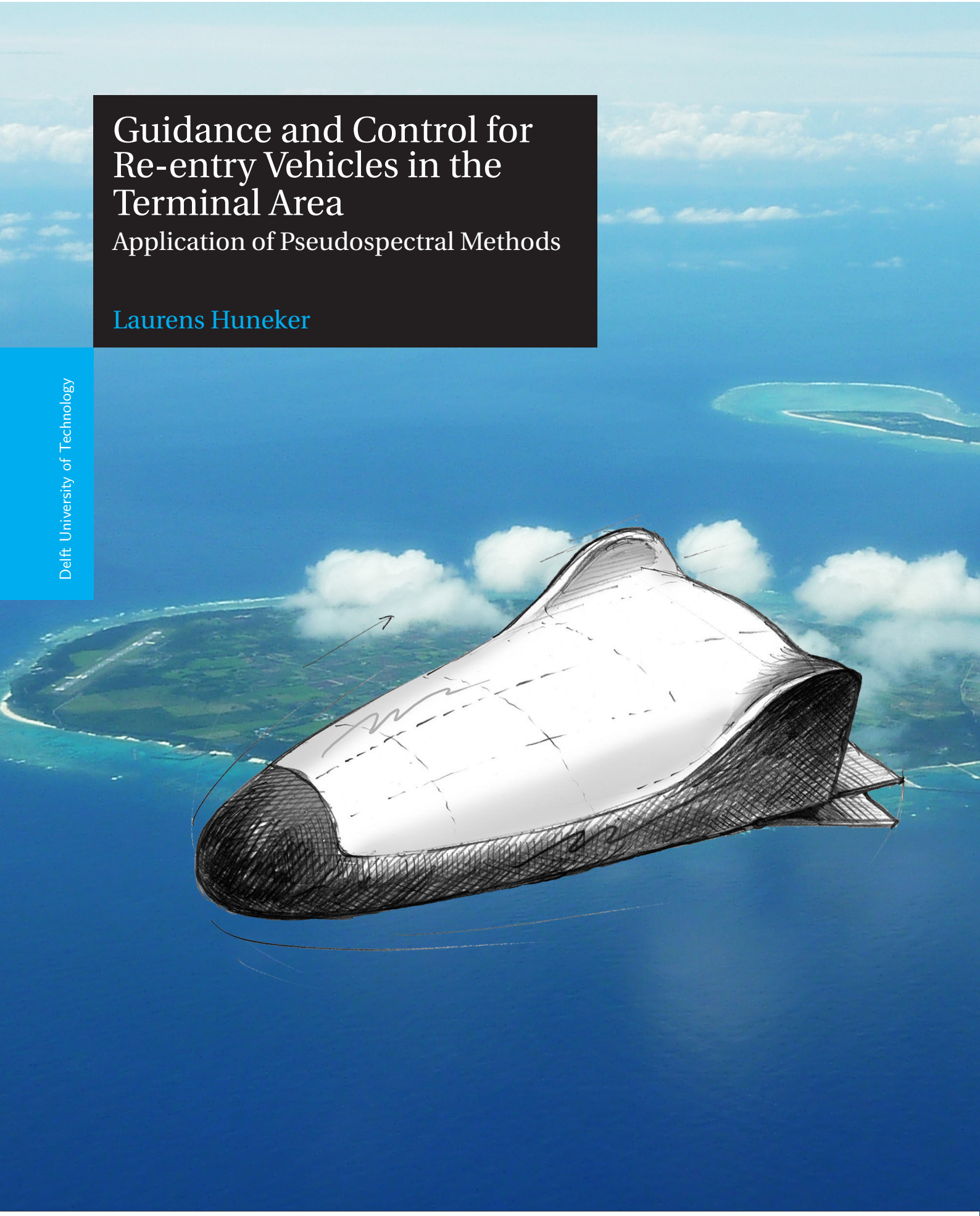
Guidance and Control for Re-entry Vehicles in the Terminal Area

Application of Pseudospectral Methods

Laurens Huneker

Delft University of Technology



TU Delft Delft University of Technology

Challenge the future

Guidance and Control for Re-entry Vehicles in the Terminal Area

Application of Pseudospectral Methods

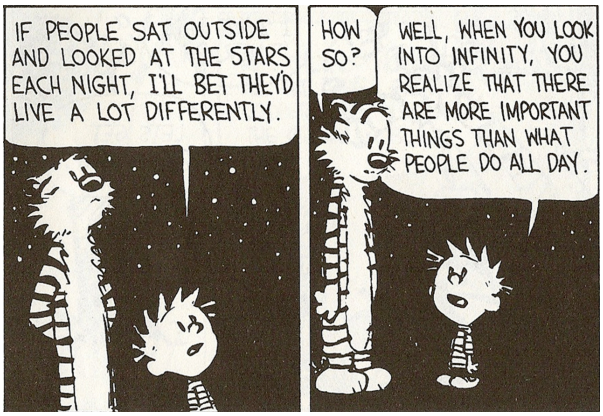
by

L.J. Huneker

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Thursday August 25, 2016 at 13:30.

Student number:	1314998	
Project duration:	May 1, 2015 – August 25, 2016	
Thesis committee:	Prof. dr. ir. P.N.A.M. Visser,	Delft Technical University
	Dr. Ir. E. Mooij,	Delft Technical University, daily supervisor
	Dr.ir. H.G. de Visser,	Delft Technical University

TU Delft Delft University of Technology



Bill Watterson ©

Frontpage:
A concept drawing of the IXV [Image source: NASA] flying over an island [artist unknown]

Preface

I had just become an adult when I came to the faculty of Aerospace Engineering to start my student life, still enthusiastic about airplanes. I discovered after a few years that airplanes were not that special and that I should aim higher, into space. This thesis is the result of all my years that I spent in the aerospace faculty in Delft.

My thanks goes out to Erwin Mooij who continued to support me during the long period that he supervised me, for being critical and keeping me on the right track. This report would not have this form if it wasn't for him. I was introduced to Pseudospectral methods by Marco Sagliano, the person that introduced me to this school of pseudospectral methods, of course I would like to thank him as well. Almost all of the simulations were run in C++, the amount of support I got from various persons to be able to compile everything was wonderful, a lot of time was spent useful with other parts of this thesis thanks to them.

This report would also not have been possible without the support of my parents, who never lost their faith in the fact that I would someday graduate, my sincerest gratitude For over a year now I have been working at the 9^{th} floor in room 9.06, meeting many new people as enthusiastic as I was about Space Exploration, making the endeavour that is called graduation far more enjoyable.

I wish you, the reader, the best of luck with understanding the intricacies of this report.

Laurens Huneker
August 16, 2016

Abstract

One part of spaceflight technology that has always been in demand is to go to space using a vehicle that can be re-used without much additional costs. It is roughly five years ago that the last Space Shuttle mission took place, and replacements are currently in development. Yet, much of the available literature is still based on architectures created during the development of the Space Shuttle. The goal of this thesis is to expand the current knowledge with modern computational methods.

This thesis demonstrates if pseudospectral methods can be used as the basis of a guidance and control architecture in the terminal area. In the past, these architectures were based by decoupling the longitudinal and lateral dynamics, using the energy available to calculate the current state without knowing what is to come, and calculating many variables and possible scenarios before the flight has even commenced. The architecture presented here does not require this, and it demonstrates an integrated package that can do this on the fly, by deploying a type of non-linear optimization method called pseudospectral methods.

The X-38 is chosen as the reference vehicle due to the compatibility of the aerodynamic data set with non-linear optimizers. A 3-DOF simulation is created to see what set of feasible trajectories can be created under nominal conditions. Subsequently, a single reference trajectory is used as a basis to determine if the guidance subsystem is able to cope with a wide variety of initial conditions. The guidance system proved to be quite robust. A set of winds and gusts have been simulated and it would return to the desired state as long as there is an approximate predictive ability. The research is later continued by extending to a 6-DOF simulation. A control subsystem based on LQR is implemented with good results. A PSM implementation is introduced after and the results are promising, but the computation time is too high to be feasible. The upside is that a result is found without calculating the deflection of the trimmed state and gains, which was required with LQR. The conclusion is that pseudospectral methods can be used as a robust real-time guidance method, but that using it as a control method is currently not feasible.

It is thus proven that pseudospectral method is able to deal without calculating predictive trajectories and gains as was done with the Space Shuttle. This reduces the complexity of re-entry problems by a substantial amount. The time it takes to solve problems with new conditions and spacecraft is reduced, which might encourage other people to continue in this direction. There is still much more possible than what is presented in this thesis.

Contents

Nomenclature

Abbreviations

Abbreviation	Description
AB	Adams-Bashfort
ALFLEX	Automatic Landing Flight Experiment
DARPA	Defense Advanced Research Project Agency
CIRA	COSPAR International Reference Atmosphere
DCM	Direction Cosine Matrix
DSMC	Discrete Sliding Mode Control
EAFB	Edwards AirForce Base
ESA	European Space Agency
fLGR	flipped Legendre Gauss Radau
FESTIP	Future European Space Transportation Investigation Programme
GNC	Guidance, Navigation and Control
GPS	Global Positioning System
GRAM	Global Reference Atmospheric Model
HAC	Heading Alignment Cylinder
HORUS	Hypersonic ORbital Upper Stage
HSFD	High Speed Flight Demonstration
HYFLEX	HYpersonic FLight Experiment
IMU	Inertial Measurement Unit
IPOPT	Interior Point OPTimizer
ISS	International Space Station
JAXA	Japanese Space Agency
KSC	Kennedy Space Center
LEO	Low Earth Orbit
LG	Legendre-Gauss
LGL	Legendre-Gauss-Lobatto
LGR	Legendre-Gauss-Radau
LIDAR	LIght Detection And Ranging
LOCV	Loss of Vehicle and Crew
MAKS	Multipurpose Aerospace System
NASA	National Aeronautics and Space Administration
NCEP	National Centers for Environmental Prediction
ODE45	Ordinary Differential Equations 45
OREX	Orbital Re-Entry EXperiment
ODSMC	Output-based Discrete Sliding Mode Control
PID	Proportional Integral Derivative
PRIME	Precision Recovery Including Manoeuvring re-Entry
PSM	PseudoSpectral Methods
RCS	Reaction Control System
RK	Runge-Kutta
RK4	Runge-Kutta 4^{th} order
RKF45	Runge-Kutta-Fehlberg 4(5)
SNOPT	Sparse Nonlinear OPTimizer
SSTO	Single Stage To Orbit
START	Spacecraft Technology And Re-entry Test
TAEM	Terminal Area Energy Management
TUDAT	TU Delft Astrodynamics Toolbox

UAV	Unmanned Aerial Vehicle
VAFB	Vandenberg Air Force Base

Roman symbols

Symbol	Unit	Description
A	-	State Matrix
B	-	Control Matrix
b_{ref}	m	Aerodynamic reference wing span
c_{ref}	m	Aerodynamic reference chord
C	-	Direction Cosine Matrix
C	-	Output Matrix
C_D	-	Aerodynamic drag coefficient
C_l	-	Aerodynamic roll coefficient
C_L	-	Aerodynamic lift coefficient
C_m	-	Aerodynamic pitch coefficient
C_n	-	Aerodynamic yaw coefficient
C_S	-	Aerodynamic sideforce coefficient
D	N	Drag
D	-	Direct Transmission Matrix
$e_{i,j}$	-	Direction cosines
F	N	Forces
g	m/s^2	Gravitational acceleration
g_0	m/s^2	Gravitational acceleration at sea level
g_r	m/s^2	Radial Gravitational acceleration
g_δ	m/s^2	Latitudinal acceleration
ge	n.a.	Global error
h	m	Height
h	s	Stepsize
I	kgm^2	Principle moment of inertia
J	-	Cost function
J_2	-	Second degree harmonic of the gravitational field
K_p	v.a.	Proportional gain
K_i	v.a.	Integral gain
K_d	v.a.	Derivative gain
K	v.a.	Gain matrix
le	v.a.	Local error
lepus	v.a.	Local error per unit step
L	N	Lift
v L	-	Lagrange function
L_{z_i}	K/m	Thermal lapse rate
$\mathcal{L}$	Nm	Rolling moment
m	kg	Vehicle Mass
M	-	Mach number
M	mol	Mean Molecular Weight
M_T	Nm	Moment generated by thrusters
M_p	-	Maximum percentage overshoot
M	Nm	Moment
$\mathcal{M}$	Nm	Pitching moment
$\mathcal{N}$	Nm	Yawing moment
u	v.a.	Control Vector
p	N/m^2	Pressure
p	rad/s	Roll rate
P	s	Period
P	-	Integrated Virtual Power
q	N/m^2	Dynamic Pressure
q	-	Quaternions
q	rad/s	Pitch rate
Q	v.a.	Maximum allowable deviation for the state vector

r	rad/s	yaw rate
$\mathbf{r_{cm}}$	m	Distance from centre of mass in the body frame
R	m	Distance from centre of Earth-reference frame
R_0	m	Radius of the Earth at an inclination of 45°
R_e	m	Equatorial radius of the Earth
R	J K^{-1} mol^{-1}	Universal gas constant
$\mathbf{R}$	n.a.	Maximum allowable deviation for the control vector
S	N	Side-force
S_{ref}	m^2	Aerodynamic reference area
t_d	s	Delay time
t_p	s	Peak time
t_r	s	Rise time
t_s	s	Settling time
T	K	Temperature
T	s	Amplitude
V	m/s	Relative velocity
$\mathbf{V}$	-	Lyapunov Function
W	-	Weight of the nodes
$\mathbf{x}$	v.a.	State vector
X	-	Reference Frame X-Axis
Y	-	Reference Frame Y-Axis
z	m	Geopotential height
Z	-	Reference Frame Z-Axis

Greek symbols

Symbol	Units	Description
α	rad	Angle of attack
α_c	rad	Commanded angle of attack
α_e	rad	Deviation of commanded angle of attack
α^*	rad	Open-loop reference angle of attack
β	rad	Side slip angle
β	1/m	Scale height
γ	rad	Flight path angle
δ	v.a.	Relative residual
δ	rad	Deflection angle
δ	rad	Latitude
δ_a	rad	Aileron deflection angle
δ_b	rad	Body-flap deflection angle
δ_e	rad	Elevator deflection angle
Δ	v.a.	Residual
ϵ	-	Machina accuracy
ζ	-	Dampening Ratio
θ	rad	Pitch angle
μ	m^3s^{-2}	Standard gravitational parameter of the Earth
μ	-	Eigenvector
ρ	kg/m^3	Atmospheric density
ρ_0	kg/m^3	Atmospheric density at sea level
σ	rad	Bank angle
σ_c	rad	Commanded bank angle
σ_e	rad	Deviation of commanded bank
σ	v.a.	Switching function
τ	rad	Longitude
τ	-	Independent variable
ϕ	rad	Roll angle
ϕ	-	Mayer function
ϕ	-	Notation of numerical methods
ϕ	-	Lagrange interpolating polynomial
Φ	n.a.	Linear reaching law
χ	rad	Flight heading angle
ψ	rad	yaw angle
ω	rad/s	Angular rate
ω_E	rad/s	Rotational velocity of the Earth
ω_n	rad/s	Natural frequency
Ω	rad/s	Angular velocity

Subscripts

Symbol	Description
0	Clean configuration (for aerodynamics)
e,l	Left elevon
e,r	Right elevon
r,l	Left rudder
r,r	Right rudder

Reference Frames

Symbol	Description
$\mathbb{F}_V$	Vertical Reference Frame
$\mathbb{F}_B$	Body Reference Frame
$\mathbb{F}_{TG}$	Trajectory Reference Frame (groundspeed based)
$\mathbb{F}_{TA}$	Trajectory Reference Frame (airspeed based)
$\mathbb{F}_{AG}$	Aerodynamic Reference Frame (groundspeed based)
$\mathbb{F}_{AA}$	Aerodynamic Reference Frame (airspeed based)
$\mathbb{F}_W$	Wind Reference Frame
$\mathbb{F}_R$	Runway Reference Frame

1 Introduction

Space has become a lot more crowded with satellites ever since the first one launched. Most satellites have been brought into orbit using conventional rocket launchers such as the Russian Soyuz rocket. A different method of bringing satellites into space is using winged re-entry vehicles (RV-W). The most famous of which was the Space Shuttle that has flown 135 times from 1981 until 2011. Recent technological advances have once again sparked the enthusiasm of engineers and policy-makers to re-introduce the concept of re-usable RV-W. There are currently two vehicles existing of interest. The first is the X-37 designed by Boeing and DARPA which is launched atop of a rocket and can perform autonomous landings several months later. The second is the Dream Chaser developed by the Sierra Nevada Company that has received the order in January 2016 to do six resupply missions to the International Space Station.

An RV-W is designed to go into space either horizontally or vertically, however, they are designed to land on a runway similar to a regular aircraft. The difference between the RV-W and an aircraft is that an RV-W is optimised to fly at extreme velocities which changes the aerodynamic properties, the L/D ratio is far lower when the spacecraft gets closer to the ground. Another important distinction is that an RV-W has no propulsion, this means that there is no second chance when landing.

An RV-W that is in orbit around the Earth that has a velocity of 8000 m/s has a specific enthalpy of 18 MJ/kg. (Regan, 1984) A quick calculation learns that the Space Shuttle that returns to the surface has an energy amount of 72 TJ, it is hard to properly imagine how much that is. To compare, the first atomic bomb ever used, Little Boy, had a blast yield of 67 TJ. It is therefore very important that the energy present in the RV-W is gradually diminished, or it will burn up in the atmosphere. The de-orbit of the Shuttle starts above the Indian ocean and the primary landing runway was the Kennedy Space Centre in Florida, with the possibility to defer to other runways. This entire re-entry from de-orbiting until landing is divided into three parts, namely (Cox, 1971)(Cooke, 1982).

- *Hypersonic atmospheric re-entry phase*
  The hypersonic atmospheric re-entry phase starts at an altitude of approximately 120 kilometres. The velocity ranges from Mach 24 to Mach 2.5. Being aware of the thermal loads during this period are paramount. The heat flow per unit area is given by $\frac{1}{2}\rho V^3 C_D$, the fact that the velocity (V) is cubed means that the thermal loads are at its largest during this part of the flight, getting as high as 1500°C (Cooke, 1982). The temperature is reduced by lowering the velocity as much as possible, and stay at a higher altitude with lower atmospheric density. This is done by flying with a high angle of attack, creating a large surface area, and thus more lift. This phase ends at an altitude of 25 km which is the terminal area entry point.
- *Terminal area management phase*
  The terminal area management phase starts at an altitude of around 25 km with a velocity of Mach 2.5. The vehicle makes a series of turns until the amount of energy is present that is suitable for landing the plane. It then flies towards the heading alignment cylinder (HAC), there it makes the final turn until it is aligned towards the runway.
- *Approach and landing phase*
  The approach and landing phase begins when the vehicle is aligned towards the runway after it has

exited the heading alignment cylinder. The distance from the runway depends on the location of the HAC to the runway, the Space Shuttle's HAC was on average 12 km away, with an altitude of 4.5 km. This part guides the RV/W to perform a proper landing on the runway.

Each of these segments require their own approach. The focus of this thesis will be on the terminal area, specifically the guidance, navigation, and control (GNC) subsystem. A major objective of future RV/W include significant improvements in operation costs, reliability, and vehicle safety (Kluever and Horneman, 2005). Both the market and governments currently demand cheaper transportation into space, and a winged re-entry vehicle could fulfil this request. Governments also demand that the citizens have a positive view of the space program, and that it is maintained. The increased interest in government spending requires missions to have a high success rate to avoid additional spending, safety is therefore a key issue. The probability for a catastrophic failure for the Space Shuttle was 1 in 500 missions. It is possible to reduce this to 1 in $10\,000$ missions using advanced guidance and control methods (Hanson, 2002). The guidance system is responsible for determining what trajectory needs to be followed, how to get from A to B. The navigation system's main job is to accurately determine what the location is at the current moment. This subsystem is not a part of this research, it is therefore assumed that the location and velocity is known at any time with infinitesimal precision. The control subsystem tries to follow the trajectory given by the guidance system by manoeuvring the RV-W to change the attitude.

There are several challenges when it comes to planning the trajectory and controlling the spacecraft during re-entry (Bollino, 2006). The primary challenge is dealing with the changing atmospheric conditions such as temperature and density during the descent. Different Mach ranges from hypersonic, supersonic, transonic to subsonic, each has different aerodynamic coefficients increasing the non-linearity of the dynamics of the system. Secondary challenges are that the high pressure, temperature, and g-loads that can cause difficulties for the thermal protection system and structural integrity if these conditions become too high, and the control surfaces are mechanically limited to move a certain degree per second and that these are not instantaneous at the required deflection angle. The goal is thus to create a robust system that can handle any vehicle and correct any disturbances along the way in the terminal area. A problem here is that the body of public knowledge that is flight-tested concerning this subject still dates from the Space Shuttle era, or is based upon the methods developed during the Shuttle Program (Mooij, 2013)(De Ridder and Mooij, 2011). Yet, as can be seen, there is interest in this subject to update this body of knowledge with the new mathematical methods and computers to this era.

A well known method for creating guidance and control solutions is optimal control theory (Ogata, 2010). One of the advantages of optimal control theory is that it is no longer required to manually adjust the control system for each phase of the flight which saves a lot of time during the design process. Another advantage is that any unforeseen circumstances such as larger gusts can be solved by automatically adapting the trajectory to the newer optimal one, thus increasing both the reliability and safety of the crew and spacecraft. Optimal control theory can take these challenges into account when calculating the optimal trajectory on the fly.

One class of optimal control that has become more popular recently is that of pseudospectral methods (PSM), a method that was developed to solve partial differential equations. Recent advances in computational power, algorithms, and theory are the drivers of the increased popularity (Qi Gong et al., 2007)(Garg, 2011). PSM was mainly developed in the 1970s to study fluid dynamics and meteorology. However, it became interesting for solving optimal control problems in the 1990s. One recent usage of PSM happened on November 5, 2006 and March 3, 2007 (Qi Gong et al., 2007). Two attitude manoeuvres were executed on the ISS called the Zero-prop Maneouvre. The first rotation induced an attitude change of 90 degrees in two hours, the second one was 180 degrees in just three hours. Previous large attitude manoeuvres were performed using thrusters, which added constraints to the structure caused by the increased load and pollution of the solar panels. The shortest kinematic manoeuvre between two attitude points is by following an eigen-axis path. This path can only be followed by countering the non-linear dynamics of the ISS by using the thrusters. The fuel usage could be reduced to zero by following a kinematically longer path and by taking advantage of the non-linear dynamics. The new manoeuvre used the momentum storage devices on-board to create a rotation, without the usage of propellant, saving NASA approximately 1.5 million dollars. A similar feat was performed by the TRACE space telescope (Karpenko et al., 2012). It was a minimum-time re-orientation manoeuvre which was successfully executed during the summer of 2010.

The first one of its kind. Another example involves the off-line trajectory calculations of SHEFEX by Sagliano and Theil (2013), which is a sub-orbital test vehicle.

Unfortunately, not much research has been done on applying PSM to RV-W in the terminal area. The exception is Bollino (2006), where a re-entry of the X-33 was simulated with success. However, improved algorithms and computational speeds could improve the result. It is the aim of this study to extend this theoretical framework on PSM, and to accelerate the study of new vehicles that are inevitably going to be build in the nearby future.

This study is divided into several research questions to find out whether pseudospectral Methods can be used with the latest numerical methods. The primary question is

> *Can pseudospectral methods be used as a robust, real-time guidance and control method for re-entry vehicles in the terminal area.*

This question is answered with the help of several sub-questions. The first sub-question can be considered to be the principal one, it states whether this method even can be used:

1. Is it possible to use pseudospectral methods as a guidance and control method?

A keyword in the research question is that the method should be robust. A robust system is capable of dealing with different initial conditions and atmospheric conditions. The set of initial conditions stated here consist of numerous points placed strategically around the runway, these could be considered entry positions during the re-entry. Different atmospheric conditions include changes in density and wind, Therefore, the next two sub-questions are,

2. For what set of initial conditions is it possible to create a guidance and control solution?
3. How robust is the designed guidance and control method during different atmospheric condition?

which is followed by another question that is also indirectly already posed in the primary question:

4. Can the designed guidance and control method be executed in real-time?

It is possible to give an answer to the primary question by answering the four sub-questions. A simulation is created in order to answer them. Figure 1.1 gives a top-level overview of the guidance and control diagram used in the simulation. The mission manager is located at the start of the simulation, it calculates the trajectory beforehand with the given initial condition and final point. The final point is just before the runway, at the end of the terminal area. This trajectory is passed to the guidance system, which then passes the required attitude angles to the attitude controller, which subsequently moves the actuators of the spacecraft. The dynamic equations are then followed which leads to a new state. This state is either passed to the attitude controller or guidance system depending on the frequency of the loops.

The simulation is divided into two loops, the inner and outer. Both have their own frequency. The reason for this is that the time-scaling separation of the control subsystem and guidance subsystem is inherently different. The deviation of the attitude of the spacecraft is subject to external influences and small changes can rapidly increase this deviation. The guidance subsystem can deal with differences in the trajectory more gradually, the non-linear effects of the dynamics of the system have a smaller impact on the trajectory.

The report starts off with discussing the current heritage in Chapter 2 to give a brief presentation of previous work and RV/W. It is then followed by an introduction to flight mechanics in Chapter 3, this chapter explains the core of the simulator with respect to how and where it flies. Much of the work in this thesis is based on mathematics which needs to be understood before it is possible to introduce the chapters about the guidance and control methods, Chapter 5 explains the most important mathematical procedures followed. The core of this thesis, which is PSM, is explained in Chapter 4, included by two examples. The RV/W used during this thesis is chosen in Chapter 7. How the software is written and verified is in Chapter 6. It is now possible to answer the research questions. The first one is answered in Chapter 8, where it is demonstrated how the guidance system responds to certain trajectories when using PSM. The extension to the 6-DOF system is presented in Chapter 9, where the PSM-method is compared to the LQR control method. The research questions posed above are finally answered in Chapter 10, where in addition to that there are also recommendations for readers who wish to continue with this work.

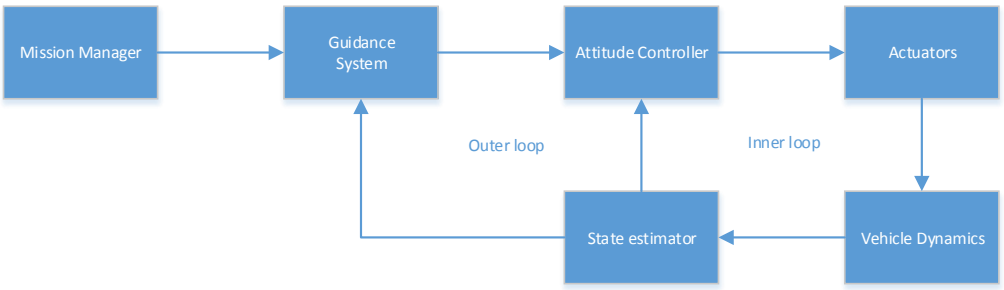


Figure 1.1: Top-level diagram of a simulator.

2

Mission Heritage

This chapter will describe the past work that has been done in our fields of interest. Section 2.1 starts with describing the different vehicles that have been thought of and created in the past, followed by some example guidance models that these vehicles used in Section 2.2.

2.1. Vehicle Heritage

re-entry vehicles have been in development for almost sixty years, and most of them never left the drawing board. Two types of reusable re-entry vehicles exist (Weiland, 2014)(Bertin and Cummings, 2003). The first one is the lifting body which generates lift with its body, wings are non-existent or very small. The second type is the winged re-entry vehicle (RV-W) which has the ability to produce lift with the body and wings like a conventional aircraft. Many nations that worked on projects during this period acted alone or together when trying to develop these vehicles. such as the United States, Soviet Union, Europe, and Japan. These projects shall be mentioned briefly from here on, since it is very important to know the foundations of today's knowledge. It is impossible to mention all the projects that were initiated, but a large part is certainly covered, more than enough to get a proper overview of the existing vehicles.

The United States has developed many programs focusing on re-entry vehicles (Jenkins et al., 2003)(da Costa, 2003). The first program was the X-20 Dyna-Soar which was a continuation of concepts developed by the German Eugen Sänger in 1928. The X-20 was continued by the Bell company with the intention of being a orbital bomber during the fifties. The plane was never built, however, the numerous tests and developments contributed a lot to the field of high-speed flight. Two other lifting bodies were the X-23 and X-24 created by the Martin Marietta company during the mid-sixties. Both were a part of the PRIME project (Precision Recovery Including Manoeuvring re-Entry) which was a part of the larger START program (Spacecraft Technology And Re-entry Test). The X-23 focused on the very high-speed re-entry phase whereas the X-24 focused on the low-speed landing characteristics. The X-23 main testing purposes where the manoeuvrability during the supersonic- and hypersonic-phase and was designed to be intercepted mid-air by parachute. Out of the four that were built only three were launcher, out of those three only one was recovered. Only two X-24s were constructed, these two, however, proved that precision landings on runways could be made with lifting bodies, the tests were so successful that the X-24B was created to test more advanced designs. The X-23 and X-24 proved that the concept of the well-known Space Shuttle was feasible.

The Space Shuttle is the best known RV-W. For good reason, the project that had its first launch in April 1981 was followed by 134 more missions (Treadwell, 2010). The fleet consisted of five Shuttles named Columbia, Endeavour, Challenger, Atlantis and Discovery that were fully functional, and the Shuttle called the Enterprise was merely used for suborbital testing purposes. Out of the 135 missions in total there were 2 catastrophes that caused a so called loss of vehicle and crew (LOCV). The first took place in 1986 in which the Challenger exploded, 73 seconds after launch, caused by a faulty O-ring. The second LOCV happened during the re-entry of the Columbia on 1 February 2003 caused by a damaged heat shield on the left wing. The Columbia disintegration resulted into being the beginning of the end for the Space Shuttle Program. The

Figure 2.1: The X-38 (image source: NASA)

final mission, STS-135, launched on July 8, 2011 was to the International Space Station [1]. The Space Shuttle was discontinued due to the expensive nature of the program.

It became clear that follow-ups of the Space Shuttle were required, and the research continued. The X-30 project led by DARPA (Defense Advanced Research Project Agency) was started with the intention to assess the feasibility for a single-stage-to-orbit (SSTO) vehicle that could take-off and land horizontally (Jenkins et al., 2003). It should be able to reach LEO or reach Tokyo within two hours. It was cancelled in 1985 just after three years due to an extreme rise in costs and technological difficulties. The X-37 and X-40 started in 1999. The first is designed to be a fully autonomous space-plane that could stay in space for long durations and then land without being controlled, the second, the X-40, existed merely as a testing platform for the X-37. The plane was originally designed to be carried within the space Shuttle's maintenance bay, this was however no longer possible after the Space Shuttle program was cancelled and it is therefore currently designed to be launched atop of a rocket. The X-37 was transferred from NASA to DARPA in 2004 and it became classified shortly thereafter, but not before dividing the project into two different sections. The X-37A is responsible for the approach and landing phase whereas the X-37B is the orbital vehicle. Two launches with the X-37B were performed in 2010 and 2011 and both made successful touchdowns at Vandenberg Airforce base. Four missions are currently confirmed. Not much information is available due to the fact that this project is highly classified. Therefore, no use can be made of this spacecraft during this research [2].

Another project that was designed roughly simultaneously as the X-40 was intended to be a crew rescue vehicle for the ISS, it was called the X-38. The X-38 project started in 1995 to provide the basis for a crew rescue vehicle replacing the Soyuz at the ISS and possibly the Space shuttle. The following features were required (Weiland, 2014)

1. Accurate and soft landing to allow the transportation of injured crew members.
2. Load factor minimisation
3. Sufficient cross range capability
4. Autonomous return from orbit

Three different versions were developed, all based on the X24-A, but scaled up to increase the volume to fit the passengers and cargo. The lifting body shape of the X-24 has limited stability during the landing, and the minimum landing speed was around 128 meters per second. The L/D ratio during the subsonic phase is of order 2, which is below the required value. The X-38 program solved this by using a parafoil during the descent, where winged re-entry vehicles would perform a landing without the parafoil. The L/D ratio during

[1] http://www.thespacereview.com/article/1887/1
[2] http://www.boeing.com/assets/pdf/defense-space/ic/sis/x37b_otv/bkgd/_x37/_0311.pdf

Figure 2.2: The Spiral (source: www.buran.ru)

the supersonic phase is of order 1.4, which is sufficient during this phase.

The first two versions were developed to test the parafoil and flight control system, with the numerical designators V131 and V132. The V131 was designed to fly for four seconds in a trimmed position before the deployment of the parafoil was tested, two flight tests were performed. The follow-up V132 did contain a full flight control system and the vehicle flew for 30 seconds after being dropped from a height of approximately 11 kilometres before the parafoil was used.

The project was cancelled in 2002, but not before an extensive database of the V201 was created using wind-tunnel experiments, Euler and Navier-Stokes equations, and the data from the V131 drop tests. it got cancelled in 2002 before a space-rated vehicle got assembled. The generated aerodynamic data has been used many times in different papers (Wacker, R. and Munday, S. and Merkle, 2001).

All these projects have made America the leading authority with respect to the knowledge of re-entry vehicles, but luckily, other nations have been active as well. One of the more noticeable competitors of the United States was of course the Soviet Union during the cold war. Initial development of the space-plane vehicles were designed with military requirements in mind, that was no different from the Americans who developed the Dyna-Soar with orbital nuclear strikes in mind. A very similar program that the Soviets had was called the Tupolev space-plane project, which had planes named the TU-136 or Zvezda ("Star"), the project got cancelled in 1963. The interest in space-planes continued and a new one was developed, which was called the Spiral (see Figure 2.2). The early design was finished in 1966 and the result was in total a 115 ton system consisting of a hypersonic boost aircraft, an orbital space plane and a two stage rocket to launch the space plane into orbit. The program faced many hindrances, mainly technological ones, but the program continued nonetheless for a long time, it is a mystery why and when it was cancelled. In fact, not much was known during that period about this project and some disclosure was finally given during the previous decade. What, however, is known, is that the Spiral project contributed a lot to another project which is more known throughout the space community, the Buran, which started in 1976 (Figure 2.3). This space plane was the Russian response to the Space Shuttle [3]. Many prototypes were built and in total 25 flight tests were performed before the final version was built. The Buran orbiter completed two full orbits before landing automatically, no-one was on-board. The data available about this operation is very limited.

Two other projects that were started by the Russians were the MAKS and Oryol, both however never left the drawing board and were cancelled nearly simultaneous with the fall of the Soviet Union (Hendrickx and Vis, 2007). The MAKS was supposed to be a follow up of the Spiral space plane, including the launch system. The other space plane, the Oryol, was designed to be a VTOL system. It is clear that the Russians tried to develop space planes as well, it is, however, quite noticeable that the need for them to develop these systems was not prevalent over the more familiar rockets such as the Soyuz that already proved their success with the requirement to gain access to space.

The United States and the Russian Federation were not the only nations involved, both Europe and Japan also tried or are trying to develop winged re-entry vehicles, albeit to a lesser extent. The HORUS (Hyper-

[3] www.buran.ru

Figure 2.3: The Buran, the Russian counter to the Space Shuttle (source: www.buran.ru)

sonic ORbital Upper Stage) is a German designed spaceplane that started development in the 1980s as an upper stage of the Ariane 5, it would later on evolve to use the Sänger first stage (Müller, 1988). The first phase is a hypersonic air-breathing plane capable of achieving a cruise velocity whilst carrying the second stage, HORUS, at an altitude of 120 km. The second stage has a rocket booster that allows the plane to deliver cargo and passengers in LEO. The program was cancelled in 1994 due to budget constraints. ESA started in 1985 with its first design of a winged vehicle called the Hermes Space Plane, which was designed to be be put on top of the Ariane 5, the program was cancelled in 1992. It followed with several studies such as FESTIP (Future European Space Transportation Investigation Program) (Ackermann et al., 2005)(Bayer, 2010). FESTIP was established in 1994 and ran until 1998 (Dujarric, 1999). The program was initiated to be able to compete with the Americans who were also aiming for reusable launch vehicles to drastically cut the costs. The top-level system requirement was that the cost of getting objects into orbit was to be reduced. This means that all options were open and re-usability is one of them. Some of the concepts are therefore semi-reusable. The performance requirements were not abundant, the only two demands were that the concept should be able to bring 7 tons of payload into LEO and 2 tons of payload into a Low Earth Polar Orbit. Logically, the launch base of all the concepts was to be Kourou, Europe's launch base in French Guiana. The idea was that the launchers should be ready somewhere around 2017-2020. In the end eight different concepts went into the detailed design phase, from vertical launches on top of a rocket to horizontal launches with some help from sleds. A few of the concepts warrant special attention.

The first concept FSSC-1 is a winged body SSTO vehicle. The vehicle is designed 59 metres long and it has a lift-off mass of 900 Mg. Eight engines are present to power the vehicle delivering a total of 144 000 kN, Four of these are booster engines and the other four are sustainer engines. One of the main problems was the control and stability of the vehicle during re-entry and landing. The main reason for this is the c.o.g position, which was at least 2 m behind a position that can be handled by the control system. A similar concept, FSSC-4, was also designed not to launch vertically, but horizontally by a sled, resulting in a weight of around 600 Mg. Both concepts had high requirements for the engines, thermal and mechanical, too high to be feasible.

The third concept to be discussed, the FSSC-12, is based on the Sänger configuration. The first stage is an airplane with a trapezoidal-winged configuration, the second stage was based on the HORUS (Hypersonic ORbital Upper Stage) design. The HORUS is a German designed spaceplane that started development in the 1980s as an upper stage of the Ariane 5, it would later on evolve to use the Sänger concept (Müller, 1988). The first phase is a hypersonic air-breathing plane capable of achieving a cruise velocity of Mach 4 whilst carrying the second stage, HORUS. The second stage has a rocket booster that allows the plane to deliver cargo and passengers in LEO. The program was cancelled in 1994 due to budget constraints. The idea of this concept is that the plane flies around Mach 5-7 at 25 km where the second stage is released.

Another concept, FSSC-15, evolved into the sub-orbital Hopper. This concept had the idea that the vehicle should launch horizontally over a 4 km long magnetic rails and land horizontally again (HTHL). The plane would launch at the Guiana Space Center up to an altitude of 100 km after which it would coast to a height

of 150 km. The height would then be lowered in a controllable fashion after the cargo was released. The horizontal landing was planned at a downrange landing site (DRLS) approximately 30 minutes after take-off, these sites were the ascension islands, the Azores or St. Pierre et Miquelon. The Hopper would then be transported back to French Guiana.

The Japanese Space Agency (NASDA, now called JAXA) had a program called the Hope-X, a project that was postponed in 2004 and is currently still on hold. The Hope-X was planned to be launched atop of the H-II rocket and was the continuation of multiple experiments called OREX, HYFLEX, ALFLEX and HSFD. Both ESA and JAXA never fully realized a winged re-entry vehicle, although the research that was done can still be useful for future missions.

It is clear by now that multiple attempts have been made to create winged re-entry vehicles and that only a few of them reached a phase of development sufficient for later research. Most notably, the Space Shuttle, HORUS, and X-38, which all have aerodynamic data publicly available.

2.2. Heritage Guidance Methods in the Terminal Area

Different guidance concepts have been developed for the terminal area over the years for various spacecraft. The goal of guidance subsystem is to guide the spacecraft from a certain initial position to a final position without for example, having excessive g-loads or heat flux, by changing the magnitude and orientation of the aerodynamic force. This is done by varying the aerodynamic force components lift, side, and drag-force. The main variables to calculate the magnitude of these components are the velocity, atmospheric density, aerodynamic coefficients, and spacecraft configuration. The first two are dependent on the state of the aerodynamic coefficients, and the spacecraft configuration can be considered a constant. The aerodynamic coefficients can be changed by varying the angle of attack and Mach number. The angle of attack can be varied by changing the attitude by pitching the nose up or down. This movement only covers the longitudinal motion, that is the motion when a straight flight is performed. The direction of the force components are changed by varying the bank angle, this is changed by moving the wings up or down. The changes in the angle of attack and bank angle are done by moving the actuators, such as the aerodynamic control surfaces. How these should be moved is calculated by the control subsystem.

There are various methods for guidance to calculate the desired attitude. The three main methods are (Mooij, 2013)

- *Pre-flight calculated trajectory*
  This method assumes that it is possible to use tracking to stay close to a trajectory that has small deviations from a given nominal trajectory. A set of initial trajectories are calculated off-line beforehand for a large range of initial conditions. A certain trajectory is chosen at the beginning that matches the initial condition of the nominal and current trajectory closely. This method has trouble dealing with unforeseen circumstances, such as damaged components.

- *On-board calculated trajectory*
  A trajectory is calculated at the initial point of the flight with the state at that time. A new trajectory is calculated if the control subsystem cannot keep up such the state has deviated so far from the ideal state that it is more efficient to follow a new trajectory.

- *Prediction guidance*
  A guidance method of this type does not calculate a reference trajectory or is dependent on trajectories calculated beforehand.It can be calculated implicitly by the amount of kinetic and potential energy that is available to calculate the required state in order to reach, for example, the runway. The guidance control variables can be found using analytical formulae (Helmersson, 1988).

There are a few notable vehicles that used these concepts. The first that is to be discussed is the Space Shuttle, which used a combination of pre-flight calculated trajectories and prediction guidance using energy control, can be found in Section 2.2.1. The second is the HORUS which has methods designed for nominal reference trajectories that are calculated pre-flight (De Ridder, 2009)(Mooij, 1998).

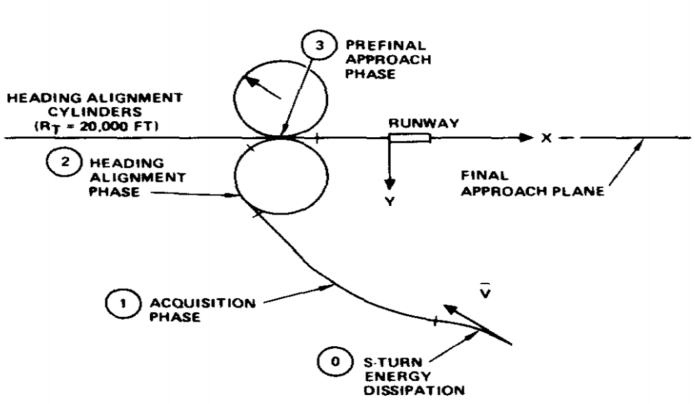


Figure 2.4: Terminal area guidance phases as used in the Shuttle program (Ehlers and Kraemer, 1977).

2.2.1. Space Shuttle

It is far from a proper gesture to not discuss the guidance heritage created by the Space Shuttle before discussing other vehicles. Even now, many re-entry models created draw their inspiration from the designs created by NASA for the Space Shuttle (Kafer, 1982)(Lu and Hanson, 1998)(Schierman et al., 2004).

