Optimal Control) 5 May 2010
% iclocs@imperial.ac.uk


%--------------------------------------------------------

close all
clear all;
format compact;
clc

rng('shuffle');

global sol;
sol=[];                                 % Initialize solution structure

[problem,guess]= MinPropellantToOrbit;     % Fetch the problem definition
%[problem,guess]= SparePropellantToOrbit;  % Fetch the problem definition
options = settings;                     % Get options and solver settings
[infoNLP,data]=transcribeOCP(problem,guess,options); % Format for NLP solver
```

C.2 MinPropellantToOrbit.m

```
function [problem,guess] = MinPropellantToOrbit

mfl= 150;                       %Lower bound for Final Mass [kg]
mfu= 150;                       %Upper bound for Final Mass [kg]

m0l = 2500;                     %Lower bound for initial Mass [kg]
m0u = 8000;                     %Upper bound for initial Mass [kg]

v_orbit = 7621.342985;          %Orbit Velocity [m/s]s

%Physical Constants
R_E=6375750;                    %Earth Radius
g_0=9.8067;                     %Gravity at sea level
mu = 398600441800000;           %Standard Gravitational Paramater

mscale = 1000;                  %Scaling of mass to ton

%Normalizers
tn = (R_E/g_0).^(.5);           %Time Normalizer
rn = R_E;                       %Position Normalizer
vn = (R_E*g_0).^(.5);           %Velocity Normalizer
an = g_0;                       %Acceleration Normalizer

%inclin = 97.38*(pi/180);       %[rad]
inclin = 82.62*(pi/180);        %[rad]

init_vertical_speed = 1;        %[m/s]
coriolis = [-24.0852 136.5951 0];             %[m/s]
%[-24.0852 136.5951 0];           %[m/s] coriolis vector @ AAU
%||v(t_0)|| = 138.7059 [m/s]

% Initial time. t0<tf. For discrete time systems is the initial index
problem.time.t0=0;

% Final time. Let tf_min=tf_max if tf is fixed.
problem.time.tf_min=1/tn;
problem.time.tf_max=1200/tn;
guess.tf=580/tn;

% Parameters bounds. pl=< p <=pu
problem.parameters.pl=[1];
problem.parameters.pu=[10];
guess.parameters=[2.2647];

% Initial conditions for system.
problem.states.x0=[];

% Aalborg radius: 6375750
% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u
problem.states.x0l=[3434202.222/rn ...
                    605537.877/rn ...
                    5337580.635/rn ...
                    ((init_vertical_speed*0.5386)+coriolis(1))./vn ...
                    ((init_vertical_speed*0.0950)+coriolis(2))./vn ...
                    ((init_vertical_speed*0.8372)+coriolis(3))./vn ...
                    m0l/mscale -24.0852./vn 136.5951./vn];% 0.5787 0.1746];

problem.states.x0u=[3434202.222/rn ...
                    605537.877/rn ...
                    5337580.635/rn ...
                    ((init_vertical_speed*0.5386)+coriolis(1))./vn ...
                    ((init_vertical_speed*0.0950)+coriolis(2))./vn ...
                    ((init_vertical_speed*0.8372)+coriolis(3))./vn ...
                    m0u/mscale -24.0852./vn 136.5951./vn];% 0.5787 0.1746];


% State bounds. xl=< x <=xu
problem.states.xl=[-(R_E+3010000)/rn -(R_E+3010000)/rn -(R_E+3010000)/rn ...
 -8000/vn -8000/vn -8000/vn mfl/mscale -inf -inf];% 0 0];
problem.states.xu=[(R_E+3010000)/rn (R_E+3010000)/rn (R_E+3010000)/rn ...
 8000/vn 8000/vn 8000/vn m0u/mscale inf inf];% pi 2*pi];

% Terminal state bounds. xfl=< xf <=xfu
%problem.states.xf=[];
problem.states.xfl=[-(R_E+510000)/rn -(R_E+510000)/rn -(R_E+510000)/rn ...
-8000/vn -8000/vn -8000/vn mfl/mscale -inf -inf];% 0 0];
problem.states.xfu=[(R_E+510000)/rn (R_E+510000)/rn (R_E+510000)/rn ...
8000/vn 8000/vn 8000/vn mfu/mscale inf inf];% pi 2*pi];

% Guess the state trajectories with [x0 xf]
% guess.states(:,:)=[x0 xf];
guess.states = [];