The terminal area is divided into four phases as seen in Figure 2.4 (Ehlers and Kraemer, 1977). The first phase is called the S-phase and is used if there is an excess amount of energy in the system. The orbiter is turned away from the tangent of the HAC whilst a maximum energy dissipation rate is achieved by flying in the maximum dynamic pressure regime and by using maximum speedbrakes. The acquisition phase is started when enough energy has dissipated by flying along the tangent of the HAC. The energy is controlled by changing the dynamic pressure and changing the speedbrake in the subsonic speed range. The third phase is entered once the HAC is reached and the Orbiter is controlled to directionally track the HAC. The prefinal phase starts once the heading is within 20 degrees of the final approach plane. The lateral motion then makes sure that the Orbiter is aligned with the extended runway centerline and the speed brake is modulated to reach the desired 150 m/s velocity to initiate the autoland interface. The terminal area ends when this inferface is initiated.

The original method to find the desired attitude was mainly by using predictive guidance with energy control and dynamic pressure as the leading variables, which is called the energy-range guidance concept (Ehlers and Kraemer, 1977). The energy control is performed by achieving a certain energy state that is close to the nominal rate. The amount of energy is modified by calculating the derivative of the energy per unit mass over the distance from the runway.

$$\frac{d(E/W)}{dR} = -\frac{D}{W} \cdot \frac{1}{\cos\gamma} \tag{2.1}$$

The rate at which the energy dissipates is thus proportional to the amount of drag and inversely proportional to the flight-path angle. There are three ways to change the dissipation rate. The first one takes place during the S-turn in which the distance from the runway is increased. The distance from the runway is estimated by the ground-track predictor. The second method is by changing the dynamic pressure, and thus the drag. The drag itself can also be altered by varying the speed brakes, which is the third way. The dynamic pressure is determined by

$$\bar{q} = \frac{W\cos\gamma}{C_L S\cos\sigma} \tag{2.2}$$

The lift coefficient is calculated with, but not only with, the angle of attack. The reason that this method was used is that it is easier and more accurate to measure the dynamic pressure than the angle of attack. It also provides proper phugoid damping and good flight path control during turns.

2.2.2. BURAN

The re-entry system of the BURAN was capable of landing the RV/W without any assistance from the pilot with a system that uses Trajectory Tubes [4]. A trajectory tube has a certain width and height in which almost all (P=0.997) of the possible trajectories exist when different disturbances are taken into account. The tube gets smaller and smaller as the BURAN gets closer to the runway. There are three stages for a full-re-entry

- Descent Stage: altitude = 100-20 km
- Pre-Landing Manoeuvring Stage = 20-4 km
- Pre-Landing Approach and Landing Stage: altitude = 4-0 km

The Pre-Landing Manoeuvring Stage resembles the terminal area, in terms of height. Figure 2.5 shows the different Trajectory Tubes that were available during the first flight. There are two tasks in the pre-landing phase. The first is to navigate the Orbiter from the final area of the Descent stage to the vicinity of the Key Point. The Key Point is located on the runway centerline at an altitude of 4 km and approximately 14.5 km from the start of the runway. The second task is to get rid of excessive energy. This can generally be done via three methods

- Change of trajectory extent with the help of turns for the coordination of flying vehicle current energy with range up to the designed point;
- Change of dynamic pressure;
- Change of Lift-to-Drag ratio.

The BURAN used a combined method. The basis was a change in trajectory by combining programmed changes in the ratio of L/D and dynamic pressure. For example, the maximum programmed dynamic pressure is used when flying with the wind to allow for maximum energy dispersion and a minimum dynamic pressure with a maximum L/D ratio is flown when there is headwind.

A 3D-trajectory is generated at the start of the Pre-Landing Manoeuvring Stage, it consists of five segments. A turn-away spiral, spiral to correct the turn, tangent to the HAC, follow the HAC outline, and a straight line to the Key Point. The trajectory is generated by comparing the required and available energy required. The first flight made use of the eastern cylinder of energy dispersion because of the wind conditions. Starting at this point, two trajectories are possible. One tube follows a southern HAC and a second tube the northern HAC. The first flight made use of the northern HAC. It later successfully arrived at the Key Point.

2.2.3. HORUS

The original design of the HORUS-2B guidance algorithm was designed in 1988 by Helmersson (1988). The research done by SAAB aerospace was done with detailed trajectory analyses on the different phases of the re-entry. They proposed a TAEM guidance system based on predicting capabilities. De Ridder (2009) later proposed a new technique based on the energy state.

The original terminal area is divided into four segments as visualized in Figure 2.6 with three possible circuit patters: a direct approach, a left-hand HAC turn or a right-hand HAC turn. This system requires that the pattern is chosen before the mission has commenced, and the design only used the fixed left-circuit mode. The four segments are now briefly discussed (Helmersson, 1988).

- *Zoom and initial turn*
  The zoom and initial turn phase is the first phase within the terminal area and it is responsible for making sure that the velocity is reduced to the required amount. This insures that the aerodynamic forces do not become too large within the denser atmosphere.

[4] http://www.buran-energia.com/documentation/documentation-akc-guidance-control.php

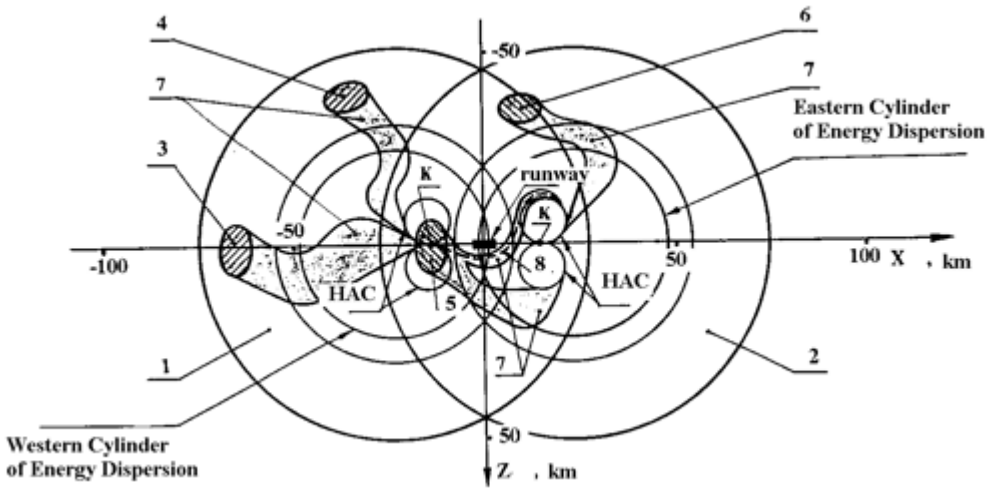


Figure 2.5: Trajectory tubes at the pre-landing manoeuvring Stage for the first flight of the BURAN

- *Straight descending flight*
  There is a straight descending flight after the initial turn that continues until the HAC is reached. The glide ensures that the velocity is lowered from supersonic to subsonic velocities, by keeping the equivalent airspeed constant, a value of around 144.0 m/s for the nominal case. The angle of attack is limited in size due to the aerodynamic shocks caused by the transonic regime.

- *Final turn around the HAC*
  The heading alignment cylinder (HAC) aligns the vehicle to the runway. The shape here is fixed conical, with a radius of 2 km that becomes proportionally larger by the inverse air density. The location is 12 km in front of the runway and 2.5 km to the left of the centerline. The turn is performed with constant equivalent airspeed and bank angle. This phase ends when the alignment is reached by a $5°$ difference.

- *Straight final approach*
  The final segment aligns the vehicle to the runway and keeps it as small as possible until the landing phase commences. The last phase is not treated in the study done by (Helmersson, 1988) and shall also not be discussed in this literature study. The final flare manoeuvre is therefore neglected.

Multiple tests were performed to test the robustness of the guidance method designed by SAAB. It proved to be reliable when variations were imposed on the initial state such as a change in altitude, velocity, and heading. The variables were only changed one at a time, and a performance assessment showed that the guidance method proved to be unreliable for off-nominal cases that required the choosing of another approach than the left-handed one Helmersson (1988). The terminal area guidance system devised by (De Ridder, 2009) attempts to circumvent this problem.

As said, the guidance system is a closed-loop one that has predicting capabilities. The required energy to reach the runway is continuously calculated and the proper actions are performed to dissipate or economize the amount of energy. The required energy itself is not calculated, but rather the excess energy that can be divided into three components: the speed energy, transonic, and the turn loss. The amount of excess energy can determine what type of longitudinal or lateral guidance is required.
Two different sets guidance laws are used to determine the trajectory. The first set determines the longitudinal trajectory by setting the required angle of attack, the second set determines the lateral ground track by calculating the bank angle. The longitudinal guidance has two different control laws, one for the maximum-range and another for the maximum-dive trajectories.The choice of guidance laws mainly depend on the initial state and the amount of excess energy present De Ridder (2009).

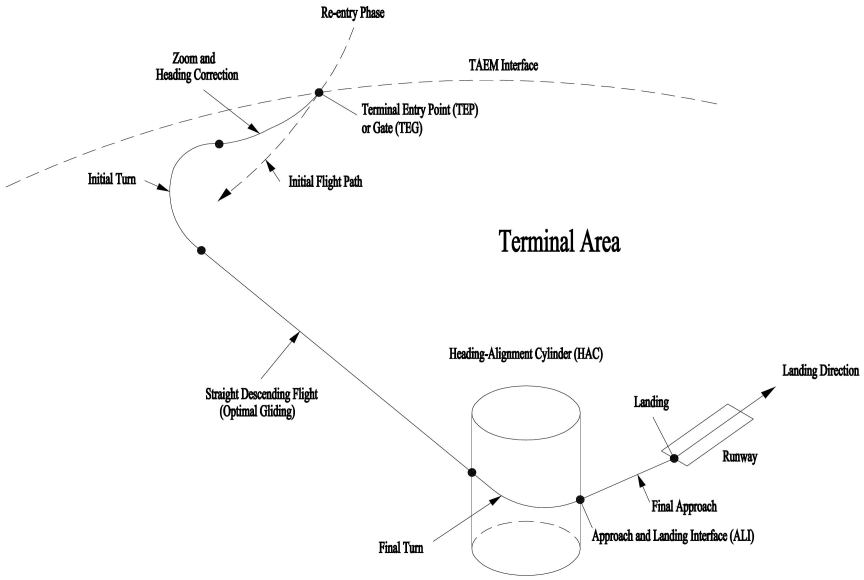


Figure 2.6: Overview of the different phases during re-entry (Mooij, 2013)

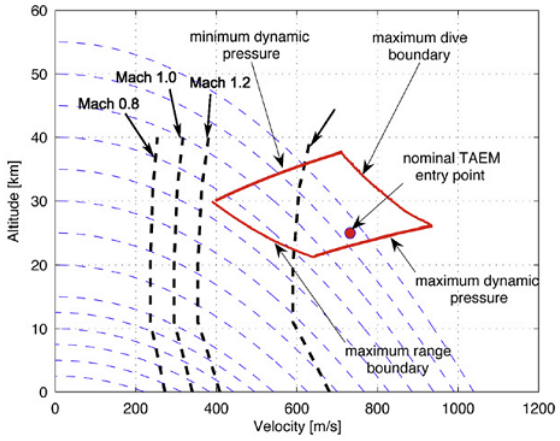


Figure 2.7: Example of a cross-section of the energy tube (De Ridder, 2009).

The energy-tube concept can be used for this to determine the flight envelope of the vehicle. The different trajectories form a boundary which can be called a vertical corridor. A target is selected, the runway for example. and the amount of energy required to get there is calculated, both the minimum and maximum. The combinations to fly a particular distance without overshoot form an energy area in the energy space. The energy area is the cross-section of the energy-tube with the desired range. Two extra boundaries are that of the stall speed and the maximal dynamic pressure. A visualisation is given in Figure 2.7. The range that is achievable is usually dictated in terms of the minimum amount of energy required, there are, however, differences in how that minimal energy is composed, differing between kinetic and potential energy. This is another reason that the cross-section is beneficial, since it gives a nice overview of what combinations of velocity and height are required. An overshoot will occur if the initial state is past the maximum-dive boundary and an undershoot will happen if it is to the left of the maximum-range boundary. The energy-tube concept does not take into account off-nominal conditions that could happen later on in the flight path. It is, however, safe to say that these conditions can be countered if the surface of the cross-section is large due to the excess energy. The choice of relying on one strategy, such as the maximum range one, is therefore undesirable.

The usual steady-state approximation to achieve maximum-range is to have an angle of attack profile that achieves a maximum L/D ratio. The usual method is viable for the supersonic phase, it is, however, proven

that using this method for the subsonic phase cannot achieve the same distance. The optimal subsonic solutions follow the energy-state approximation which states that the maximum range is achieved by minimizing the drag as much as possible. Different initial states all converge to the same trajectory guaranteeing this minimal drag by performing transient manoeuvres to reach it. The turn around the HAC is performed in the subsonic region with an optimal turn that has a higher angle of attack and lower velocity compared to a wings-level flight. The trajectory that aims towards the HAC is affected by the radius of it by changing the dynamic pressure in the drag valley (the path of minimal drag). The optimal angle of attack is calculated with the simple PD-controller during the subsonic and optimal turn around the HAC (De Ridder, 2009)

$$\alpha = \alpha^* + K_p(\bar{q}_{ref} - \bar{q}) + K_d\dot{\bar{q}} \tag{2.3}$$

α^* is the open-loop reference angle of attack for an equilibrium flight at the required dynamic pressure and the current energy height based on maximizing the lift over drag ratio and $\bar{q}_{ref}$ is the optimal constant reference dynamic pressure, the dynamic pressure of the drag valley. α^* is a function of the turn radius and the energy height, and $\bar{q}_{ref}$ is a function of the turn radius. The optimal gain values K_p and K_d differ for each initial state, a sub-optimal set is, however, available in (De Ridder and Mooij, 2009) that still provides a result very close to the optimal one. A maximum-dive trajectory can be reached by minimizing the dynamic pressure, this however results in less control capabilities making this option less favourable. The maximum dive shall therefore be done with the maximum dynamic pressure. The angle of attack is calculated with the same method as the maximum-range problem, Equation 2.3, but $\bar{q}_{ref}$ is now the maximal dynamic pressure. The speedbrake is deflected by $25°$ when in this mode, still leaving margin of $15°$ for the rudder to properly function. The angle of attack is limited by two different filters (De Ridder, 2009). The first filter is a limitation in the trim range, this a restriction, based upon the Mach number. The next filter constrains the derivative of the angle of attack by $3°/s$ to simulate realistic flyable trajectories.

The lateral guidance laws are used to calculate the bank angle that can in turn be used to follow the required ground track. The ground track can be split into three parts, the acquisition phase, a heading alignment phase and a pre-final phase. The acquisition phase is the time when the vehicle should align to a heading tangent to the HAC, followed by a wing-level flight. The bank command angle for this phase is simply a proportional controller

$$\sigma = K_p(\chi_{req} - \chi) \tag{2.4}$$

in which χ is the actual heading and χ_{ref} is the heading required. The next phase is the heading-alignment phase. The bank command angle should make sure that the vehicle makes the turn that follows the HAC. The commanded bank angle is proportional to the distance from the cylinder ΔR and the position-rate error $\Delta\dot{R}$, thus

$$\sigma = \tan^{-1}\left(\frac{V^2\cos\gamma}{gR_{hac}}\right) + K_p\Delta R + K_d\Delta\dot{R} \tag{2.5}$$

The first term represents the bank angle required to make a turn that is equal to the curvature of the HAC. This phase continues until the heading comes within $5°$ of the heading required for the final-approach phase, it then switches. The lateral guidance steers the vehicle towards the extension of the runway by looking at the lateral deviation Δy and its derivative $\Delta\dot{y}$.

$$\sigma = K_p\Delta y + K_d\Delta\dot{y} \tag{2.6}$$

A filter is applied after the calculation is made to insure that the allowable bank angle is not exceeded and the change in bank angle is not larger than $15°$/s.

The trajectory of a nominal case is shown in Figure 2.8. The heading angle changes towards the edge of the HAC to perform a right-handed turn later on until it is aligned towards the runway. The angle of attack and bank angle to acquire this are seen in Figure 2.9. The angle of attack slowly becomes smaller until the equivalent airspeed is reached until the transonic regime is entered where it performs a dive and pull-up manoeuvre. The angle of attack later on becomes higher to counter the increase in equivalent airspeed bank angle change during the turn around the HAC. Attempts were made to reproduce the results produced by De Ridder (2009), this only succeeded partially, but the nominal results were acceptable. The problem was found that the longitudinal movement is based on energy control, which has difficulties with different initial positions and external disturbances. There was often an undershoot or overshoot of the runway. Further research on this subject was therefore abandoned.

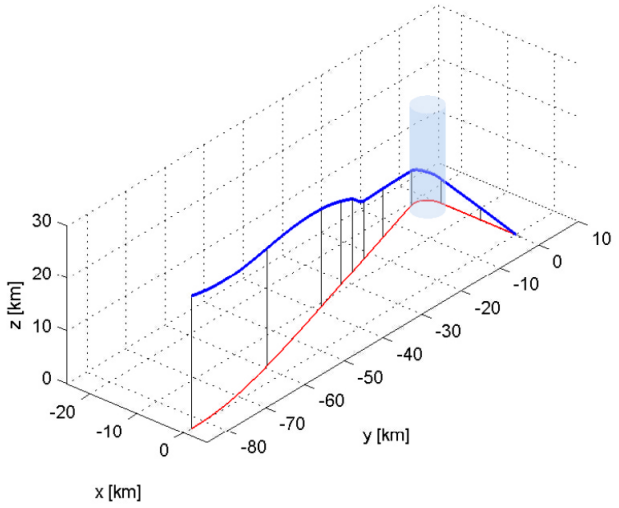


Figure 2.8: The trajectory of a nominal case for HORUS (De Ridder, 2009).

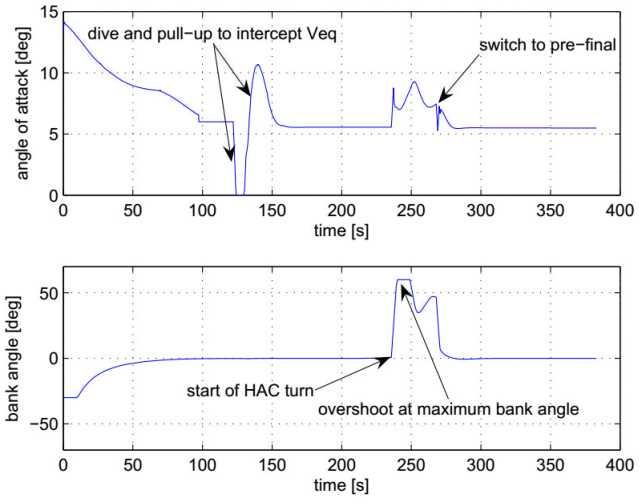


Figure 2.9: The angle of attack and bank angle for a nominal case for HORUS (De Ridder, 2009).

2.2.4. X-33

The X-33 is a commonly used test vehicle for advanced guidance and control methods (Kluever and Horneman, 2005) since the aerodynamic data and vehicle paramaters are widely available.

One example of this research is Bollino (2006). PSM were employed to solve re-entry trajectory optimisation problems. The goal was to generate an integrated guidance and control system by coupling the inner and outer loop for a 6-DOF model, thus introducing a single non-linear problem. Traditional methods assume that the goal is to safely reach the landing site at the correct attitude and energy, while jeopardizing the safety of the vehicle because that target must be reached (Bollino, 2006). Optimal footprint generation is identified as a critical capability to improve the reliability and effectiveness in case it is required to reach an alternate landing site in the case of any contingencies. An autonomous on-board capability was designed that can reach a Flight Approach Corridor (FAC) from any given entry point. The FAC is seen in Figure 2.10. It can be imagined as a box in front of the runway, the trajectory should end within this box. This example requires an average flight-path angle of $-10°$ from the FAC until the beginning of the runway.

Several initial conditions were tested using open-loop, using PSM combined with the FAC. It was shown that it was possible to have an accurate and reliable method. This method was later on integrated into the 6-DOF model after the initial tests showed a viable concept. Anti-aliasing techniques were used to calculate the optimal control surface deflections which limited the differences between the different frequencies of the rotational and translational motion. An example of such a trajectory is seen in Figure 2.11. The trajectory shows resemblances to ones using a regular HAC. The robustness of the model was tested by imposing various winds and gust loads, the model could handle wind speeds up to 7.7% of the flight speed. The conclusion in the end was that it is feasible for PS methods to be used as a guidance and control technique for re-entry systems.

The thought of the research was that recent advances in computational power and numerical methods made it possible to use PSM. Yet, there were still computational and modelling issues. It is time to see what possible improvements are possible 10 years later. There are recommendations at the end of (Bollino, 2006) that this research shall attempt to improve. The first one is that DIDO was limited to a discretisation amount of 30 nodes, the resulting trajectories for simulations that had higher nodes were infeasible. Increasing the amount of nodes should theoretically increase the accuracy of the results. The second problem was with the difference between the slow translational dynamics and the fast rotational dynamics. An increase in nodes should alleviate this problem, but that is dependent on the total time of the trajectory. An anti-aliasing technique was used to cover the loss of the accuracy of the higher frequency domain, this solved the problem, but it no longer was an optimal solution, albeit close to one. A 6-DOF solution can be found by solving one problem, but this problem can be reduced to two simpler ones by performing an actual separation of the time scales. It should be theoretically possible to reach the FAC with PS methods when this is done. The third recommendation is that it is possible to reduce the computation time a hundred fold by moving away from Matlab and use a simulation that has a C or C++ architecture. These are recommendations that will be used during this research.

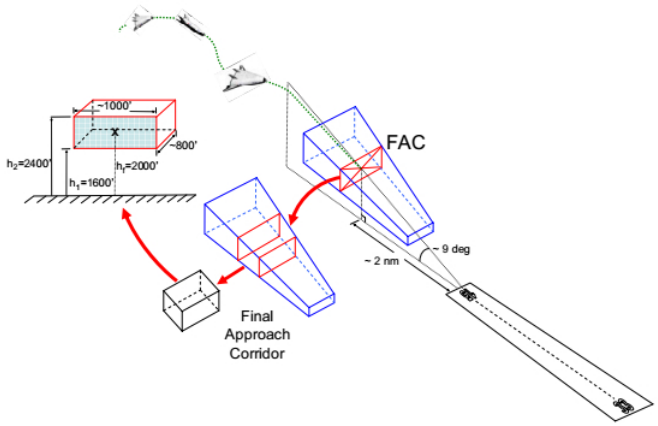


Figure 2.10: The flight approach corridor as used in (Bollino et al., 2006).

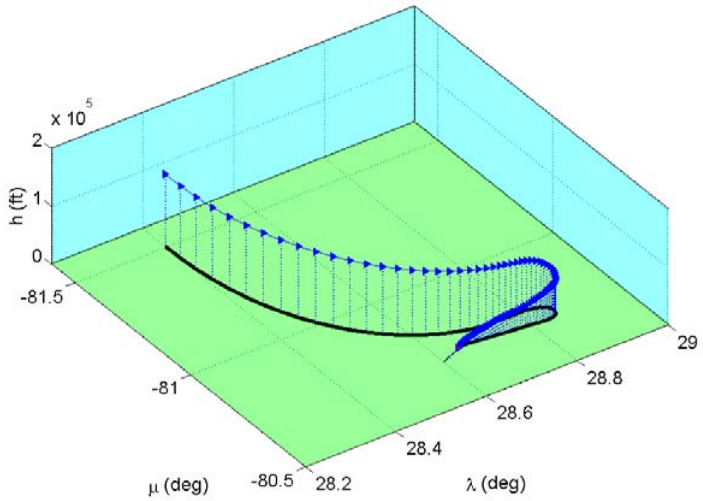


Figure 2.11: The flight approach corridor as used in (Bollino et al., 2006).

2.3. Study objectives

The introduction stated that the research objective is to find out what the viability is of PSM in the terminal area and how robust this is. The aerodynamic data of two spacecraft are available, the X-38 and HORUS. The first goal is to assess which one has the potential to be used in conjunction with PSM, before moving on to the primary question.

The heritage studied in this chapter has made clear that there is a broad set of knowledge available when designing re-entry vehicles. Two types were discussed. The first type is the creation of reference trajectories. A reference trajectory is followed by tracking the difference between the current state and the nominal state, and by trying to reduce this difference. The downside of this system is that it lacks flexibility due to a change in situation caused by expected and unexpected elements that cause deviations of the current state. There are many initial trajectory calculated pre-flight to be prepared for different scenarios, but there are only so many scenarios that can be thought up beforehand. The second type is based on energy control. The current state of the RV-W is used to calculate the angle of attack and ground path.

The goal of this thesis is thus to find a model that has pseudospectral methods as a replacement. The initial reference trajectory is calculated beforehand, similar to the first method. How well does PSM track this trajectory, and how can it handle disturbances. The follow-up question is how similar the results obtained by using PSM when compared to heritage methods. The HAC is a method that is used throughout, is there an approach that is similar, and does the ground-path follow an arc? We can make a comparison by using similar initial conditions as used by De Ridder (2009).

This research can then be extended to how well it performs in combination with control methods. What is the difference between following the reference trajectory, and how well does it fare when the reference trajectory is updated to the current state.

3

Flight Mechanics

This chapter introduces the concepts that are used when building the flight dynamics model. Section 3.1 starts with the reference frames used, followed by transformations between the reference frames in Section 3.2. Section 3.3 gives an overview of the different state variables that are used in this research. How the environment of the RV-W is modelled is presented in Section 3.4, and what effect the environment on the RV-W has in terms of forces and moments is shown in Section 3.5. The equation of motion are presented in Section 3.6 and 3.7. Finally, the two reference vehicles are presented in Section 3.8.

3.1. Reference Frames

Many different kinds of reference frames are used throughout this report. These are described below. (Mooij, 2013)

- *Vertical reference frame* ($\mathbb{F}_V$)
  The vertical reference is also known as the vehicle-carried normal Earth reference frame. The origin of this reference frame is located at the centre of mass of the vehicle of discussion. The X_V-axis is directed north, the Y_V-axis east and the Z_V-axis is pointed down towards the origin of the frame.

- *Body reference frame* ($\mathbb{F}_B$)
  This frame is fixed to the vehicle and has its origin at the centre of mass of the vehicle. The X_B-axis is pointed towards the nose of the vehicle, the Z_B-axis is pointed downwards and the Y_B-axis is pointed towards the right wing. The difference between the vertical reference frame and the body reference frame is defined by the so called Euler-angles. These are ϕ, θ and ψ for the XYZ-axes respectively. See Figure 3.1.

- *Trajectory reference frame (groundspeed based)* ($\mathbb{F}_{TG}$)
  The trajectory reference frame has its origin at the centre of mass of the vehicle. The X_{TG}-axis is defined by the direction of the velocity vector corresponding to the ground, the Z_{TG}-axis lies in the vertical frame pointing down. The Y_{TG}-axis is found by completing the right-handed system.

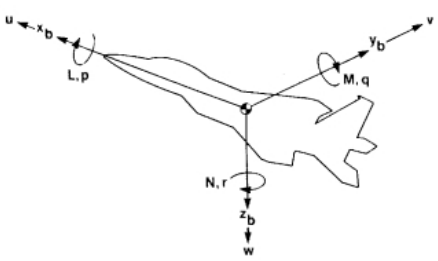


Figure 3.1: The body reference frame (Duke, 1994).

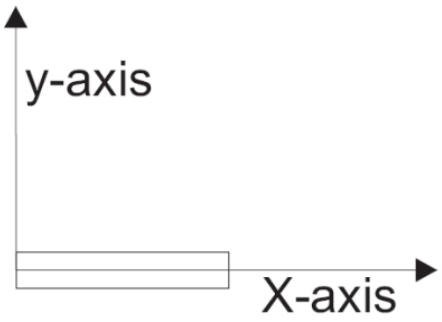


Figure 3.2: The runway reference frame (De Ridder, 2009).

- *Trajectory reference frame (airspeed based)* ($\mathbb{F}_{TA}$)
  A reference frame that is similar to the previous one except for the fact that the X_{TA} is defined as the velocity vector compared to the surrounding atmosphere. The Z_{TA}-axis points downwards and the Y_{TA}-axis completes the right-handed frame.
- *Aerodynamic reference frame (ground speed based)* ($\mathbb{F}_{AG}$)
  The aerodynamic reference frame is defined to have the X_{AG}-axis the same as the velocity vector relative to the ground and the Z_{AG}-axis is to be collinear with the lift vector but opposite in direction. The Y_{AG} completes the right-handed system.
- *Wind reference frame* ($\mathbb{F}_W$)
  The X_W-axis is collinear with the wind vector, it is positive in the direction from which the wind is coming. The Z_W-axis is pointing upwards if the wind comes from below and the Y_W-axis finishes the right-handed system.
- *Runway reference frame* ($\mathbb{F}_R W$)
  The last reference frame that is used involves the runway. The reference frame has its center at the beginning of the runway with the X-axis pointing towards the end of the runway straight through the middle of the runway. The Y-axis points to the right and the Z-axis points upwards. See Figure 3.2.

All these reference frames will be used throughout this report. The method of changing from one frame to another one will be discussed in Section 3.2.

3.2. Reference Frame transformations

Different reference frames are used during the simulation. The location is determined in the vertical and runway frame, and the attitude uses the aerodynamic frame. It is important that these frames can be used side by side, this is done by transforming the values of the vectors within those frames to the correct frame with transformation matrices. This section explains how these transformation matrices are created, and they are shown for the relevant reference frame transformations.

Imagine a reference frame A that is right-handed with three unit vectors [$\mathbf{a}_1$,$\mathbf{a}_2$,$\mathbf{a}_3$] and another reference frame B that has a similar reference frame but with unit vectors [$\mathbf{b}_1$,$\mathbf{b}_2$,$\mathbf{b}_3$]. The vectors of B can be expressed using the vectors of frame A. This is done via (Wie, 2008):

$$\mathbf{b}_1 = C_{11}\mathbf{a}_1 + C_{12}\mathbf{a}_2 + C_{13}\mathbf{a}_3 \tag{3.1}$$
$$\mathbf{b}_2 = C_{21}\mathbf{a}_1 + C_{22}\mathbf{a}_2 + C_{23}\mathbf{a}_3 \tag{3.2}$$
$$\mathbf{b}_3 = C_{31}\mathbf{a}_1 + C_{32}\mathbf{a}_2 + C_{33}\mathbf{a}_3 \tag{3.3}$$

C_{ij} is the direction cosine, the cosine of the angle between $\mathbf{b}_i$ and $\mathbf{a}_j$ ($C_{ij} \equiv \mathbf{b}_i \cdot \mathbf{a}_j$). These three expresions can also be written in a matrix form

$$\begin{pmatrix}\mathbf{b}_1\\ \mathbf{b}_2\\ \mathbf{b}_3\end{pmatrix} = \begin{bmatrix}C_{11} & C_{12} & C_{13}\\ C_{21} & C_{22} & C_{23}\\ C_{31} & C_{32} & C_{33}\end{bmatrix}\begin{pmatrix}\mathbf{a}_1\\ \mathbf{a}_2\\ \mathbf{a}_3\end{pmatrix} = \mathbf{C}_{B,A}\begin{pmatrix}\mathbf{a}_1\\ \mathbf{a}_2\\ \mathbf{a}_3\end{pmatrix} \tag{3.4}$$

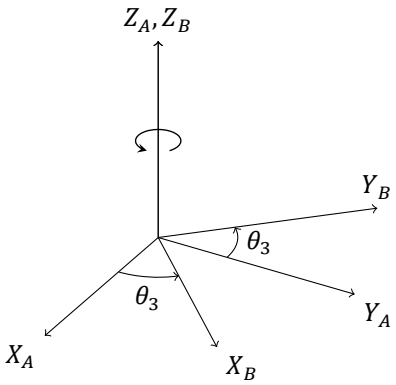


Figure 3.3: Rotation around the Z_A-axis from reference frame A to reference frame B

$\mathbf{C}_{B,A}$ is known as the direction cosine matrix (DCM), it describes the relative orientation of B to A. Other terms for this are the rotation matrix or coordinate transformation matrix. It can also be represented with $\mathbf{C}_{B,A} : B \leftarrow A$ and

$$\mathbf{C}_{B/A} = \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \end{pmatrix} \cdot \begin{pmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \mathbf{a}_3 \end{pmatrix} \tag{3.5}$$

It is quite obvious that since both **a** and **b** are orthogonal unit vectors that the resulting direction cosine matrix **C** is an orthonormal matrix. We can therefore apply the following math

$$\mathbf{C}^{-1} = \mathbf{C}^T \tag{3.6}$$

$$\mathbf{C}\mathbf{C}^T = \mathbf{I} = \mathbf{C}^T\mathbf{C} \tag{3.7}$$

Following this, and knowing that Equation 3.5 can be rewritten to $\mathbf{C}_{A,B} = \mathbf{a}_i \cdot \mathbf{b}_j$ we can say that

$$\mathbf{C}_{A,B}^{-1} = \mathbf{C}_{A,B}^T = \mathbf{C}_{B,A} \tag{3.8}$$

$$\mathbf{C}_{B,A}^{-1} = \mathbf{C}_{B,A}^T = \mathbf{C}_{A,B} \tag{3.9}$$

Both DCM's can be multiplied to get the transformation order of $A \leftarrow B \leftarrow A$ and that an identity matrix as in Equation 3.7 is found. This makes sense, since the original reference frame A is both used at the start and should be the result.

It is possible to rotate around a single axis of a reference frame by using the following transformation matrices:

$$\mathbf{C}_1(\theta_1) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_1 & \sin\theta_1 \\ 0 & -\sin\theta_1 & \cos\theta_1 \end{bmatrix} \tag{3.10}$$

$$\mathbf{C}_2(\theta_2) = \begin{bmatrix} \cos\theta_2 & 0 & -\sin\theta_2 \\ 0 & 1 & 0 \\ \sin\theta_2 & 0 & \cos\theta_2 \end{bmatrix} \tag{3.11}$$

$$\mathbf{C}_3(\theta_3) = \begin{bmatrix} \cos\theta_3 & \sin\theta_3 & 0 \\ -\sin\theta_3 & \cos\theta_3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.12}$$

The i in $\mathbf{C}_i$ denotes the rotation around that axis. A visualization of this is given in Figure 3.3 where a rotation occurs around the third axis, usually called the Z-axis. The direction cosine matrix is the basis for the Euler angle method as seen in Section 3.6.2. The rest of the section focuses on how to transform the different reference frames presented in Section 3.1.

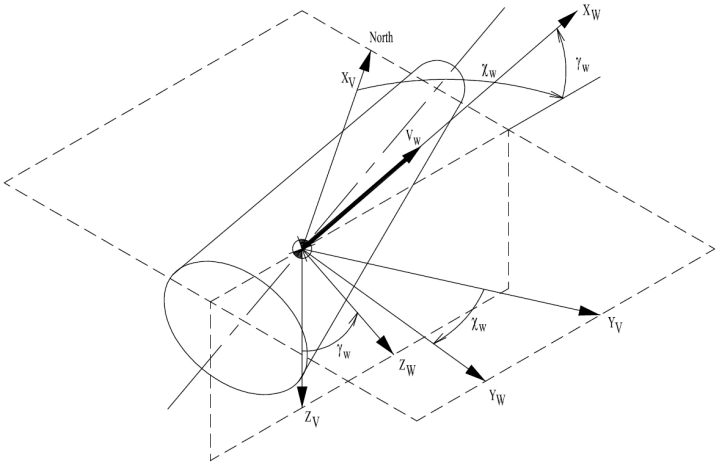


Figure 3.4: Switching from the vertical to the wind reference frame (Mooij, 2013).

Wind to vertical frame

This transforms the wind direction into the axes of the vertical frame. χ_W is the heading of the wind vector and γ_W is the flight-path angle for the wind vector. Figure 3.4 represents the transformation.

$$\mathbf{C}_{V,W} = \mathbf{C}_3(-\chi_W)\mathbf{C}_2(-\gamma_W) = \begin{bmatrix} \cos\chi_W \cos\gamma_W & -\sin\chi_W & \cos\chi_W \sin\gamma_W \\ \sin\chi_W \cos\gamma_W & \cos\chi_W & \sin\chi_W \sin\gamma_W \\ \sin\gamma_W & 0 & \cos\gamma_W \end{bmatrix} \tag{3.13}$$

Trajectory to vertical frame

This transformation changes the trajectory frame to the vertical frame as seen in Figure 3.5. γ_G is the flight-path angle as required for the ground speed and χ_G is the flight-path angle required. The subscripts of the headings can be changed to the aerodynamic frame so that the airspeed-based trajectory frame is calculated.

$$\mathbf{C}_{V,TG} = \mathbf{C}_3(-\chi_G)\mathbf{C}_2(-\gamma_G) = \begin{bmatrix} \cos\chi_G \cos\gamma_G & -\sin\chi_G & \cos\chi_G \sin\gamma_G \\ \sin\chi_G \cos\gamma_G & \cos\chi_G & \sin\chi_G \sin\gamma_G \\ \sin\gamma_G & 0 & \cos\gamma_G \end{bmatrix} \tag{3.14}$$

$$\mathbf{C}_{V,TA} = \mathbf{C}_3(-\chi_A)\mathbf{C}_2(-\gamma_A) = \begin{bmatrix} \cos\chi_A \cos\gamma_A & -\sin\chi_A & \cos\chi_A \sin\gamma_A \\ \sin\chi_A \cos\gamma_A & \cos\chi_A & \sin\chi_A \sin\gamma_A \\ \sin\gamma_A & 0 & \cos\gamma_A \end{bmatrix} \tag{3.15}$$

Airspeed-based aerodynamic to airspeed-based trajectory frame

The difference between the airspeed-based aerodynamic and that of the airspeed-based trajectory frame is that of the bank angle (σ_A) made by the plane. See Figure 3.6.

$$\mathbf{C}_{TA,AA} = \mathbf{C}_1(\sigma_A) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\sigma_A & \sin\sigma_A \\ 0 & -\sin\sigma_A & \cos\sigma_A \end{bmatrix} \tag{3.16}$$

Body to aerodynamic frame

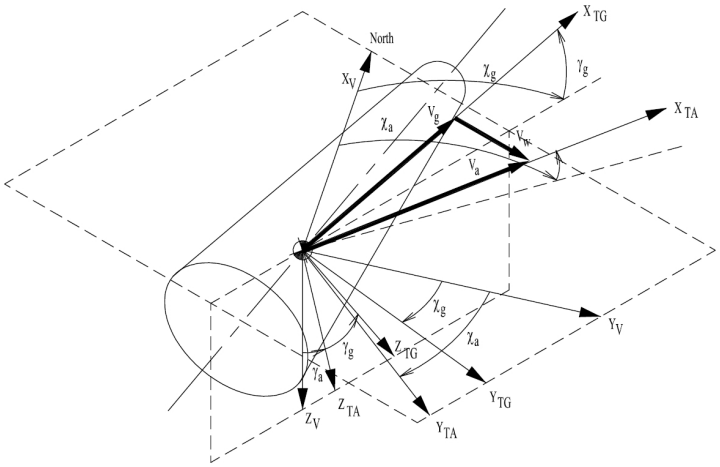


Figure 3.5: Switching from the vertical to the ground or aerodynamic trajectory reference frame (Mooij, 2013).

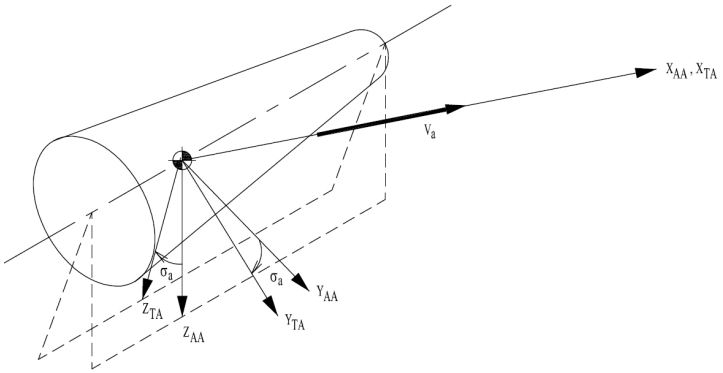


Figure 3.6: Relation between the airspeed-based aerodynamic frame and the airspeed-based trajectory frame (Mooij, 2013).

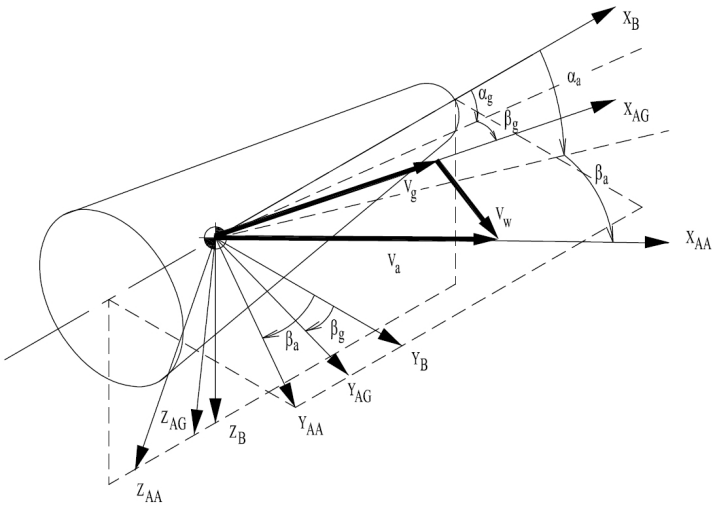


Figure 3.7: Relation between the body frame and the aerodynamic frames. (Mooij, 2013).

The relation with the body frame and the aerodynamic frame can be found with the angle of attack (α) and the angle of sideslip (β). This can be done for both the aerodynamic frame and the airspeed-based (subscript A) and the groundspeed-based aerodynamic frame (subscript G)

$$\mathbf{C}_{AA,B} = \mathbf{C}_3(\beta_A)\mathbf{C}_2(-\alpha_A) = \begin{bmatrix} \cos\alpha_A \cos\beta_A & \sin\alpha_A & \sin\alpha_A \cos\beta_A \\ -\cos\alpha_A \sin\beta_A & \cos\beta_A & -\sin\alpha_A \sin\beta_A \\ -\sin\alpha_A & 0 & \cos\alpha_A \end{bmatrix} \tag{3.17}$$

$$\mathbf{C}_{AG,B} = \mathbf{C}_3(\beta_G)\mathbf{C}_2(-\alpha_G) = \begin{bmatrix} \cos\alpha_G \cos\beta_G & \sin\alpha_G & \sin\alpha_G \cos\beta_G \\ -\cos\alpha_G \sin\beta_G & \cos\beta_G & -\sin\alpha_A \sin\beta_G \\ -\sin\alpha_G & 0 & \cos\alpha_G \end{bmatrix} \tag{3.18}$$

The transformation is depicted in Figure 3.7.

Wind to runway reference frame

It is required for the wind reference frame to be projected in the runway reference frame since the dynamic equations are in this frame as well. Two initial rotations are required. The first one flips the axis around the y-axis, such that the z-axis points upwards. The second rotates the z-axis with $-90°$ to switch the zonal and meridonial axis, the wind-reference system that is used switches the north and east direction.

$$\mathbf{C}_{RW,W} = \begin{bmatrix} \cos\chi_{RW} & -\sin\chi_{RW} & 0 \\ \sin\chi_{RW} & \cos\chi_{RW} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.19}$$

3.3. State Variables

This section gives the state variables that are of interest during this study. The state variables are the representation of the state of the vehicle with respect to a certain reference frame. Three different sets of states are of interest, the position, velocity, and attitude.

Two different coordinate systems are used regularly to denote the state, these are the spherical and the Cartesian coordinate system (Mooij, 1998). The spherical system gives an interpretable location on the Earth with the longitude and latitude, which is beneficial if large distances are traversed. However, this is not the case in this thesis, the simulation starts at roughly 80 kilometres from the runway. It is therefore easier to interpret the position when using the Cartesian coordinate system using the runway reference frame. Also, a smaller number of calculations are required to find out the next state, reducing the amount

of time it takes to calculate the solution. There are three states that define the position in the runway reference system. These are

- x = Centerline distance
- y = Centerline offset
- z = Height

The x-position is the perpendicular distance from the centreline of the runway and the y-position is the offset of the runway centreline. x is negative if it is before the start of the runway, positive if it past it. y is positive if the location is to the left of the centerline, and negative if it is to the right. A 0-value means that the spacecraft is perfectly aligned (assuming that it is pointing the right way).

The velocity vector can either be given as the derivative of the Cartesian components $(\dot{x}, \dot{y}, \dot{z})$ or with respect to the vertical frame as angles. The reason the latter is used is twofold. First, the variables are easier to interpret. Second, the computations are easier, there is less dependency between the variables (Bollino, 2006). The importance of this will become apparent in Chapter 4. The following three variables are of interest

- V = Velocity
- γ = Flight-path angle
- χ = Heading angle

The velocity is equal to the velocity on the ground with respect to the vertical frame (Mooij, 2013). The direction of the velocity vector is defined with two angles. The flight-path angle is the angle between the velocity vector and the local horizontal. This angle ranges from $-90°$ to $90°$ and it is negative when the velocity vector points below the horizon. The heading angle is defined as the angle between the projection of the velocity vector on the local horizontal and the local north direction and the angle ranges from $-180°$ to $180°$.

The attitude of the vehicle is defined as the orientation of the body-fixed reference frame with respect to an external reference frame (De Ridder, 2009). There are two popular methods that define the attitude, Euler angles (defined in Section 3.6) and quaternions. Each method has its own specific advantage. Euler angles have interpretable values if used in the right reference frame. Quaternions are harder to interpret and extra calculations are required to visualise the results. The advantage of quaternions however, is that there are no singularities, and that the computations are linear instead of non-linear, which results in faster computation times (Singla et al., 2005). Euler angles are chosen as the attitude representatives due to the facts that the singularity is not reached during re-entry and that these are easy to interpret. A specific set of Euler angles are the aerodynamic angles, these are

- α = Angle of attack
- β = Sideslip angle
- σ = Bank angle

The angle of attack is positive for nose-up attitudes, the sideslip angle is positive for nose-left attitudes, and the bank angle is positive when banking to the right. The angle of attack and bank angle ranges from $-180°$ to $180°$, the sideslip angle ranges from $-90°$ to $90°$.

3.4. Environmental Models

The environment around the spacecraft is of major importance during the re-entry on Earth. The gravitational acceleration experienced by the vehicle is calculated here, as well as the atmospheric model that is used. The landing takes place at Kennedy Space Centre, the local wind conditions around that runway are also given.

3.4.1. Gravitational Model

One of the main forces acting on the spacecraft is that of gravity. The magnitude of this force depends on the position around the globe. The trajectory analysis focusses on trajectories using the flat-Earth approximation that are created to land the spacecraft at Kennedy Space Center. It is not a valid strategy to take

the nominal value from the equator, the difference can be a few percent. The requirement is thus that it should be calculate the gravitational acceleration at any point around the Earth at a certain height with an acceptable precision.

The gravitational acceleration is calculated using Newton's law of gravitational acceleration (Wakker, 2015)

$$g = -\frac{\mu}{R^2} \tag{3.20}$$

where μ is the gravitational parameter and R the distance from the spacecraft to the centre of the Earth. The value of μ depends on the location on the Earth and the largest difference is noticeable when one moves from north to south or vice versa. This is due to the eccentricity of the Earth's surface. The radius to the centre of mass of the Earth is approximately 21 kilometres lower than at the poles than at the equator. μ can be calculated with

$$\mu = \mu_0 \left(1 - \frac{3}{2} J_2 \left(\frac{R_e}{R_s}\right)^2 (3\sin^2\delta - 1)\right) \tag{3.21}$$

μ_0 is the standard gravitational parameter at the equator, J_2 is the second zonal harmonic coefficient, R_e the equatorial radius, R_s the radius of the Earth at the runway, and δ is the latitude. The surface radius is calculated with

$$R_s = R_e(1 - e\sin^2\delta) \tag{3.22}$$

The following constants were used

- $e = 0.0033528$
- $R_e = 6\,378\,106$ m
- $J_2 = -1.08263529 \cdot 10^{-3}$
- $\mu_0 = 398600.4418 \cdot 10^9$ m^3/s^2

The distance from the spacecraft to the centre of the Earth's mass is simly calculated with

$$R = R_s + h \tag{3.23}$$

It is possible to calculate g with higher precision but the flat-earth induces a larger error since the curvature of the Earth is not taken into account. Therefore, this is not required. It is assumed that the location around the runway has a constant gravitational parameter.

3.4.2. Atmospheric Model

The atmospheric model used during all the simulations is the US76-model from NOAA et al. (1976), it is an international defined standard that can be used for first-order analyses . This section is a summary from this reference. The model works up to an altitude of 86 kilometres, well within our limit, and it calculates, but not limited to, the temperature, pressure, and atmospheric density. The model can also include other elements, such as temporal variations, but that is outside the scope of this thesis. The air is assumed to be completely homogeneous and dry up to 86 kilometres and it can therefore be assumed that up to that height that the atmosphere is a perfect gas, which means that the ideal gas law can be used, as well as the hydrostatic equilibrium theory.

The height that is used in the calculations is not the geometrical, but the geopotential altitude, symbolised with z. The difference between the two is that the geopotential altitude assumes that the gravitational acceleration (g_0) is constant throughout the atmosphere, found at 45.5° latitude, with a value of 9.80655 m/s^2. The geopotential altitude is calculated with

$$z = \frac{R_0 h}{R_0 + h} \tag{3.24}$$

Table 3.1: Different layers of the atmospheric US76-model (NOAA et al., 1976).

Layer b	Geopotential height z_i [km']	Temperature gradient L_{z_i} [K/km']	Temperature at interval T_i [Kelvin]	Pressure at interval p_i [Pascal]
0	0.0	-6.5	288.15	101325
1	11.0	0.0	216.65	22632.1
2	20.0	+1.0	216.65	5474.89
3	32.0	+2.8	228.65	868.019

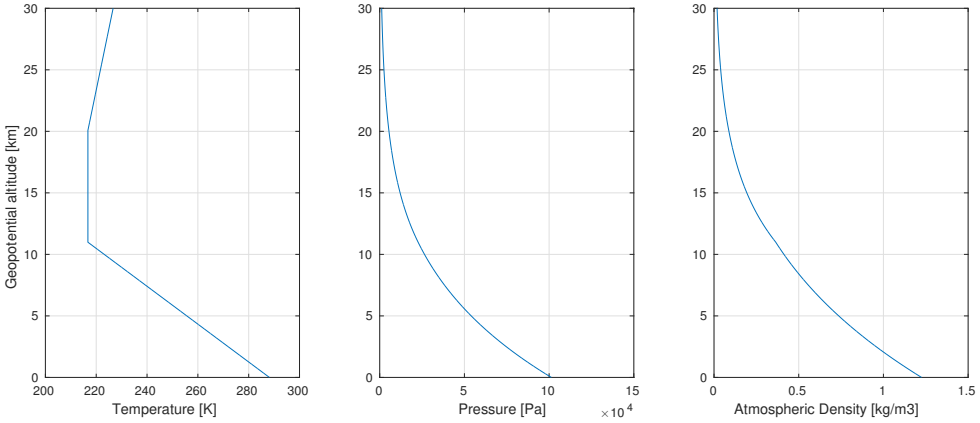


Figure 3.8: Atmospheric properties according to the US76-model.

with the radius of the Earth being $R_0 = 6356.766$ km. The geopotential altitude can be split into multiple intervals,l or layers, as seen in Table 3.1, up to an altitude of a little bit more than 84 kilometres, the subscript i refers to the initial altitude of an interval. The first atmospheric variable, the temperature, is calculated:

$$T = T_i + L_{z_i}(z - z_i) \tag{3.25}$$

L_{z_i} is called the thermal lapse rate and T_i is the temperature at the beginning of each layer , as seen in Table 3.1. There are two ways to calculate the static pressure, the method differs whether the thermal lapse rate is zero or not. If this rate is zero then,

$$p = p_i \exp\left[\frac{-g_0(z - z_i)}{RT_i}\right] \tag{3.26}$$

in which R is the universal gas constant and for intervals in which the thermal lapse rate is not zero

$$p = p_i\left[\frac{T_i}{T_i + L_{z_i}(z - z_i)}\right]^{\frac{g_0}{RL_{z_i}}} \tag{3.27}$$

with p_i the pressure at the beginning of each interval. Finally, the density can then be calculated with the equation of state

$$\rho = \frac{p}{RT} \tag{3.28}$$

The results can be seen in Figure 3.8.

3.4.3. Wind Model

A wind model is included to establish the robustness of the chosen controllers. The vehicle's response to atmospheric disturbances can determine whether the flight remains safe within the operational limits, both for the, but not limited to, the structural integrity and the guidance and flight control design, this thesis shall only discuss the latter. The handbook for atmospheric disturbances (Johnson, 2008) provides

various models that can be used to simulate the wind velocity up to an altitude of 28 kilometres. This section uses this handbook, as it is used by NASA for preliminary studies regarding the design of aerospace vehicles.

Two types of models exist to create wind profiles: synthetic (both scalar and vector) and measured profiles samples. The oldest method is that of synthetic profiles. It provides a reasonable approach when simulating not only the wind speed, but also gusts and wind layers. The measured profile samples are created by various methods and sensors such as the rawinsonde, radar wind profiler, rocketsonde, and radar. A large amount of measurements is required in order for this method to be viable, therefore, a great amount of measurements were performed at the Kennedy Space Centre (KSC), enough to perform statistical analyses of the temporal and height variations, these are also called range reference atmospheres.

The general method presented here is the meteorological coordinate system and it does not calculate the magnitude of the wind, but rather just two components. The zonal wind is depicted with the symbol u with the positive direction is eastwards and the meridional wind is given as v with the positive direction northwards, as seen in Figure 3.9. Vertical wind shears are not taken into account in this system.

A combination of the two methods is proposed here. The samples shall be used to test the response of the vehicle to the steady-state wind, and the synthetic profiles to create gusts, to test the guidance and control limits. The steady-state wind is a nominal wind profile that has been created by combining over 1,800 different profiles measured throughout the year. Table 3.2 provides these values. Linear interpolation is used to calculate the intermediate values. It can be seen that there is a large amount of wind at an altitude of around 12 kilometres, which is the subtropical jet stream, something that is prevalent throughout the year and therefore has to be taken into account. A wind gust is simulated with the NASA 1997 Discrete Gust Model. This method creates a sinusoidal-like gust calculated with

$$V = \frac{V_m}{2}\left(1 - \cos\left(\frac{\pi d}{d_m}\right)\right) \tag{3.29}$$

in which V_m is the wind magnitude, d is the height of the gust, and d_m is the gust half-width. This formula is only used if $0 \leq d \leq 2d_m$. Figure 3.10 shows two different results. The left figure shows only the nominal wind as seen in Table 3.2, the right figure has a gust superimposed on the steady-state wind. The gust has a maximum has a gust magnitude of 30.0 m/s at a height of 12,000 m with a distance of 3000 m. The heading angle of the gust is 45°. This value would be seen as an extreme value but it is purely as an example.

The wind should still be converted to the wind reference frame. This is all rather easy, especially since the magnitude is already known. The new wind angles are

$$\gamma_W = \arccos\left(\frac{\sqrt{u^2 + v^2}}{V}\right) \tag{3.30}$$

$$\chi_W = \tan\left(\frac{u}{v}\right) \tag{3.31}$$

The wind is also represented in the vertical frame. This is done with

$$W_x = u \tag{3.32}$$

$$W_y = v \tag{3.33}$$

$$W_z = w \tag{3.34}$$

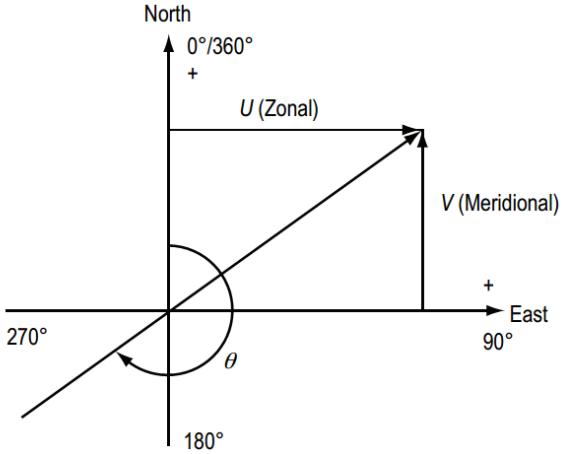


Figure 3.9: The wind reference frame (Johnson, 2008).

Table 3.2: The mean wind values at Kennedy Space Center throughout the year (Johnson, 2008).

Altitude [km]	u [m/s]	v [m/s]	Altitude [km]	u [m/s]	v [m/s]
0	-0.54	-0.64	14	25.71	-0.62
1	2.84	1.31	15	22.63	0.29
2	6.27	1.58	16	20.40	0.86
3	9.00	1.80	17	17.38	0.90
4	12.39	2.34	18	13.84	0.98
5	16.10	3.00	19	8.59	0.97
6	19.46	3.61	20	3.67	0.71
7	22.59	4.23	21	0.99	0.13
8	25.79	4.56	22	1.79	-0.42
9	28.83	4.48	23	1.03	-0.26
10	31.16	3.65	24	2.45	0.06
11	31.56	2.02	25	4.14	0.42
12	30.98	0.52	26	4.97	0.64
13	27.80	-0.38	27	5.86	1.26

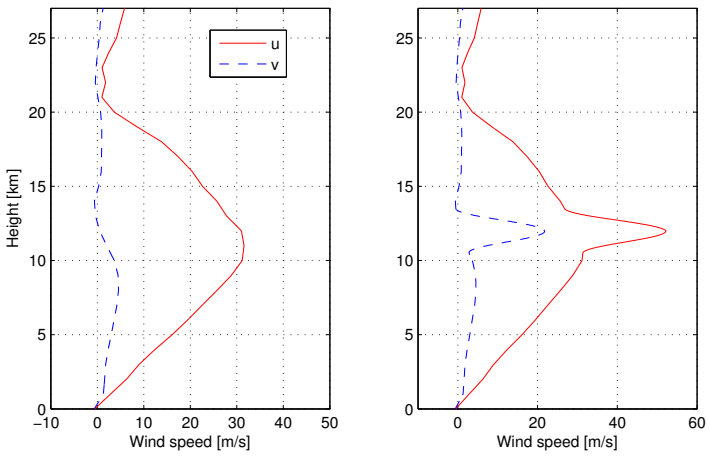


Figure 3.10: The results of the wind models with on the left side the steady-state wind, a gust is added on the right side.

3.5. External Forces and Moments

Three different types of forces and moments are usually to be considered when it involves re-entry, these are of the aerodynamic, gravitational and propulsive kind (Mooij, 1998). The last one, propulsive, shall not be used. This is because the Horus has no propulsion when it is re-entering the atmosphere. Other forces such as magnetic forces and solar radiation are ignored, their magnitudes are significantly lower than the remaining two. Figure 3.11 represents the forces and moments acting on the centre of mass in the body frame.

One of the main forces working on the vehicle is that of the aerodynamic force. The well known equation is given as

$$\mathbf{F}_{A,AA} = \begin{pmatrix} -D \\ -S \\ -L \end{pmatrix} = \begin{pmatrix} -C_D \bar{q} S_{ref} \\ -C_S \bar{q} S_{ref} \\ -C_L \bar{q} S_{ref} \end{pmatrix} \tag{3.35}$$

The forces are divided into three directions: the drag D, side force S and lift L. The forces are given in the airspeed-based aerodynamic frame. The forces are thus pointed in the negative direction with respect to the frame. The variables in Equation 3.35 are defined as

$$\bar{q} = \frac{1}{2}\rho V_A^2 = \text{dynamic pressure (N/m}^2\text{)} \tag{3.36}$$

$$V_A = \text{airspeed (m/s)} \tag{3.37}$$

$$S_{ref} = \text{aerodynamic reference area (m}^2\text{)} \tag{3.38}$$

The coefficients C_D, C_S and C_L are functions of the Mach number, aerodynamic angles, and the deflection angles of the aerodynamic surfaces. How the coefficients are calculated is found in Section 3.8. The aerodynamic forces can also be defined in the body frame. Either by transforming it to this frame, or the coefficients are given in this frame. The transformation can be achieved by using Equation 3.17. This results in

$$\mathbf{F}_{A,B} = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} C_X \bar{q} S_{ref} \\ C_Y \bar{q} S_{ref} \\ C_Z \bar{q} S_{ref} \end{pmatrix} \tag{3.39}$$

The forces are located on reference points, which is usually not in the centre of mass of the vehicle. This means that a moment, in the body frame, is generated, that is given by

$$\mathbf{M}_{A_F,B} = \mathbf{r}_{cm} \times \mathbf{F}_{A,B} \tag{3.40}$$

r_{cm} is the distance between the reference points and the centre of mass. The force created by the thrusters are a prime example when this is used. The aerodynamic moment is given by

$$\mathbf{M}_{A_M,B} = \begin{pmatrix} \mathcal{L} \\ \mathcal{M} \\ \mathcal{N} \end{pmatrix} = \begin{pmatrix} C_l \bar{q} S_{ref} b_{ref} \\ C_m \bar{q} S_{ref} c_{ref} \\ C_n \bar{q} S_{ref} b_{ref} \end{pmatrix} \tag{3.41}$$

with $\mathcal{L}$ as the rolling moment, $\mathcal{M}$ as the pitching moment and $\mathcal{N}$ as the yawing moment. b_{ref} and c_{ref} are the aerodynamic reference lengths. The total aerodynamic moment vector is then

$$\mathbf{M}_{A,B} = \mathbf{M}_{A_F,B} + \mathbf{M}_{A_M,B} \tag{3.42}$$

The gravitational force that is effecting the vehicle is equal to

$$F_{G,R} = mg \tag{3.43}$$

m is the mass of the vehicle and g is the gravitational acceleration vector working on the centre of mass of the vehicle. The value of g is calculated in Section 3.4.

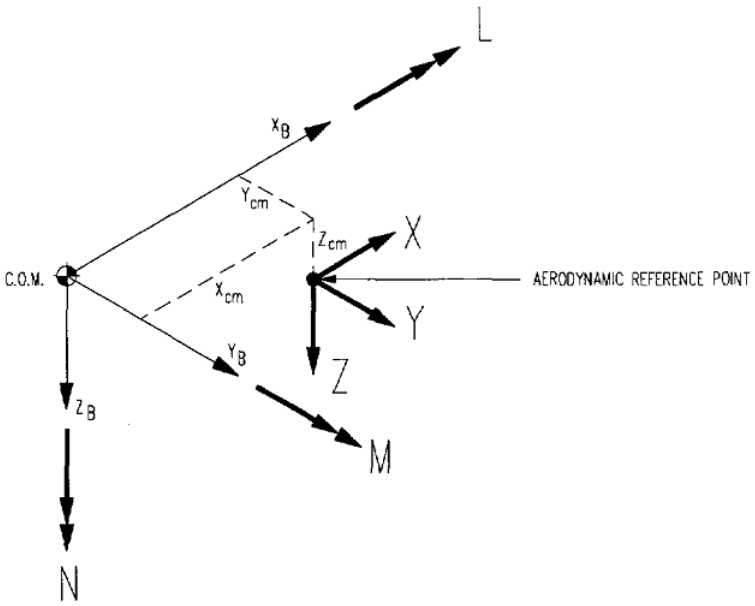


Figure 3.11: Overview of the aerodynamic forces and moments (Mooij, 1998)

3.6. Equations of Motion

This section discusses the motion of the vehicle, both how it moves around in the atmosphere and the rotation of the vehicle. It is assumed that the vehicle can be modelled as a point mass, a point that is the centre of mass of the vehicle. This section starts of with describing the general equations of motion, before splitting that up into the translational and rotational motion. The effect of wind on both types of motion is discussed last.

3.6.1. Translational Motion

Different types of equations of motion exist, from spherical to flat Earth equations (Vinh, 1981). It is beneficial to make proper assumptions to choose the correct set and simplify these equations when it is allowed. The path followed in the terminal area traverses a small part of the Earth, this means that the effect of the curvature of the Earth is relatively small in a small time-frame. The flat-earth approximation will therefore be used. The derivations of the equations of motion in the vertical reference frame are given in (Bollino, 2006). A graphical representation of the variables given here are seen in Figure 3.12. Two different sets of equations are given, the first one in which sideslip is equal to zero ($\beta = 0$) and the second one in which there is sideslip ($\beta \neq 0$). The first set is (Bollino, 2006)

$$\dot{x}_g = V \cos\gamma \sin\chi \tag{3.44}$$
$$\dot{y}_g = V \cos\gamma \cos\chi \tag{3.45}$$
$$\dot{z}_g = \dot{h} = V \sin\gamma \tag{3.46}$$
$$\dot{V}_g = -\frac{D}{m} - g \sin\gamma \tag{3.47}$$
$$\dot{\gamma}_g = \frac{L \cos\sigma}{mV} - \frac{g}{V} \cos\gamma \tag{3.48}$$
$$\dot{\chi}_g = \frac{L \sin\sigma}{mV \cos\gamma} \tag{3.49}$$

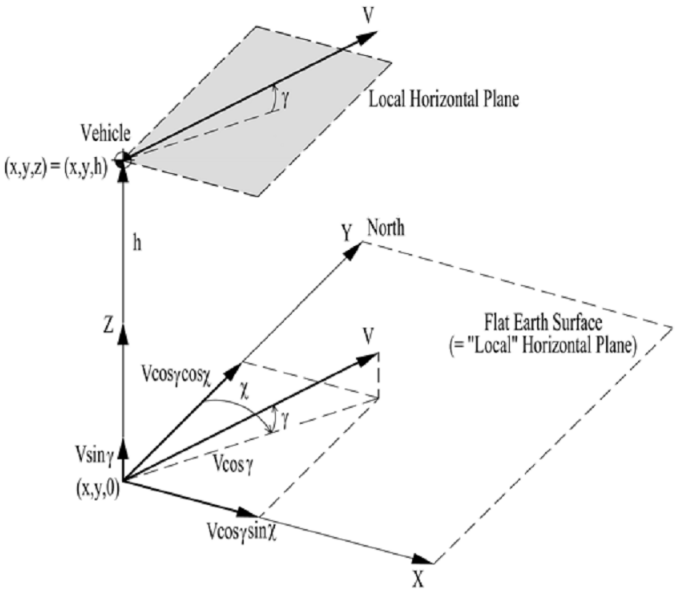


Figure 3.12: An overview of the different elements in the vertical frame. (Mooij, 2013)

and the second set, with sideslip (Bollino, 2006):

$$\dot{x}_g = V \cos\gamma \sin\chi \tag{3.50}$$

$$\dot{y}_g = V \cos\gamma \cos\chi \tag{3.51}$$

$$\dot{z}_g = \dot{h} = V \sin\gamma \tag{3.52}$$

$$\dot{V}_g = \frac{1}{m}(S \sin\beta - D \cos\beta) - g \sin\gamma \tag{3.53}$$

$$\dot{\gamma}_g = \frac{1}{mV}(L \cos\sigma - D \sin\beta \sin\sigma - S \cos\beta \sin\sigma) - \frac{g}{V} \cos\gamma \tag{3.54}$$

$$\dot{\chi}_g = \frac{1}{mV \cos\gamma}(L \sin\sigma + D \sin\beta \cos\sigma + S \cos\beta \cos\sigma) \tag{3.55}$$

These equations are altered to be in the vertical reference frame that is used here in which the heading angle is zero when it points north. The definition used in (Bollino, 2006) was that a zero degree heading angle points east.

3.6.2. Rotational Motion

The second type is that of the rotational motion which shall be further discussed here. This motion is induced by adding a moment to the c.o.m. which then causes a rotational acceleration. It is assumed that the body is rigid and that there is no change in mass. The Euler equations are (Mooij, 1998)

$$\dot{\omega} = I^{-1}(M_{cm} - \omega \times I\omega) \tag{3.56}$$

with

$\mathbf{M}_{cm} = \left(M_x, M_y, M_z\right)^T$ — Sum of external moments about the body axis (Nm)

$\mathbf{I} = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{xy} & -I_{yy} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{zz} \end{bmatrix}$ — The intertia tensor of the vehicle (kgm^2)

$\omega = (p, q, r)^T$ — Rotation vector of the body frame with respect to the inertial frame, expressed in components along the body axes [rad/s]

The tensor can be simplified since Horus has a plane of mass symmetry in the $X_B Y_B$-plane, in short, this means that the inertial tensor can be simplified by setting $I_{xy} = I_{xz} = I_{yz} = 0$. Equation 3.56 is then solved into the following set of equations

$$\dot{p} = \frac{I_{zz}}{I^*}M_x + \frac{I_{xz}}{I^*}M_z + \frac{\left(I_{xx} - I_{yy} + I_{zz}\right)I_{xz}}{I^*}pq + \frac{(I_{yy} - I_{zz})I_{zz} - I_{xz}^2}{I^*}qr \tag{3.57}$$

$$\dot{q} = \frac{M_y}{I_{yy}} + \frac{I_{xz}}{I_{yy}}\left(r^2 - p^2\right) + \frac{I_{zz} - I_{xx}}{I_{yy}}pr \tag{3.58}$$

$$\dot{r} = \frac{I_{xz}}{I^*}M_x + \frac{I_{xx}}{I^*}M_z + \frac{(I_{xx} - I_{yy})I_{xx} + I_{xz}^2}{I^*}pq + \frac{\left(-I_{xx} + I_{yy} - I_{zz}\right)I_{xz}}{I^*}qr \tag{3.59}$$

with $I^* = I_{xx}I_{zz} - I_{xz}^2$. The value of I_{xz} is relatively low and it can be assumed that HORUS is rotationally symmetric along the X_B-axis. The set of equations can then be simplified into the following dynamical equations (Mooij, 2013)

$$\dot{p} = \frac{M_x}{I_{xx}} + \frac{(I_{yy} - I_{zz})}{I_{xx}}qr \tag{3.60}$$

$$\dot{q} = \frac{M_y}{I_{yy}} + \frac{(I_{zz} - I_{xx})}{I_{yy}}pr \tag{3.61}$$

$$\dot{r} = \frac{M_z}{I_{zz}} + \frac{(I_{xx} - I_{yy})}{I_{zz}}pq \tag{3.62}$$

There are at least two consequences due to this simplification. First, the linearisation of the attitude dynamics becomes a lot easier, which will be found in Section 3.7. Second, there is less correlation between the different variables, which means that the pseudospectral method requires less time to find the optimal solution.

The differential equations are stated in (Mooij, 2013)

$$\dot{\alpha} = q - (p\cos\alpha + r\sin\alpha)\tan\beta - \frac{L - mg\cos\gamma\cos\sigma}{mV\cos\beta} \tag{3.63}$$

$$\dot{\beta} = p\sin\alpha - r\cos\alpha - \frac{S + mg\cos\gamma\sin\sigma}{mV} \tag{3.64}$$

$$\dot{\sigma} = -\frac{p\cos\alpha + r\sin\alpha}{\cos\beta} - \frac{L - mg\cos\gamma\cos\sigma}{mV}\tan\beta + \frac{L\sin\sigma + S\cos\sigma}{mV}\tan\gamma \tag{3.65}$$

There is a clear relation between α and σ, the longitudinal and lateral motion are thus coupled. Most studies assume that the lateral motion and longitudinal one can be decoupled from each other. This assumption is valid if the aerodynamic angles α and σ are relatively small. It is doubtful whether this is the case with re-entry vehicles due to their unusual shape and high angle of attack. Horus enters the terminal area with an angle of attack of roughly 20 degrees, it is therefore safe to assume that coupling effects do occur and should be taken for granted.

The initial values of the angular rates are dependent on different variables such as the angle of attack, velocity, and flight-path angle.

$$p_0 = \frac{g_0}{V_0}\cos\gamma_0\sin\sigma_0\sin\alpha_0 + \frac{L_0}{mV_0}\tan\gamma_0\sin\sigma_0\cos\alpha_0 \tag{3.66}$$

$$q_0 = \frac{L_0}{mV_0} - \frac{g_0}{V_0}\cos\gamma_0\cos\sigma_0 \tag{3.67}$$

$$r_0 = -\frac{g_0}{V_0}\cos\gamma_0\sin\sigma_0\cos\alpha_0 + \frac{L_0}{mV_0}\tan\gamma_0\sin\sigma_0\sin\alpha_0 \tag{3.68}$$

This insures that $\dot{alpha}$, $\dot{beta}$, and $\dot{sigma}$ start as 0 and that the flight is stable, if properly trimmed.

3.6.3. Wind Equations

One disturbance that has not yet been put in equational form in this chapter is that of wind. The translational and rotational dynamics have so far not taken into account any of this, something that can severely

alter the trajectory of the vehicle as well as the aerodynamic properties due to a change in which the direction of the wind is coming from. The wind is generated by using the models discussed in Section 3.4.3. It is thus assumed that the direction of the wind is known, as well as the magnitude. The wind is modeled with wind velocity components in the vertical reference frame (W_x,W_y,W_z).
The effect on the kinematical equations are (Bollino, 2006)

$$\dot{x}_g = V\cos\gamma\cos\chi + W_x \tag{3.69}$$
$$\dot{y}_g = V\cos\gamma\sin\chi + W_y \tag{3.70}$$
$$\dot{z}_g = V\sin\gamma + W_z \tag{3.71}$$

And the effect on the dynamical equations of motion is (Bollino, 2006)

$$\dot{V}_g = -\frac{1}{m}\left(S\sin\beta - D\cos\beta\right) - g\sin\gamma - \left(\dot{W}_x\cos\gamma + \dot{W}_z\sin\gamma\right) \tag{3.72}$$
$$\dot{\gamma}_g = \frac{1}{mV}\left(L\cos\sigma - D\sin\beta\sin\sigma - S\cos\beta\sin\sigma\right) - \frac{g}{V}\cos\gamma + \frac{1}{V}\left(\dot{W}_x\sin\gamma + \dot{W}_z\cos\gamma\right) \tag{3.73}$$
$$\dot{\chi}_g = \frac{1}{mV\cos\gamma}\left(L\sin\sigma + D\sin\beta\cos\sigma + S\cos\beta\cos\sigma\right) \tag{3.74}$$

The effect of the wind on the rotational motion is calculated differently. The wind is transformed from the vertical to the body frame in order to calculate the disturbed aerodynamic angles. The body axis rates are summed up with the wind (Bollino, 2006)

$$u = u_b + W_{b,u} \tag{3.75}$$
$$v = v_b + W_{b,v} \tag{3.76}$$
$$w = w_b + W_{b,w} \tag{3.77}$$

with the velocity vector defined as

$$V = \sqrt{u^2 + v^2 + w^2} \tag{3.78}$$

These new axis rates then allow for the new aerodynamic angles to be calculated

$$\alpha_{AA} = \tan^{-1}\left(\frac{w}{u}\right) \tag{3.79}$$
$$\beta_{AA} = \sin^{-1}\left(\frac{v}{V}\right) \tag{3.80}$$

This allows for the calculation of the aerodynamic coefficients, which should be the $\mathbb{F}_a a$ frame.

3.7. Linearised Rotational Motion

This section shall linearise the rotational equations of motion. The reason that this is required is that classical control methods such as the linear quadratic regulator (LQR) or other control methods that also require that the system has equations of motion that are Linear Time Invariant (LTI) (Mooij, 1998). The eigenvalues and eigenvectors that describe the open-loop motion can only be calculated if the system is LTI. The rest of this section is based upon equations found in (Mooij, 2013), but the equations here are limited to the rotational motion only.

The outcome of the linearisation is a set of coupled linear differential equations, written as (Mooij, 2013)

$$\Delta\dot{\mathbf{x}} = \mathbf{A}\Delta\mathbf{x} + \mathbf{B}\Delta\mathbf{u} \tag{3.81}$$

where $\Delta\mathbf{x} \in \mathbb{R}^n$ is the error state vector, $\Delta\mathbf{u} \in \mathbb{R}^m$ is thenoise input vector, $\mathbf{A} \in \mathbb{R}^n \times \mathbb{R}^n$ the system or state matrix, and $\mathbf{B} \in \mathbb{R}^n \times \mathbb{R}^m$ the input matrix. The system and input matrix can be derived from the non-linear equations of motions with

$$\mathbf{A} = \begin{bmatrix} \frac{\delta f_1}{\delta x_1} & \cdots & \frac{\delta f_1}{\delta x_n} \\ \vdots & \vdots & \vdots \\ \frac{\delta f_n}{\delta x_1} & \cdots & \frac{\delta f_n}{\delta x_n} \end{bmatrix} \tag{3.82}$$

and

$$\mathbf{B} = \begin{bmatrix} \frac{\delta f_1}{\delta u_1} & \cdots & \frac{\delta f_1}{\delta u_n} \\ \vdots & \vdots & \vdots \\ \frac{\delta f_n}{\delta u_1} & \cdots & \frac{\delta f_n}{\delta u_n} \end{bmatrix} \tag{3.83}$$

The functions are evaluated at the nominal conditions.

The linearisation of the states and controls is done by using Taylor series. This method is a useful tool when it is required when differentiable functions need to be linearised. if there is a function y with variable x then the Taylor series look like

$$y = f(x_0) + f'(x_0)(x_0 - x) + \frac{f''(x_0)}{2!}(x_0 - x)^2 + \cdots + \frac{f^n(x_0)}{n!}(x_0 - x)^n \tag{3.84}$$

It is usually the case that the series are stopped when $n = 1$, higher derivatives are then depicted with $\mathcal{O}(n)$, meaning that the higher-ups are ignored. Higher order-terms can often be ignored due to their lower value, which is also the case here. It is also usually the case that more variables are involved, the Taylor series then become

$$y = f(x_0) + \sum_i \frac{\partial x_0}{\partial x_i}\Delta x_i + \frac{1}{2}\sum_i\sum_j \frac{\partial^2 x_0}{\partial x_i \partial x_j}\Delta x_i \Delta x_j \tag{3.85}$$

where i and j are any of the variables present within x. The continuation of this series can easily be deduced from the single state Taylor expansion.

The rotational motion is thus linearised with the first-order Taylor series. The basic form is

$$p = p_0 + \Delta p \tag{3.86}$$
$$q = q_0 + \Delta q \tag{3.87}$$
$$r = r_0 + \Delta r \tag{3.88}$$
$$\alpha = \alpha_0 + \Delta\alpha \tag{3.89}$$
$$\beta = \beta_0 + \Delta\beta \tag{3.90}$$
$$\sigma = \sigma_0 + \Delta\sigma \tag{3.91}$$

The subscript 0 means in this context the nominal state. The nominal rates of the rotational vector are calculated with
and the nominal state of the aerodynamical angles are given by the guidance system. It can be said that we are not per se interested in the states itself, but in the deviations from the nominal state, since the goal is to minimize these deviations. Having nominal states also means that there are nominal aerodynamic forces and moments, and deviations thereof. How to calculate the deviations from the states is shown below:

$$\Delta\dot{p} = \frac{\Delta M_x}{I_{xx}} \tag{3.92}$$
$$\Delta\dot{q} = \frac{\Delta M_y}{I_{yy}} \tag{3.93}$$
$$\Delta\dot{r} = \frac{\Delta M_z}{I_{zz}} \tag{3.94}$$
$$\Delta\dot{\alpha} = \Delta q - \frac{1}{mV_0}\Delta L - \frac{g_0}{V_0}\cos\gamma_0 \sin\sigma_0 \Delta\sigma \tag{3.95}$$
$$\Delta\dot{\beta} = \sin\alpha_0 \Delta p - \cos\alpha_0 \Delta r - \frac{\Delta S}{mV_0} - \frac{g_0}{V_0}\cos\gamma_0 \cos\sigma_0 \Delta\sigma \tag{3.96}$$
$$\Delta\dot{\sigma} = -cos\alpha_0 \Delta p - \sin\alpha_0 \Delta r - \left(\frac{L_0}{mV_0} - \frac{g_0}{V_0}\cos\gamma_0 \cos\sigma_0\right)\Delta\beta \ldots$$
$$+ \frac{\tan\gamma_0}{mV_0}(\sin\sigma_0 \Delta L + \cos\sigma_0 L_0 \Delta\sigma + \cos\sigma_0 \Delta S) \tag{3.97}$$

The nominal position (R_0), velocity (V_0), and flight path angle (γ_0) are calculated seperately and it is assumed that these variables remain constant.

These equations contain the force and moment variations that can be converted in functions that have state and control variables. The Equations 3.114-3.113 show the effect of different aerodynamic components on the coefficients. The drag-, side- and lift-force are functions of the state and control inputs of the spacecraft. The functions are linearised into ΔD, ΔS, and ΔL and a few assumptions are made along the way. The drag caused by the rudder only becomes noticeable when the deflection is $30°$ or higher, it is assumed that the deflection does not reach that number, so that coefficient is neglected. The elevon does have a non-negligible contribution at angles above ten degrees and lower speeds. This however hardly happens according to (Mooij, 1998) and it can therefore be ignored. The change in drag can then be written as

$$\Delta D = \frac{\partial C_D}{\partial \alpha} \bar{q}_0 S_{ref} \Delta\alpha \tag{3.98}$$

The sideslip and lift forces also ignore the deflection of the control surfaces resulting in

$$\Delta S = \frac{\partial C_S}{\partial \beta} \bar{q}_0 S_{ref} \Delta\beta \tag{3.99}$$

$$\Delta L = \frac{\partial C_L}{\partial \alpha} \bar{q}_0 S_{ref} \Delta\alpha \tag{3.100}$$

The variation of moments consist of two parts , that of the aerodynamic one and of that of the moment generated by the reaction control system

$$\Delta M_x = \Delta\mathcal{L} + \Delta M_{T,x} \tag{3.101}$$

$$\Delta M_y = \Delta\mathcal{M} + \Delta M_{T,y} \tag{3.102}$$

$$\Delta M_z = \Delta\mathcal{N} + \Delta M_{T,z} \tag{3.103}$$

The thrusters are already control variables so these do not have to be converted. The aerodynamic contributions however is not and has to be treated in the same way as the forces just calculated. The control surfaces this time are not neglected since that would nullify the effect of the actuators. The result is

$$\Delta\mathcal{L} = \frac{\partial C_l}{\partial \beta} \bar{q}_0 S_{ref} b_{ref} \Delta\beta + \frac{\partial C_l}{\partial \delta_a} \bar{q}_0 S_{ref} b_{ref} \Delta\delta_a \tag{3.104}$$

$$\Delta\mathcal{M} = \frac{\partial C_m}{\partial \alpha} \bar{q}_0 S_{ref} b_{ref} \Delta\alpha + \frac{\partial C_m}{\partial \delta_e} \bar{q}_0 S_{ref} b_{ref} \Delta\delta_e \tag{3.105}$$

$$\Delta\mathcal{N} = \frac{\partial C_n}{\partial \beta} \bar{q}_0 S_{ref} b_{ref} \Delta\beta + \frac{\partial C_n}{\partial \delta_r} \bar{q}_0 S_{ref} b_{ref} \Delta\delta_r + \frac{\partial C_n}{\partial \delta_e} \bar{q}_0 S_{ref} b_{ref} \Delta\delta_e \tag{3.106}$$

All the equations are combined below for clarity

$$\Delta\dot{p} = \frac{1}{I_{xx}} \left(\left(\frac{\partial C_l}{\partial \beta} \Delta\beta + \frac{\partial C_l}{\partial \delta_a} \Delta\delta_a \right) \bar{q}_0 S_{ref} b_{ref} + \Delta M_{T,x} \right) \tag{3.107}$$

$$\Delta\dot{q} = \frac{1}{I_{yy}} \left(\left(\frac{\partial C_m}{\partial \alpha} \Delta\alpha + \frac{\partial C_m}{\partial \delta_e} \Delta\delta_e \right) \bar{q}_0 S_{ref} b_{ref} + \Delta M_{T,y} \right) \tag{3.108}$$

$$\Delta\dot{r} = \frac{1}{I_{zz}} \left(\left(\frac{\partial C_n}{\partial \beta} \Delta\beta + \frac{\partial C_n}{\partial \delta_r} \Delta\delta_r + \frac{\partial C_n}{\partial \delta_e} \Delta\delta_e \right) \bar{q}_0 S_{ref} b_{ref} + \Delta M_{T,z} \right) \tag{3.109}$$

$$\Delta\dot{\alpha} = \Delta q - \frac{1}{mV_0} \frac{\partial C_L}{\partial \alpha} \bar{q}_0 S_{ref} \Delta\alpha - \frac{g_0}{V_0} \cos\gamma_0 \sin\sigma_0 \Delta\sigma \tag{3.110}$$

$$\Delta\dot{\beta} = \sin\alpha_0 \Delta p - \cos\alpha_0 \Delta r - \frac{1}{mV_0} \frac{\partial C_S}{\partial \beta} \bar{q}_0 S_{ref} \Delta\beta - \frac{g_0}{V_0} \cos\gamma_0 \cos\sigma_0 \Delta\sigma \tag{3.111}$$

$$\begin{aligned} \Delta\dot{\sigma} = &- cos\alpha_0 \Delta p - \sin\alpha_0 \Delta r + \left(\frac{g_0}{V_0} \cos\gamma_0 \cos\sigma_0 - \frac{L_0}{mV_0} \right) \Delta\beta \ldots \\ &+ \frac{\tan\gamma_0}{mV_0} \left(\sin\sigma_0 \frac{\partial C_L}{\partial \alpha} \bar{q}_0 S_{ref} \Delta\alpha + \cos\sigma_0 L_0 \Delta\sigma + \cos\sigma_0 \frac{\partial C_S}{\partial \beta} \bar{q}_0 S_{ref} \Delta\beta \right) \end{aligned} \tag{3.112}$$

these equations are combined into a state space system in Appendix A.