% Number of control actions N
% Set problem.inputs.N=0 if N is equal to the number of integration steps.
% Note that the number of integration steps
% defined in settings.m has to be divisible
% by the  number of control actions N whenever it is not zero.
problem.inputs.N=0;

% Input bounds
problem.inputs.ul=[0 0 0.025];
problem.inputs.uu=[pi 2*pi-0.00000000000000000000001 1];

% Guess the input sequences with [u0 uf]
guess.inputs=[];

% Choose the set-points if required
problem.setpoints.states=[];
problem.setpoints.inputs=[];


% Bounds for path constraint function gl =< g(x,u,p,t) =< gu
problem.constraints.gl=[1 0 0];
problem.constraints.gu=[1.5 5 5.3610e-04.*10000];

% Bounds for boundary constraints bl =< b(x0,xf,u0,uf,p,t0,tf) =< bu
```

```
problem.constraints.bl=[0 0 0 0];
problem.constraints.bu=[0 0 0 0];

% Store the necessary problem parameters used in the functions
problem.data.R_E=R_E;
problem.data.g_0=g_0;
problem.data.mu = mu;
problem.data.v_orbit = v_orbit;
%Normalizers
problem.data.mscale = mscale;
problem.data.tn = tn;
problem.data.rn = rn;
problem.data.vn = vn;
problem.data.an = an;
problem.data.inclin = inclin;

problem.functions={@L,@E,@f,@g,@b};


%Lagrangian:
function stageCost=L(x,xr,u,ur,p,t,data)
% Syntax:  stageCost = L(x,xr,u,ur,p,t,data)
stageCost = 0*t;


%Boundary Cost:
function boundaryCost=E(x0,xf,u0,uf,p,tf,data)
% Syntax:  boundaryCost=E(x0,xf,u0,uf,p,tf,data)
boundaryCost=x0(7);


%System Dynamics:
function dx = f(x,u,p,t,data)
%Physical Constants
R_E = data.R_E;             %[m]
g_0 = data.g_0;             %[m/s^2]

%Normalizers
vn = data.vn;               %Velocity [.]
an = data.an;               %Acceleration [.]

%State Labels
r1 = x(:,1);
r2 = x(:,2);
r3 = x(:,3);
v1 = x(:,4);
v2 = x(:,5);
v3 = x(:,6);
m = x(:,7);
vcoryx = x(:,8);
vcoryy = x(:,9);

%Input Labels
theta = u(:,1);
phi = u(:,2);
```

```
eta = u(:,3);

%Maximum Thrust
Tmax = x(1,7).*p.*g_0;      %[kgm/sÂ²]

%Air Density
rhoair = ((1<(r1.^2+r2.^2+r3.^2).^(.5)).*1.277 ...
  .*exp(-761.2645*((r1.^2+r2.^2+r3.^2).^(.5)-1)));

%Control Vector (aka LV Attitude)
cv1 = sin(theta).*cos(phi);
cv2 = sin(theta).*sin(phi);
cv3 = cos(theta);

vmagsquare = ((v1-vcoryx).*vn).^2+((v2-vcoryy).*vn).^2+(v3.*vn).^2;

%Atmosphere Movement
acorix = 2*v2.*vn.*((2*pi)/(24*3600)).*(rhoair./1.277);
acoriy = -2*v1.*vn.*((2*pi)/(24*3600)).*(rhoair./1.277);

%Angle of Attack (LV attitude vs velocity)
vnorm = ((v1-vcoryx).^2+(v2-vcoryy).^2+v3.^2).^(.5);
vnorm1 = (v1-vcoryx)./vnorm;
vnorm2 = (v2-vcoryy)./vnorm;
vnorm3 = v3./vnorm;

cvn1 = cv2.*(cv1.*vnorm2-cv2.*vnorm1)-cv3.*(cv3.*vnorm1-cv1.*vnorm3);
cvn2 = cv3.*(cv2.*vnorm3-cv3.*vnorm2)-cv1.*(cv1.*vnorm2-cv2.*vnorm1);
cvn3 = cv1.*(cv3.*vnorm1-cv1.*vnorm3)-cv2.*(cv2.*vnorm3-cv3.*vnorm2);

alphaprime = vnorm1.*cv1+vnorm2.*cv2+vnorm3.*cv3;         %[.]
betaprime = vnorm1.*cvn1+vnorm2.*cvn2+vnorm3.*cvn3;

%Gravity Disturbance
gravity = 1./(((r1.^2+r2.^2+r3.^2).^(.5)).^3);           %[.]

% radius = .363;
% CDA = 0.0611
% CDN = 0.7513



%State Equations
dx(:,1) = v1;
dx(:,2) = v2;
dx(:,3) = v3;

dx(:,4) = eta.*(Tmax./(m.*an)).*cv1-gravity.*r1-...
((0.0938.*alphaprime.*cv1+1.0346.*x(:,7).*betaprime.*cvn1)...
.*vmagsquare.*rhoair)./(m.*1000.*an);
dx(:,5) = eta.*(Tmax./(m.*an)).*cv2-gravity.*r2-...
((0.0938.*alphaprime.*cv2+1.0346.*x(:,7).*betaprime.*cvn2)...
.*vmagsquare.*rhoair)./(m.*1000.*an);
dx(:,6) = eta.*(Tmax./(m.*an)).*cv3-gravity.*r3-...
((0.0938.*alphaprime.*cv3+1.0346.*x(:,7).*betaprime.*cvn3)...
.*vmagsquare.*rhoair)./(m.*1000.*an);

cn = (258.13+24.68.*(1-(rhoair./1.277)))/((R_E/(g_0^3)).^(.5));
dx(:,7) = -eta.*Tmax./cn;               %Mass flow [.]

dx(:,8) = acorix./an;   %Coriolis state
dx(:,9) = acoriy./an;   %Coriolis state


%Path Constraints:
function c=g(x,u,p,t,data)
% Syntax:  c=g(x,u,p,t,data)
%State Labels
r1 = x(:,1);
r2 = x(:,2);
r3 = x(:,3);
v1 = x(:,4);
v2 = x(:,5);
v3 = x(:,6);
m = x(:,7);
vcoryx = x(:,8);
vcoryy = x(:,9);

%Input Labels
eta = u(:,3);

Tmax = x(1,7).*p;

c(:,1) = (r1.^2+r2.^2+r3.^2).^(.5);             %Surface of the Earth constraint
c(:,2) = eta.*(Tmax./m);                              %Maximum g-load constraint
c(:,3) = (1<(r1.^2+r2.^2+r3.^2).^(.5)).*0.6385 ...
 .*exp(-761.2645*((r1.^2+r2.^2+r3.^2).^(.5)-1))...
 .*((v1-vcoryx).^2+(v2-vcoryy).^2+v3.^2).*10000;   %Dynamic Pressure


%Boundary Constraints:
function bc=b(x0,xf,u0,uf,p,tf,data)
% Syntax:  bc=b(x0,xf,u0,uf,p,tf,data)
rn = data.rn;
vn = data.vn;
R_E = data.R_E;
v_orbit = data.v_orbit;
inclin = data.inclin;

bc=[(xf(1)*xf(1)+xf(2)*xf(2)+xf(3)*xf(3)).^(.5)-1.0784;  %Orbit Height
    (xf(4)*xf(4)+xf(5)*xf(5)+xf(6)*xf(6)).^(.5)-0.9638;  %Orbit Velocity
    xf(1)*xf(4)+xf(2)*xf(5)+xf(3)*xf(6);                 %Eccentricity
    (xf(1)*xf(5)-xf(2)*xf(4))-0.1335];                   %Inclination

%------------- END OF CODE --------------|
```

C.3 SparePropellantToOrbit.m

```
function [problem,guess] = SparePropellantToOrbit

mfl= 150;                       %Lower bound for Final Mass [kg]
mfu= 400;                       %Upper bound for Final Mass [kg]

m0l = 8000;                     %Lower bound for initial Mass [kg]
m0u = 8000;                    %Upper bound for initial Mass [kg]

v_orbit = 7621.3429849192434630822390317918;  %Orbit Velocity [m/s]s

%Physical Constants
R_E=6375750;                    %Earth Radius
g_0=9.8067;                     %Gravity at sea level
mu = 398600441800000;           %Standard Gravitational Paramater

mscale = 1000;                  %Scaling of mass to ton

%Normalizers
tn = (R_E/g_0).^(.5);           %Time Normalizer
rn = R_E;                       %Position Normalizer
vn = (R_E*g_0).^(.5);           %Velocity Normalizer
an = g_0;                       %Acceleration Normalizer

%inclin = 97.38*(pi/180);       %[rad]
inclin = 82.62*(pi/180);        %[rad]

init_vertical_speed = 1;        %[m/s]
coriolis = [-24.0852 136.5951 0];             %[m/s]
%[-24.0852 136.5951 0];           %[m/s] coriolis vector @ AAU