3.8. Reference Vehicles

This section deals with the basic characteristics of both RV-W, that are required for the further development of the guidance and control method.

3.8.1. HORUS

The first reference vehicle is the HORUS. The reason for choosing this spaceplane is quite straightforward, a lot of research has been performed in the astrodynamics department of the Delft University of Technology. this research includes, but is not limited to, the following: (Mooij, 1998)(Papp, 2014)(De Ridder, 2009)(Lammens, 2015). The available data generated by these references can be used to validate the simulation. Basic characteristics of HORUS are given in Table 3.4.
Having a proper aerodynamic database is important in terms of how the vehicle reacts to the surrounding atmosphere, and to what moment is generated by the control surfaces when it is deflected by a certain amount. How to calculate the coefficients for the HORUS is given by (Müller, 1988) and is repeated here for good measure

$$C_L = C_{L_0} + \Delta C_{L_{e,l}} + \Delta C_{L_b} + \Delta C_{L_{e,r}} \tag{3.113}$$

$$C_D = C_{D_0} + \Delta C_{D_{r,l}} + \Delta C_{D_{e,l}} + \Delta C_{D_b} + \Delta C_{D_{r,r}} + \Delta C_{D_{e,r}} - C_{D_h} \tag{3.114}$$

$$C_S = \Delta C_{S_{r,l}} + \Delta C_{S_{e,l}} + \Delta C_{S_{r,r}} + \Delta C_{S_{e,r}} + \left[\Delta\left(\frac{\partial C_S}{\partial \beta}\right)_0 + \Delta\left(\frac{\partial C_S}{\partial \beta}\right)_{e,l} + \Delta\left(\frac{\partial C_S}{\partial \beta}\right)_{e,r}\right]\beta \tag{3.115}$$

$$C_m = C_{m_0} + \Delta C_{m_{e,l}} + \Delta C_{m_b} + \Delta C_{m_{e,r}} \tag{3.116}$$

$$C_n = \Delta C_{n_{r,l}} + \Delta C_{n_{e,l}} + \Delta C_{n_{e,r}} + \Delta C_{n_{r,r}} + \ldots \left[\Delta\left(\frac{\partial C_n}{\partial \beta}\right)_0 + \Delta\left(\frac{\partial C_n}{\partial \beta}\right)_{r,l} + \Delta\left(\frac{\partial C_n}{\partial \beta}\right)_{e,l} + \Delta\left(\frac{\partial C_n}{\partial \beta}\right)_{e,r} + \Delta\left(\frac{\partial C_n}{\partial \beta}\right)_{r,r}\right]\beta \tag{3.117}$$

$$C_l = \Delta C_{l_{e,l}} + \Delta C_{l_{e,r}} + \left(\frac{\partial C_S}{\partial \beta}\right)_0 \beta \tag{3.118}$$

All the functions are either a function of the angle of attack (α), the Mach number (M), and for cases that involve a control surface, that specific deflection angle (δ). A more specific list is given in (Mooij, 2013). Both the lift and the drag coefficients can be simplified to assume that the lift and drag coefficients are close to the trimmed state. The lift coefficient is therefore simply reduced to

$$C_L = C_{L_{trim}} \tag{3.119}$$

The drag coefficient becomes

$$C_D = C_{D_{trim}} - C_{D_h} + C_{D_{SPB}} \tag{3.120}$$

Viscous flows at altitudes below 20 kilometres reduce the drag and the speedbrake is present during the subsonic phase, where both the rudders that act as the speedbrakes are set to 15°. Both coefficients have been generated using the tables from De Ridder (2009). The aerodynamic data used in the simulation in De Ridder (2009) were made using linear interpolation. Optimisation programs are unable to deal with the sudden changes in the jacobian that are created if inputs use discrete data. A new method to calculate the aerodynamic coefficients had therefore be devised. Three methods were tested: third degree polynomials, cubic splines, and hermite cubic splines. The data is divided into a subsonic, transonic, and supersonic part. The challenge is therefore not only to find a right fit, but also to match the values and derivatives at the transition phases. The polynomials were generated with some overlap of the adjacent tables to increase the fit. Both the fit and the derivative are bad, but the computation time required is very small. The cubic splines were more successful with both the fit and the derivative. The best result was found using Hermite splines.The coefficients calculated this way are approximately twenty times more expensive than with linear interpolation. The results for a continuous angle of attack of 7.5° can be seen in Figure 3.13. Both the cubic splines and Hermite cubic splines had a continuous first derivative.

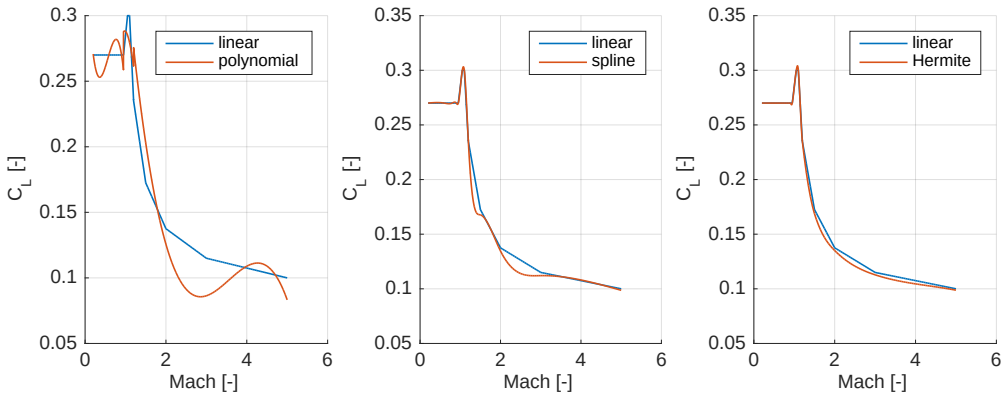


Figure 3.13: Three different interpolation techniques to calculate the lift coefficient for an angle of attack of 7.5 degrees.

Table 3.4: Some characteristics of HORUS-2B. (Helmersson, 1988)

Total vehicle length	25 m
wing span	13 m
wing chord	23 m
wing area	110 m^2
re-entry and landing mass	26029 kg
$\tilde{I}_{xx}$	119,000 kgm^2
$\tilde{I}_{yy}$	769,000 kgm^2
$\tilde{I}_{zz}$	806,000 kgm^2

The aerodynamic database is split into three segments, supersonic, transonic, and subsonic. Each database is limited by a certain angle of attack, with the transonic database being limited to an angle of attack of $10°$, which coincides with the structural limitation during this phase (Helmersson, 1988). The corridor is thus narrow in the transonic region, in which the controllability of the vehicle is limited. The $C_{D_{trim}}$ coefficients are given as a function of L/D. There is an asymptote towards infinity at an edge moving from the lower transonic angle of attack at around $1°$ degree towards $5°$ at the end of the supersonic region where the coefficients switch sign from positive to negative. This region should be avoided to improve the optimal solution convergence. The angle of attack corridor is limited to the data points given in Table 3.5, and the points are connected using Hermite interpolation. Another problem that this poses is that pseudospectral methods are not able to cope with varying boundaries, therefore, the angle of attack corridor is mapped to the $[0, 1]$ domain by performing linear scaling

$$\tilde{\alpha} = \frac{\alpha - \alpha_{min}}{\alpha_{max} - \alpha_{min}} \tag{3.121}$$

3.8.2. X-38

The second reference vehicle is the X-38, of which an extensive aerodynamic database is available. The X-38 is depicted in Figure 3.15 and the required characteristics are shown in Table 3.6.

The X-38 is controlled by moving 4 control surfaces. Two of which are the elevon, the other two are rudders. There is a left- and right-version of each one. The elevator-,aileron-, rudder-, and speedbrake-deflection

Table 3.5: Angle-of-attack limits by Mach number (De Ridder, 2009).

Mach	0	0.75	0.95	1.2	1.5	2.0	3.0	5.0
α_{max}	30.0	30.0	10.0	10.0	15.5	22.5	32.5	45.0
α_{min}	0.0	0.0	3.0	3.0	3.0	3.5	6.5	10.0

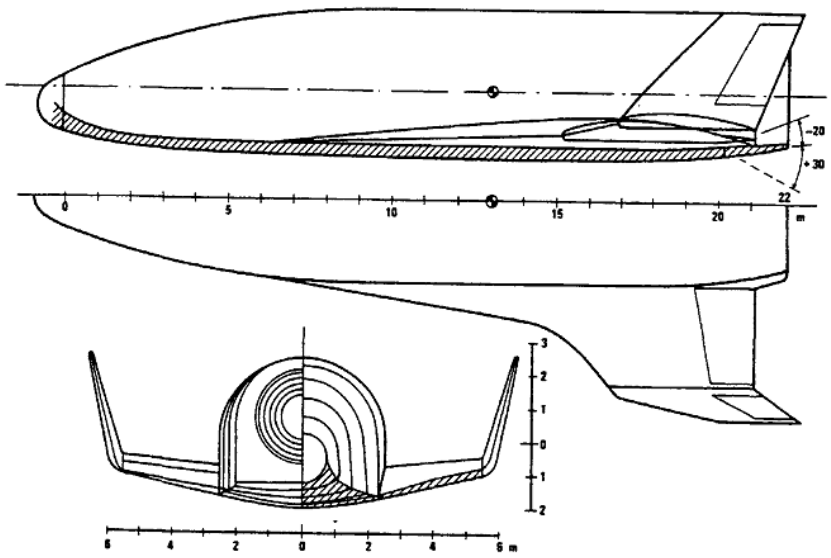


Figure 3.14: The Horus 2B-7 reference vehicle (Müller, 1988)

are calculated with

$$\delta_e = 0.5(\delta_{e,l} + \delta_{e,r}) \tag{3.122}$$
$$\delta_a = 0.5(\delta_{e,l} - \delta_{e,r}) \tag{3.123}$$
$$\delta_r = 0.5(\delta_{r,l} + \delta_{r,r}) \tag{3.124}$$
$$\delta_{spbr} = 0.5(\delta_{r,l} - \delta_{r,r}) \tag{3.125}$$
$$\tag{3.126}$$

The speedbrake is already used during the trajectory generation phase. The other three will be used in the attitude control chapter.

The lift and drag coefficients are calculated with (Dassault, 2001)

$$C_L = C_{L20s} + \Delta C_{L,e} + \Delta C_{L,spbr} \tag{3.127}$$
$$C_D = C_{D20s} + \Delta C_{D,e} + \Delta C_{D,spbr} \tag{3.128}$$

The basic lift coefficient assumes that the elevons have an extension of 20 degrees during the terminal area. The other two coefficients are based on the extension of the elevon and the speedbrake, both coefficients are ignored for now, due to their lower values. All the coefficients also assume that the hatch that releases the parafoil is closed. The aerodynamic coefficients used for attitude control are (Dassault, 2001),

$$C_S = (C_{S_{20s}}(\alpha, M) + \Delta C_{S_{\delta_e}}(\alpha, M, \delta_e))\beta + \Delta C_{S_{\delta_a}}(\alpha, M, \delta_e)\Delta a + \Delta C_{S_{\delta_r}}(\alpha, M)\Delta r \tag{3.129}$$
$$C_l = (C_{m_{20s}}(\alpha, M) + \Delta C_{l_{\delta_e}}(\alpha, M, \delta_e))\beta + \Delta C_{l_{\delta_a}}(\alpha, M, \delta_e)\Delta a + \ldots \tag{3.130}$$
$$+ \Delta C_{l_{\delta_r}}(\alpha, M)\Delta r + (C_{l_p}(M)p + C_{l_r}(M)r)\frac{L_{ref}}{V}$$
$$C_m = C_{l_{20s}}(\alpha, M) + \Delta C_{m_{\delta_e}}(\alpha, M, \delta_e) + \Delta C_{m_{\delta_{sb}}}(\alpha, M, \delta_{sb}) + C_{m_q}(M)\frac{L_{ref}}{V} \tag{3.131}$$
$$C_n = C_{n_{20s}}(\alpha, \beta, M, \delta_e) + \Delta C_{n_{\delta_a}}(\alpha, M, \delta_e)\Delta a + \Delta C_{n_{\delta_r}}(\alpha, M)\Delta r + (C_{n_p}(M)p + C_{n_r}(M)r)\frac{L_{ref}}{V} \tag{3.132}$$

It can no longer be assumed that the sideslip angle β is zero when attitude control is introduced, as was done when calculating the reference trajectory. The other coefficients presented here should be controlled such that β is minimised at all times. The cubic Hermite interpolation method is used to calculate the values of the coefficients, as was done with the HORUS. The angle-of-attack corridor is constant, the minimum

Table 3.6: Some characteristics of X38 required for attitude control.

Reference surface area	24.15 m^2 (Georgie et al., 2002)
Reference length	8.37 m (Georgie et al., 2002)
re-entry and landing mass	7977 kg (Georgie et al., 2002)
$\tilde{I}_{xx}$	$10,968$ kgm^2 (Georgie et al., 2002)
$\tilde{I}_{yy}$	$35,116$ kgm^2 (Georgie et al., 2002)
$\tilde{I}_{zz}$	$39,589$ kgm^2 (Georgie et al., 2002)
Δx_{cp}	0.5154 m (Juliana, 2003)
Δy_{cp}	0.00112 m (Juliana, 2003)
Δz_{cp}	0.3069 m (Juliana, 2003)
$\dot{\delta}_{defl}$	30 deg/s (Juliana, 2003)

value is $-14°$ and the maximum value is $40°$. The moment reference point is not on the X-axis center-line, but slightly to the right. A constant non-zero rudder deflection is therefore required when stabilising the RV-W.

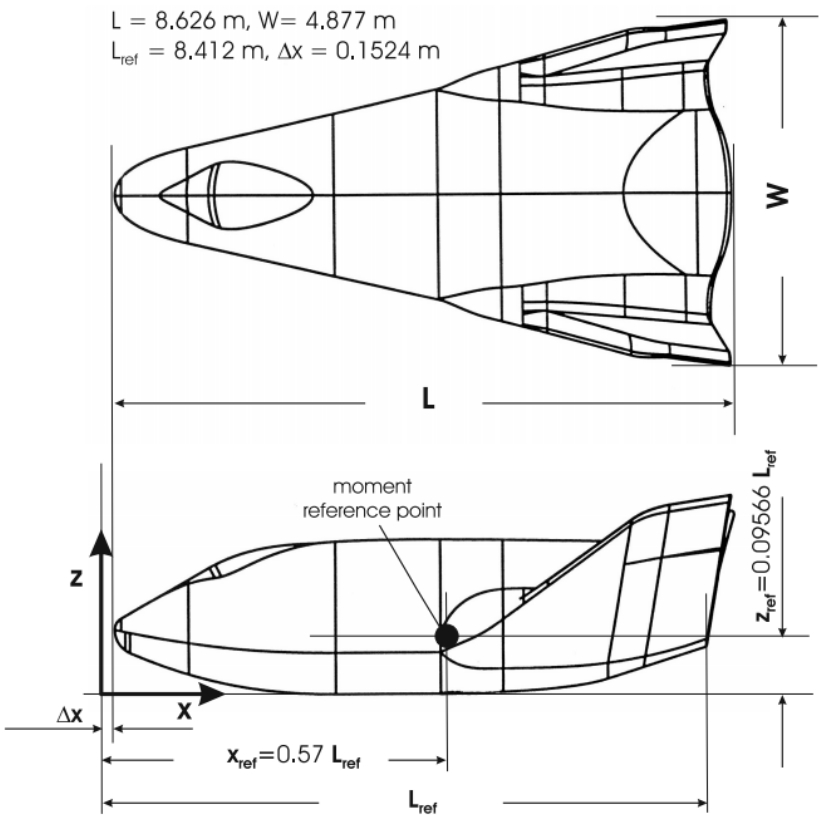


Figure 3.15: Shape of the X-38 vehicle (Georgie et al., 2002).

4

Pseudospectral Methods

This chapter introduces the concept of pseudospectral methods and how it can be applied to problems with non-linear differential equations. Chapter 5 shall give an introduction to various numerical methods, all of which shall be used here.

Two branches exist in optimal control theory, the direct method and the indirect method (Rao, 2009). Both have different philosophies as how to solve a problem, but both convert the problem into a non-linear programming problem (NLP). NLP is a method that discretises a function into several nodes. These nodes are then connected using differential equations. Direct methods optimise the cost function by changing the control inputs directly. Indirect methods solve the problem by converting it to a boundary value problem in which there are several nodes that all should satify the boundary conditions or interior point conditions. The calculus of variations is used to transform the optimal control problem into a Hamilton boundary-value problem. Direct methods have the advantage that the problem transcription is usually small which reduces the complexity of the solver. Two major downsides are that the computational power increases rapidly when the number of nodes or variables are increased, and that it cannot be calculated that the solution is actually the optimal solution. The introduction to optimal control is given in Section 4.1 and the introduction to NLP in Section 4.2.

The functions of this method are based on Lagrange polynomials and the selection of the location of the nodes is based on Gaussian quadrature rules. This provides a faster, exponential convergence rate compared to more traditional direct methods. It is a concept that was first introduced by Chebyshev in the 1980s. No analytical solutions have been found as of yet to solve a feedback closed-loop simulation, this is because the Hamilton-Jacobi-Bellman equation has to be solved. A problem which becomes harder to solve analytically as the problem increases in size, since the dimensions of the equation increases. An alternative to find this closed-loop solution is to solve the problem online and open-loop. The fast convergence rates of PSM is capable of generating the solution real-time. However, two disadvantages are that the method cannot deal with problems that have discontinuities or singular arcs, but both are not present during the re-entry of the atmosphere.

Diagram 4.1 gives an overview of the processes involved to calculate the solution using PSM. The rest of this introduction shall introduce the concepts mentioned in the boxes, and it will refer to the section where information about it can be found if required.

There are a few statements required at the beginning of the trajectory of calculating the solution, these are given in the ”Set variables”-box. The main part that defines the problem is the non-linear dynamic equations and the cost function related to the states calculated with the dynamic equations, the variables. These variables are limited by certain boundaries, the lower and upper limit, those are also given here. Another variable not directly associated with the dynamic equations is time, this can either be set fixed or loose by providing the boundaries. The variables also require a starting point, and a final point which is an approximate of the solution.The last important part is that the number of nodes are given, this mainly defines the accuracy of the solution and the computation time. These statements are the minimum requirement in order to be able to solve the non-linear problem. It is possible to add extra elements such as

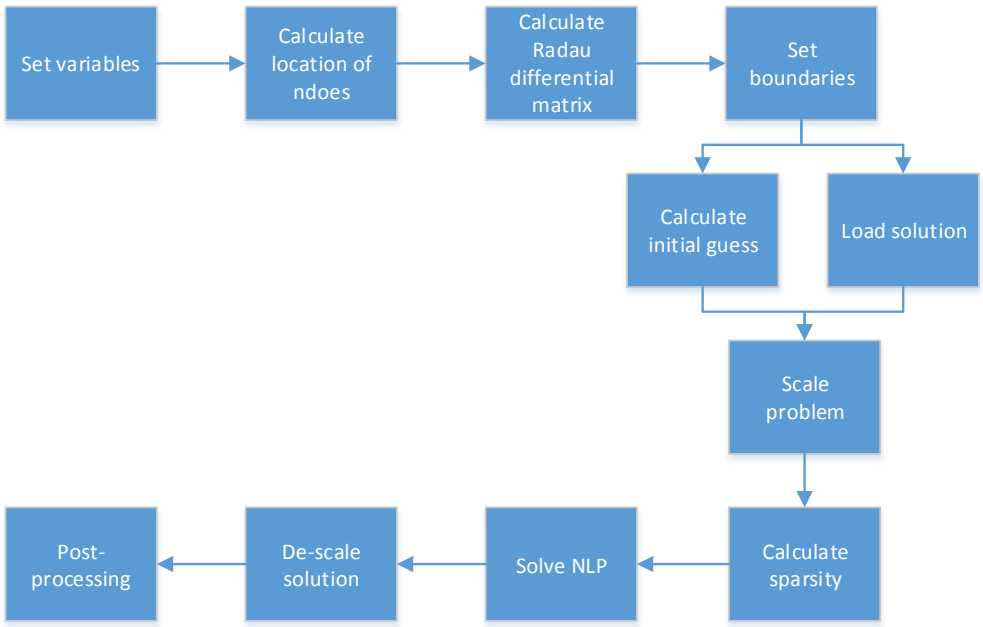


Figure 4.1: A diagram that shows the processes involved to calculate a solution using pseudospectral methods.

dynamic constraints, maximum acceleration, or by changing the accuracy of the solution. The rest of the solving procedure is independent from the user.

The next blocks are dependent on the location of the nodes within the $[-1, 1]$ domain are calculated. There are various types of node locations, these are found in Section 4.3, and the derivative of each point is called the Radau differential matrix, how to calculate both is found in Section 4.4, this is a section that also introduces the basic mathematical concepts of PSM. All of the nodes each has its own set of variables, and these are limited by the lower and upper boundaries, these can now be specified. The set of variables at each node also has a certain initial state which has to be approximated beforehand in order to find the correct local optimum. This approximation can be calculated by using linear interpolation or by having an old solution which is somewhat close to the optimal solution. Using an old solution greatly reduces the required computation time.

It is required for a problem to be scaled if the variables vary with several orders of magnitude, it is then too ill-conditioned. The scaling procedure is given in Section 4.6. There are two versions of the program, one with and one without scaling. The last step before the computation of the solution is that not all of the variables, or their derivatives, of the problem need to be calculated, how this is determined and used is found in Section 4.7. The non-linear problem can now be solved using SNOPT (Gill et al., 2008). The scaling of the problem is removed once the solution has been found, and then finally the nodes are connected to each other using cubic spline interpolation during post-processing which then allows for the figures to be created.

The chapter ends with two examples in Section 4.8 to give a bit of feeling to the reader of what can be achieved when using PSM.

4.1. Introduction to Optimal Control

It is important to discuss the basics of optimal control before PSM is further investigated. The objective of optimal control is to find the state and control of a certain problem that optimises the performance index while staying within the boundaries of the constraints imposed on the system (Garg, 2011). This section

shows how this problem is defined in an analytical way, to better understand the following sections where the optimal control problem is discretised.

An optimal control problem can be written down mathematically in the following way

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

subject to the following dynamic constraints

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

the boundarx conditions

$$\phi(\mathbf{x}(t_0), t_0, \mathbf{x}(t_f), t_f) = \mathbf{0} \tag{4.3}$$

and the inequality path constraints

$$\mathbf{C}(\mathbf{x}(t), \mathbf{u}(t), t) \leq \mathbf{0} \tag{4.4}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ is the state, $\mathbf{u}(t) \in \mathbb{R}^m$ is the control, and $t \in [t_0, t_f]$ is the independent variable. The cost function consists of two terms: First is the Mayer cost, $\Phi : \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$, and second, the Lagrangian $g : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R} \to \mathbb{R}$. The dynamic function $\mathbf{f}$ has dimension $\mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R} \to \mathbb{R}^n$ which defines the dimensions of the dynamic constraints, $\mathbf{C} : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R} \to \mathbb{R}^s$ and the path constraints, and $\phi : \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^q$ defines the boundary conditions.

The performance index is said to be minimised if at least the first-order necessary conditions of Equations 4.1-4.4 are met. This means that a variation of the cost function at an optimal path $\mathbf{x}^*$ with a small $\delta\mathbf{x}$ is 0, or

$$\delta J(\mathbf{x}^*, \delta\mathbf{x}) = \mathbf{0} \tag{4.5}$$

This simple example only holds for an unconstrained problem. The cost function described above is subject to an undefined number of constraints, which should be combined with the performance index to get the augmented cost function. This is defined as

$$\begin{aligned} J_a =& \Phi(\mathbf{x}(t_0), t_0, \mathbf{x}(t_f), t_f) - \varphi^T \phi(\mathbf{x}(t_0), t_0, \mathbf{x}(t_f), t_f) \\ &+ \int_{t_0}^{t_f} \left[g(\mathbf{x}(t), \mathbf{u}(t), t) - \lambda^T(t)(\dot{\mathbf{x}}(t) - \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) - \gamma^T(t)\mathbf{C}(\mathbf{x}(t), \mathbf{u}(t), t) \right] dt \end{aligned} \tag{4.6}$$

where $\varphi \in \mathbb{R}^n$, $\lambda(t) \in \mathbb{R}^q$, and $\gamma(t) \in \mathbb{R}^s$ are the Lagrange multipliers. $\lambda(t)$ is also called the costate or adjoint variable, it can be used to verify the optimality of the solution. The following set of equations is then formed by applying the first-order optimality conditions to J_a (Garg, 2011):

$$\dot{\mathbf{x}} = \mathbf{f} \tag{4.7}$$

$$\frac{\partial g}{\partial \mathbf{x}} + \lambda^T \frac{\partial \mathbf{f}}{\partial \mathbf{x}} - \gamma^T \frac{\partial \mathbf{C}}{\partial \mathbf{x}} = -\dot{\lambda} \tag{4.8}$$

$$\frac{\partial g}{\partial \mathbf{u}} + \lambda^T \frac{\partial \mathbf{f}}{\partial \mathbf{u}} - \gamma^T \frac{\partial \mathbf{C}}{\partial \mathbf{u}} = 0 \tag{4.9}$$

$$-\frac{\partial \Phi}{\partial \mathbf{x}(t_0)} + \varphi^T \frac{\partial \phi}{\partial \mathbf{x}(t_0)} = \lambda^T(t_0) \tag{4.10}$$

$$\frac{\partial \Phi}{\partial \mathbf{x}(t_f)} - \varphi^T \frac{\partial \phi}{\partial \mathbf{x}(t_f)} = \lambda^T(t_f) \tag{4.11}$$

$$\frac{\partial \Phi}{\partial t_0} - \varphi^T \frac{\partial \Phi}{\partial t_0} = (g + \lambda^T \mathbf{f} - \gamma^T \mathbf{C})|_{t=t_0} \tag{4.12}$$

$$-\frac{\partial \Phi}{\partial t_f} - \varphi^T \frac{\partial \Phi}{\partial t_f} = (g + \lambda^T \mathbf{f} - \gamma^T \mathbf{C})|_{t=t_f} \tag{4.13}$$

$$\phi = \mathbf{0} \tag{4.14}$$

These equations can be simplified by introducing the Hamiltonian, which is the integral part of the cost function. The augmented Hamiltonian is defined as

$$H(\mathbf{x}(t),\mathbf{u}(t),\lambda(t),\gamma(t),t) = g(\mathbf{x}(t),\mathbf{u}(t),t) - \lambda^T(t)(\dot{\mathbf{x}}(t) - \mathbf{f}(\mathbf{x}(t),\mathbf{u}(t),t) - \gamma^T(t)\mathbf{C}(\mathbf{x}(t),\mathbf{u}(t),t) \quad (4.15)$$

and thus

$$\dot{\mathbf{x}}^T(t) = \frac{\partial H}{\partial \lambda} \quad (4.16)$$

$$\dot{\lambda}^T(t) = -\frac{\partial H}{\partial \mathbf{x}} \quad (4.17)$$

$$\mathbf{0} = \frac{\partial H}{\partial \mathbf{u}} \quad (4.18)$$

$$\lambda^T(t_0) = -\frac{\partial \Phi}{\partial \mathbf{x}(t_0)} + \varphi^T \frac{\partial \phi}{\partial \mathbf{x}(t_0)} \quad (4.19)$$

$$\lambda^T(t_f) = \frac{\partial \Phi}{\partial \mathbf{x}(t_f)} - \varphi^T \frac{\partial \phi}{\partial \mathbf{x}(t_f)} \quad (4.20)$$

$$H|_{t=t_0} = \frac{\partial \Phi}{\partial t_0} - \varphi^T \frac{\partial \Phi}{\partial t_0} \quad (4.21)$$

$$H|_{t=t_f} = \frac{\partial \Phi}{\partial t_f} - \varphi^T \frac{\partial \Phi}{\partial t_f} \quad (4.22)$$

$$\phi = \mathbf{0} \quad (4.23)$$

The Lagrange multiplier γ can already be further defined using the complementary slackness condition.

$$\gamma_i(t) = 0 \quad \text{when} \quad C_i(\mathbf{x}(t),\mathbf{u}(t),t) < \mathbf{0} \quad \text{for} \quad 1 \leq i \leq s \quad (4.24)$$

$$\gamma_i(t) < 0 \quad \text{when} \quad C_i(\mathbf{x}(t),\mathbf{u}(t),t) = \mathbf{0} \quad \text{for} \quad 1 \leq i \leq s \quad (4.25)$$

This means that if $C_i < 0$ than the path constraint C_i is not active and the constraint is ignored since $\gamma_i(t) = 0$. The negative γ_i results in the fact that the cost can only be improved by violating the constraint. Slackness conditions are used to create a feasible solution. The slack condition is minimised to zero during the optimisation

The first-order necessary conditions defines that the set of equations above result in the derivative of the cost function being 0 at $\mathbf{y}^*$. The problem however is that this derivative does not determine whether the cost function is at a minimum, maximum, or saddle point extremal. This can be solved by inspecting the second-order necessary conditions, however, solving this analytically would result in pages filled with equations. The lowest extremal is therefore chosen as the solution. This is done numerically, and how this is done is shown in Section 4.2. The information given here should provide a solid basis to understand optimisation to better grasp PSM.

4.2. Numerical Optimisation

The previous section described optimal control as a continuous problem. This section shows how a similar problem can be solved by creating a non-linear programming problem (NLP), which is the discretised form of the continuous problem.

The unconstrained cost function is written once again as $J(\mathbf{x})$. The state vector $\mathbf{x}$ is discretised into n points so $\mathbf{x} = (x_1, \dots, x_n)^T \in \mathbb{R}^n$ and $\mathbf{x}^*$ is the state vector that has the optimal solution such that a small deviation from the optimal state, $\bar{\mathbf{x}} = \mathbf{x}^* + \delta\mathbf{x}$, yields the following definition

$$J(\bar{\mathbf{x}}) > J(\mathbf{x}^*) \quad (4.26)$$

The first-order necessary condition is stated as

$$\mathbf{g}(\mathbf{x}^*) = \mathbf{0} \quad (4.27)$$

where $\mathbf{g}(\mathbf{x})$ is defined as

$$\mathbf{g}(\mathbf{x}) \equiv \nabla_x J^T = \begin{pmatrix} \frac{\partial J}{\partial x_1} \\ \frac{\partial J}{\partial x_2} \\ \vdots \\ \frac{\partial J}{\partial x_n} \end{pmatrix} \tag{4.28}$$

This gradient condition consists of n conditions to determine the n unknown variables which is then a solution for the first necessary condition, but it still only defines the extremal which could be a minimum, maximum, or saddle point. The sufficient condition is used to determine which of these three extrema it is, this is determined by first calculating the Taylor series expansion around $\bar{\mathbf{x}}$ to find the second derivative.

$$\mathbf{J}(\bar{\mathbf{x}}) = J(\mathbf{x}^*) + \mathbf{g}^T(\mathbf{x}^*)(\bar{\mathbf{x}} - \mathbf{x}^*) + \frac{1}{2}(\bar{\mathbf{x}} - \mathbf{x}^*)^T\mathbf{H}(\mathbf{x}^*)(\bar{\mathbf{x}} - \mathbf{x}^*) + \mathcal{O}^3 \tag{4.29}$$

where H is the Hessian matrix which is defined as

$$\mathbf{H}(\mathbf{x}) \equiv \nabla_{xx} J \equiv \frac{\partial^2 J}{\partial \mathbf{x}^2} = \begin{bmatrix} \frac{\partial^2 J}{\partial x_1^2} & \frac{\partial^2 J}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 J}{\partial x_1 \partial x_n} \\ \frac{\partial^2 J}{\partial x_2 \partial x_1} & \frac{\partial^2 J}{\partial x_2^2} & \cdots & \frac{\partial^2 J}{\partial x_2 \partial x_n} \\ \vdots & & \ddots & \\ \frac{\partial^2 J}{\partial x_n \partial x_1} & \frac{\partial^2 J}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 J}{\partial x_n^2} \end{bmatrix} \tag{4.30}$$

Equation 4.29 can be partially ignored, since $\bar{\mathbf{y}} = \mathbf{y}^*$, this means that the higher order terms can be ignored, and the necessary condition is equal to 0, hence

$$\mathbf{J}(\bar{\mathbf{x}}) = J(\mathbf{x}^*) + \frac{1}{2}(\bar{\mathbf{x}} - \mathbf{x}^*)^T\mathbf{H}(\mathbf{x}^*)(\bar{\mathbf{x}} - \mathbf{x}^*) \tag{4.31}$$

which is followed by Equation 4.26

$$\begin{aligned} J(\mathbf{x}^*) + \frac{1}{2}(\bar{\mathbf{x}} - \mathbf{x}^*)^T\mathbf{H}(\mathbf{x}^*)(\bar{\mathbf{x}} - \mathbf{x}^*) &> J(\mathbf{x}^*) \\ (\bar{\mathbf{x}} - \mathbf{x}^*)^T\mathbf{H}(\mathbf{x}^*)(\bar{\mathbf{x}} - \mathbf{x}^*) &> \mathbf{0} \end{aligned} \tag{4.32}$$

A local minima is assured when Equation 4.32 is added as a constraint. If the result was that it is smaller than 0 then there is a local maximum. A saddle point occurs when the condition vary between smaller, larger or when it is equal to 0. The result here is the minimum for the unconstrained problem.

Section 4.1 showed that the optimal control problem is bound by different constraints which resulted in the augmented cost function (Equation 4.6). The Lagrangian is then defined to include both the equality and inequality contraints to form

$$\mathcal{L}(\mathbf{x}, \lambda.\varphi^A) = J(\mathbf{x}) - \lambda^T\mathbf{f}(\mathbf{x}) - (\varphi^A)^T\mathbf{C}^A(\mathbf{x}) \tag{4.33}$$

where λ are the Lagrange multipliers for the equality constraints and φ^A) the Lagrange multipliers for the active set of inequality constraints. The inactive set of inequality constraints are ignored by setting $\varphi^{A'} = \mathbf{0}$. The necessary conditions for a minimum are then

$$\nabla_x \mathcal{L}(\mathbf{x}^*, \lambda^*, \varphi^{A*}) = \mathbf{0} \tag{4.34}$$

$$\nabla_\lambda \mathcal{L}(\mathbf{x}^*, \lambda^*, \varphi^{A*}) = \mathbf{0} \tag{4.35}$$

$$\nabla_\varphi \mathcal{L}(\mathbf{x}^*, \lambda^*, \varphi^{A*}) = \mathbf{0} \tag{4.36}$$

These equations are also known as the first-order necessary Karush Kuhn-Tucker conditions (KKT). The gradients to calculate the minimum is

$$\nabla_x \mathcal{L} = \mathbf{g}(\mathbf{x}) - \mathbf{G}_\lambda^T(\mathbf{x})\lambda - \mathbf{G}_\varphi^T(\mathbf{x})\varphi \tag{4.37}$$

$$\nabla_\lambda \mathcal{L} = -\mathbf{f}(\mathbf{x}) \tag{4.38}$$

$$\nabla_\varphi \mathcal{L} = -\mathbf{C}^A(\mathbf{x}) \tag{4.39}$$

Remember that $\boldsymbol{\lambda} \in \mathbb{R}^q$ and $\boldsymbol{\mu} \in \mathbb{R}^s$, and that $\mathbf{f}(\mathbf{x}) = \mathbf{C}(\mathbf{x}) = \mathbf{0}$ is a necessary requirement. The Jacobian matrix $\mathbf{G}_{\boldsymbol{\mu}}(\mathbf{x})$ for the costate section is defined as

$$\mathbf{G}_{\boldsymbol{\mu}}(\mathbf{x}) \equiv \frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \dots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \dots & \frac{\partial f_2}{\partial x_n} \\ \vdots & & & \\ \frac{\partial f_q}{\partial x_1} & \frac{\partial f_q}{\partial x_2} & \dots & \frac{\partial f_q}{\partial x_n} \end{bmatrix} \tag{4.40}$$

The Jacobian matrix for the inequality constraints has a similar shape, but then of dimensions $[s, n]$. Once again, it is not yet determined whether this extremal is a minimum, the Hessian is therefore declared as

$$\mathbf{H}_L = \nabla_{xx}\mathcal{L} = \nabla_{xx}J - \sum_{i=1}^{q} \lambda_i \nabla_{xx} f_i - \sum_{i=1}^{s} \varphi_i \nabla_{xx} C_i \tag{4.41}$$

A sufficient condition for a minimum is then

$$\mathbf{v}^T \mathbf{H}_L \mathbf{v} > \mathbf{0} \tag{4.42}$$

for any vector $\mathbf{v}$ that is within the constrained space such that the non-active set $\boldsymbol{\mu}^{A'}$ is rightfully not active.

Numerous toolboxes are available to solve the non-linear programming problem that use a few well known methods to solve NLP's are conjugate direction methods, sequential quadratic programming, and interior-point methods. There are two main requirements when choosing a possible solver. The first one is that it is easily compatible with the C++ framework, and second, that it generates a solution as fast as possible. Well-known solvers that can deal with large-scale sparse non-linear problems are IPOPT (Kawajir et al., 2015), SNOPT (Gill et al., 2008) and KNITRO (Ghaffari and Hribar, 2007). SNOPT is a better candidate than IPOPT for sparse non-linear problems similar to the one presented here (D'Onofrio et al., 2016), and KNITRO is in the same performance league as IPOPT (Ghaffari and Hribar, 2007). One of the possible minima is found faster by an order of magnitude, especially when the solution is far from the desired one, which is possible in case the attitude angles cannot reach the desired attitude that is calculated in the guidance component.

The method to solve the NLP used by SNOPT is that of sequential quadratic programming (SQP) (Gill et al., 2008). It is especially useful when the calculation of the functions and jacobians thereof are relatively expensive. Sequential quadratic programming uses Newton's method to find the KKT points (Wassel, 2013). This means that certain properties are inherited, such as local quadratic convergence and the dependency of the initial guess. It is therefore important that the initial guess already has a somewhat similar shape as the solution. One possible way to do this is to estimate the end-state, and linearly interpolate to get the impromptu values of the other points, or use the solution from a similar problem.

4.3. The Different Pseudospectral Methods

Pseudospectral methods can be divided into three different families that each has its own set of discretisation points. These are Legendre-Gauss-Lobatto (LGL) (Elnagar et al., 1995), Legendre-Gauss (LG) (Herman and Conway, 1996), and Legendre-Gauss-Radau (LGR) (Garg, 2011). All three are direct methods to solve non-linear programming problems, the basis of which is explained in Sections 4.1 and 4.2.

Each set is defined within the $[-1, 1]$ domain, but are different as in that the LG points do not reach any of the endpoints, LGR is located at the -1 point, and LGL reaches both the endpoints. A variation of LGR is when the domain is $(-1, 1]$, this is called the flipped Legendre-Gauss-Radau (LGR-f) method. An overview of these points, also called nodes or collocation points, is shown in Figure 4.2. The methods might all seem nearly the same, but they are very different on a mathematical level. It is beyond the scope of this thesis to elaborate on the specifics, however, a global overview is given.

The sets of discretisation points are all based on quadrature rules based on Legendre polynomials. The location of the nodes (τ) of the three different methods in the $[-1, 1]$ domain are calculated with the following

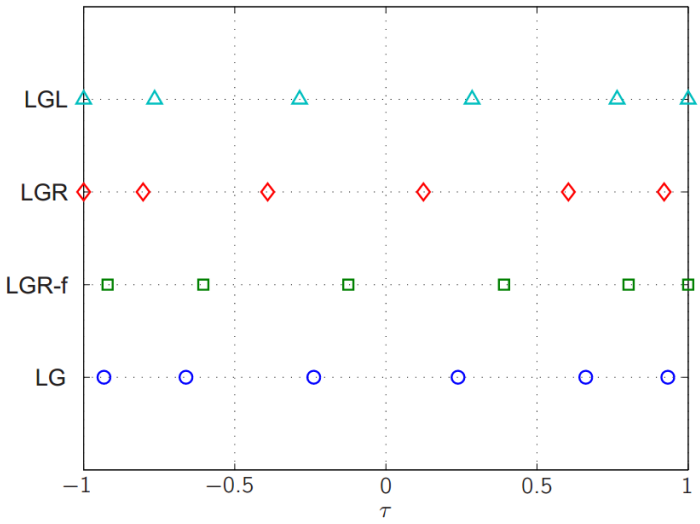


Figure 4.2: An overview of the different discretisation sets (Garg, 2011).

relationships

$$\tau_{LG} = L_N(\tau) \tag{4.43}$$
$$\tau_{LGR} = L_{N-1}(\tau) + L_N(\tau) \tag{4.44}$$
$$\tau_{LGL} = L_N(\tau) \text{ with points } -1 \text{ and } 1 \text{ added} \tag{4.45}$$

The result is that there are two symmetrical sets of nodes (LG and LGL) and one asymmetrical set (LGR). The advantage of having an asymmetrical set of nodes is that it has a faster convergence rate towards the optimal solution. This means that fewer collocation points are required to find the same solution. The solution difference can be several magnitudes, depending on the problem.

It is possible that there are problems that do not have a distinct time interval. This is the case if there is a certain target that needs to be reached, or any other state, which results in another variable, final time. A requirement for problems such as these is that there is a collocation point with location 1. It is not possible for the optimisation program to interpolate to the end of the spectrum with the current calculus in an efficient manner. The calculation of the trajectory is an example for such a problem.

One of the recommendations at the end is that it is possible to create a multi-step pseudospectral Method, but it is not required for the type of problem presented in this thesis. Each node presented in Figure 4.2 alse presents a node with which calculations are performed. It is obviously beneficial that each node is calculated just once. This is another downside for LGL, since there are nodes at both ends of the spectrum.

It is shown later in Section 4.1 that the costates should have the values defined in Equations 4.17-4.20 in order for the solution to have optimality. An example of these costates is seen in Figure 4.3 where these are calculated with LGR or LG, and LGL respectively. These are the costates of a problem very similar to what is used as a verification example in Section 4.8. The left solution closely follows the analytical solution, the right solution clearly oscillates around the analytical solution. This oscillation increases the inaccuracy even further when more nodes are used, one of the problems of Bollino (2006) as discussed in Section 2.2.4. Choosing another method has thus the potential to solve this problem. A downside of both LG and LGL is that it is required to calculate the costates seperately for each problem, this is not required for LGR. The result is that implementing a new problem, or changing one, is easier when LGR is used. The costate calculation is relatively easy for simple problems, these calculations become puzzling when the problem becomes more complex. It is still possible to solve the costate equations for LGR using the Karuhn Kuhn-Tucker equations, if so required.

In short, it can be said that both LG and LGR are superior to LGL when it involves finding the accuracy of the solution, but it were primarily the advantages of not having to calculate the costate and being able to

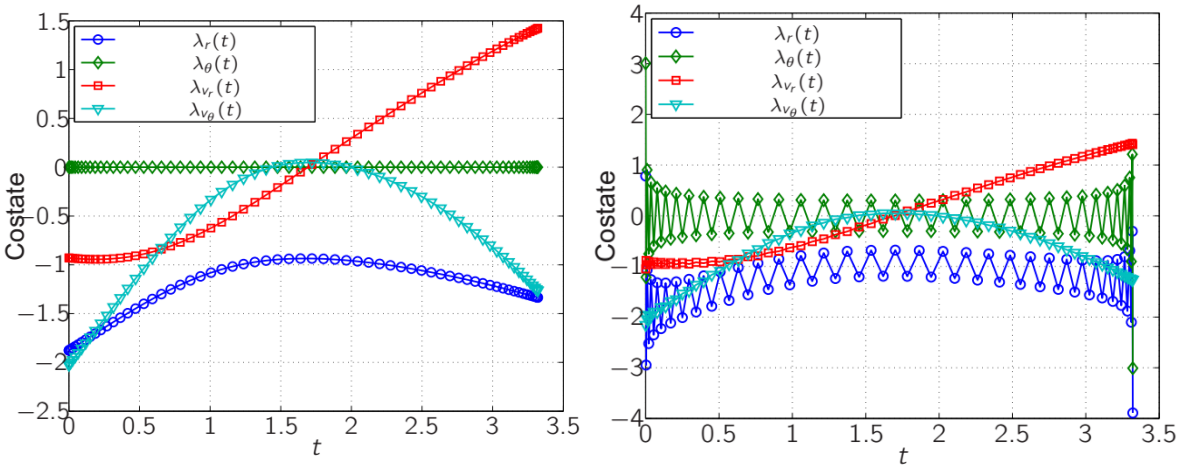


Figure 4.3: The costate solution of the Orbit Raising problem calculated with LGR and LGL respectively (Garg, 2011).

calculate the final time is too have defined the choice of which pseudospectral Method to use is LGR-f. The mathematics and transcription of the flipped Legendre Gauss Radau method is explained in Section 4.4, but not before an introduction of optimal control is given in Section 4.1.

4.4. Flipped Radau Pseudospectral Method

It is discussed in Section 4.3 that the flipped Radau pseudospectral method is to be used as optimalisation methods during the simulation. The mathematical basis is the same for LGR and LGR-f, the only difference is that the locations of the nodes and weights are mirrored. This basis is thoroughly explained in (Garg, 2011), but the essentials and results are explained here. Pseudospectral methods are a form op optimal control theory of which the optimality conditions are found in Section 4.1. These conditions are not yet discretised or scaled to the $[-1,1]$ domain. The discrete points are mapped in the $(-1,1]$ domain from the time domain $[t_0, t_f]$, this is done with the following affine transformation

$$\tau = \frac{2}{t_f - t_0} t - \frac{t_f + t_0}{t_f - t_0} \tag{4.46}$$

and to map this domain back to the time domain can be done with

$$t = \frac{t_f - t_0}{2} \tau + \frac{t_f + t_0}{2} \tag{4.47}$$

The cost function stated in Equation 4.1 is then defined in the $[-1,1]$ domain to

$$J = \Phi(\mathbf{x}(-1), t_0, \mathbf{x}(1), t_f) + \frac{t_f - t_0}{2} \int_{-1}^{1} \Psi(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau; t_0, t_f) d\tau \tag{4.48}$$

subject to the dynamic constraints

$$\frac{d\mathbf{y}(\tau)}{d\tau} = \dot{\mathbf{x}}(\tau) = \frac{t_f - t_0}{2} f(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau; t_0, t_f) \tag{4.49}$$

and the boundary conditions

$$\phi(\mathbf{x}(-1), t_0, \mathbf{x}(1), t_f) = \mathbf{0} \tag{4.50}$$

and the inequality constraints

$$\frac{t_f - t_0}{2} \mathbf{C}(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau; t_0, t_f) \leq \mathbf{0} \tag{4.51}$$

This cost function and its constraints can then be used to create the Hamiltonian

$$\begin{aligned} H(\mathbf{x}(\tau), \mathbf{u}(\tau), \lambda(\tau), \gamma(\tau), \tau; t_0, t_f) =& \Psi(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau; t_0, t_f) + \langle \lambda, \mathbf{f}(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau; t_0, t_f) \rangle \\ & - \langle \gamma, \mathbf{C}(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau; t_0, t_f) \rangle \end{aligned} \tag{4.52}$$

And the corresponding first-order optimality conditions are

$$\dot{\mathbf{x}}(\tau) = \frac{t_f - t_0}{2}\nabla_\lambda H \tag{4.53}$$

$$\dot{\lambda}(\tau) = -\frac{t_f - t_0}{2}\nabla_y H \tag{4.54}$$

$$\mathbf{0} = \nabla_u H \tag{4.55}$$

$$\lambda(-1) = -\nabla_{y(-1)}(\Phi - \langle\varphi, \phi\rangle) \tag{4.56}$$

$$\lambda(+1) = \nabla_{y(1)}(\Phi - \langle\varphi, \phi\rangle) \tag{4.57}$$

$$\nabla_{t_0}(\Phi - \langle\varphi, \phi\rangle) = \frac{1}{2}\int_{-1}^{1} H d\tau - \frac{t_f - t_0}{2}\int_{-1}^{1}\frac{\partial H}{\partial t_0}d\tau \tag{4.58}$$

$$-\nabla_{t_f}(\Phi - \langle\varphi, \phi\rangle) = \frac{1}{2}\int_{-1}^{1} H d\tau + \frac{t_f - t_0}{2}\int_{-1}^{1}\frac{\partial H}{\partial t_0}d\tau \tag{4.59}$$

$$\gamma_i(\tau) = 0 \text{ when } C_i(\mathbf{y}(\tau), \mathbf{u}(\tau)) < 0, \text{ for } 1 \leq i \leq N \tag{4.60}$$

$$\gamma_i(\tau) < 0 \text{ when } C_i(\mathbf{y}(\tau), \mathbf{u}(\tau)) = 0, \text{ for } 1 \leq i \leq N \tag{4.61}$$

$$\phi = \mathbf{0} \tag{4.62}$$

The location of the nodes τ are calculated by combining the Radau quadrature with the Newton-Raphson method to calculate Equation 4.44 in the following manner

$$\tau_{i+1} = \tau_i - \frac{R_{n-1}(\tau_i)}{R'_{n-1}(\tau_i)} \tag{4.63}$$

with

$$R_{n-1}(\tau_i) = \frac{L_{n-1}(\tau_i) + L_n(\tau_i)}{1 + \tau_i} \tag{4.64}$$

$$R'_{n-1}(\tau_i) = \frac{2n}{1 - \tau_i^2}L_{n-1}(\tau_i) \tag{4.65}$$

and $L_n(\tau)$ are the Lagrange Polynomials which are also found in Section 5.2:

$$L_n(\tau) = \prod_{\substack{k=0 \\ k\neq i}}^{N}\frac{\tau - \tau_k}{\tau_n - \tau_k} \tag{4.66}$$

Equation 4.63 can then be rewritten when using the symmetric properties of the Legendre polynomials $L_{n-1}(\tau) = -L_n(\tau)$ into

$$\tau_{i+1} = \tau_i - \left(\frac{1 - \tau_i}{2}\right)\frac{L_{n-1}(\tau_i) + L_n(\tau_i)}{L_{n-1}(\tau_i) - L_n(\tau_i)} \tag{4.67}$$

This process is repeated until the difference of the next solution is within machine accuracy ($|\tau_{n+1} - \tau_n| < 10^{-15}$). The number needs to be as low as possible, since any numerical error made here propagates throughout the entire optimisation. The initial guess of the node locations is calculated with the Chebyshev polynomials

$$\tau_k = \cos\left(\frac{2(k-1)}{2n}\pi\right) \tag{4.68}$$

The result is a set of n LGR nodes. A simple trick is then used to get the LGR-f nodes

$$\tau_{LGR-f} = -flip(\tau_{LGR}) \tag{4.69}$$

An example of a set of 5 flipped Legendre-Gauss-Radau nodes is given in Table 4.1. The state at each of these nodes is found by $\mathbf{y}(\tau)$ and approximated by using the Lagrange polynomial of degree N, the amount of nodes, with

$$\mathbf{x}(\tau) \approx \mathbf{X}(\tau) = \sum_{i=0}^{N}\mathbf{X}(\tau)L_i(\tau) \tag{4.70}$$

Table 4.1: The locations of a set of 5 flipped Legendre-Gauss-Radau points.

−0.885791	−0.446313	−0.167180	0.7204802	1.000000

Table 4.2: The results of the differential matrix for five nodes for LGR-f.

−5.5193	4.3780	1.4460	−0.4376	0.1999	−0.0669
1.7915	−3.5807	0.9030	1.1819	−0.4330	0.1372
−1.1721	2.0611	−2.2480	0.4284	1.2592	−0.3285
1.1413	−1.9393	1.6966	−2.5942	0.2906	1.4050
−2.5000	4.2062	−3.4851	4.3886	−9.1096	6.5000

The cost function is subject to the dynamic constraints (Equation 4.50). It is therefore necessary to also approximate the derivative of **y** in a similar manner

$$\dot{\mathbf{x}}(\tau) \approx \dot{\mathbf{X}}(\tau) = \sum_{i=0}^{N} \mathbf{X}(\tau)\dot{L}_i(\tau) \tag{4.71}$$

$\dot{L}_i(\tau)$ is the derivative of the Lagrange polynomials which is renamed to D_{ki} and it is called the Radau Discrete matrix that has dimensions $[n \times n+1]$. This is then inserted into Equation 4.50 to get

$$\sum_{i=0}^{N} D_{ij}\mathbf{X}_j = \frac{t_f - t_0}{2}\mathbf{f}(\mathbf{X}_i, \mathbf{U}_i, \tau; t_0, t_f) \tag{4.72}$$

The summation starts at $\tau = 0$, this is not a node, but it is used in the state approximation. The following algorithm is used to calculate the Radau Discrete matrix (Sagliano and Theil, 2013)

$$\hat{D}_{ij} = \frac{\sum\limits_{\substack{k=0 \\ k\neq j}}^{N} \left(\prod\limits_{\substack{m=0 \\ m\neq j,k}}^{N} (\tau_i - \tau_m) \right)}{\prod\limits_{\substack{k=0 \\ k\neq j}}^{N} (\tau_j - \tau_k)} \tag{4.73}$$

The result is a matrix that has dimensions $[n+1 \times n+1]$ which can either be used for LGR or LGR-f. The matrix is reduced to suit one of both, the upper row is removed for LGR-f, and the lowest row for LGR. The summation of each row equals to 0.

The cost function is also discretised into different nodes and the Lagrange term is calculated at all of the nodes. Each node has a seperate weight in the cost function since the distance between the nodes is not equal. The weight for LGR is calculated with

$$\tilde{w}_k = \frac{1}{(1-\tau_j)\dot{L}_{n-1}^2} \text{ for } k = 1, 2, .., n \tag{4.74}$$

$$\tilde{w}_0 = \frac{2}{n} \tag{4.75}$$

The weights are then flipped for them to be used for LGR-f

$$w = flip(\tilde{w}) \tag{4.76}$$

The discretised cost function that has to be minimized is written as

$$J = \Phi(\mathbf{X}(\tau_0), \tau_0, \mathbf{X}(\tau_N), \tau_N) + \frac{t_f - t_0}{2} \sum_{i=1}^{N} w_k \Psi(\mathbf{X}_i, \mathbf{U}_i, \tau; t_0, t_f) \tag{4.77}$$

subject to the following dynamic and inequality constraints

$$\mathbf{D}_{ij}\mathbf{X}_j - \frac{t_f - t_0}{2}\mathbf{f}(\mathbf{X}_j, \mathbf{U}_j, \tau; t_0, t_f) = \mathbf{0} \tag{4.78}$$

$$\varphi(\mathbf{X}(\tau_0), \tau_0, \mathbf{X}(\tau_N), \tau_N) = \mathbf{0} \tag{4.79}$$

$$\frac{t_f - t_0}{2}\mathbf{C}(\mathbf{X}_j, \mathbf{X}_j, \tau; t_0, t_f) \leq \mathbf{0} \tag{4.80}$$

This non-linear programming problem can now be solved by the numerical optimizer SNOPT (Gill et al., 2008) to find the local optimum. It is numerically impossible to reach exactly zero, the limits are therefore set to 10^{-15}. Changing this number to be higher increases the solution space, which results in faster solutions, but a loss of accuracy.

This section described the transcription method of the radau flipped pseudospectral method. Two examples are demonstrated in Section 4.8. The first example is a simple orbit raising problem that has a fixed time and no constraints. The second example demonstrates the maximum-range problem with the Space Shuttle, this problem requires scaling, constraints, and a variable final time.

4.5. Transcription of the Flipped Radau Pseudospectral Method

Section 4.4 transcribed a continuous time-based optimal control problem into a non-linear programming problem. This section follows up on that with a more practical approach with an understandable transcription, and how it is implemented to calculate the trajectories and attitude control in the simulation.

Once again, the following cost function is minimised

$$J = \Phi(\mathbf{x}(t_0), t_0, \mathbf{x}(t_f), t_f) + \frac{t_f - t_0}{2}\int_{-1}^{1} \Psi(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau; t_0, t_f)d\tau \tag{4.81}$$

subject to the dynamics

$$\dot{x} = f_c(t, x, u) \tag{4.82}$$

and the states and controls are bound within

$$x_L \leq x(t) \leq x_U \tag{4.83}$$

$$u_L \leq u(t) \leq u_U \tag{4.84}$$

and the global constraints are, if any,

$$g_L \leq g(x, t, u) \leq g_U \tag{4.85}$$

The states and controls are discretized to a set of N nodes in the $[-1, 1]$ spectrum based on the locations determined in Equations 4.63-4.68.

$$x(t_i) \cong X_i,\ i \in [0, N] \tag{4.86}$$

$$u(t_i) \cong U_i,\ i \in [1, N] \tag{4.87}$$

The states at X_0 are bound at the initial point but are not a part of the calculation at the nodes, that starts at X_1. This is a direct result from using LGR-f. Most problems only begin with an initial state and a guess of what the final state might be. An initial guess can be created by linearly interpolating between the initial and final state. The quality of the initial guess of the solution determines the time it takes at which the solution is found, but it does not have to be very accurate, a solution can still be found most of the times. The final state can be set equal to the initial one if it is unknown what the outcome is. Each node is bound to the boundaries from Equations 4.83-4.84

$$\mathbf{x}_L \leq \mathbf{X}_i \leq \mathbf{x}_u,\ i \in [0, N] \tag{4.88}$$

$$\mathbf{u}_L \leq \mathbf{U}_i \leq \mathbf{u}_L,\ i \in [1, N] \tag{4.89}$$

The cost function based on Equation 4.77 can be further discretised into

$$J = \sum_{i=1}^{N} v_i \Phi(X_i, U_i, \tau_i) + \frac{t_f - t_0}{2} w_i \Psi(X_i, U_i, \tau_i),\ i \in [1, N] \tag{4.90}$$

where Φ is the Mayer term and Ψ is the Lagrange term. The Mayer function only has a value at the first or last node, v is used to enforce this

$$\mathbf{v} = \left[v_1, v_2, \dots, v_N\right]^T \tag{4.91}$$

with v_1 and v_N being either 0 or 1, the rest of the vector is 0. The weights of the Lagrange term are calculated with Equations 4.74 and 4.75. The dynamics of the problem are rewritten as with Equation 4.78, with the residual f added.

$$\mathbf{f}_i = \sum_{j=0}^{N} \mathbf{D}_{ij} \mathbf{X}_j - \frac{t_f - t_0}{2} \mathbf{f}_c(\mathbf{X}_i, \mathbf{U}_i, \tau; t_0, t_f) = 0,\ i \in [1, N] \text{ and } j \in [0, N] \tag{4.92}$$

and the discretised global constraints are

$$\mathbf{g}_L \leq \mathbf{g}(\tau_i, \mathbf{X}_i, \mathbf{U}_i) \leq \mathbf{g}_U,\ i \in [1, \dots, N] \tag{4.93}$$

The non-linear programming problem of Equations 4.88-4.93 can be solved by SNOPT directly (Gill et al., 2008). The program is able to calculate the derivatives of the equation by itself using automatic differentiation. The sparsity is calculated as well. However, the computation time can be improved by roughly 40% by calculating the Jacobian at each step for the program. There are a few steps needed to calculate the jacobian in an efficient manner. First, the different variables are put in the order of usage per node, thus

$$\mathbf{X}_{NLP} = \left[\mathbf{X}_0 \,|\, \mathbf{X}_1 \mathbf{U}_1 \,|\, \mathbf{X}_2 \mathbf{U}_2 \,|\, \dots \,|\, \mathbf{X}_N \mathbf{U}_N \,|\, t_f\right] \tag{4.94}$$

Note that the final time added at the end. Some problem examples have a loose endtime, which means that time it becomes a variable as well. These states all have an effect of course on Equations 4.90-4.93. These functions are all turned into inequality constraints

$$\mathbf{C}(\mathbf{X}_{NLP}) = \left[J \,|\, \mathbf{f}_1 \mathbf{f}_2 \,\dots\, \mathbf{f}_N \,|\, \mathbf{g}_1 \mathbf{g}_2 \,\dots\, \mathbf{g}_N\right] \tag{4.95}$$

It might not make sense at first glance that the cost function is turned into an inequality constraint. The reason is that it becomes a part of the NLP and that requires that even the cost function is bounded to stay within certain values, this is set to a large quantity such that it does not influence the outcome. The Jacobian is then defined as the derivative of the inequality constraints by the variables

$$Jac = \left[\frac{\partial \mathbf{C}}{\partial \mathbf{X}_{NLP}}\right] = \begin{bmatrix} \nabla \mathbf{J} \\ \nabla \mathbf{F} \\ \nabla \mathbf{G} \end{bmatrix} = \begin{bmatrix} \frac{\partial J}{\partial \mathbf{x}_0} & \frac{\partial J}{\partial \mathbf{x}_1} & \frac{\partial J}{\partial \mathbf{u}_1} & \frac{\partial J}{\partial \mathbf{x}_2} & \frac{\partial J}{\partial \mathbf{u}_2} & \cdots & \frac{\partial J}{\partial \mathbf{x}_N} & \frac{\partial J}{\partial \mathbf{u}_N} & \frac{\partial J}{\partial t_F} \\ \frac{\partial \mathbf{f}_1}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{f}_1}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{f}_1}{\partial \mathbf{u}_1} & \frac{\partial \mathbf{f}_1}{\partial \mathbf{x}_2} & \frac{\partial \mathbf{f}_1}{\partial \mathbf{u}_2} & \cdots & \frac{\partial \mathbf{f}_1}{\partial \mathbf{x}_N} & \frac{\partial \mathbf{f}_1}{\partial \mathbf{u}_N} & \frac{\partial \mathbf{f}_1}{\partial t_F} \\ \frac{\partial \mathbf{f}_2}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{f}_2}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{f}_2}{\partial \mathbf{u}_1} & \frac{\partial \mathbf{f}_2}{\partial \mathbf{x}_2} & \frac{\partial \mathbf{f}_2}{\partial \mathbf{u}_2} & \cdots & \frac{\partial \mathbf{f}_2}{\partial \mathbf{x}_N} & \frac{\partial \mathbf{f}_2}{\partial \mathbf{u}_N} & \frac{\partial \mathbf{f}_2}{\partial t_F} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots \\ \frac{\partial \mathbf{f}_N}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{f}_N}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{f}_N}{\partial \mathbf{u}_1} & \frac{\partial \mathbf{f}_N}{\partial \mathbf{x}_2} & \frac{\partial \mathbf{f}_N}{\partial \mathbf{u}_2} & \cdots & \frac{\partial \mathbf{f}_N}{\partial \mathbf{x}_N} & \frac{\partial \mathbf{f}_N}{\partial \mathbf{u}_N} & \frac{\partial \mathbf{f}_N}{\partial t_F} \\ \frac{\partial \mathbf{g}_1}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{g}_1}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{g}_1}{\partial \mathbf{u}_1} & \frac{\partial \mathbf{g}_1}{\partial \mathbf{x}_2} & \frac{\partial \mathbf{g}_1}{\partial \mathbf{u}_2} & \cdots & \frac{\partial \mathbf{g}_1}{\partial \mathbf{x}_N} & \frac{\partial \mathbf{g}_1}{\partial \mathbf{u}_N} & \frac{\partial \mathbf{g}_1}{\partial t_F} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots \\ \frac{\partial \mathbf{g}_N}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{g}_N}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{g}_N}{\partial \mathbf{u}_1} & \frac{\partial \mathbf{g}_N}{\partial \mathbf{x}_2} & \frac{\partial \mathbf{g}_N}{\partial \mathbf{u}_2} & \cdots & \frac{\partial \mathbf{g}_N}{\partial \mathbf{x}_N} & \frac{\partial \mathbf{g}_N}{\partial \mathbf{u}_N} & \frac{\partial \mathbf{g}_N}{\partial t_F} \end{bmatrix} \tag{4.96}$$

The dimension of this Jacobian is

$$dim(\mathbf{Jac}) = [N \cdot (n_s + n_g) + 1] \times [(n + 1) \cdot n_s + N \cdot n_c + 1] \tag{4.97}$$

4.6. Scaling

It can happen that a problem has state variables with significant different order of magnitudes (Betts, 2010)(Sagliano, 2014) . A re-entry Space Shuttle problem, for example, has attitude angles varying between $-180°$ and $180°$, and a height that stays between 0 and $260\,000$ feet. The solver than has problems with minimising the problem, or even finding a solution, it can then be said that the problem is ill-conditioned. It is then required that the problem is scaled, and there are a few steps required to be able to implement this, but in essence it boils down to adapting the variables to the $[0, 1]$-domain and to change the boundaries, dynamic and inequality constraints, and the jacobian to match this. There are several ways to do this, and the projected Jacobian Rows Normalisation (PJRN) shall be used here, since this proved to give the best results (Sagliano, 2014).

The variables are changed first using the standard linear transformation (Betts, 2010), the scaled state is given as

$$\tilde{\mathbf{X}} = \mathbf{K_x} \cdot \mathbf{X} + \mathbf{b_x} \tag{4.98}$$

with

$$\mathbf{K_{x_{ii}}} = \frac{1}{\mathbf{X_{U_i}} - \mathbf{X_{L_i}}} \tag{4.99}$$

$$\mathbf{b_{x_{ii}}} = -\frac{\mathbf{X_{L_i}}}{\mathbf{X_{U_i}} - \mathbf{X_{L_i}}} \tag{4.100}$$

The boundaries of the variables are then changed to

$$\tilde{\mathbf{x}}_\mathbf{L} = \mathbf{K_x} \cdot \mathbf{x_L} + \mathbf{b_x} \tag{4.101}$$

$$\tilde{\mathbf{x}}_\mathbf{U} = \mathbf{K_x} \cdot \mathbf{x_U} + \mathbf{b_x} \tag{4.102}$$

It is not essential to calculate this since the boundaries should be between 0 and 1. However, this is useful to use such that it can be seen that $\mathbf{K_x}$ and $\mathbf{b_x}$ are calculated correctly.

The scaled dynamic and inequality constraints are also calculated using the simple linear technique

$$\tilde{\mathbf{F}} = \mathbf{K_f} \cdot \mathbf{F} \tag{4.103}$$

$$\tilde{\mathbf{G}} = \mathbf{K_g} \cdot \mathbf{G} \tag{4.104}$$

where the scaling factors $\mathbf{K_f}$ and $\mathbf{K_g}$ are found with the PJRN-technique (Sagliano, 2014)

$$\mathbf{K_{f}}_{ii} = \frac{1}{|\nabla \mathbf{F} \cdot \mathbf{K_x}^{-1}|_i} \tag{4.105}$$

$$\mathbf{K_{g}}_{ii} = \frac{1}{|\nabla \mathbf{G} \cdot \mathbf{K_x}^{-1}|_i} \tag{4.106}$$

Both the scaling factors are diagonal matrices and each point is calculated using one row of the jacobian from Equation 4.96. These constants are calculated before the actual simulation, it is possible to do it in the mean-time as well, but this increases the computation time and there is not much added benefit in terms of finding an accurate solution.

The boundaries of the constraints have to be scaled as well, this is done by linearly scaling them

$$\tilde{\mathbf{f}}_L = \mathbf{K_f} \mathbf{f}_L \tag{4.107}$$

$$\tilde{\mathbf{f}}_U = \mathbf{K_f} \mathbf{f}_U \tag{4.108}$$

$$\tilde{\mathbf{g}}_L = \mathbf{K_g} \mathbf{g}_L \tag{4.109}$$

$$\tilde{\mathbf{g}}_U = \mathbf{K_g} \mathbf{g}_U \tag{4.110}$$

The scaled Jacobian is calculated with the following

$$\tilde{\mathbf{Jac}} = \begin{pmatrix} \tilde{\nabla} \tilde{J} \\ \tilde{\nabla} \tilde{\mathbf{F}} \\ \tilde{\nabla} \tilde{\mathbf{G}} \end{pmatrix} = \begin{pmatrix} K_j \cdot \nabla J \cdot \mathbf{K_x}^{-1} \\ \mathbf{K_F} \cdot \nabla \mathbf{F} \cdot \mathbf{K_x}^{-1} \\ \mathbf{K_G} \cdot \nabla \mathbf{G} \cdot \mathbf{K_x}^{-1} \end{pmatrix} \tag{4.111}$$

4.7. Sparsity of the Jacobian

It is inefficient to calculate the derivative at each point of the Jacobian since the matrix is a sparse one. A matrix is said to be sparse if there is a matrix **a** that has many entries $a_{ij} = a_{ji} = 0$ (Kreyszig, 2011). There are two types of derivatives present in the Jacobian, linear and non-linear. It is beneficial to separate both types to increase the efficiency of the solver by giving the linear derivatives at the start of the program. The derivatives are calculated using the Complex step method (Martins et al., 2003) as seen in Section 5.4.

The sparse Jacobian can be seen as a summation of three parts: pseudospectral, numerical, and theoretical.

$$Jac = Jac_{Pseudospectral} + Jac_{Numerical} + Jac_{Theoretical} \tag{4.112}$$

The seperation of the three allows the jacobian to be implemented without much hassle and it increases the feeling for the problem if implemented. All three Jacobians are discussed below in the same order.

First up is the pseudospectral Jacobian, which is inherited from the usage of pseudospectral methods. This Jacobian is formed by the discrete differential matrix used in Equation 4.92

$$\mathbf{D} \cdot \mathbf{X} \tag{4.113}$$

The result is a Jacobian filled with constant entries that represent the nodal values of each state that are calculated at the start of the NLP. The pseudospectral Jacobian has the following form

$$Jac_{Pseudospectral} = \begin{bmatrix} & \mathbf{O}_{1\times[(n+1)\cdot n_s+n\cdot n_c+1]} & & \\ \tilde{\mathbf{D}}_{1,0} & .. & \tilde{\mathbf{D}}_{1,n} & \\ .. & .. & .. & \mathbf{O}_{[n\cdot(n_s+n_g)+1]\times 1} \\ \tilde{\mathbf{D}}_{n,0} & .. & \tilde{\mathbf{D}}_{n,n} & \\ & \mathbf{O}_{n_g\times[(n+1)\cdot n_s+n\cdot n_c+1]} & & \end{bmatrix} \tag{4.114}$$

with

$$\tilde{\mathbf{D}}_{i,j} = \mathbf{D}_{i,j} \cdot I_{n_s},\ i \in [1, N], j \in [0, N] \tag{4.115}$$

The values of D are calculated with Equation 4.73 and I_{n_s} is an identity matrix that has sides equal to the amount of states.

The entries of the numerical Jacobian differ between linear and non-linear entries. It is beneficial to know what part of the Jacobian is linear or zero to significantly increase the speed of the computation. A simple procedure is presented here loosely based on (Sagliano and Theil, 2013) to do just that.

A different set of states is created within the boundaries x_L and x_U with

$$\mathbf{X}_r \sim U([\mathbf{x}_L, \mathbf{x}_U]) \tag{4.116}$$

$$\mathbf{U}_r \sim U([\mathbf{u}_L, \mathbf{u}_U]) \tag{4.117}$$

The values have a uniform distribution and there is no correlation between the different states or sets. The random states are used to calculate the Jacobian of the cost, dynamics and global constraints function. These Jacobians are compared to each other to see if there are differences between them, thus, if the derivatives are linear or non-linear. An example of this procedure is given for the dynamics. First, the Jacobian of the dynamics function is defined as

$$\mathbf{A}_i = \left[\frac{\partial \mathbf{f}}{\partial(\mathbf{X}_R)_i}\ \frac{\partial \mathbf{f}}{\partial(\mathbf{U}_R)_i}\right], \text{ for } i \in [1, \dots, n] \tag{4.118}$$

where **A** is a set of Jacobians of size n is compared to each other in the following manner

$$(M_{dyn})_{m,n} = \begin{cases} 0 \text{ if } (A_1)_{m,n} = (A_i)_{m,n} = 0 & \text{for } i = 2, \dots, n \\ 1 \text{ if } (A_1)_{m,n} = (A_i)_{m,n} \neq 0 & \text{for } i = 2, \dots, n \\ 2 \text{ if } (A_1)_{m,n} \neq (A_i)_{m,n} & \text{for } i = 2, \dots, n \end{cases} \tag{4.119}$$

Zero entries of the Jacobian are denominated with a 0, linear entries with a 1 and non-linear entries with a 2. The numbers are used for identification only, they are not used in calculations. The orbit raising problem

required two sets to be able to recognise the elements, but different problems might require more, five sets were used to create a system that is robust. A similar matrix is made for the cost function, $\mathbf{M}_{cost,mayer}$, $M_{cost,lagr}$ and the global constraints $\mathbf{M}_{con}$ to form the following matrix

$$\mathbf{M}_{num} = \begin{bmatrix} M_{cost,lagr} & \dots & M_{cost,lagr} & M_{cost,lagr} + M_{cost,mayer} & \\ M_{dyn} & \dots & M_{dyn} & M_{dyn} & \\ \dots & \dots & \dots & \dots & \\ M_{dyn} & \dots & M_{dyn} & M_{dyn} & O_{[n\cdot(n_s+n_g)+1]\times 1} \\ M_{con} & \dots & M_{con} & M_{con} & \\ \dots & \dots & \dots & \dots & \\ M_{con} & \dots & M_{con} & M_{con} & \end{bmatrix} \tag{4.120}$$

The matrix is split into two parts, one linear and one for the non-linear elements, $\mathbf{M}_{num,lin}$ and $\mathbf{M}_{num,nonlin}$, the reason for this is that the elements are directly multiplied with the numerical Jacobian, and the non-linear derivatives would be multiplied by two otherwise. The complete numerical Jacobian can easily be distilled from Equations 4.90-4.93 by setting the differential matrix to 0 and the final time to a constant. The result is

$$Jac_{Continuous} = \left[\frac{\partial \mathbf{C}}{\partial \mathbf{X}_{NLP}}\right]_{D=0} = -\frac{t_F - t_0}{2}\begin{bmatrix} \frac{\partial J}{\partial \mathbf{x}_0} & \frac{\partial J}{\partial \mathbf{x}_1} & \frac{\partial J}{\partial \mathbf{u}_1} & \frac{\partial J}{\partial \mathbf{x}_2} & \frac{\partial J}{\partial \mathbf{u}_2} & .. & .. & \frac{\partial J}{\partial \mathbf{x}_N} & \frac{\partial J}{\partial \mathbf{u}_N} \\ \frac{\partial \mathbf{f}_1}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{f}_1}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{f}_1}{\partial \mathbf{u}_1} & \frac{\partial \mathbf{f}_1}{\partial \mathbf{x}_2} & \frac{\partial \mathbf{f}_1}{\partial \mathbf{u}_2} & .. & .. & \frac{\partial \mathbf{f}_1}{\partial \mathbf{x}_N} & \frac{\partial \mathbf{f}_1}{\partial \mathbf{u}_N} \\ \frac{\partial \mathbf{f}_2}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{f}_2}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{f}_2}{\partial \mathbf{u}_1} & \frac{\partial \mathbf{f}_2}{\partial \mathbf{x}_2} & \frac{\partial \mathbf{f}_2}{\partial \mathbf{u}_2} & .. & .. & \frac{\partial \mathbf{f}_2}{\partial \mathbf{x}_N} & \frac{\partial \mathbf{f}_2}{\partial \mathbf{u}_N} \\ .. & .. & .. & .. & .. & .. & .. & .. & .. \\ \frac{\partial \mathbf{f}_N}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{f}_N}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{f}_N}{\partial \mathbf{u}_1} & \frac{\partial \mathbf{f}_N}{\partial \mathbf{x}_2} & \frac{\partial \mathbf{f}_N}{\partial \mathbf{u}_2} & .. & .. & \frac{\partial \mathbf{f}_N}{\partial \mathbf{x}_N} & \frac{\partial \mathbf{f}_N}{\partial \mathbf{u}_N} \\ \frac{\partial g_1}{\partial \mathbf{x}_0} & \frac{\partial g_1}{\partial \mathbf{x}_1} & \frac{\partial g_1}{\partial \mathbf{u}_1} & \frac{\partial g_1}{\partial \mathbf{x}_2} & \frac{\partial g_1}{\partial \mathbf{u}_2} & .. & .. & \frac{\partial g_1}{\partial \mathbf{x}_N} & \frac{\partial g_1}{\partial \mathbf{u}_N} \\ .. & .. & .. & .. & .. & .. & .. & .. & .. \\ \frac{\partial g_N}{\partial \mathbf{x}_0} & \frac{\partial g_N}{\partial \mathbf{x}_1} & \frac{\partial g_N}{\partial \mathbf{u}_1} & \frac{\partial g_N}{\partial \mathbf{x}_2} & \frac{\partial g_N}{\partial \mathbf{u}_2} & .. & .. & \frac{\partial g_N}{\partial \mathbf{x}_N} & \frac{\partial g_N}{\partial \mathbf{u}_N} \end{bmatrix} O_{[n\cdot(n_s+n_g)+1]\times 1} \tag{4.121}$$

The Jacobian above is multiplied with Equation 4.120 with the Hadamard product.

$$Jac_{Numerical,lin} = Jac_{Continuous} \circ \mathbf{M}_{num,lin} \tag{4.122}$$

$$Jac_{Numerical,nonlin} = Jac_{Continuous} \circ \mathbf{M}_{num,nonlin} \tag{4.123}$$

The third part of the Jacobian is only relevant if there is an unknown final time. The entries of the Jacobian are checked for non-linearity in a similar manner as the numerical Jacobian to create $\mathbf{M}_{time}$.

$$Jac_{Theoretical} = -\frac{1}{2}\begin{bmatrix} O_{[n\cdot(n_s+n_g)+1]\cdot[n_s+n\cdot n_c+1]} & \begin{matrix} 0 \\ \mathbf{f}_{C,1} \\ \mathbf{f}_{C,2} \\ .. \\ \mathbf{f}_{C,N} \end{matrix} \\ & O_{n\cdot n_g\times 1} \end{bmatrix} \tag{4.124}$$

All three of the matrixes are added to form the Jacobian. It is important to make a distinction here between problems that are scaled and those that are not. Problems that do not have to be scaled can use the two split jacobians, linear and non-linear, which decreases the computation time. This however, is not possible for scaled problems. The variables are scaled to the $[0, 1]$ domain which inherently also means that the jacobian is scaled to these proportions, in this case there is just one jacobian. How this scaling is performed can be found in Section 4.6.

4.8. Examples

The theory presented in this chapter is supplemented by two examples which are presented here to demonstrate how simple and practical the algorithm is. The first example is that of raising to the highest orbit with

a constant thrust factor, the second example involves the Space Shuttle and to reach the highest latitude. These examples also serve as a way to validate the program.

4.8.1. Orbit Raising

The Orbit Raising example (Fumenti et al., 2013)(Sagliano, 2014) has been used to verify that the core of the program is working, it has no constraint functions, time is invariable, and there is no specific final answer. It is an example that optimises the trajectory of a satellite around a planet or another gravitational object. The goal is to get as high as possible while limiting the velocity. The system is normalised to involve length units [LU] and time units [TU]. The cost function is

$$J = \frac{1}{r(t_f)} - \left(\frac{1}{2} \left(V_r^2(t_f) + V_t^2(t_f) \right) \right) \tag{4.125}$$

and the corresponding dynamic equations are

$$\dot{r} = V_r \tag{4.126}$$

$$\dot{\theta} = \frac{V_t}{r} \tag{4.127}$$

$$\dot{V}_r = \frac{V_t^2}{r} - \frac{\mu}{r^2} + T \sin(\delta) \tag{4.128}$$

$$\dot{V}_t = -\frac{V_r V_t}{r} + T \cos(\delta) \tag{4.129}$$

where T is the specific force set to 0.01, μ is the normalised gravitational parameter ($\mu = 1$), δ is the control parameter that can be used by the optimisation program freely to find the optimal solution. The initial condition is set to $\mathbf{x}_i = [1.1,\ 0.0,\ 0.0,\ 0.95]$ and the final guess to $\mathbf{x}_f = [4.0,\ 20.0,\ 0.15,\ 0.5]$. The states and control are limited to the following

$$\begin{pmatrix} 1.0 \\ 0.0 \\ -0.05 \\ 0.0 \\ -0.05 \end{pmatrix} \leq \begin{pmatrix} r \\ \theta \\ V_r \\ V_t \\ \delta \end{pmatrix} \leq \begin{pmatrix} 5.0 \\ 25.0 \\ 0.25 \\ 1.0 \\ 0.35 \end{pmatrix} \tag{4.130}$$

The final time is set to 50 TU. Different simulations have been run with nodes varying between 20 and 400, all with similar outcomes. The result of a 50-node simulation can be seen in Figure 4.4 and 4.5, the calculation time for this problem is around 3 seconds. The optimal solution here has identical results to (Sagliano and Theil, 2013) and (Fumenti et al., 2013) which verifies that the solution is correct. The maximum radius achieved is 4.3163. The small discrepancy at the beginning of the thrust angle is due to the slight perturbation initial conditions.

4.8.2. Space Shuttle

The Space Shuttle problem (Betts, 2010) is the second example to demonstrate that the pseudospectral methods subprogram is working. The difference compared to the Orbit Raising program is that this one involves a variable end-time, has constraints, has a required end-point, and the differences in value requires that the problem is scaled. The imperial unit system is used here, but that does not change whether or not the program works.

The goal of the problem is to maximise the longitude, this is done with the following cost function

$$J = -\theta(t_f) \tag{4.131}$$

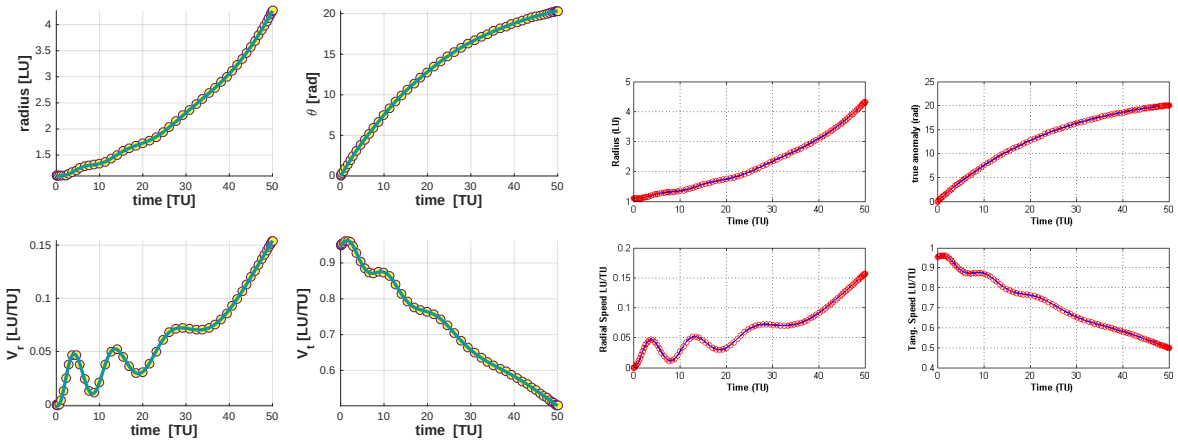


Figure 4.4: The states of the orbit raising problem from the simulation (left), compared to literature (Sagliano, 2014).