% Initial time. t0<tf. For discrete time systems is the initial index
problem.time.t0=0;

% Final time. Let tf_min=tf_max if tf is fixed.
problem.time.tf_min=1/tn;
problem.time.tf_max=1200/tn;
guess.tf=700/tn;

% Parameters bounds. pl=< p <=pu
problem.parameters.pl=[];
problem.parameters.pu=[];
guess.parameters=[];

% Initial conditions for system.
problem.states.x0=[];

% Aalborg radius: 6375750
% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u
problem.states.x0l=[3434202.222/rn ...
                    605537.877/rn ...
                    5337580.635/rn ...
                    ((init_vertical_speed*0.5386)+coriolis(1))./vn ...
                    ((init_vertical_speed*0.0950)+coriolis(2))./vn ...
                    ((init_vertical_speed*0.8372)+coriolis(3))./vn ...
                    m0l/mscale -24.0852./vn 136.5951./vn];

problem.states.x0u=[3434202.222/rn ...
                    605537.877/rn ...
                    5337580.635/rn ...
                    ((init_vertical_speed*0.5386)+coriolis(1))./vn ...
                    ((init_vertical_speed*0.0950)+coriolis(2))./vn ...
                    ((init_vertical_speed*0.8372)+coriolis(3))./vn ...
                    m0u/mscale -24.0852./vn 136.5951./vn];


% State bounds. xl=< x <=xu
problem.states.xl=[-(R_E+3010000)/rn -(R_E+3010000)/rn -(R_E+3010000)/rn ...
 -8000/vn -8000/vn -8000/vn mfl/mscale -inf -inf];% 0 0];
problem.states.xu=[(R_E+3010000)/rn (R_E+3010000)/rn (R_E+3010000)/rn ...
  8000/vn 8000/vn 8000/vn m0u/mscale inf inf];% pi 2*pi];

% Terminal state bounds. xfl=< xf <=xfu
%problem.states.xf=[];
problem.states.xfl=[-(R_E+510000)/rn -(R_E+510000)/rn -(R_E+510000)/rn ...
-8000/vn -8000/vn -8000/vn mfl/mscale -inf -inf];% 0 0];
problem.states.xfu=[(R_E+510000)/rn (R_E+510000)/rn (R_E+510000)/rn ...
 8000/vn 8000/vn 8000/vn mfu/mscale inf inf];% pi 2*pi];

% Guess the state trajectories with [x0 xf]
% guess.states(:,:)=[x0 xf];
guess.states = [];

% Number of control actions N
% Set problem.inputs.N=0 if N is equal to the number of integration steps.
% Note that the number of integration steps
% defined in settings.m has to be divisible
% by the  number of control actions N whenever it is not zero.
problem.inputs.N=0;

% Input bounds
problem.inputs.ul=[0 0 0.025];
problem.inputs.uu=[pi 2*pi 1];

% Guess the input sequences with [u0 uf]
guess.inputs=[];


% Choose the set-points if required
problem.setpoints.states=[];
problem.setpoints.inputs=[];


% Bounds for path constraint function gl =< g(x,u,p,t) =< gu
problem.constraints.gl=[1 0 0];
problem.constraints.gu=[1.5 5 5.3610e-04.*10000];
% 700/vn^2 = 5.3610e-04
% 400/vn^2 = 6.3974e-06
% Bounds for boundary constraints bl =< b(x0,xf,u0,uf,p,t0,tf) =< bu
problem.constraints.bl=[0 0 0 0 -inf];
problem.constraints.bu=[0 0 0 0 0];

% Store the necessary problem parameters used in the functions
problem.data.R_E=R_E;
problem.data.g_0=g_0;
problem.data.mu = mu;
problem.data.v_orbit = v_orbit;
%Normalizers
problem.data.mscale = mscale;
problem.data.tn = tn;
problem.data.rn = rn;
problem.data.vn = vn;
problem.data.an = an;
problem.data.inclin = inclin;

problem.functions={@L,@E,@f,@g,@b};


%Lagrangian:
function stageCost=L(x,xr,u,ur,p,t,data)
% Syntax:  stageCost = L(x,xr,u,ur,p,t,data)
stageCost = 0*t;


%Boundary Cost:
function boundaryCost=E(x0,xf,u0,uf,p,tf,data)
% Syntax:  boundaryCost=E(x0,xf,u0,uf,p,tf,data)
boundaryCost=-xf(7);


%System Dynamics:
function dx = f(x,u,p,t,data)
%Physical Constants
R_E = data.R_E;             %[m]
g_0 = data.g_0;             %[m/s^2]

%Normalizers
vn = data.vn;               %Velocity [.]
an = data.an;               %Acceleration [.]