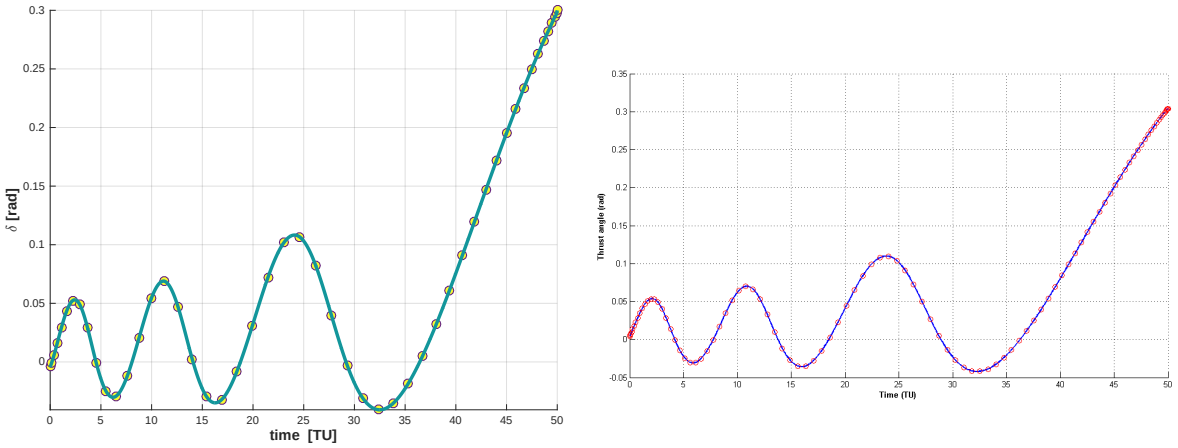


Figure 4.5: The control of the orbit raising problem from the simulation (left), compared to literature (Sagliano, 2014).

subject to the following dynamic constraints

$$\dot{h} = V \sin\gamma \tag{4.132}$$
$$\dot{\phi} = \frac{V}{r}\frac{\cos\gamma \sin\varphi}{\cos\theta} \tag{4.133}$$
$$\dot{\theta} = \frac{V}{r}\cos\gamma\cos\varphi \tag{4.134}$$
$$\dot{V} = -\frac{D}{m} - g\sin\gamma \tag{4.135}$$
$$\dot{\gamma} = \frac{L}{mV}\cos\sigma + \cos\gamma\left(\frac{V}{r} - \frac{g}{V}\right) \tag{4.136}$$
$$\dot{\chi} = \frac{L\sin\sigma}{mV\cos\gamma} + \frac{V}{r\cos\theta}\cos\gamma\sin\varphi\sin\theta \tag{4.137}$$

where h is the height, ϕ is the latitude, θ is the longitude, V is the velocity, γ the flight path angle, and χ is the heading angle. The two control vectors are the angle of attack α and the bank angle σ. A constraint is added to simulate a maximum heat rate, this is calculated with

$$q = q_a q_r \tag{4.138}$$

The following equations are used to aid the dynamic and constraint equations with the constants as given in Table 4.3

$$L = \frac{1}{2}\rho V^2 S C_L \tag{4.139}$$
$$D = \frac{1}{2}\rho V^2 S C_D \tag{4.140}$$
$$C_L = a_0 + a_1\hat{\alpha} \tag{4.141}$$
$$C_D = b_0 + b_1\hat{\alpha} + b_2\hat{\alpha}^2 \tag{4.142}$$
$$r = R_e + h \tag{4.143}$$
$$m = w/g_0 \tag{4.144}$$
$$g = \frac{\mu}{r^2} \tag{4.145}$$
$$\rho = \rho_0 e^{-h/h_r} \tag{4.146}$$
$$q_a = c_0 + c_1\hat{\alpha} + c_2\hat{\alpha}^2 + c_3\hat{\alpha}^3 \tag{4.147}$$
$$q_r = 17700\sqrt{\rho}(0.0001v)^{3.07} \tag{4.148}$$

The simulation is ended when the Shuttle has reached the following three states

$$h_f = 80\,000 \text{ ft} \quad v_f = 2500 \text{ ft/s} \quad \gamma_f = -5 \text{ deg}$$

and the variables are limited during the simulation with the following boundaries

$$\begin{pmatrix} 0 \\ -89 \text{ deg} \\ -89 \text{ deg} \\ 1 \\ -89 \text{ deg} \\ -179 \text{ deg} \end{pmatrix} \leq \begin{pmatrix} h \\ \phi \\ \theta \\ V \\ \gamma \\ \chi \end{pmatrix} \leq \begin{pmatrix} h_0 \\ 89 \text{ deg} \\ 89 \text{ deg} \\ V_0 \\ 89 \text{ deg} \\ 179 \text{ deg} \end{pmatrix} \tag{4.149}$$

The results have been successfully found as seen in Figure 4.6 that shows all six states, Figure 4.7 that shows the angle of attack and bank angle, and Figure 4.8 that shows the heat rate and how it is limited during a large part of the flight. The results are compared with Betts (2010). The dashed lines are the results when the trajectory is constrained to stay within the heat flux boundaries. The total flight-time is 2182.96 seconds and the maximum latitude achieved is 30.62 degrees.

Table 4.3: The constants used in the Space Shuttle Example (Betts, 2010).

$\rho_0 = 0.002738$	$w = 203000$
$h_r = 23800$	$b_0 = 0.7854$
$\hat{\alpha} = 180\alpha/\pi$	$b_1 = -0.61592 \cdot 10^{-2}$
$R_e = 20902900$	$b_2 = 0.621408 \cdot 10^{-3}$
$S = 2690$	$c_0 = 1.0672181$
$\mu = 0.14076539 \cdot 10^{17}$	$c_1 = -0.19213774 \cdot 10^{-1}$
$a_0 = -0.20704$	$c_2 = 0.21286289 \cdot 10^{-3}$
$a_1 = 0.029244$	$c_3 = -0.10117249 \cdot 10^{-5}$

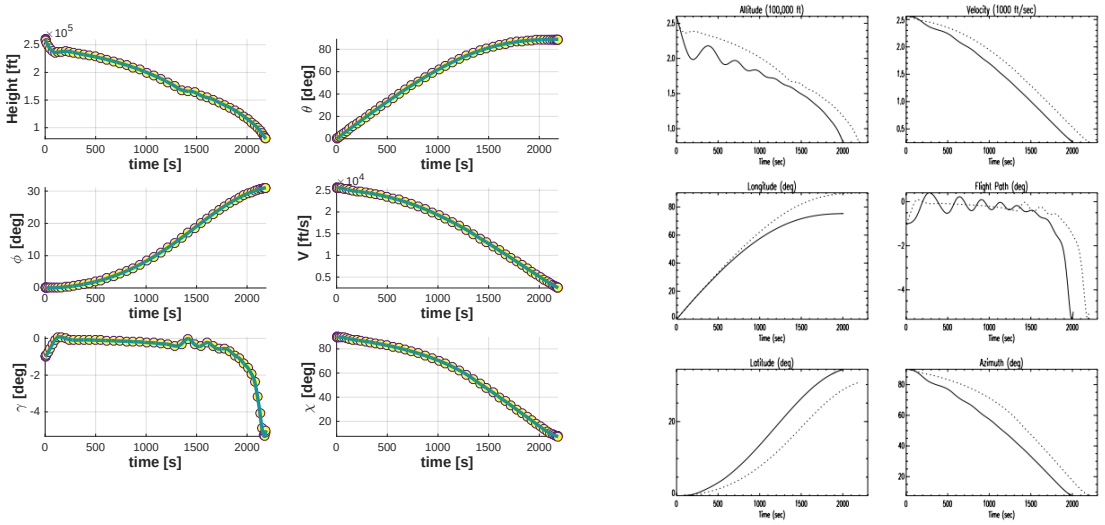


Figure 4.6: The states of the space shuttle problem from the simulation (left), compared to literature (Betts, 2010).

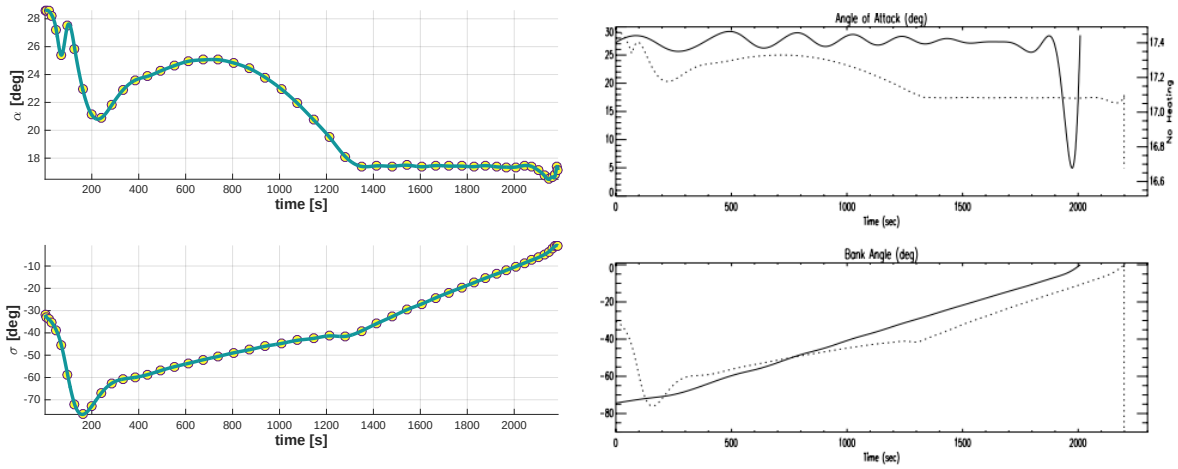


Figure 4.7: The controls of the space shuttle problem from the simulation (left), compared to literature (Betts, 2010).

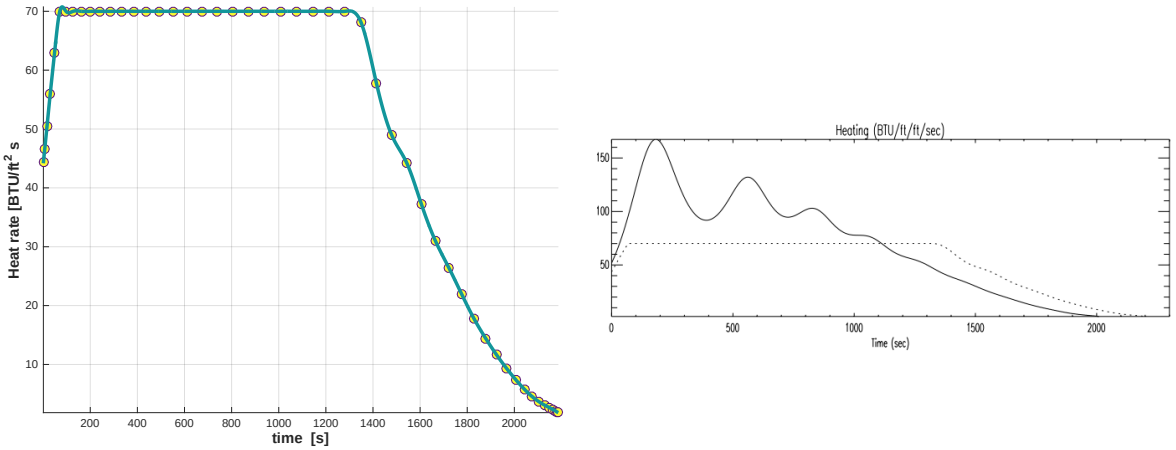


Figure 4.8: The constraint of the Space Shuttle problem from the simulation (left), compared to literature (Betts, 2010).

5

Numerical Tools

This chapter provides the numerical tools that are used during the simulation. Linear Quadratic Control is explained first which is used to validate the PSM control scheme in Section 5.1. Basic techniques are presented such as interpolation, integration and differentiation. How those disciplines are used during this research is found in Section 5.2, 5.3, and 5.4 respectively.

5.1. Linear Quadratic Control

The method of Linear Quadratic Control (LQR) is used for the initial attitude simulations. It is a quick method that can be easily applied to make sure that the model has been correctly programmed. In essence, this method calculates the response of the control vector by how a linearised model would behave by calculating certain multipliers called gains which are subsequently multipled with the values of states, such as the angle of attack. (Mooij, 1998)

LQR is based on optimal control as well, and the cost function is formulated as

$$J = \int_0^\infty (\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{u}^T\mathbf{R}\mathbf{u})\mathbf{dt} \tag{5.1}$$

where $\mathbf{x}(\mathbf{t})$ is subject to

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}, \mathbf{t}) \tag{5.2}$$

$\mathbf{x}^T\mathbf{Q}\mathbf{x}$ is the state deviation and $\mathbf{u}^T\mathbf{R}\mathbf{u}$ is the control effort, the goal is to minimize the cost value and thus minimize both the deviation and effort. $\mathbf{Q}$ and $\mathbf{R}$ are initially defined in the following way, according to Bryson's rule

$$\mathbf{Q} = \text{diag}\left\{\frac{1}{\Delta x_{1_{max}}^2} \frac{1}{\Delta x_{2_{max}}^2} \dots \frac{1}{\Delta x_{n_{max}}^2}\right\} \tag{5.3}$$

$$\mathbf{R} = \text{diag}\left\{\frac{1}{\Delta u_{1_{max}}^2} \frac{1}{\Delta u_{2_{max}}^2} \dots \frac{1}{\Delta u_{n_{max}}^2}\right\} \tag{5.4}$$

$\Delta u_{1_{max}}^2$ in $\mathbf{R}$ is the maximum allowable deviation for that control vector, it is similar with $\mathbf{Q}$ but for the states. The cost function can never be negative since $\mathbf{Q}$ is a real positive semi-definite matrix and $\mathbf{R}$ is the same, except it being definite. The weights can further be iterated with the method in (Luo and Lan, 1995). To solve for $\mathbf{K}$ we make use of the Lyapunov function

$$\mathbf{V}(\mathbf{x}) = \mathbf{x}^T\mathbf{P}\mathbf{x} \tag{5.5}$$

This function is the negative derivate of the cost function and can be seen as virtual energy that is added to the state vector. This reduces the problem to decrease the integrated virtual power $\mathbf{P}$. So by equating Equation 5.5 with a slightly modified Equation 5.1

$$\mathbf{x}^T(\mathbf{Q} + \mathbf{K}^T\mathbf{R}\mathbf{K})\mathbf{x} = -\frac{\mathbf{d}}{\mathbf{dt}}(\mathbf{x}^T\mathbf{P}\mathbf{x}) \tag{5.6}$$

and integrating that gives

$$(\mathbf{A}-\mathbf{BK})^{\mathbf{T}}\mathbf{P}+\mathbf{P}(\mathbf{A}-\mathbf{BK})=-(\mathbf{Q}+\mathbf{K}^{\mathbf{T}}\mathbf{RK}) \tag{5.7}$$

The result is called the Lyapunov Equation. It is possible to calculate **P** by rewriting this equation to

$$\mathbf{A}^{\mathbf{T}}\mathbf{P}+\mathbf{PA}-\mathbf{PBR}^{-1}\mathbf{BP}+\mathbf{Q}=\mathbf{0} \tag{5.8}$$

There are numerous software packages that can calculate **P** from this so-called Ricatti equation. The gain matrix **K** is subsequently calculated with

$$\mathbf{K}=\mathbf{R}^{-1}\mathbf{B}^{\mathbf{T}}\mathbf{P} \tag{5.9}$$

And finally, the control output is then determined with

$$\mathbf{u}=-\mathbf{Kx} \tag{5.10}$$

5.2. Interpolation Methods

Numerical interpolation is a method to obtain the values between the data points of a discretised function. One of the main applications is to calculate the aerodynamic coefficients which is given as a table, another application is when using pseudospectral methods to get the values between the collocation points. The outcome of the interpolation determines for example, what angle of attack the controller should follow, it is therefore important to pick the right interpolator, since each one generates a different shape. Four different forms of interpolation shall be discussed here: Linear interpolation, Lagrange interpolation, Hermite cubic interpolation, and cubic-piecewise-spline interpolation.

The easiest form of interpolation is that of linear interpolation (Kreyszig, 2011). This method takes the two closest points available, and it averages between them resulting in a straight line:

$$q(x)=q(x_0)+\frac{q(x_j)-q(x_{j-1})}{x_j-x_{j-1}}(x-x_{j-1}) \tag{5.11}$$

This method is not very accurate for non-linear functions, especially when there are a limited amounts of nodes on the playing field. It is therefore only useful for predictable functions in which the distance of the nodes are not that far apart. Another downside is that the derivative of the function at the nodes is discontinuous. The reason that it is still mentioned here is because of the low computational requirement. Linear interpolation is solely used when calculating the aerodynamic coefficients. The discontinuity of the derivative does not pose any problems and the coefficients have to be calculated very frequently during the simulation. The minor error caused by the loss of non-linearity of the aerodynamics database needs to be taken for granted.

The next method of interpolation is by using Lagrange polynomials. This method finds the smallest polynomial by which all the data points ranging from (y_0,x_0) until (y_k,x_k). The interpolated result is calculated with

$$Y(x)=\sum_{n=0}^{N} y_n L_n(x) \tag{5.12}$$

where

$$L_n(x)=\prod_{\substack{k=0\\k\neq i}}^{N}\frac{x-x_k}{x_n-x_k} \tag{5.13}$$

is the Lagrange polynomial. One problem of Lagrange polynomials is the Runge phenomenon (Kreyszig, 2011). This is a problem that occurs at the boundaries of the solution where y_j spikes to a minimum or maximum to adher to the polynomial as seen in Figure 5.1. It can be avoided by choosing data-points that do not have an equidistant spacing, such as the Chebyshev polynomials or by using the Radau quadrature. A further downside of this method is that it is computationally expensive, especially if the amount of points increases.

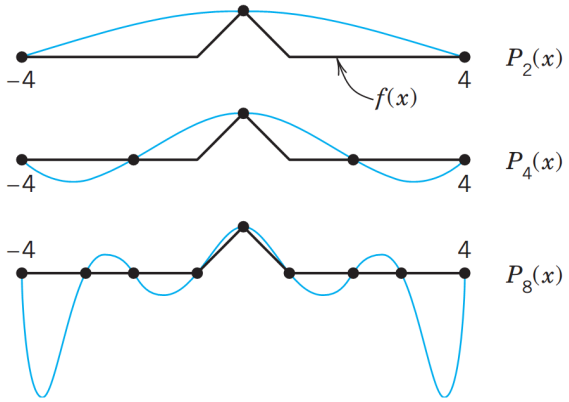


Figure 5.1: A linear interpolation compared to Lagrange interpolation for n points. The Runge phenomenon is clearly visible when increasing the points. (Kreyszig, 2011)

The third method used is the Hermite cubic interpolation method. This method is unique compared to the others presented here that it not only requires the value of the data points left and right of the requested value, but also their derivatives, which results in a interpolated line that has a continuous first derivative. The interpolated value $q(x)$ is calculated with

$$q(x) = h_{00}q_{j-1}(x) + h_{10}(x_j - x_{j-1})m_{j-1} + h_{01}q_j(x) + h_{11}(x_j - x_{j-1})m_j \tag{5.14}$$

with

$$h_{00} = 2t^3 - 3t^2 + 1 \tag{5.15}$$
$$h_{10} = t^3 - 2t^2 + t \tag{5.16}$$
$$h_{01} = -2t^3 + 3t^2 \tag{5.17}$$
$$h_{11} = t^3 - t^2 \tag{5.18}$$

The variable t in these equations is a result from a transformation to the $[0, 1]$ domain of x between x_{j-1} and x_j with

$$t = \frac{x - x_{j-1}}{x_j - x_{j-1}} \tag{5.19}$$

The two tangents m_{j-1} and m_j are, if not available, calculated by the first-order derivative between the points left and right

$$m_{j-1} = \frac{q_j - q_{j-2}}{x_j - x_{j-2}} \tag{5.20}$$
$$m_j = \frac{q_{j+1} - q_{j-1}}{x_{j+1} - x_{j-1}} \tag{5.21}$$

The tangent between q_j and q_{j-1} is used for boundary cases. Extrapolation is performed by using the tangent of the boundary points.

The fourth and last form of interpolation used is cubic spline interpolation. This method involves using a n-degree (3 in the case of cubic) polynomial instead of using a high degree polynomial by splitting up the available data points into smaller sets (splines). The advantage here is that, unlike the lagrange polynomial, not all the data points are used, and the Runge phenomenon is avoided. The requirement here is that the function should be smooth, this means that it is differentiable in the first and second degree.
So for every spline there exists a polynomial $q(x)$ that has a degree not higher than three, such that there are two points

$$q_j(x_j) = f(x_j), \quad q_{j+1}(x_{j+1}) = f(x_{j+1}) \quad (j = 0, 1, \ldots, n-1) \tag{5.22}$$

The cubewise polynomial has the well-known form

$$q_j(x) = a_j(x - x_j)^3 + b_j(x - x_j)^2 + c_j(x - x_j) + f(x_j) \tag{5.23}$$

with the derivatives

$$q'_j(x) = 3a_j(x - x_j)^2 + 2b_j(x - x_j) + c_j \tag{5.24}$$
$$q''_j(x) = 6a_j(x - x_j) + 2b_j \tag{5.25}$$

The following relationships are present since it is twice continuously differentiable.

$$q_j(x_{j+1}) = f(x_j) : f(x_{j+1}) - f(x_j) = a_j(h_j)^3 + b_j(h_j)^2 + c_j(h_j) \tag{5.26}$$
$$q'_{j-1}(x_j) = q'_j(x_j) : 3_{a-1}h^2_{j-1} + 2b_{j-1}h_{j-1} + c_{j-1} = c_j \tag{5.27}$$
$$q''_{j-1}(x_j) = q''_j(x_j) : 6a_{j-1}h_{j-1} + 2b_{j-1} = 2b_j \tag{5.28}$$

with $h_j = x_{j+1} - x_j$. The next step is to solve for a, b, and c.

$$q''_j(x_{j+1}) = q''_{j+1}(x_{j+1}) : a_i = \frac{b_{j+1} - b_j}{3h_j} \tag{5.29}$$
$$q_j(x_{j_1}) = f(x_{j+1}) : c_i = \frac{y_{j+1} - y_j}{h_j} - \frac{1}{3}(2b_j + b_{j+1})h_j \tag{5.30}$$
$$q'_{j-1}(x_j) = q'_j(x_j) : \frac{1}{3}h_{j-1}b_{j-1} + \frac{2}{3}(h_{j-1} + h_j)b_j + \frac{1}{3}h_j b_{j+1} = \frac{f(x_{j+1}) - f(x_j)}{h_j} - \frac{f(x_j) - f(x_{j-1})}{h_{j-1}} \tag{5.31}$$

The first step is to solve b with Equation 5.31, and then solve Equations 5.29 and 5.30. The polynomial from Equation 5.23 is now known. The result is a smooth curve that is not prone to the Runge phenomenon and the effect of points that is further in the axis does not is not present.

5.3. Numerical Integration

During the simulation it is important to predict what the result of a function is at a later time step. It is often impossible to obtain a solution when working with differential equations analytically. Numerical integration is then used to calculate the value of the next discrete point by using the values of the points before the requested point. The basic form is that the integral can be approximated by sampling a finite number of points and combine that with a weighted sum (Garg, 2011)

$$x_0 + \int_{x_0}^{x_f} g(x)dx \approx x_0 + \sum_{i=1}^{N} w_i g(x_i) \tag{5.32}$$

where $g(x)$ are the differential equations and w_i is the weight associated with that sampling point. The amount of points selected and the weighing of each points determines the accuracy of the solution. There are two methods of numerical integration discussed here, that done with the Runge Kutta method, and collocation integration. The Runge Kutta method is used throughout the simulator to calculate the next state of Horus, and it shall be further explained in this section. Integration by collocation is only used in the Pseudospectral method. There, the discrete points are multiplied with weights calculated by a certain quadrature, but this is discussed in Chapter 4.

The Runge-Kutta (RK) method evaluates the first derivative at multiple points to predict the solution. Different weights are given to each point and from there on out the secant of the slope is calculated by averaging out the different points. The standard form of explicit RK methods is given with

$$\mathbf{x_{n+1}} = \mathbf{x_n} + \mathbf{h}\sum_{\mathbf{i=1}}^{\mathbf{s}} \mathbf{b_i k_i} \tag{5.33}$$

where k_i is represented as

$$\mathbf{k_1} = \mathbf{g}(\mathbf{t_n}, \mathbf{x_n}) \tag{5.34}$$
$$\mathbf{k_2} = \mathbf{g}(\mathbf{t_n} + \mathbf{c_2 h}, \mathbf{x_n} + \mathbf{a_{21} k_1}) \tag{5.35}$$
$$\mathbf{k_3} = \mathbf{g}(\mathbf{t_n} + \mathbf{c_3 h}, \mathbf{x_n} + \mathbf{a_{31} k_1} + \mathbf{a_{32} k_2}) \tag{5.36}$$
$$\vdots \tag{5.37}$$
$$\mathbf{k_s} = \mathbf{g}(\mathbf{t_n} + \mathbf{c_s h}, \mathbf{x_n} + \mathbf{a_{s1} k_1} + \mathbf{a_{s2} k_2} + \ldots + \mathbf{a_{s,s-1} k_{s-1}}) \tag{5.38}$$

a_i and b_i are both the weights of each dependent step, and c_i increases the step-size. c_1 is always zero since the first point is taken at the beginning of the evaluation. The same holds for $a_{1,i} = 0, i = 1, 2, ..., s$, since the evaluation of k_i cannot depend on previously calculated values. a_i, b_i, and c_i can be combined into a Butcher Tableau in the following form:

$$\begin{array}{c|c} \mathbf{c} & \mathbf{A} \\ \hline & \mathbf{b^T} \end{array} = \begin{array}{c|cccccc} 0 & 0 & 0 & 0 & \cdots & 0 \\ c_2 & a_{21} & 0 & 0 & \cdots & 0 \\ c_3 & a_{31} & a_{32} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ c_2 & a_{s1} & a_{s2} & \cdots & a_{s,s-1} & 0 \\ \hline & b_1 & b_2 & b_3 & \cdots & b_s \end{array} \tag{5.39}$$

The non-relevant zero's are usually left out as blanks. The presented equations here have the general form of many different type of integrators that have been developed over the years. Two shall be discussed here, the Runge-Kutta Fehlberg 45 (RKF45) and Runge-Kutta Fehlberg 78 (RKF78) (Montenbruck and Gill, 2012). Both methods have proven their usefulness in literature and are the go-to integrators for simulations when trying to reduce the complexity but still have accurate results. The RKF45 butcher tableau is

$$\begin{array}{c|ccccc} 0 & & & & & \\ \frac{1}{4} & \frac{1}{4} & & & & \\ \frac{3}{8} & \frac{3}{32} & \frac{9}{32} & & & \\ \frac{12}{13} & \frac{1932}{2197} & -\frac{7200}{2197} & \frac{7296}{2197} & & \\ 1 & \frac{439}{216} & -8 & \frac{3680}{513} & -\frac{845}{4104} & \\ \frac{1}{2} & -\frac{8}{27} & 2 & -\frac{3544}{2565} & \frac{1859}{4104} & -\frac{11}{40} \\ \hline & \frac{16}{135} & 0 & \frac{6656}{12825} & \frac{28561}{56430} & -\frac{9}{50} & \frac{2}{55} \\ & \frac{25}{216} & 0 & \frac{1408}{2565} & \frac{2197}{4104} & -\frac{1}{5} & 0 \end{array} \tag{5.40}$$

The two different set of weights b are used to determine the local error by comparing the upper solution which is a 5$^{\text{th}}$ order solution, to the lower one which is the 4$^{\text{th}}$ order. The local error is approximated with

$$\mathbf{le} = |\mathbf{x}^*_{\mathbf{n+1}} - \mathbf{x_{n+1}}| \tag{5.41}$$

The step-size can then be increased or decreased depending on whether the tolerances are met. The tolerance ϵ is dependent on the application, but is set to $1e^{-15}$ during the simulation. The new step-size is calculated if either the tolerance is not met, or when the step-size is equal to the requested time step with

$$h^* = \eta \sqrt[p+1]{\frac{\epsilon}{le}} h \tag{5.42}$$

where p is the order of the method and η is a constant set at 0.84 to increase the stability of the new step-size in order to avoid another unsuccessful step. The initial step-size is set at the same frequency as the inner loop, which is 20 hertz. This frequency already proved to be accurate enough to avoid doing multiple iterations in one time-step, but it should still be kept in place during the regions of the simulation where non-linearity is higher than normal to make sure that the solution is accurate enough. The same method is applied for RKF78 and the equivalent Butcher tableau is found in (Montenbruck and Gill, 2012).

5.4. Numerical Differentiation

This section discusses numerical differentiation techniques, which is used a lot during the calculations of the jacobians of the dynamic equations when using pseudospectral methods, it is thus one of the most

frequently used method during the simulation. When choosing a numerical differentiation technique it is therefore important to assess two different parameters, that of the accuracy of the result, and how long it takes to calculate the derivative of the function. Another requirement that can be added but is not as important, is the difficulty of the implementation, but it not the first priority since the same sub-routine is called continuously. It is also closely related to the first requirement, computational speed. Difficult methods usually require more calculations, which show up negatively when one assesses the primary requirement.

There are two ways to calculate the derivative of a function, the continuous and the discrete approach (Corless and Fillion, 2013). The continuous method is far more accurate, only round-off errors are present, but it requires that all the used equations are differentiated by hand. The work required to derive the equations, which are numerous and sometimes complex, is massive, and this is therefore not feasible. The discrete approach does not require this, and a single program can calculate the derivative by discretising the function and subsequently differentiate the results to get an approximation of the derivative. Different methods are available, such as forward differencing and n-point stencils, also called finite-differencing techniques, or packages can be found on the internet that involve automatic differentiation, implementing these into the existing c++ framework can however be complicated, and these are therefore not implemented.

One of the most popular finite differencing-technique is the forward technique. This method is very easy to implement and can provide first-order results, it is however not very accurate. The formula is derived by expanding the taylor series around a small deviation h at a certain point.

$$f(x+h) = f(x) + hf'(x+h) + \mathcal{O}(h) \tag{5.43}$$

which can then be rewritten to, assuming that h is relatively small into

$$f'(x) \approx \frac{f(x+h) - f(x)}{h} + \mathcal{O}(h) \tag{5.44}$$

The central diference technique is similarly calculated by subtracting the forward difference with the backward difference ($f(x-h)$) which results in

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h} + \mathcal{O}(h^2) \tag{5.45}$$

It is possible to expand the taylor series to get techniques such as the five-point stencil or higher. These equations however require more processing power, and are therefore not discussed. Another method than taylor series to calculate the derivative is by using complex numbers (Martins et al., 2003). Assume that there is a function $f = u + iv$ which has a complex variable $z = x + iy$. It can safely be assumed that f is differentiable in the complex plane and that the Cauchy-Riemann equations apply and thus

$$\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y} \tag{5.46}$$

$$\frac{\partial u}{\partial y} = -\frac{\partial v}{\partial x} \tag{5.47}$$

These equations give the relations between the real and imaginary part of the function. The first Cauchy-Riemann equation can be derived into

$$\frac{\partial u}{\partial x} = \lim_{h \to 0} \frac{v(x + i(y+h)) - v(x+iy)}{h} \tag{5.48}$$

The only part that is of interest in these equations is that of the real part of the function and variables, which means that $y = 0$, $u(x) = f(x)$, and $v(x) = 0$. Equation 5.48 can then be rewritten to

$$f'(x) = \frac{\mathrm{Im}[f(x+ih)]}{h} \tag{5.49}$$

This is called the complex-step derivative approximation (Martins et al., 2003). The error order is calculated similarly as the previous examples by using Taylor series, the result is that the error order is $\mathcal{O}(h^2)$, which is the same as the central difference method. The main difference however, is that the complex-step does

Table 5.1: The results of different differentiation methods in C++.

	stepsize [-]	relative error [-]	time [s]
Forward difference	$10 \cdot 10^{-8}$	$1.64 \cdot 10^{-7}$	2.86
Central Difference	$10 \cdot 10^{-5}$	$8.58 \cdot 10^{-10}$	3.47
Complex-step	$10 \cdot 10^{-10}$	$1.20 \cdot 10^{-15}$	2.98

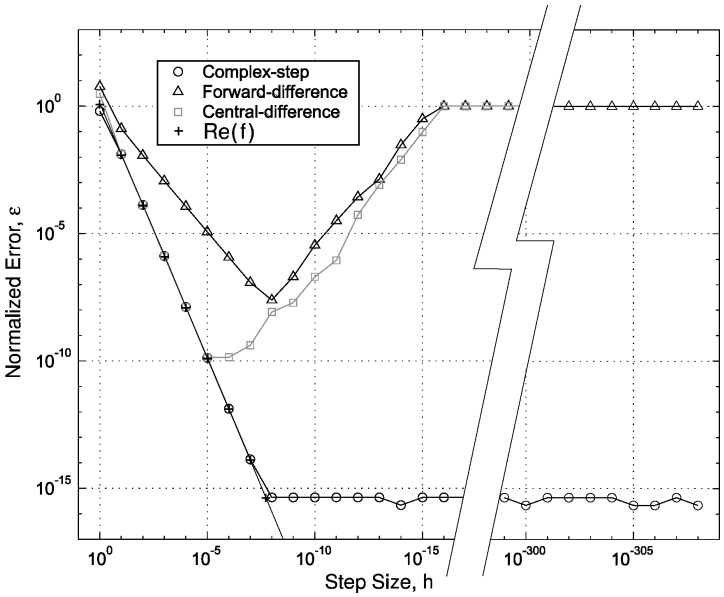


Figure 5.2: Relative error of different differentiation methods with decreasing stepsize h (Martins et al., 2003).

not have subtractive cancellation errors. An example of the errors of all the methods discussed above per stepsize is given in Figure 5.2. It shows that the relative error for both the forward- and central-difference method far exceed that of the complex-step by several orders. The relative error is calculated with

$$\epsilon = \frac{|f'(x) - f'(x)_{ref}|}{|f'(x)_{ref}|} \tag{5.50}$$

Both show a decrease in error as expected until the subtractive cancellation error becomes significant, the absence of this type of error in the complex-step avoids the up-going phenomenon.

A small program has been written in C++ to simulate the obtainment of the derivatives as would be done during the actual simulation to test the computational requirement. The following simple equation has been used

$$f(x) = e^{3x} \cos x + 2x^2 e^x \tag{5.51}$$

The advantage of this equation is that it the derivation can easily be calculated by hand, to test the accuracy as well. The result of acquiring the derivate of 10 million different points between -5 and 5 is seen in Table 5.1. The relative error is what is to be expected as it is shown in Figure 5.2. The processing power required by all is relatively equal, considering the scale, with the central differencing method losing by a small margin. The results for using the five-points stencil are not shown in the table, but it required twice as much time as the other three, whilst having the same relative error as the central difference method.

The complex-step derivative method is the clear winner when analysing both the requirements, accuracy and speed, as seen above. The accuracy is the determining factor, since the speed is nearly the same. It is also very easy to implement .

6

Software Development

6.1. Software Architecture

The simulator is written in C++ and different algorithms have been built and verified. This section shall provide how the most important algorithms are built and what external software is used.

The top layer of the simulation is shown in Figure 6.1, which is very similar to other frameworks (Mooij, 2013). The outer loop frequency is set to 1 Hz, and the inner loop frequency to 20 Hz.

The simulation starts with setting the variables such as runway location and initial state. A reference trajectory is then created which takes about 1 second after the steps have been taken that are in Figure 6.2. The line in this flowchart has to be followed precisely, or else the program might not work. It is possible to add both a Lagrange and Mayer function, the rest of the options are either left or right, and not both. The striped boxes are optional, and can be skipped without a problem. The controller segment can be divided into two different options. The first option is that a cubic Hermite interpolator is used to find the control solution from the calculated trajectory. This is used in Chapter 8. The other option is that a Lagrange cost function is used in a seperate pseudospectral generator to determine the elevon and rudder angles, as seen in Chapter 9. The controls are then applied to the dynamics of the vehicle with an integrator (Figure 6.4). The relevant flight data is stored to an array, which is converted into a .csv file when the simulation ends. The new state is used in the Guidance segment (Figure 6.3) as the initial point for a new trajectory, with the old former calculated trajectory as a basis. Almost all of the settings from the reference trajectory are re-used. The simulation is ended once the terminal conditions are reached, the flight approach corridor, or if the trajectory generator can no longer find a solution.

The three main external packages that were used are Boost, Eigen, and SNOPT. Boost was used as a way to save and load data, and Eigen was used to create an mathematical environment that is close to Matlab. Adding the SNOPT library was not without hiccups. A new version (7.2.12) was required because the old version would not compile with current-day compilers, which did compile quickly, but it could only be added to the Cmake file after building a seperate static library. The original code incorporated TUDAT (Delft University of Technology, 2012) in its design process. This was later removed because the framework did not comply with the TUDAT requirements, and it was no longer possible to compile because of the implementation of new compilers, this error is resolved in the newer versions. It was however, too late to return back to TUDAT. Multi-core features were added, this distributed the CPU load across several cores. This however, did not speed up the process and was dropped because of the increased complexity. The beginning of the development phase was quite tough, since the knowledge of developing specifically in C++ was very limited. Tasks that could easily be performed in Matlab within half an hour could take up to days in C++ because there was some illogical or pointing error.

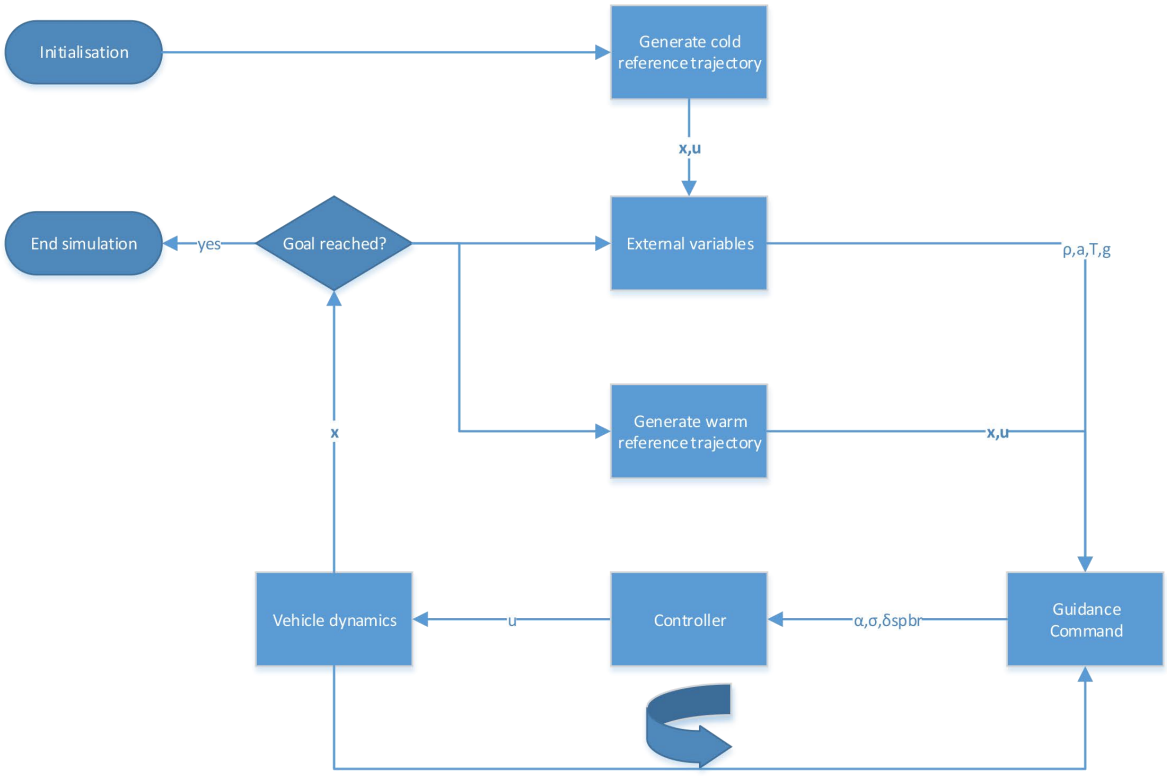


Figure 6.1: Framework of the simulator

6.2. Software Validation

The simulation consists of different subroutines working together to get the required result. It is important to know what the requirements in terms of input is of each subroutine and what the expected result is before they are linked together. The last part is done through the method of verification and validation. The definitions of both are given by (Oberkampf and Trucano, 2002) and are

- **Verification** The process of determining that a computational mode accurately represents the underlying mathematical model and its solution.
- **Validation** The process of determining the degree to which a model is an accurate representation of the real world from the perspective of the intended use of the model.

In other words, verification is done to insure that there are no bugs present in the code that prevent the models created to behave as they are supposed to. Validation looks at the result of the model, after it is verified, and compares it to experiments to check whether or not it can be considered realistic. No experiments shall be performed in the foreseeable future, validation is therefore not applied in the thesis. Verification, however, still plays a big role. A bottom-up approach is used when verifying the results of the different subroutines, this means that each one is tested separately first with unit tests, and then when working in unison. Sanity checks were performed along the way during the creation of the code, preventing majors errors.

The Atmosphere and Gravitational Field
Matlab has been used to verify the results obtained from the equations as presented in Section 3.4.2. The gravitational acceleration is calculated using the `gravityspericalharmonic`-function with the inclusion of the J_2-term. The pressure, temperature, and atmospheric density are calculated using the `atmoscoesa`-function. The C++-code produces nearly identical results to the matlab function as seen in Figure 6.5.

Numerical Interpolation

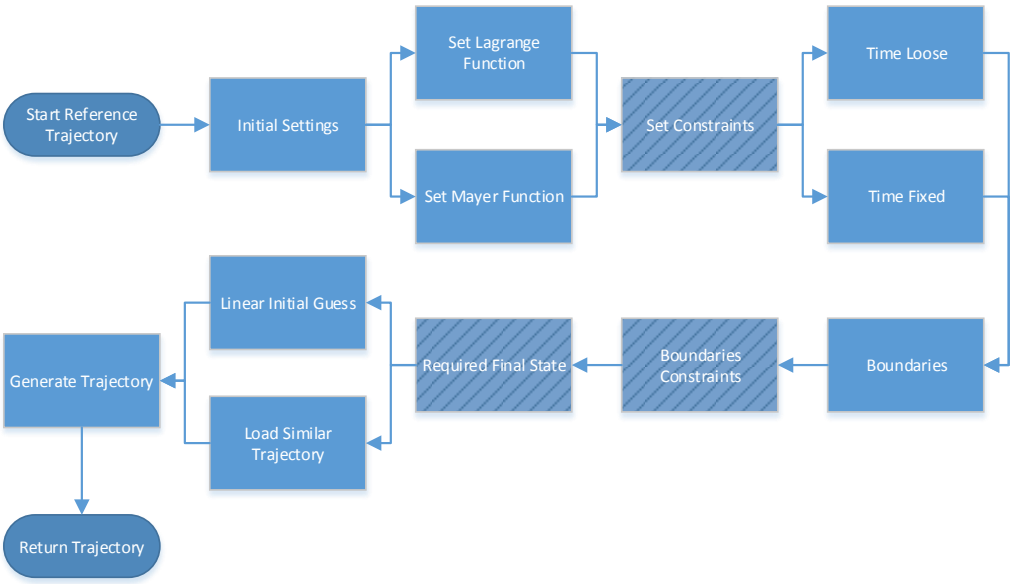


Figure 6.2: The required steps to initialise the trajectory generator. The striped boxes are not required, and one of the two boxes have to be used when it is splitted.

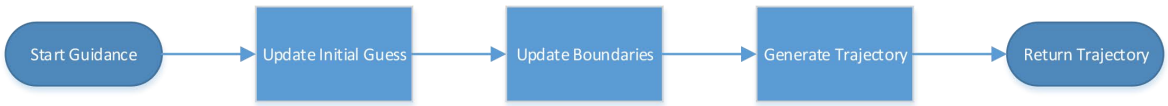


Figure 6.3: The trajectory generator has fewer steps later in the simulator.

The interpolation was tested by comparing the results of the C++ code with the `interp1`-function. The different techniques used here could be used by adding the following command: `linear`,`pchip`, and `spline` for linear, cubic Hermite, and cubic spline interpolation respectively.

Numerical Differentiation

The numerical differentiators that were seen in Section 5.4 were all tested by comparing the result with an analytical answer. All the results were inside the expected error margins.

Numerical Integration

A numerical integration simulation is used to test the integration schemes RKF45 and RKF78 as seen in Section 5.3. Three different re-entres with the Apollo shuttle is simulated with simplified variables, by varying the initial flight path angle. The non-linear dynamical equations used are the following

$$\dot{V} = -\frac{D}{m} - g\sin\gamma \tag{6.1}$$

$$V\dot{\gamma} = \frac{L}{m} - g\cos\gamma + \frac{V^2}{R} \tag{6.2}$$

$$\dot{x} = V\cos\gamma \tag{6.3}$$

$$\dot{h} = V\sin\gamma \tag{6.4}$$

These are nearly similar to the ones used in Section 3.6, with the exception of the change in flight path angle, which now takes into account that the Earth is spherical, and not flat. The variables found in these equations as well as in the lift and drag formulae are found in Table 6.1. It is assumed that the lift and drag coefficient remain constant during the entire flight. This simulation does not use the spherical gravity

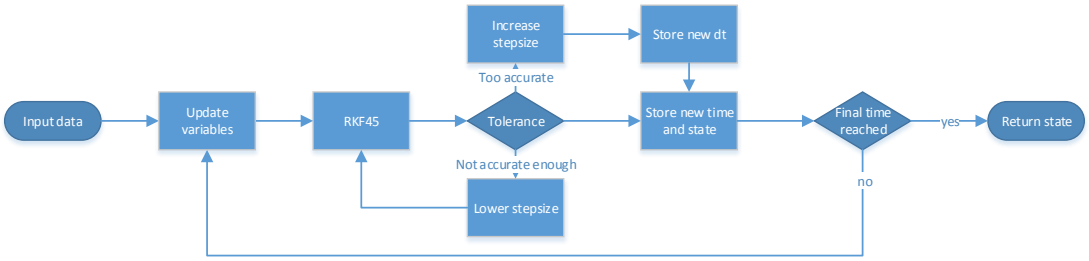


Figure 6.4: The RKF45 integrator flowchart.

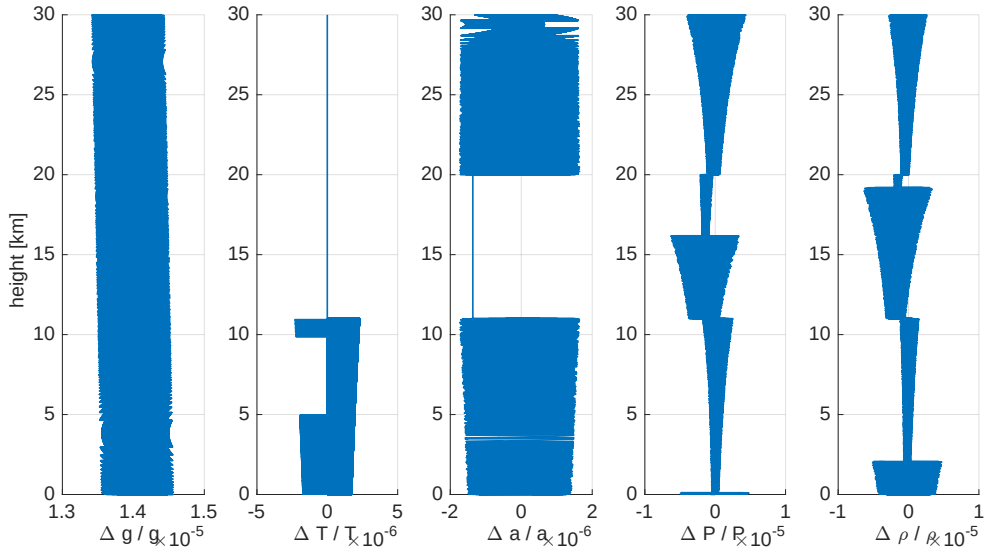


Figure 6.5: Validation of the US76 atmospheric model.

field nor the US76 atmospheric model. Instead, a simplified inversed square gravity field is used and an exponential atmospheric model to calculate the density of the atmosphere. The gravity is calculated with

$$g(h) = g_0 \left(\frac{R_E}{R_E + h} \right)^2 \tag{6.5}$$

and the atmospheric density with

$$\rho(h) = \rho_0 \exp^{-\beta h} \tag{6.6}$$

with $\beta = 1.40845 \cdot 10^{-4}$ m^{-1}. The results can be seen in Figure 6.6, which are identical to those found in (Hirschel and Weiland, 2009). These results are created with the RKF78 integrator, but the RKF45 integrator has the same results. It can therefore be said that both the integration schemes are working correctly.

Aerodynamics

The aerodynamics of both the vehicles originated in Matlab. The results could therefore be compared between the Matlab and C++ results. Other tests to see if there were any hiccups were creating L/D plots and $C_L(\alpha, M)/C_D(\alpha, M)$ plots. Any errors would immediately be noticeable due to the sudden jumps. The results are verified with the supplied software from Dassault (2001).

Pseudospectral Methods

Table 6.1: Values of variables used in the Apollo simulation (Hirschel and Weiland, 2009).

Symbol	Quantity	Value
A_{ref}	Reference area [m^2]	12.02
C_L	Lift coefficient	0.374
C_D	Drag coefficient	1.247
m	Total vehicle mass [kg]	5 470.0
V_e	Flight velocity at entry	7 670.0
γ_e	Flight path angle at entry [°]	$-0.75, -1.50, -3.50$
h_e	Height at entry [km]	120.0

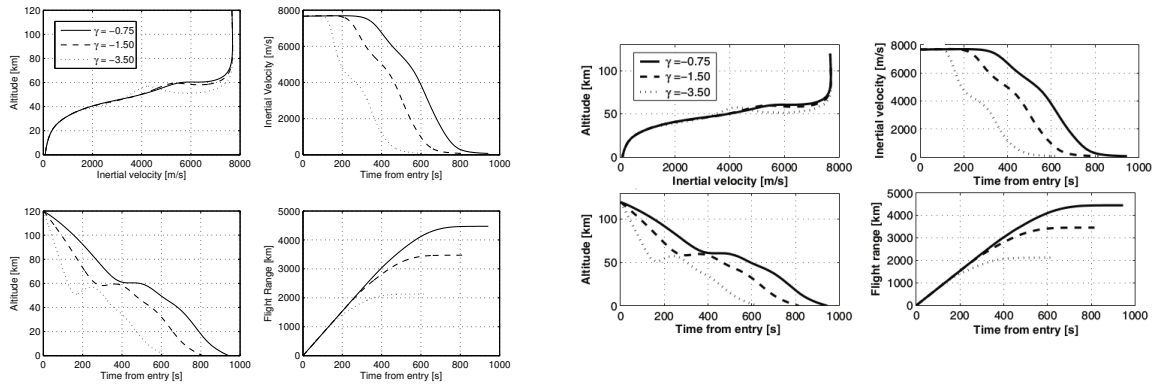


Figure 6.6: Simulation of the Apollo trajectory from C++ compared to (Hirschel and Weiland, 2009).

It is vital to make sure that the core method that is used to simulate the results is working properly. The following example problems were tested, and each has a small description of the added difficulty of each problem

- Orbit Raising This test was there to ensure that the basic principles and the Mayer part of the cost function were working (Sagliano and Theil, 2013)
- Space Shuttle This problem has variables that vary greatly, requiring scaling (Betts, 2010).
- Hang Glider The final state is fixed in this problem (Betts, 2010).
- Benham-Bryson The performance index here is verified with a Lagrange function.
- Constrained Orbit Raising The thrust angle vector is constrained to 1, it is almost similar to the unconstrained version.

The first two are demonstrated in Section 4.8.

Wind Models

The wind models that were extracted from (Johnson, 2008) were interpolated using Cubic Hermite Polynomials. Figure 3.10 shows the results of doing so, and the results are directly interpolated from Table 3.2.

Attitude Control

Both control methods were tested by creating a simulation that only involved the rotational motion. The aerodynamic coefficients are validated as well by doing this test. Different angle of attack and bank angles commands were tested and performed well as seen in Figure 6.7 and 6.8. The test was performed at the following point and initial states

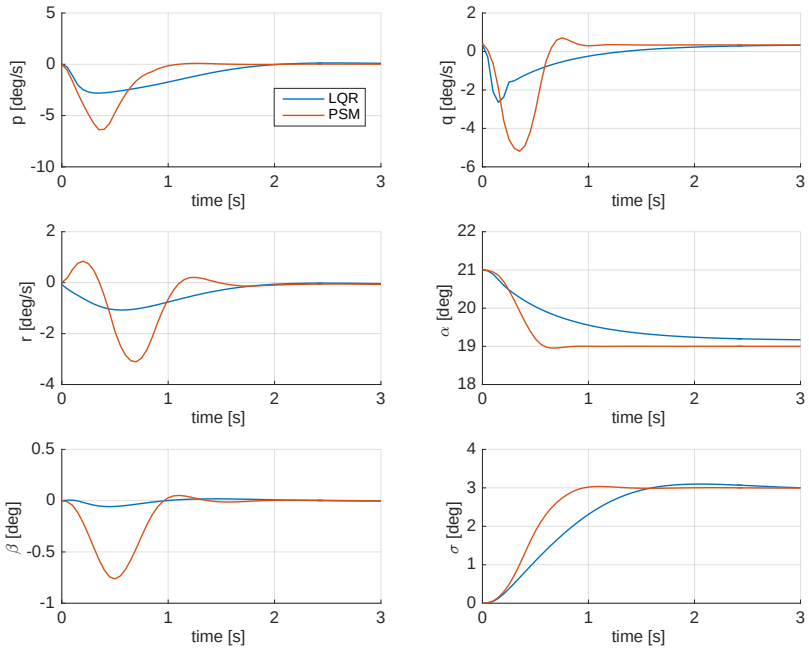


Figure 6.7: States of the attitude control test.

$$\begin{pmatrix} M \\ \gamma \\ h \\ \alpha \\ \beta \\ \sigma \\ \delta_{spbr} \end{pmatrix} = \begin{pmatrix} 1.4 \\ -10° \\ 20000\,\text{m} \\ 21° \\ 0° \\ 0° \\ -9° \end{pmatrix}$$

The goal was to reach the following aerodynamic angles

$$\begin{pmatrix} \alpha_c \\ \sigma_c \end{pmatrix} = \begin{pmatrix} 19° \\ 3° \end{pmatrix} \tag{6.7}$$

This means that there is a step of α of 2° and 3° of σ. The following allowable deviations were set for the LQR: $\Delta p_{max} = 10°$, $\Delta q_{max} = 10°$, $\Delta r_{max} = 10°$, $\Delta\alpha_{max} = 5°$, $\Delta\beta_{max} = 5°$, $\Delta\sigma_{max} = 5°$, and $\Delta\delta_{e_{max}} = \Delta\delta_{a_{max}} = \Delta\delta_{r_{max}} = 30°$. These functions were chosen after several iterations, and are quite effective as can be seen in the figures. The cost function used with PSM is

$$J = \int_{t_0}^{t_f} (\Delta\alpha)^2 + (\Delta\beta)^2 + (\Delta\sigma)^2 dt \tag{6.8}$$

Both methods have converged to the desired state after two seconds. There are some minor differences between the two. The elevator has a spike at the beginning for LQR, followed later by PSM, but this quickly returns to the trimmed state. The reactions of the aileron and rudder are similar, but there it is the PSM that reacts like this. Furthermore, PSM makes use of the sideslip angle to quickly reach σ_c, the result is that σ is aligned half a second faster. There remains a small offset for α with LQR. All in all there is a correct result, although it seems that the allowable deviations could be tweaked further to avoid the offset of α and slower reaction of LQR across the board.

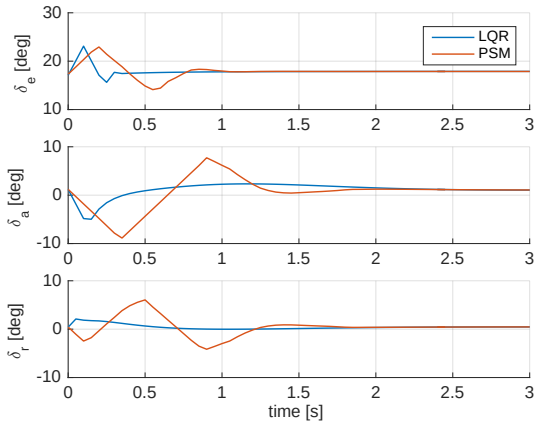


Figure 6.8: Controls of the attitude control test.

7
Vehicle Selection

A very important aspect when doing research similar to this one is to have a suitable spacecraft with enough data to perform the research. The goal of this thesis was to initially use the HORUS 2B-7.It became apparent as the research continued that the available aerodynamic database was unable to be used effectively with optimisation methods based on Newtonian methods such as pseudospectral methods. The X-38 proved to be a suitable replacement, as seen in this chapter. The comparisons made here are all with respect ot finding optimal trajectories, using 3-DOF simulations. There is no rotational aerodynamic data available for HORUS for $M < 1.2$ (Mooij, 1995). The data differs too much when flying in the transonic regime, it can therefore not be simply interpolated. The X-38 was assigned to fill this gap. A 3-DOF analysis is shown here in order to select the RV-W that is going to be used in the subsequent chapters.

Two flight cases are compared to test the suitability of both vehicles with the optimiser and pseudospectral methods. A relatively simple subsonic case is shown first in Section 7.1, it is then followed by a supersonic case in Section 7.2.

7.1. Maximum Range Subsonic Flight

The first test to compare the performance of the HORUS and the X-38 is a simulation of a maximum range flight with a turn in the subsonic regime.

The cost function is set to

$$J = -x \tag{7.1}$$

and the initial state is

$$\begin{pmatrix} x_0 \\ y_0 \\ h_0 \\ V_0 \\ \gamma_0 \\ \chi_0 \end{pmatrix} = \begin{pmatrix} 0.0\,\text{m} \\ 0.0\,\text{m} \\ 10000.0\,\text{m} \\ 250.0\,\text{m/s} \\ -10.0^\circ \\ 0^\circ \end{pmatrix}$$

The initial heading angle is pointed north, and the cost function is set to maximise the distance travelled in eastern direction, the simulation is set in the vertical frame, which means that a turn has to be performed. Therefore, the final heading angle is 90°. The final height is set to 500 metres.

The results can be seen in Figure 7.1 and 7.2. Both trajectories show the same tendencies. The flight for HORUS is 306.3 seconds and that of the X-38 is 217.6 seconds. The reduced flight time is inherent to the different configuration of both vehicles. The HORUS is a winged re-entry vehicle which has a higher L/D ratio than the X-38. The angle of attack pattern is similar with respect to that both angles stabilise at a certain angle (8° for HORUS and 23° for X-38). A spike can be seen at both the RV/W at the end of the control scheme, this manoeuvre attempts to get a few extra metres. A limiter is required that dampens this effect. The turn is performed at the start with similar bank angles, although HORUS requires a kilometre

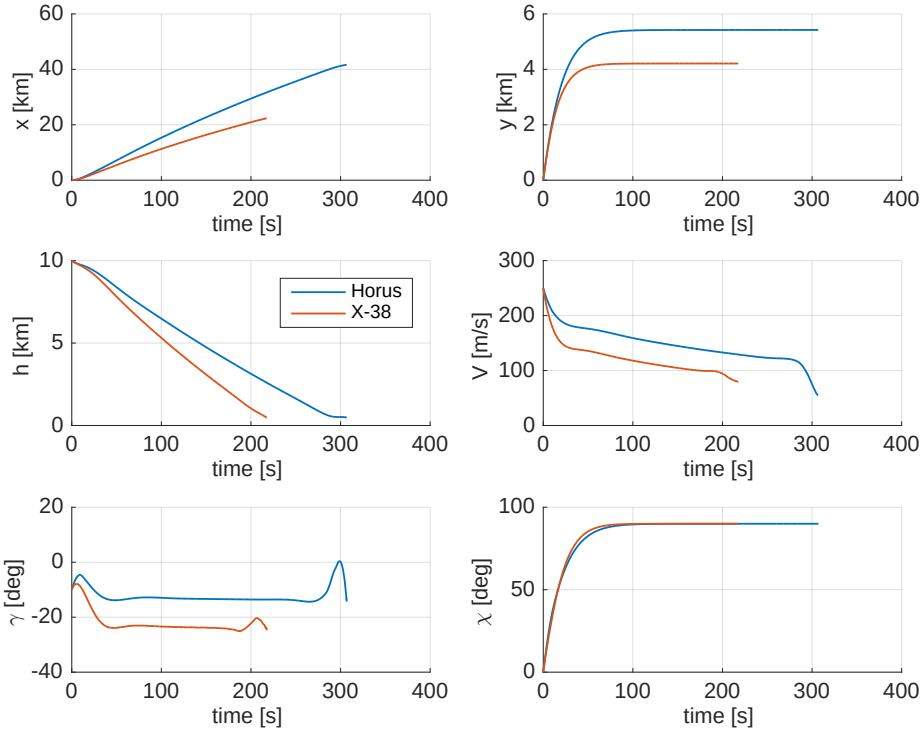


Figure 7.1: The states of a subsonic flight of the X-38 and HORUS.

more in the g-direction to reach $90°$. This is because the velocity drop for HORUS is smaller, which results in a smaller negative flight-path angle.

The validity of the result can be tested by using the steady state glide theory that uses the performance curve or the velocity polar (Ruijgrok, 2009). This curve gives the relationship between the rate of descent (vertical velocity) and the horizontal velocity. The lenght of the vector from the origin to the point of the curve is the velocity and the angle between the vector and the horziontal velocity axis is the flight path angle. Each curve is only valid for that specific state. Furthermore, it is assumed that there is a straight flight ($\sigma = 0$), a constant velocity ($\dot{V} = 0$), and that the flight path angle is constant ($\dot{\gamma} = 0$). The dynamic equations from Equations 3.47 and 3.48 then become the following force equilibrium

$$L - mg\cos\gamma = 0 \tag{7.2}$$

$$-D - mg\sin\gamma = 0 \tag{7.3}$$

The aerodynamic forces can be rewritten to

$$\frac{1}{2}\rho V C_L S = mg\sin\gamma \tag{7.4}$$

$$\frac{1}{2}\rho V C_D S = -mg\sin\gamma \tag{7.5}$$

The velocity is then

$$V = \sqrt{\frac{mg}{S}\frac{2}{\rho}\frac{1}{C_L}\cos\gamma} \tag{7.6}$$

and the flight path angle is calculated by dividing Equation 7.4 by Equation 7.5

$$\tan\gamma = -\frac{C_D}{C_L} \tag{7.7}$$

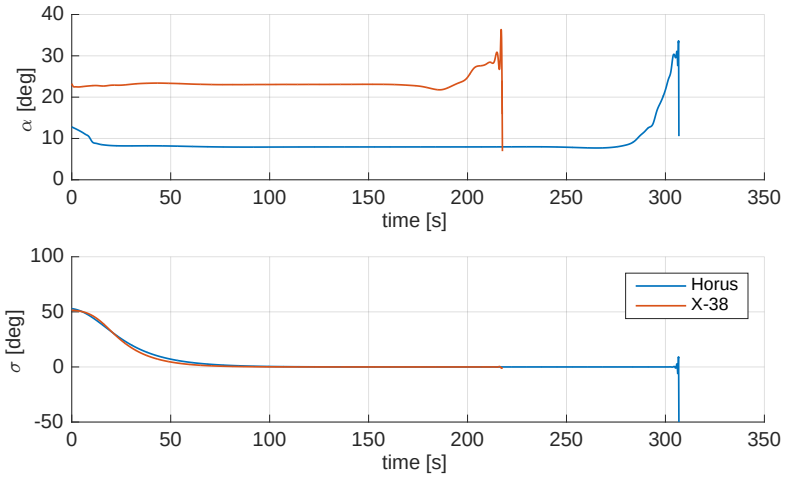


Figure 7.2: The controls of a subsonic flight of the X-38 and HORUS.

The rate of descent can then finally be calculated by

$$RD = V \sin\gamma \tag{7.8}$$

The curve can now be found by calculating the above equations with the interval of possible angle of attacks. Three points are of interest when analysing these curves (Ruijgrok, 2009). The first one is the smallest velocity possible, which is the stall speed and it can be found at the end of the upper part of the line. The second part is the point of maximum endurance, or maximum flight time. This point coincides with the smallest RD in the graph, where the tangent is horizontal. The third one is the maximum range steady-state, this is flown when the ratio between the horizontal and vertical speed is the largest. The state can be found by calculating the tangent from a line to the origin to a point on the curve, and finding an identical tangent.

The performance curves are created from the state at around $t = 125$ s, since the bank angle is zero here, and it is before any pull-up manoeuvres. The diagrams are given in Figure 7.3. The difference between the two is very apparent, the increased L/D of HORUS in this regime shows a curve that is wider and smooth. The maximum endurance point is calculated, which is found by following the black dotted line. The ideal flight path angle for HORUS at this state is $14.88°$ and that of the X-38 $25.07°$, very close to the flight path angle calculated using pseudospectral methods. Th performance curve method only takes the current state into account, and does not look at other parts of the trajectory, so small differences are to be expected. It can therefore be concluded that the program works. A small remark is that the X-38 is very close to stall speed at the maximum range point, further proof that the stability and manoeuvrability in the subsonic regime is limited.

7.2. Maximum Range Supersonic Flight

The point of entry at the terminal area occurs in the supersonic regime, it is therefore not sufficient to only validate the results in the subsonic region as done in Subsection 7.1. The cost function is once again set to maximise the distance in eastern direction

$$J = -x \tag{7.9}$$

and the initial state is at the same height and velocity at the entry point of the terminal area

$$\begin{pmatrix} x_0 \\ y_0 \\ h_0 \\ V_0 \\ \gamma_0 \\ \chi_0 \end{pmatrix} = \begin{pmatrix} 0.0\,\text{m} \\ 0.0\,\text{m} \\ 25000.0\,\text{m} \\ 732.0\,\text{m/s} \\ -10.0° \\ 0° \end{pmatrix}$$

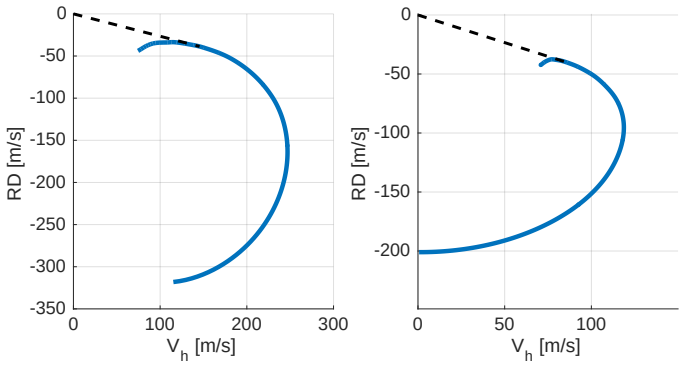


Figure 7.3: The performance diagrams of HORUS on the left and the X-38 on the right at $t = 125$ s.

The results are to be found in Figure 7.4 and 7.5. The results look similar at first glance, yet there are some concerns, especially for the flight path angle and the angle of attack from HORUS. There is an oscillation in these states emanating from the transonic phase that does not occur with the X-38. The only difference between the two simulations is the different vehicle, all the other parameters are similar.

Originally, the solution could not be found for HORUS. The HORUS aerodynamic database from De Ridder (2009) was used as described. After examination it was observed that there is a negative drag coefficient region in the $\alpha < 5$ region for $M > 1.1$. This is of course physically not possible, it was created by interpolating the original aerodynamic database from Helmersson (1988) which did not give the data for this region for a lower angle of attack. The solution presented here is by setting α_{min} at 5° for all values lower than 6.5° in Table 3.5. It is likely that the remaining part of the aerodynamic database is responsible for the oscillations of α and γ, since this movement does not occur with the X-38. Further research is required to validate this assumption. De Ridder (2009) could also not find a solution for this problem during the transonic phase where a constant α was used.

Other attempts to create trajectories, such as runway alignments, did not yield a solution for HORUS. The X-38 continuously found a solution in which it was viable or not that the intended target was reached whereas HORUS did find a solution, or it came across numerical difficulties. It thus proved that the database was too unreliable to be used due to this inconsistency and further simulations will be done with the X-38 as a basis. The performance of the X-38 has to be improved mathematically such that it gets the characteristics of a winged re-entry vehicle, similar to HORUS, this is done in the next section.

7.3. Adjusting the Performance of the X-38

It has been established in Subsection 7.2 that the X-38 is used as the spacecraft of choice during the simulations. The X-38 however, does not have the aerodynamic properties to function in the terminal area as a winged re-entry vehicle (RV-W), the properties should therefore be adjusted in such a way that it assumes the value of one such vehicle such as the HORUS or Space Shuttle.

The aerodynamic performance of an RV-W can be measured with the lift over drag (L/D) ratio (Weiland, 2014). RV-W typically have an $L/D \approx 1.5 - 2$ in the high Mach number regime, and an $L/D \approx 4.5$ in the low subsonic regime. The X-38 lift coefficient has been modified by a factor of 1.2 and 1.75 for the supersonic and subsonic regime respectively to reach these values, and mimic the aerodynamic behaviour of HORUS. The L/D ratio is seen in Figure 7.7. It is seen that the new values of the L/D ratios correspond to the requirements as stated in (Weiland, 2014). The method is far from perfect, since the L/D ratio changes at each angle of attack, the results are however acceptable, and deemed useful to use with further investigations. An interesting note is that the L/D ratio of the HORUS does not correspond to the required values. Recreations of the maximum range simulations for both the subsonic and supersonic situation show that the new maximum range flight path angle is around 12°, which is more in line with the HORUS or Space

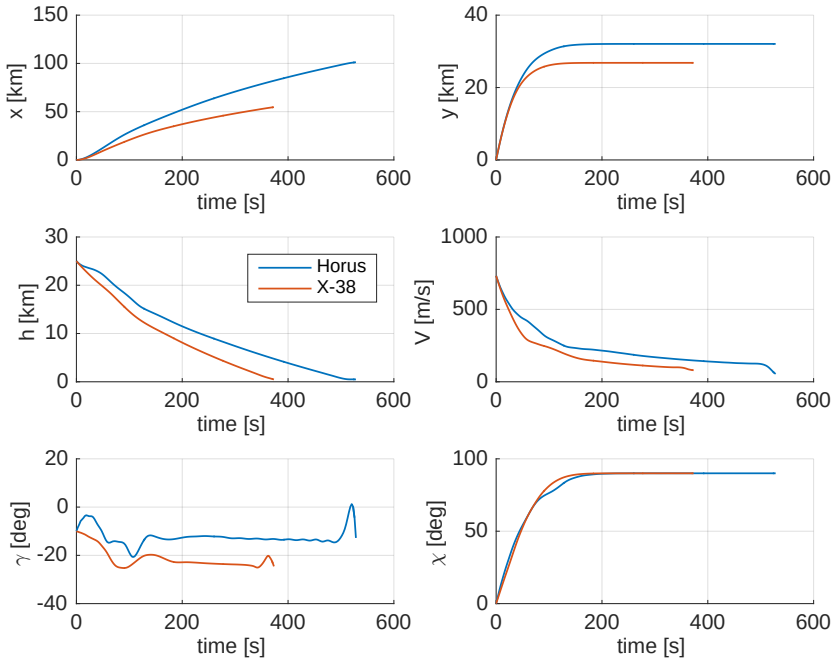


Figure 7.4: The states of a supersonic flight of the X-38 and HORUS.

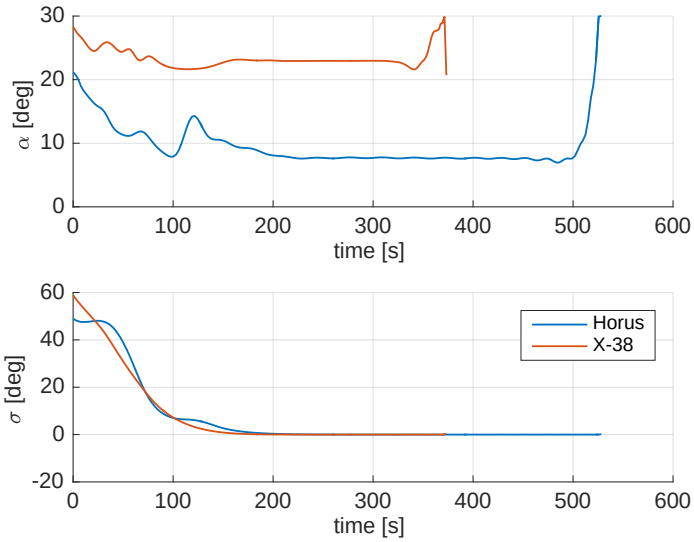


Figure 7.5: The controls of a supersonic flight of the X-38 and HORUS.

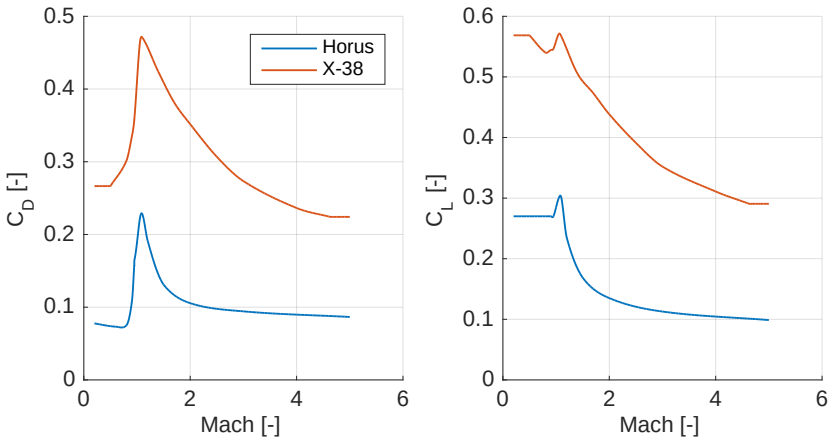


Figure 7.6: The lift and drag coefficients for $\alpha = 7.5°$ and $\alpha = 22.5°$ for HORUS and X-38 respectively.

Shuttle (Weiland, 2014).

A new supersonic maximum range flight simulation is performed with the new aerodynamic coefficients. The results of this are given in Figure 7.8 and 7.9. The flight path angle has a high oscillation at the beginning, which is then followed by oscillations that get smaller and smaller. The initial oscillation is caused by the rapid changes of the angle of attack. The oscillations that were present in the original trajectory have been increased due to the changes. Figure 7.10 shows the lift coefficient over the relevant Mach range. There is a jump in the transonic region that is caused by the changes of the coefficients. A trade-off had to be made between a smooth L/D or lift coefficient, this result was acceptable.The downside is that the flight-path angle oscillation remains throughout the rest of this research. A recommendation for future research is that this oscillation should be reduced by implementing an improved aerodynamic database, when it becomes available. It is beyond the scope of this research to fully research a new database.

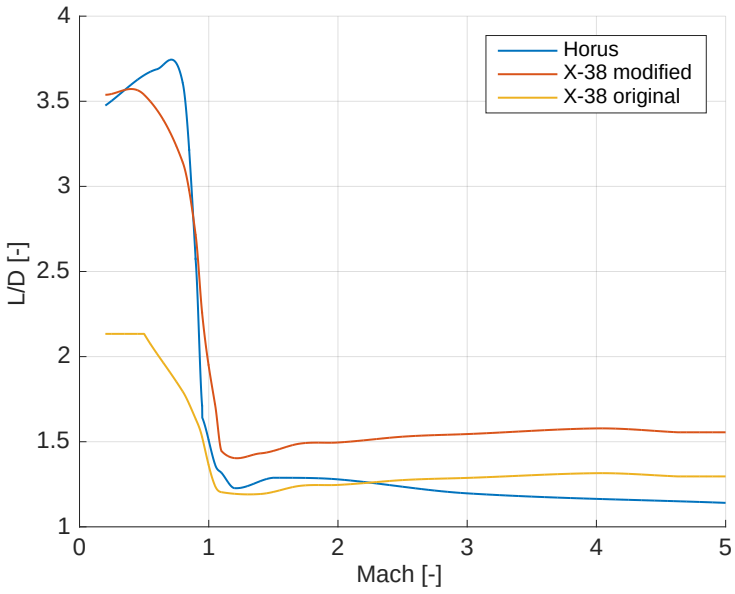


Figure 7.7: The L/D ratio of HORUS and the original and modified X-38.

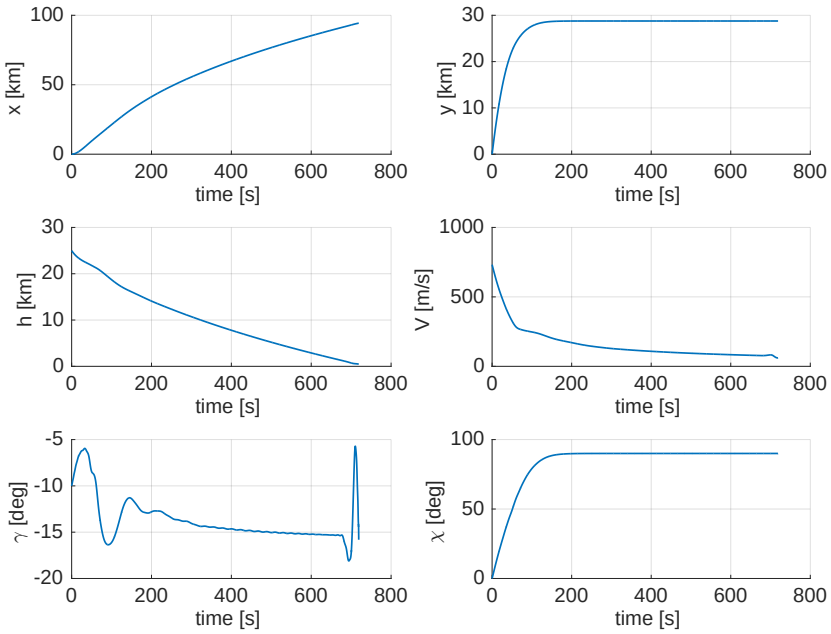


Figure 7.8: The states of a maximum range flight of X-38 with the new coefficients.

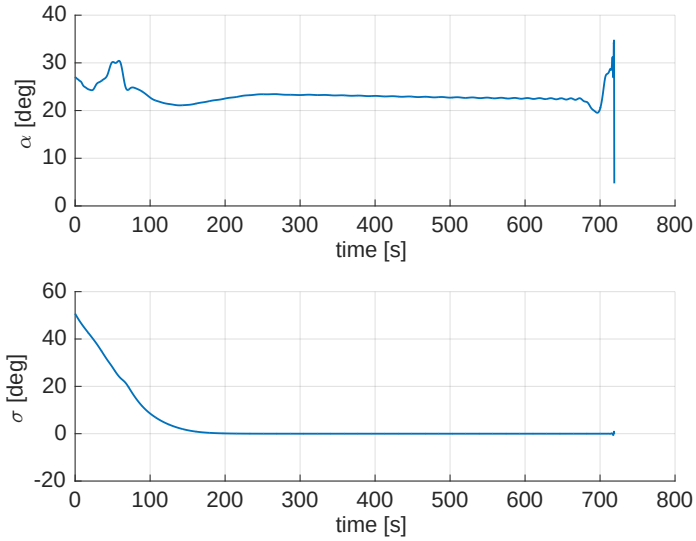


Figure 7.9: The controls of a maximum range flight of X-38 with the new coefficients.

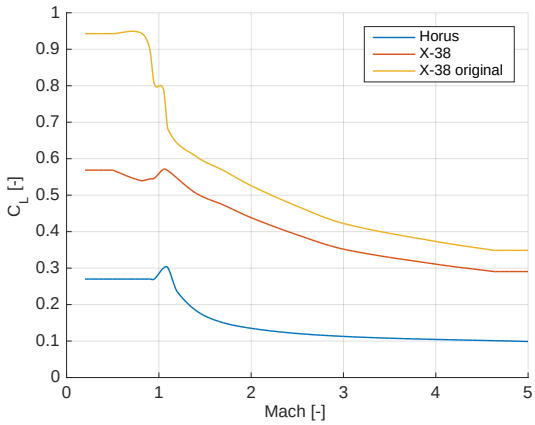


Figure 7.10: The lift coefficients for $\alpha = 7.5°$ and $\alpha = 22.5°$ for HORUS and X-38 respectively.

8

Trajectory Generation

This chapter demonstrates the possible results that can be achieved when using the technique of pseudospectral methods to find the trajectory required to align towards the runway.

The simulation starts with the calculation of the trajectory from an initial guess that does not guarantee to be the correct solution. The outer loop is entered when this trajectory is calculated successfully, of which the first step is to calculate a control solution that is interpolated using the Hermite interpolation method. This control solution is a collection of α, σ, and δ_{sb} points at the designated time steps as calculated with the time relationship between the inner and outer frequency. This control solution is applied to the dynamics of the model within the inner loop. The next step is to determine whether the exit conditions have been met. There are two possible outcomes. The first outcome is that the conditions have not been met and this means that a new optimal trajectory is calculated with the current state as the initial point and by using the old trajectory as the initial guess. The usage of the former trajectory greatly reduces the computation time of the new one. The loop is then continued to the control solution. The second outcome is that the final point has been met and that the spacecraft is within the flight approach corridor. The simulation is then ended.

The chapter starts with the problem definition of the reference trajectory in Section 8.1, followed by what variable of the problem is optimised in Section 8.2. It is then possible to create a nominal trajectory, which is shown in Section 8.3. There is no such thing as a nominal trajectory when performing a re-entry in reaility, therefore, this nominal trajectory is subjected to wind in Section 8.5. Finally, the robustness of the guidance system to start with various starting positions is shown in Section 8.6.

8.1. Problem Formulation

This section gives the problem definition that is used in this chapter. It includes the initial conditions, the dynamics of the system, and the constraints.

The goal is to create a trajectory that can be flown with the X-38 as the reference vehicle from the entry point of the terminal area until the spacecraft is aligned in front of the runway within a reasonable distance. This trajectory should be updated online whilst the simulation is in progress.

The equations of motion were already given in Section 3.6.1, but are repeated here for good measure.

$$\dot{x}_E = V \cos\gamma \sin\chi \tag{8.1}$$
$$\dot{y}_E = V \cos\gamma \cos\chi \tag{8.2}$$
$$\dot{z}_E = \dot{h} = V \sin\gamma \tag{8.3}$$
$$\dot{V} = -\frac{D}{m} - g \sin\gamma \tag{8.4}$$
$$\dot{\gamma} = \frac{L \cos\sigma}{mV} - \frac{g}{V} \cos\gamma \tag{8.5}$$
$$\dot{\chi} = \frac{L \sin\sigma}{mV \cos\gamma} \tag{8.6}$$

The initial condition is set at the entry point of the terminal area at a distance of 100 kilometres from the runway.

$$\begin{pmatrix} x_0 \\ y_0 \\ z_0 \\ V_0 \\ \gamma_0 \\ \chi_0 \end{pmatrix} = \begin{pmatrix} 0\,\text{m} \\ -100\,000\,\text{m} \\ 25\,000\,\text{m} \\ 732\,\text{m/s} \\ -10^\circ \\ 0^\circ \end{pmatrix} \tag{8.7}$$

The initial angle of attack is set to 22° which is close to the maximum range angle of attack. The bank angle is set to 0°, it is assumed that the spacecraft is flying straight towards the beginning of the runway.

The final node is set to be limited to be within the Flight Approach Corridor (FAC) as seen in Figure 2.10. The location of the FAC depends on the problem itself, and the following final states are required

$$\begin{pmatrix} x_f \\ y_f \\ z_f \\ V_f \\ \gamma_f \\ \chi_f \end{pmatrix} = \begin{pmatrix} -3000\,\text{m} \pm 100\,\text{m} \\ 0\,\text{m} \pm 20\,\text{m} \\ 500\,\text{m} \pm 20\,\text{m} \\ 100\,\text{m/s} \pm 10\,\text{m/s} \\ \text{n.a.} \\ 90^\circ \pm 2^\circ \end{pmatrix} \tag{8.8}$$

This location is relatively close to the runway, but the heading angle requirement ensures that the spacecraft is aligned and the required flight path angle towards the beginning of the runway is roughly 10 degrees. The states are constrained to be within the following values

$$\begin{pmatrix} -100\,000\,\text{m} \\ -140\,000\,\text{m} \\ 0\,\text{m} \\ 1\text{m/s} \\ -89^\circ \\ -179^\circ \\ -14^\circ \\ -50^\circ \\ -9^\circ \end{pmatrix} \leq \begin{pmatrix} x_E \\ y_E \\ z_E \\ V \\ \gamma \\ \chi \\ \alpha \\ \sigma \\ \delta_{spbr} \end{pmatrix} \leq \begin{pmatrix} 100\,000\,\text{m} \\ 100\,000\,\text{m} \\ 28\,000\,\text{m} \\ 800\,\text{m/s} \\ 89^\circ \\ 179^\circ \\ 40^\circ \\ 50^\circ \\ 9^\circ \end{pmatrix} \tag{8.9}$$

and the derivatives of the controls are limited to stay within

$$\begin{pmatrix} -3^\circ\text{/s} \\ -15^\circ\text{/s} \\ -10^\circ\text{/s} \end{pmatrix} \leq \begin{pmatrix} \dot{\alpha} \\ \dot{\sigma} \\ \dot{\delta}_{spbr} \end{pmatrix} \leq \begin{pmatrix} 3^\circ\text{/s} \\ 15^\circ\text{/s} \\ 10^\circ\text{/s} \end{pmatrix} \tag{8.10}$$

These values are selected based on the maximum rates of the HORUS (De Ridder, 2009). No maximum rates for the X-38 were found.

The initial conditions and constraints of the reference trajectory presented here shall be used in the subsequent sections.

8.2. Cost-Function Analysis

Defining the correct cost function is essential when defining the problem that is to be solved. It can determine whether it is cheaper to have a certain angle of attack or bank angle. One might prove to be minimal when an S-turn manoeuvre is performed, or when the angle of attack remains large, when the system has to dissipate energy for example.

Different cost functions were considered that could improve the trajectory from the entry point towards the transition of the terminal area and the approach and landing phase. The following were considered

$$J = (\gamma - \gamma_f)^2 \tag{8.11}$$

$$J = \int \dot{\alpha} dt \tag{8.12}$$

$$J = c_1(\gamma - \gamma_f)^2 + c_2 \int \dot{\alpha} dt \tag{8.13}$$

$$J = \sqrt{(x - x_f)^2 + (y - y_f)^2 + (z - z_f)^2} \tag{8.14}$$

There is a number of reasons why these were considered. Having a selected final flight-path angle (γ_f) assures that the bank angle is close to zero, the flight-path angle decreases during a turn unless it is countered with a high angle of attack, and that the velocity is high enough to sustain the required lift. The value of γ_F is set to $-9°$ as in line with the angle towards the beginning of the runway. The derivative of the angle of attack is used to smoothen out the trajectory to reduce the change in flight-path angle and thus get a more uniform load factor. The values of c_1 and c_2 are introduced as weight factors. The value of $\dot{\alpha}$ is lower than γ, this difference should be compensated in the cost function, to prevent that one value overshadows the other. The purpose of the last cost function is aim the final point of the trajectory to be as much in the middle of the flight-approach corridor as possible. Note that some of the final states are already limited by Equation 8.8.