%State Labels
r1 = x(:,1);
r2 = x(:,2);
r3 = x(:,3);
v1 = x(:,4);
v2 = x(:,5);
v3 = x(:,6);
m = x(:,7);
vcoryx = x(:,8);
vcoryy = x(:,9);

%Input Labels
```

```
theta = u(:,1);
phi = u(:,2);
eta = u(:,3);

%Maximum Thrust
Tmax = 101.2311;

%Air Density
rhoair = ((1<(r1.^2+r2.^2+r3.^2).^(.5)).*1.277 ...
.*exp(-761.2645*((r1.^2+r2.^2+r3.^2).^(.5)-1)));

%Control Vector (aka LV Attitude)
cv1 = sin(theta).*cos(phi);
cv2 = sin(theta).*sin(phi);
cv3 = cos(theta);

vmagsquare = ((v1-vcoryx).*vn).^2+((v2-vcoryy).*vn).^2+(v3.*vn).^2;

%Atmosphere Movement
acorix = 2*v2.*vn.*((2*pi)/(24*3600)).*(rhoair./1.277);
acoriy = -2*v1.*vn.*((2*pi)/(24*3600)).*(rhoair./1.277);

%Angle of Attack (LV attitude vs velocity)
vnorm = ((v1-vcoryx).^2+(v2-vcoryy).^2+v3.^2).^(.5);
vnorm1 = (v1-vcoryx)./vnorm;
vnorm2 = (v2-vcoryy)./vnorm;
vnorm3 = v3./vnorm;

cvn1 = cv2.*(cv1.*vnorm2-cv2.*vnorm1)-cv3.*(cv3.*vnorm1-cv1.*vnorm3);
cvn2 = cv3.*(cv2.*vnorm3-cv3.*vnorm2)-cv1.*(cv1.*vnorm2-cv2.*vnorm1);
cvn3 = cv1.*(cv3.*vnorm1-cv1.*vnorm3)-cv2.*(cv2.*vnorm3-cv3.*vnorm2);

alphaprime = vnorm1.*cv1+vnorm2.*cv2+vnorm3.*cv3;         %[.]
betaprime = vnorm1.*cvn1+vnorm2.*cvn2+vnorm3.*cvn3;

%Gravity Disturbance
gravity = 1./(((r1.^2+r2.^2+r3.^2).^(.5)).^3);            %[.]

% radius = .363;
% CDA = 0.0611
% CDN = 0.7513

%State Equations
dx(:,1) = v1;
dx(:,2) = v2;
dx(:,3) = v3;

dx(:,4) = eta.*(Tmax./(m.*an)).*cv1-gravity.*r1- ...
 ((0.0938.*alphaprime.*cv1+1.0346.*x(:,7).*betaprime.*cvn1) ...
 .*vmagsquare.*rhoair)./(m.*1000.*an);
dx(:,5) = eta.*(Tmax./(m.*an)).*cv2-gravity.*r2- ...
 ((0.0938.*alphaprime.*cv2+1.0346.*x(:,7).*betaprime.*cvn2) ...
 .*vmagsquare.*rhoair)./(m.*1000.*an);
dx(:,6) = eta.*(Tmax./(m.*an)).*cv3-gravity.*r3- ...
 ((0.0938.*alphaprime.*cv3+1.0346.*x(:,7).*betaprime.*cvn3) ...
 .*vmagsquare.*rhoair)./(m.*1000.*an);

cn = (258.13+24.68.*(1-(rhoair./1.277)))/((R_E/(g_0^3)).^(.5));
dx(:,7) = -eta.*Tmax./cn;        %Mass flow [.]

dx(:,8) = acorix./an;
dx(:,9) = acoriy./an;


%Path Constraints:
function c=g(x,u,p,t,data)
% Syntax:  c=g(x,u,p,t,data)
%State Labels
r1 = x(:,1);
r2 = x(:,2);
r3 = x(:,3);
v1 = x(:,4);
v2 = x(:,5);
v3 = x(:,6);
m = x(:,7);
vcoryx = x(:,8);
vcoryy = x(:,9);

g_0 = data.g_0;

%Input Labels
eta = u(:,3);

%Tmax = x(1,7).*p;
Tmax = 101.2311/g_0;

c(:,1) = (r1.^2+r2.^2+r3.^2).^(.5);    %Surface of the Earth constraint
c(:,2) = eta.*(Tmax./m);               %Maximum g-load constraint
c(:,3) = (1<(r1.^2+r2.^2+r3.^2).^(.5)).*0.6385 ...
.*exp(-761.2645*((r1.^2+r2.^2+r3.^2).^(.5)-1))...
.*((v1-vcoryx).^2+(v2-vcoryy).^2+v3.^2).*10000;  %Dynamic Pressure


%Boundary Constraints:
function bc=b(x0,xf,u0,uf,p,tf,data)
% Syntax:  bc=b(x0,xf,u0,uf,p,tf,data)
rn = data.rn;
vn = data.vn;
R_E = data.R_E;
v_orbit = data.v_orbit;
inclin = data.inclin;

bc=[(xf(1)*xf(1)+xf(2)*xf(2)+xf(3)*xf(3)).^(.5)-1.0784;  %Orbit Height
    (xf(4)*xf(4)+xf(5)*xf(5)+xf(6)*xf(6)).^(.5)-0.9638;  %Orbit Velocity
    xf(1)*xf(4)+xf(2)*xf(5)+xf(3)*xf(6);                 %Eccentricity
    (xf(1)*xf(5)-xf(2)*xf(4))-0.1335;                    %Inclination
    (xf(3)*xf(4)-xf(1)*xf(6))];

%------------- END OF CODE --------------
```

C.4 settings.m

```
function options = settings
%SETTINGS - General and solver-specific settings are selected here
% Unless specified otherwise the options are set using 0 => no and 1 => yes

%------------- BEGIN CODE --------------

% Transcription Method:
%---------------------------------------
options.transcription='trapezoidal';
% Discrete-time model       ('discrete')
% Multiple shooting method  ('multiple_shooting')
%                            WARNING: The 'quasi-newton' option for the
%                            hessian computation has to be selected.
%                            This setting can only be used of tf is not a
%                            decision variable.
%
% Euler method              ('euler')
% Trapezoidal method        ('trapezoidal')
% Hermite-Simpson method    ('hermite')


% Derivative generation :
%---------------------------------------
options.derivatives='numeric';
% Whenever the analytic differentiation is enabled it is necessary to
% specify the available analytic forms for the cost function, the dynamic equations
% and  the constraints in the appropriate files .m
% Numerical differentiation: finite differences  ('numeric')
% Analytic differentiation: analytic gradients   ('analytic')


% Numeric generation of the Hessian:
%----------------------------------------------------------------
options.hessianFD='forward';
% Whenever the numeric differentiation is enabled it is necessary to
% specify which kind of finite difference approximation to use  between
% the following ones:
%
% Central difference ('central')
% forward difference ('forward')

% (32*eps)^(1/3)
% (2*eps)^(1/3)
options.perturbation.H=[];
options.perturbation.J=[];
%  The perturbation size for numerical second derivatives
%  can be set in options.perturbation.H. The perturbation size for
%  numerical first derivatives can be set in  options.perturbation.J.
%
%  It is possible to select default values for the perturbations by setting
%  options.perturbation.H and options.perturbation.J to the empty matrix.
%  The default values for the gradient approximation is (eps/2)^(1/3)
%  while for the  second derivative is (8*eps)^(1/3).


% NLP solver
%---------------------------------------
% IPOPT: recommended but needs ipopt.mex        ('ipopt')
% fmincon                                       ('fmincon')
options.NLPsolver='ipopt';


%Tolererances
relTOL = 1.e-50;
absTOL = 1.e-50;
ipoptTOL = 1.e-3;


% IPOPT settings (if required)
options.ipopt.tol=ipoptTOL;            % Absolute tolerance

options.ipopt.print_level=5;
  % Print level. The valid range for this integer
  %option is [0,12] and its default value is 5.

options.ipopt.max_iter=3000;
  % Maximum number of iterations. The default value is 3000.

options.ipopt.mu_strategy ='adaptive';
  % Determines which barrier parameter update strategy is to be used.
  % The default value for this string option is "monotone".
  % Possible values:
  %   'monotone': use the monotone (Fiacco-McCormick) strategy
  %   'adaptive': use the adaptive update strategy

options.ipopt.hessian_approximation='limited-memory';
  %  Indicates what information for the Hessian of the Lagrangian function is
  %  used by the algorithm. The default value is 'exact'.
  %  Possible values:
  %   'exact': Use second derivatives provided by ICLOCS.
  %   'limited-memory': Perform a limited-memory quasi-Newton approximation
  %  implemented inside IPOPT


% fmincon settings (if required)
%options.fmincon=optimset('GradObj','on','GradConstr','on');
%Alternate:
options.fmincon=optimset;

% Automatic scaling (recommended)
options.scaling=1;


% Output settings
%---------------------------------------
% Display relative local discretization error
%(recommended for direct transcription)
```

```
options.print.relative_local_error=1;
options.print.time=1;          % Display computation time
options.print.cost=1;          % Display cost

options.plot.states=1;         % Plot states
options.plot.inputs=1;         % Plot inputs
options.plot.multipliers=1;    % Plot Lagrange multipliers


% Direct transcription settings
%---------------------------------------
options.nodes=1001;
% Number of   integration nodes in the interval t=[0,tf]; nodes=steps+1.
% The quantity steps/N (N number of control actions) must be a positive
% integer.


options.tau=0;
% Distribution of integration steps. Set tau=0 for equispaced steps.
% Otherwise: tau is a vector of length M-1 with 0<tau(i)<1 and sum(tau)=1.
% For discrete time system  set  tau=0.


options.ODEsolver='cvodes';
% Multiple shooting settings
%---------------------------------------
% N/S=normal/stiff. H/M/L=high/medium/low accuracy
%
% 'cvodes'   N/S    H    High accuracy, difficult problems(slow)
%
%
%  Note: 'cvodes' requires the sundialsTB.


% CVODES settings (if required)
Method='Adams';         % Method: BDF, Adams
Solver='Functional';    % Solver: Newton, Functional (requires dfdx)

%Recommended: Combine {BDF,Newton} or {Adams,Functional}


% Options for forward integration
options.cvodes = CVodeSetOptions('RelTol',relTOL,...
                          'AbsTol',absTOL,...
                          'LinearSolver','Dense',...
                          'MaxNumSteps',10000,...
                          'LMM',Method,...
                          'NonlinearSolver',Solver);

% Forward sensitivity options
options.cvodesf=CVodeSensSetOptions('ErrControl',true,'method',...
'Staggered','RelTol',relTOL,'AbsTol',absTOL); % FSA initialization
```

Bibliography

[Ahem 09] Judson L. Ahem. *Geomagnetic Field Illustration.* 2009. http://geophysics.ou.edu/solid_earth/notes/mag_earth/magnetic_field_a.gif.

[Arianespace 11a] Arianespace. *Ariane 5 user manual.* 2011. http://www.arianespace.com/launch-services-ariane5/Ariane5_users_manual_Issue5_July2011.pdf.

[Arianespace 11b] Arianespace. *Ariane 5 User Manual.* 2011. http://www.arianespace.com/launch-services-ariane5/Ariane5_users_manual_Issue5_July2011.pdf.

[Blau 14] Patrick Blau. *Ariane 5 - VA216 Launch Updates.* 2014. http://www.spaceflight101.com/ariane-5-va216-launch-updates.html.

[Boedker 15] Boedker. *Polyethylene Specifications.* 2015. http://www.boedeker.com/polye_p.htm.

[Bradford 13] John Bradford & Dominic DePasquale. *Microsatellite Market Assessment.* 2013. http://www.sei.aero/eng/papers/uploads/archive/SpaceWorks_NanoMicrosat_Market_Feb2013.pdf.

[Committee 02] Inter-Agency Space Debris Coordination Committee. *IADC Space Debris Mitigation Guidelines.* pages 9–10, 2002. http://www.iadc-online.org/Documents/IADC-2002-01,%20IADC%20Space%20Debris%20Guidelines,%20Revision%201.pdf.

[Cornelisse 79] J.W. Cornelisse, H.F.R Schöyer & K.F. Wakker. Rocket propulsion and spaceflight dynamics. Pitman Publishing, 1979.

[Curtis 05] Howard D. Curtis. Orbital mechanics for engineering students. Elsevier Butterworth-Heinemann, 2005.

[Devices 14] Analog Devices. *ADIS16448.* 2014. http://www.analog.com/media/en/technical-documentation/data-sheets/ADIS16448.pdf.

[Encyclopedia 04] of World Biography Encyclopedia. *Konstantin Eduardovich Tsiolkovsky.* 2004. http://www.encyclopedia.com/topic/Konstantin_Eduardovich_Tsiolkovsky.aspx.

[Falugi 10] Paola Falugi, Eric Kerrigan & Eugene van Wyk. *ICLOCS User Manual.* 2010. http://www.ee.ic.ac.uk/ICLOCS/user_guide.pdf.

[Grewal 01] M.S. Grewal & A.P. Andrews. Kalman filtering - theory and practice using matlab. John Wiley and Sons, Inc., 2001.

[Jackson 01] Doug Jackson. *Space Shuttle Max-Q*. 2001. http://www.aerospaceweb.org/question/aerodynamics/q0025.shtml.

[JAXA 03] Japan Aerospace Exploration Agency JAXA. *Piggyback Payloads*. 2003. http://global.jaxa.jp/countdown/f15/overview/sub_payload_e.html.

[Khalil 02] Hassan K. Khalil. Nonlinear systems. Prentice Hall, 2002.

[Lethbridge 98] Cliff Lethbridge. *Space Shuttle Main Engines Fact Sheet*. 1998. http://spaceline.org/rocketsum/main-engines.html.

[Leveque 11] Daniel L. Oltrogge & Kyle Leveque. *An Evaluation of CubeSat Orbital Decay*. page 9, 2011. https://www.agi.com/downloads/resources/white-papers/20110808_SmallSat_CubeSat_Orbit_Lifetime_10.pdf.

[Maestro 13] Maestro. *GPS Receiver A2035-H User's Manual*. 2013. http://www.farnell.com/datasheets/1794015.pdf.

[Martinez-Sanchez 05] Manuel Martinez-Sanchez. *Lecture 32: Orbital Mechanics: Review, Staging*. 2005. http://ocw.mit.edu/courses/aeronautics-and-astronautics/16-512-rocket-propulsion-fall-2005/lecture-notes/lecture_32.pdf.

[NASA 76] NASA, NOAA & USAF. *U.S. Standard Atmosphere*. 1976. http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19770009539.pdf.

[NASA 09] NASA. *NASA CECE Test*. 2009. http://www.nasa.gov/home/hqnews/2009/jan/HQ_09-005_Cryo_engine_test.html.

[NASA 14] NASA. *Rocket Control*. 2014. http://exploration.grc.nasa.gov/education/rocket/rktcontrl.html.

[NOAA 15] NOAA. *National Centers for Environmental Information*. 2015. http://www.ngdc.noaa.gov/geomag/.

[of State 15] U.S. Department of State. *The International Traffic in Arms Regulations (ITAR)*. 2015. https://www.pmddtc.state.gov/regulations_laws/itar.html.

[Ponomarenko 15] Alexander Ponomarenko. *Rocket Propulsion Analysis User Manual*. 2015. http://www.propulsion-analysis.com/downloads/2/docs/RPA_2_User_Manual.pdf.

[Ravtrack 15] Ravtrack. *Coordinating Committee for Multilateral Export Controls*. 2015. http://ravtrack.com/GPStracking/cocom-gps-tracking-limits/469/.

[Rodriguez 14] Ferdinand Rodriguez. *Encyclopædia Britanicca*. 2014. http://global.britannica.com/EBchecked/topic/463684/plastic#ref625153.

[SpaceX 09] SpaceX. *Falcon 9 Launch Vehicle Payload User's Guide*. 2009. http://www.spaceflightnow.com/falcon9/001/f9guide.pdf.

[Sutton 10] George P. Sutton & Oscar Biblarz. Rocket propulsion elements. John Wiley & Sons, Inc., 2010.

[Turner 01] Martin J. L. Turner. Rocket and spacecraft propulsion. Springer in association with Praxis Publishing, 2001.

[Wertz 78] James R. Wertz. Spacecraft attitude determination and control. D. Reidel Publishing Company, 1978.

[Yemets 08a] Vitaly Yemets. *Launch Vehichle with Combustible Polyethylene Case Gasification Chamber Design Basis.* pages 32–38, 2008.

[Yemets 08b] Vitaly Yemets, Fedir Sanin, Yevgen Dzhur, Mykola Masliany, Oleg Kostritsyn & Grygory Minteev. *Single-stage small satellite launcher with combustible tank of polyethylene.* pages 28–32, 2008.

[Zak 13] Anatoly Zak. *Sputnik Rocket.* 2013. `http://www.russianspaceweb.com/sputnik_lv.html`.

List of Corrections