An attempt has been made to simulate the entire closed loop trajectory with all the presented cost functions. Function 8.14 did not create a feasible trajectory and was therefore discarded, the consequence of this is limited, since the final point is already constrained to end within the box. The other three did generate results as seen in Figure 8.1 where three important states are shown. The first component of the remaining three, γ_f, provides good results. A relatively minor S-turn is flown to reduce the energy before the final point is reached well within the required bounds. The trajectory calculated with solely the second component, $\dot{\alpha}$, does not have this S-turn but it flies directly towards the final turn with a larger variation of the angle of attack during the transonic phase. A change in σ requires a correction of α, σ stays therefore close to zero. It can already be seen that there are some irregularities with this trajectory. There is an oscillation of α present that does not occur with γ_f. The cause of this is that the first few nodes jump around and are thus not stable, creating a seesaw like movement as seen in the graph. The hypothesis is that the solution is on a singular arc which limits the convergence towards a proper solution (Garg, 2011). This means that there is not one correct solution, but multiple. The creation of the initial trajectory also takes five times as long when compared to γ_f, five seconds in total. Attempts were made to combine both the components to get the best of both worlds and avoid the possibility of a hypothetical singular arc. The final result is close to the solution of just $\dot{\alpha}$, with sometimes a minor oscillation during the later part of the flight. However, the trajectories found were unstable since no solution could sometimes be found during the flight, depending on the constants chosen. This example has $c_1 = 1$ and $c_2 = 0.01$, but alternating c_2 with values with a difference of 10% already caused problems. This combination is therefore considered unstable, and it is dropped.

The chosen cost function for all the trajectory calculations is thus the one containing just γ_F.

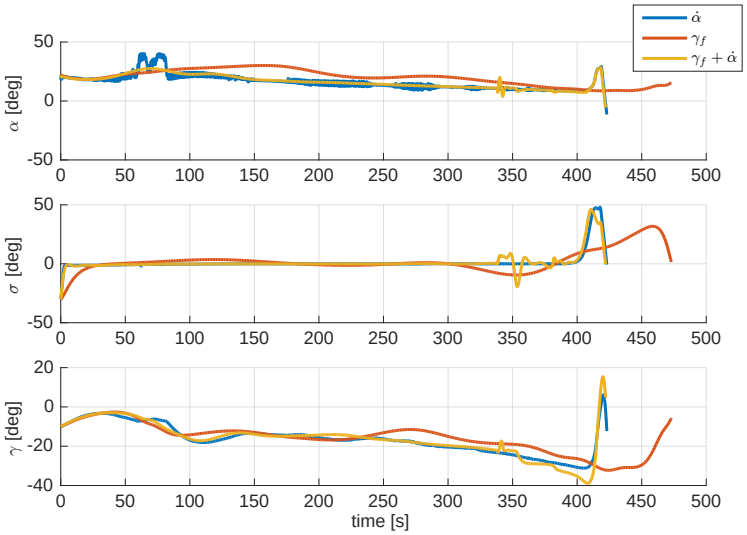


Figure 8.1: Three different cost functions compared

8.3. Nominal Trajectory Analysis

This section gives an analysis of the reference trajectory followed using a closed-loop without any disturbances. The initial and final point, and boundaries are given in Section 8.1. The corresponding cost function of the problem is given in Section 8.2.

There are two ways in which the accuracy of the solution can be altered. The first one is by changing the amount of collocation points used, where the higher the amount of nodes mean a more accurate solution, at the cost of computation time (Garg, 2011). The second method is changing the frequency of the outer loop. Each cycle requires that a new optimal trajectory is calculated.

The analysis to determine the number of nodes is shown first. A broad amount was attempted starting with a set of 20 nodes. Viable solutions were only found from 50 nodes and onwards. Solutions before this were not found from the cold start or it failed halfway in the simulation with a outer frequency of 1 Hz. Figure 8.2 shows three sets of controls flown with 50, 75, and 100 nodes. The first two are relatively close and they show a smooth result. The 100-nodes trajectory is already becoming unstable, yet it arrives at the destination. This instability is even more present if the number of nodes is further increased.This result is a stark constrast of what is to be expected in sutuations like this. Examples such as the Space Shuttle problem showed that an increase in nodes clearly did improve the accuracy. The conclusion is therefore that the model used is the limiting factor. Note that this problem already occurs during the early trajectory generation phase, it is therefore not a problem caused by the integrator. The cause of this is most likely the aerodynamic database or use thereof.

The amount of computation time required is also an important factor. One of the research sub questions is to determine whether it is possible to perform the calculations in real-time. Table 8.1 shows the time required to calculate the trajectory from a simple initial guess and the entire simulation time. The entire simulation is approximately 450 seconds, the limitation is therefore to stay within this simulation time. The added benefit of 75 nodes over 50 is not clear, no precision is gained since both trajectories are stable. Having a lower time also increases the flexibility of the system when suddenly new trajectories are required. Henceforth, 50 nodes are therefore used. The lowest row, with 150 nodes, is added to show the sheer impracticality of using this many nodes. The simulations were performed on a 4-year old laptop with an Intel I7-2670QM running at 2.2 GHz. It stands to reason therefore that this time can be improved by using a dedicated board computer. About 90% of the time is spent within SNOPT, which only works with one core. The sequential nature of the simulation does not permit the usage of multiple cores manually.

nodes	cold start time [s]	simulation time [s]
50	1.06	122.01
75	2.32	323.32
100	17.81	861.92
150	90.75	3147.21

Table 8.1: The time it took to complete the simulation.

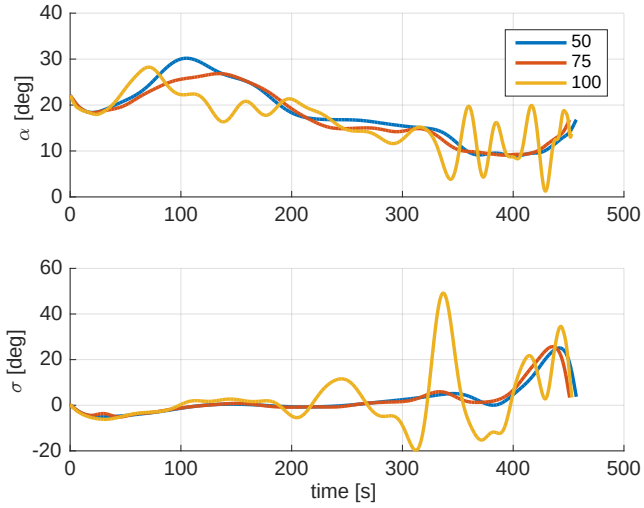


Figure 8.2: α and σ for different amount of nodes during the nominal trajectory.

The second element is the effect of changing the frequency of the outer loop. This analysis is performed on an undisturbed flight, the discussion about how often a new trajectory is required is therefore solely based on the computation time required and . Figure 8.3 is a graph that shows the difference of the angle of attack with different frequencies, from a base frequency of 0.5. The differences between the sets is relatively small with the difference getting higher as the frequency increases. The difference of $\Delta\alpha\%$ is caused by the different positions of the nodes each session and the difference of the interpolation method used to calculate the aerodynamic coefficients (Hermite polynomial interpolation) and the LGR method (Lagrange polynomial interpolation). The largest differences also occur where the spacecraft is banking. A comparison is seen in Figure 8.4 that the difference is very small. The choice is a trade-off between avoiding high spikes, computation times, and required refreshing when external forces are active. A frequency of 1 Hz is chosen because the spikes are manageable, the computation time is available (see Table 8.1) and it is still frequent enough to adjust the trajectory when it is windy.

The number of collocation points is set to 50 and the guidance frequency is 1 Hz. The resulting states and controls are seen in Figure 8.5 and 8.6 respectively. The 3-dimensional version of this trajectory to get a better grasp on the trajectory is in Figure 8.7. Two other variables are also analysed in Figure 8.8, the load factor and the energy height. The first item can be used to analyse the structural stresses on the spacecraft and it can also be described as the apparent gravitational acceleration felt by the passengers aboard. The load factor is calculated with

$$n_z = \frac{L \sin\alpha + D \cos\alpha}{mg} \tag{8.15}$$

For reference, commercial transportation vehicles such as the Boeing 747 should be able to cope with load factors in the range of $-1.52 \leq n_z \leq 3.8$. Figure 8.8 shows that the load factor stays well within these margins at all times. The second item is the presence of the energy in the system, both kinetic and potential. It can be used to see where the largest drops of available energy are and to see if the simulator does not add

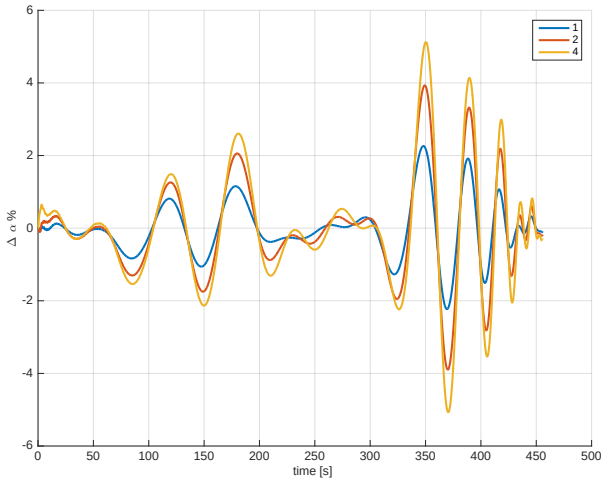


Figure 8.3: The percentual difference of different guidance frequencies.

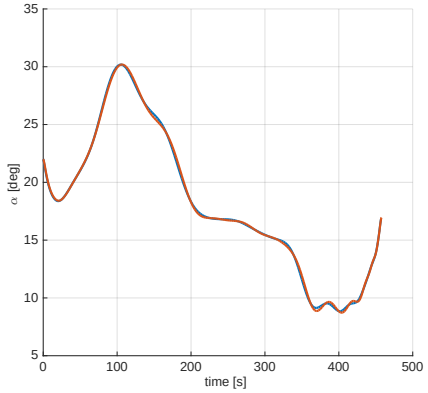


Figure 8.4: Comparison of α with different guidance frequencies.

	Final state
x [m]	-2903.02
y [m]	-19.2
z [m]	487.55
V [m/s]	93.42
γ [°]	-5.0
χ [°]	87.99
α [°]	16.86
σ [°]	2.15

Table 8.2: The final states of the nominal trajectory.

energy to the system.

The spacecraft starts at an altitude of 25 kilometres aimed straight towards the beginning of the runway located at point $(0,0)$. The first phase is used to lose energy by increasing the flight path angle and creating a turn. Then, the transonic phase happens around 60 seconds which seems quite uneventful except for a bump in the load factor, which is only natural because of the behaviour of the aerodynamic coefficients in this region as seen in Figure 7.7. A relatively minor S-turn is flown in the subsonic region to lose even more energy before the alignment in front of the runway is complete after 473 seconds. The final state can be found in Table 8.2. What is interesting is that the speedbrake stays at -9 degrees, or close to, during the entire flight which means that it is fully retracted. The increase in drag at the cost of lift is not worth it, or the optimiser preferably tweaks the controls that have more impact, α and σ. Further analysis with a fully deployed speedbrake should be performed.

The states and controls all have a smooth curve except for the flight path angle, γ goes up and down during the flight without a direct visible correlation with the other states and controls. The three important variables that impact the derivative of γ are the velocity, bank angle, and lift coefficient. The first two are relatively constant compared to the lift coefficient (see the load factor). The increase of the lift coefficient by a factor of 1.75 has a destabilising effect on γ. Early tests that had an increase of a factor of 2 had higher spikes up and down of γ. The factor was brought down to reduce this effect, even though the L/D ratio is now lower than what is expected of a winged re-entry vehicle (Weiland, 2014). A recommendation is that an un-modified aerodynamic database is used of a winged re-entry vehicle to see whether this sinusoidal curve in γ is still there. The final velocity is around 100 m/s, which is still quite high, but the handling in the subsonic phase at lower velocities has to be further researched, since the original design intents to descent the final part with a parafoil. The aerodynamics for that section of the flight has not been researched. The values of the coefficients are stretched out from the higher Mach regions.

It can be concluded that the simulator is able to create valid trajectories when the spacecraft starts at the initial point under nominal conditions.

8.4. Change in Lift and Drag Coefficients

The aerodynamic database that is generated by flight-testing and CFD is not a 100% accurate, that is impossible. Therefore, it is important to see if the guidance system is robust enough to deal when the lift and drag coefficients are changed without knowing. These do not differ that much, but it should be able to handle margins of 5%, which are reasonable (Weiland, 2014).

Figure 8.9 and 8.10 show how the nominal angle of attack and bank angle are affected by the coefficient changes in the vehicle dynamics block. The impact of a decrease in the lift coefficient is close to that of an increase in the drag coefficient, and vice versa, which makes sense considering that one decreases the loss of potential energy, and the other one in kinetic energy. The flight-time is reduced or increased by roughly 3%. The shorter flight-time does not pose any problems for the guidance method, this reduction also means that more energy is available. The angle of attack becomes a bit erratic for the one with the

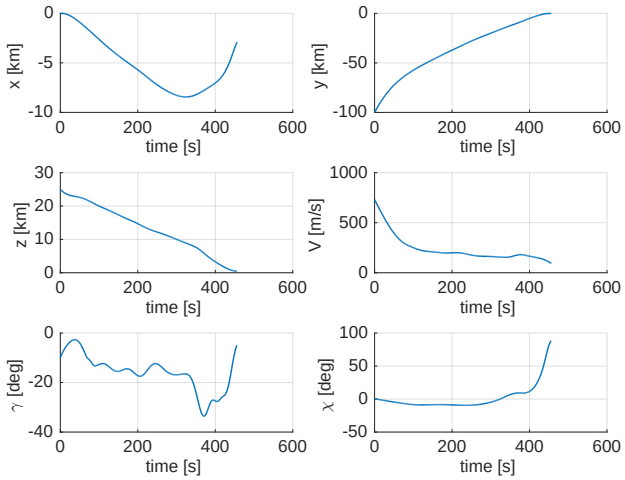


Figure 8.5: The states of the nominal trajectory

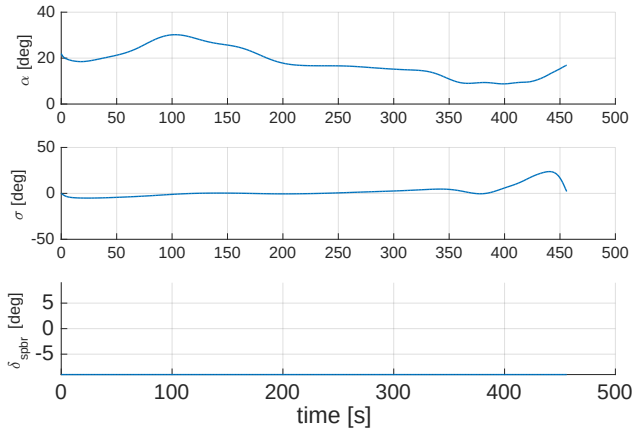


Figure 8.6: The ground-based aerodynamic angles of the reference solution

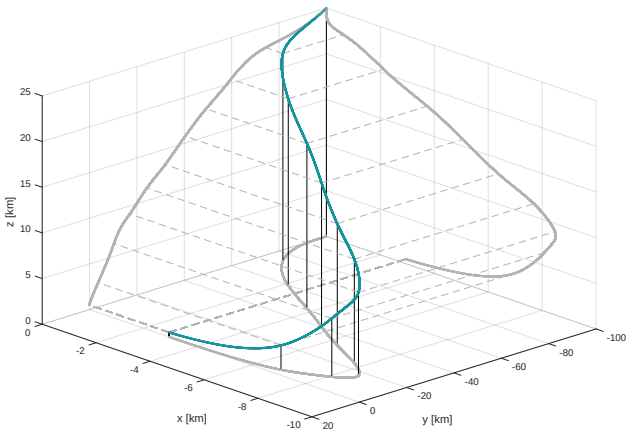


Figure 8.7: The 3-dimensional depiction of the reference trajectory.

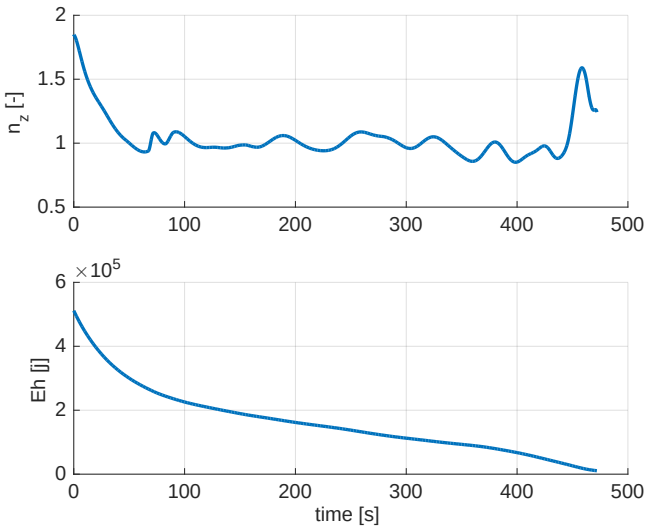


Figure 8.8: The 3-dimensional depiction of the reference trajectory.

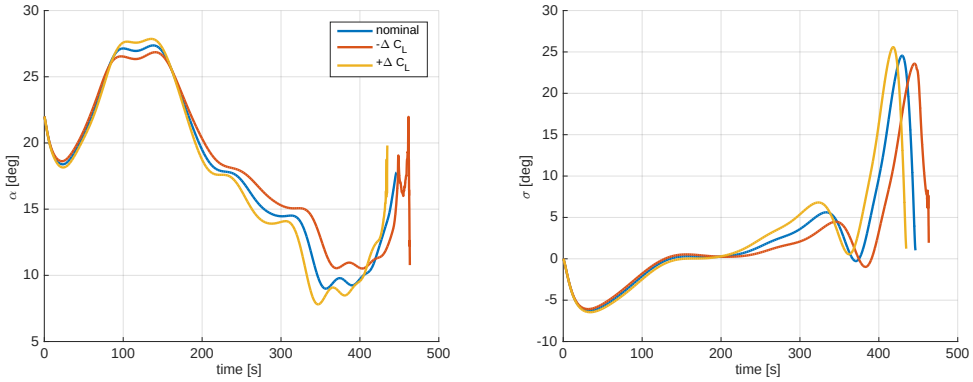


Figure 8.9: The change of α and σ when a different lift coefficient is used in the vehicle dynamics section.

longer flight-duration. The guidance system needs to re-adjust to the new state which has less energy than expected, which results in a bumpy α, and to a lesser extent, σ. The response to these changes are very similar to similar situations in Section 8.5. An example is given when C_L is increased by 5% and C_D decreased by 5% by magnifying the angle of attack at the last few seconds of the trajectory in Figure 8.4. The derivative is at times discontinuous, which is not an issue when it is small, but there are pieces that do stand out. A recommendation is that the guidance commands are smoothed out before it is passed on to the control subsystem.

8.5. Introducing Wind

This section introduces the effect that external disturbances have on the trajectory. The concept of wind was introduced in Section 3.6.3 and this theoretical wind is added to the trajectory to test the robustness of the guidance subsystem. First, the average wind is added to the environment to test standard wind conditions, and second, gust loads are imposed on the spacecraft during small portions of the flight to test the response.

The data from Table 3.2 used for nominal mission examples taken from (Johnson, 2008) is calculated by interpolating different measurements throughout the year at Kennedy Space Centre. The availability of this data is the reason that the X-38 lands at this airport. The runway that was used for various Shuttle landings

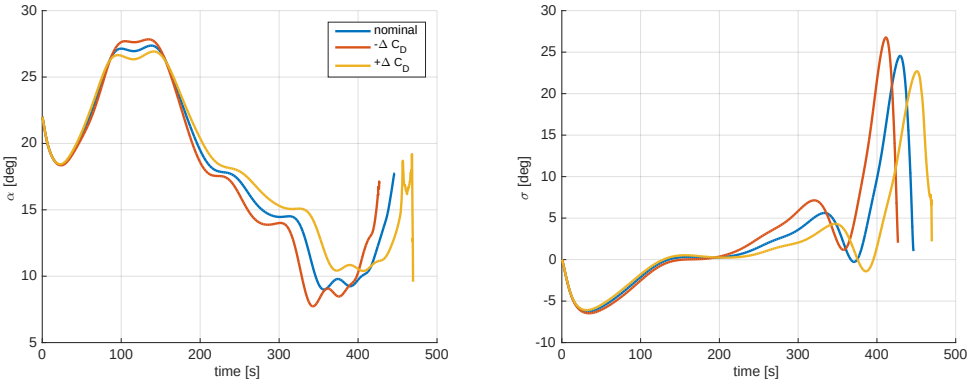


Figure 8.10: The change of α and σ when a different drag coefficient is used in the vehicle dynamics section.

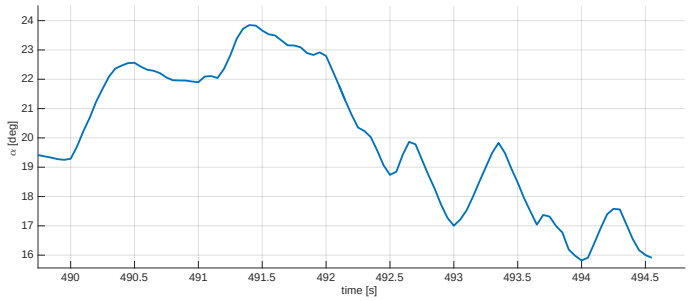


Figure 8.11: The angle of attack during the last few seconds of the trajectory with $+\Delta C_D$ and $-\Delta C_L$.

is designated with the number 15 or 33, which means that the approach vector requires a heading angle of either 150 or 330 degrees, the first one is to be used as the required heading angle. The wind data is given in the wind reference frame, this should first be transformed to the runway reference frame with $\mathbf{C}_{RW,W}$. The effect of this wind is not known by the guidance system, it calculates a new trajectory every step along the way unknown of the changes in the outside world. It is assumed that the sideslip angle is either zero, or close to zero. The control subsystem should be able to mitigate this. Furthermore, the test performed here is a first-order analysis of the guidance capabilities.

Four simulations are run with the wind coming from different direction every time, and these are compared to the nominal situation from Section 8.3. The first one is run without a rotation of the wind in Table 3.2, the other three are rotated with $90°$, $180°$ and $270°$. The wind does usually not rotate with the $90°$ angles presented here but it has more or less the same wind angle at the higher altitudes (Johnson, 2008), this situation is therefore unlikely, but is interesting to see how the guidance system responds to these unknowns.

Figure 8.12 gives a top-down look of the different trajectories flown. The original trajectory can be seen in the middle finding its way to the runway unhindered by the wind. The greatest shift comes from the meridian winds at altitudes 5 kilometres and up where the winds move an average of 20 m/s. All the trajectories differ greatly from the nominal one, the $90°$ rotation cannot even reach the runway. The simulation is stopped because the guidance system cannot create a new feasible trajectory to the runway since too much energy is dissipated because of the extra kilometres flown in the negative y-direction. The error margin for lower energy entries is small, and deviations from the nominal trajectory should therefore be avoided as much as possible. It is more interesting to see how the spacecraft handles during the flight than where it is actually flying. Figure 8.13 shows the various load factors. The nominal situation did not require the constraint that load factor is limited from -1 to 2.5 g's, the peaks demonstrate that this is a requirement for off-nominal conditions. Another noticeable change is that some of the trajectories have an oscillating load factor, which is both structurally and for the passengers unacceptable. These peaks are created by the last-minute attempts of PSM to reach the desired final state. It is clear that a guidance

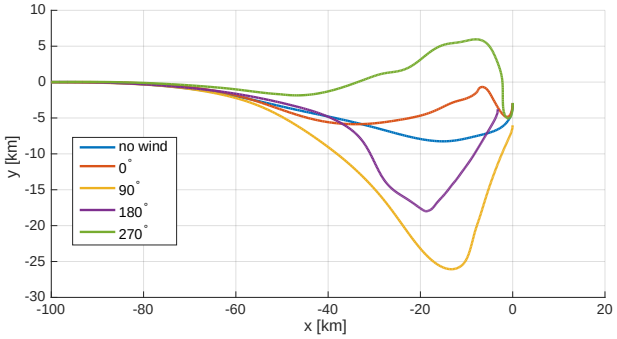


Figure 8.12: Trajectories flown under the influence of wind from different directions.

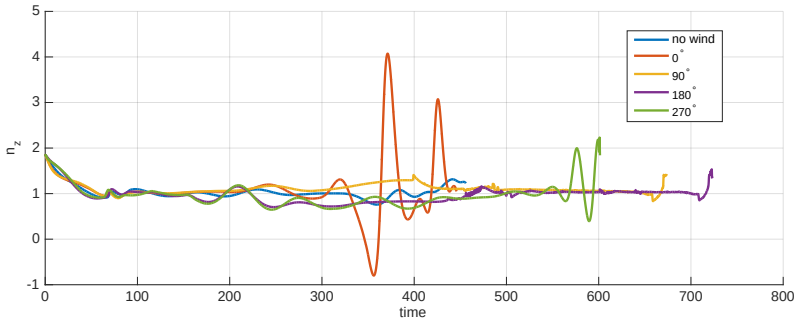


Figure 8.13: The load factor when flying under the influence of wind from different directions.

system that does not have the possibility to predict the upcoming wind is likely to have a performance that is less than adequate.
It is assumed for now that the guidance system is aware of the environment before the terminal area is entered. This information can be taken from weather forecasts and observations. Figure 8.14 shows what trajectory is created when it can be planned in advance what the conditions are going to be. The load factor is also added as a constraint, but it does not seem to be required in this phase since the curve is almost identical to Figure 8.8, although there are minor differences in the trajectory, but these are not worth any further inspection. The new trajectory brings the spacecraft further in the negative y-direction, yet, the duration is 425 seconds, instead of 449 seconds. The conclusion is that it is possible to include external disturbances with PSM with success.

Unfortunately, it is not always possible to know the exact magnitude and heading of the wind. It is therefore required that a number of trajectories is created that have perturbed conditions away from the average. The contents from Table 3.2 is modified with a probability generator. The probability curve of the magnitude has a 1σ of 10% of the original value, and the heading and flight-path angle have a 1σ of 10 degrees. Each altitude is changed separately with different values. The new wind is calculated with

$$\tilde{W}_h = C_{V,W}(\sigma_\gamma, \sigma_\chi) W_h(\sigma_F) \tag{8.16}$$

The transformation matrix is used that also transforms the wind vector to the vertical frame, the rotations required are equal.

Figure 8.15 shows α, σ, and n_z after 150 simulations. All of the simulations successfully made it to the final flight approach corridor.The control states are close to the original conditions until halfway, where the deviations gradually grow bigger. The solution of handling the different wind conditions seems to be to increase α, which in turns also increases the lift coefficient, increasing the amount of ground that can be covered to get back to the right trajectory. The increase in drag seems is relatively smaller. The bank angle has in interesting curve in the end, where all of the new trajectories follow the same pattern of a lower σ. This means in an absolute sense that it is smaller, but positive, and that, as a result, the required turn has a

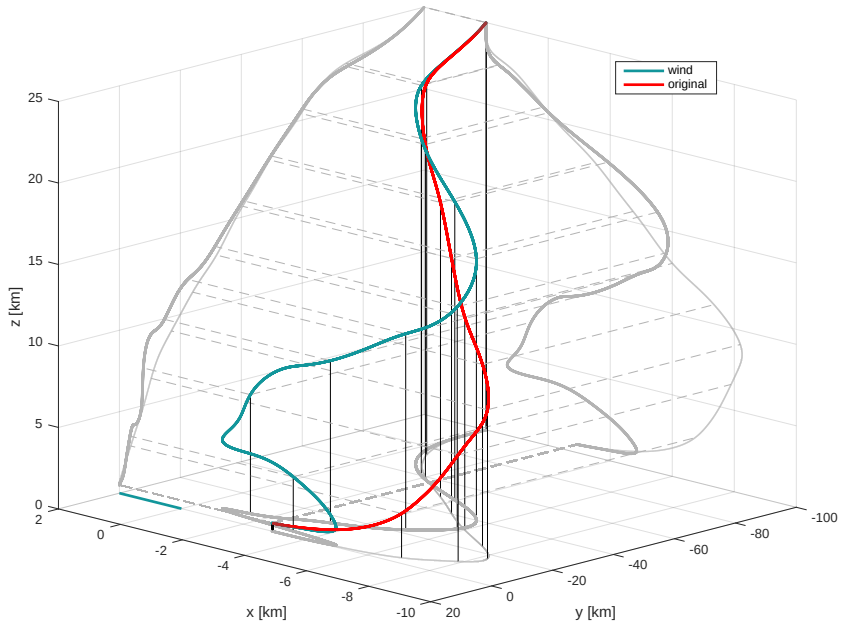


Figure 8.14: Trajectories flown under the influence of wind from different directions.

smaller inner radius than before. The load factor strays away from zero as well over the course of the trajectory. It first becomes slightly positive, which means that the small deviations away from the expected trajectory increases the overall gravitational acceleration felt by the passengers, followed by a decrease during the final reduced turn. The final seconds of almost all the trajectories show the same final spikes as in Figure 8.13. The guidance system tries to reach the final state, and even though it is within the constraints given, it is far from the desired effect. This occurrence seems to be inherent to the system, as the FAC is a requirement. The effect could be reduced by widening the criteria that form this box, making it such should reduce the effect of what is seen here. Another option would be to start the approach & landing phase right before this happens, smoothing the transition.

A wind gust is added as a final demonstration, simulated by the NASA 1997 Discrete Gust Model (Section 3.4.3), and Figure 8.16 shows it. The gust will take place at an average height of three kilometres, with a width of 500 metres. The flight path angle is set to $10°$, and the heading angle measured in the vertical frame is $25°$. The gust velocity is set to 40 m/s, which is quite extreme. For reference, this can happen within a category 3 hurricane (Johnson, 2008). No spacecraft would ever attempt a landing under such conditions, but it is shown here as a demonstration. The numbers are chosen arbitrarily, just for this example.

The gust hits the vehicle close after the 400 second mark, shown in Figure 8.17, where the gust hits the vehicle almost head on. The result is that the trajectory in the x-y direction is nearly identical, but the height is maintained near the peak of the gust. The speed drops rapidly, and there is an attempt to increase it again by decreasing the angle of attack. There is a quick descent after 60 seconds where γ reaches close to -50 degrees, this is done to regain the lost velocity to reach the target, which it does successfully after roughly 30 seconds more than the original. Figure 8.18 shows a very interesting load factor. There is near weightlessness for the first 40 seconds, followed by twice the normal gravitational load. This movement has all the looks of a parabolic flight. The guidance system is unaware of the existence of the peak, a smart one should quickly fly through this and return to the original state. These are all realistic responses when encountering this type of atmospheric disturbances, even though the magnitude of the gust is a bit exaggerated.

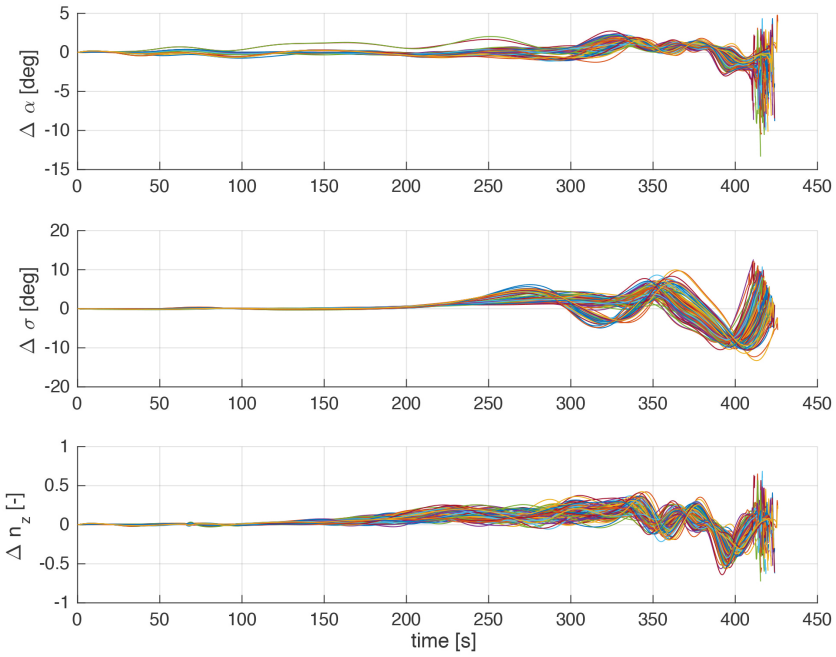


Figure 8.15: The differences in states for 150 trajectories under different wind-vectors.

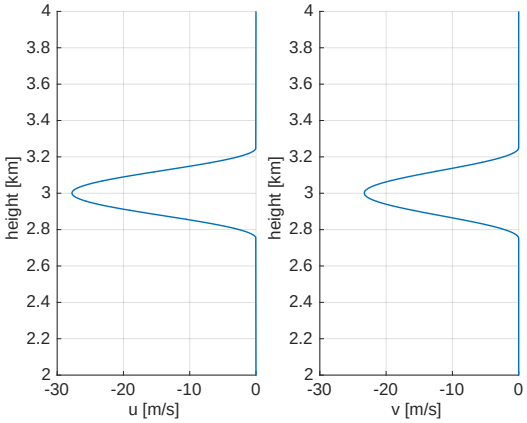


Figure 8.16: The simulated gust.

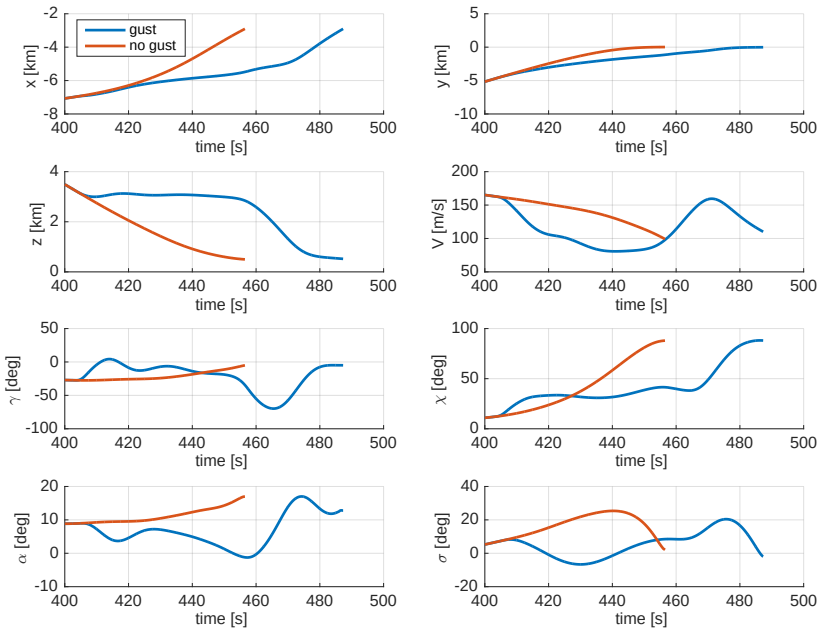


Figure 8.17: The different states during an extreme gust.

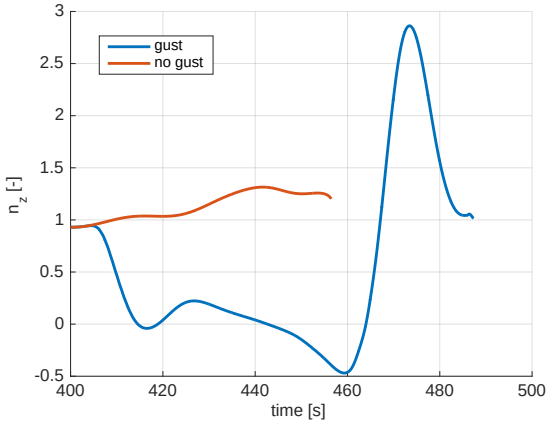


Figure 8.18: The load factor during the encounter with the extreme gust.

		χ_{RW}
$y > 0$	$x > 0$	-$\tan^{-1}(\frac{y}{x}) - 90°$
	$x < 0$	$\tan^{-1}(\frac{-y}{x}) - 90°$
	$x = 0$	$180°$
$y < 0$	$x > 0$	$\tan^{-1}(\frac{-y}{x}) - 90°$
	$x < 0$	$\tan^{-1}(\frac{-y}{x}) + 90°$
	$x = 0$	$0°$
$y = 0$	$x > 0$	$-90°$
	$x < 0$	$90°$
	$x = 0$	n.a.

Table 8.3: Ways to calculate the initial heading angle towards the runway to avoid the quadrant ambiguities.

8.6. Terminal Area Entry Footprint

This chapter has so far discussed different trajectories that all have the same initial conditions. It is however, not certain where the start of the terminal area will take place. Therefore, it is important to also assess how the guidance system is able to handle different starting positions, even when the runway approach is on the other side of the current shortest route. This section considers trajectories with nominal wind.

A range of x and y positions is given in the left of Figure 8.19, it stretches 150 kilometres in each direction, with a new attempt every 10 kilometres. Each dot is an element from which a trajectory was trying to be found. The initial heading angle is such that it is aimed at the beginning of the runway. It is not trivial to calculate the initial heading angle because of the quadrant ambiguities (De Ridder, 2009). A way to calculate the heading angle from each starting position is in Table 8.3. The other states are equal to these presented in Section 8.1.

The right side of Figure 8.19 shows where it is possible by the optimiser to find a suitable trajectory, It is possible for most of the positions close to the runway, and a circle can be imagined that contains the successful ones. It seems that the positions outside of this circle require more energy to reach the final point than is available. It is not possible to reach the runway from all points within the imaginary circle, which is wrong since there is enough energy present to reach the runway, as proven by the successful attempts around these points. The lower-left corner has more difficulties than the lower-right corner, which is strange since the results should be mirrored. One of these two locations are investigated in more detail. These are $(-50,-50)$ and $(-50,50)$. An investigation is performed to see whether different heading angles might make a difference. Each point starts with 12 different initial headings ($-180°$ mod $30°$), Figure 8.21 shows the successful trajectories. Both locations have possible trajectories when pointing roughly towards the runway. However, there are two issues. The first one is that the results are once again, not mirrored. The change in angle is due to the wind, so that is to be expected. Second, the left location fails at an heading angle of $45°$, which does not show up here. This means that the trajectory generator is inconsistent with providing correct results.
The time to find the different solutions is given in Figure 8.20. The cold start time differs greatly from under 1 second to up to 30 seconds. A large swath of the feasible trajectories are calculated within a few seconds, which is reasonable. However, a 20-seconds timespan to calculate a feasible trajectory is too long. The longer calculations are found in the lower part of the region, the points in the upper part are all around 1 second.

It is beneficial to expand this research by providing viable trajectories to compare the results, since the current implementation lacks robustness, the percentage of successes is below 100%. The guidance system is therefore not flight-qualified at this point.

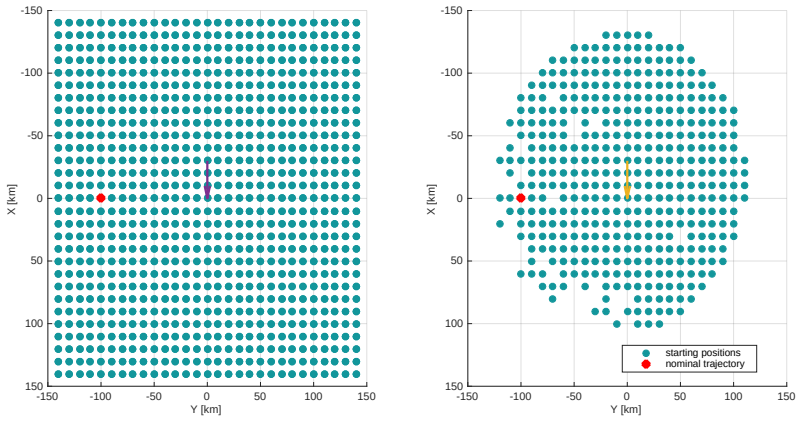


Figure 8.19: The left figure shows a grid that has possible starting locations, the right figure shows from which of these points feasible trajectories can be found.

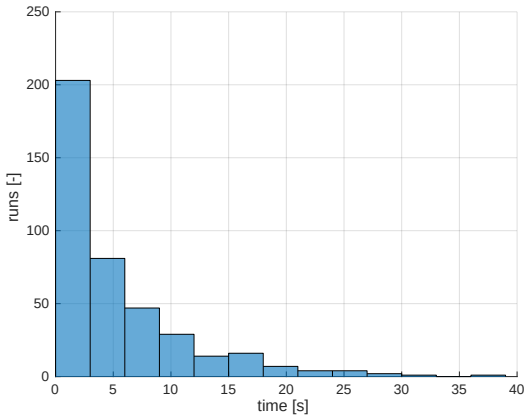


Figure 8.20: The time it takes to find a successful solution from the starting grid in Figure 8.19.

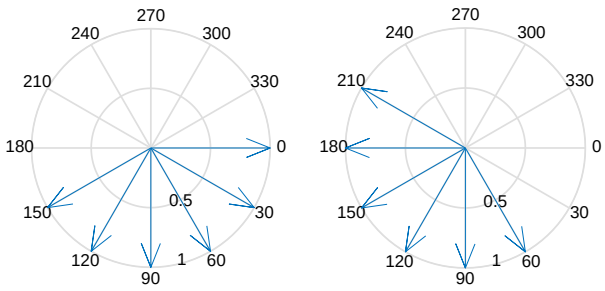


Figure 8.21: Viable headings when starting at $(-50,-50)$ and $(-50,50)$ respectively.

9

Attitude Control

This chapter will demonstrate if it is feasible to apply PSM not just for trajectory generation, but also attitude control, creating a full 6-DOF model. Section 9.1 gives the first demonstration, it is shown here how an LQR controller can handle the non-linear dynamics in the terminal area. Section 9.2 finally show what the results are when PSM is utilised as a control subsystem.

9.1. Linear Quadratic Regulator

It is important to stress that it is hard to find the exact cause of an error when calculating solutions with pseudospectral methods. The usual error is that no solution can be found, without giving any specifics. Therefore, it is important to start with knowing that the model and software is correctly set up, this is done by using a simple linear quadratic regulator (LQR). The theory behind this method was discussed in Section 5.1.

The first step is to calculate the gains such that the deflection of the elevator, aileron, and rudder can be calculated. The matrices **A** and **B** are defined by the linearised motion of the spacecraft which is first shown in Section 3.7, and it shown as a state-space matrix in Appendix A. The **Q** and **R** matrices are defined as per Bryson's rule and it signifies the maximum allowable deviation of the state and control vector respectively (Mooij, 1998). The **Q**-matrix is defined as

$$\mathbf{Q} = \text{diag}\left\{\frac{1}{\Delta p_{max}^2}, \frac{1}{\Delta q_{max}^2}, \frac{1}{\Delta r_{max}^2}, \frac{1}{\Delta \alpha_{max}^2}, \frac{1}{\Delta \beta_{max}^2}, \frac{1}{\Delta \sigma_{max}^2}\right\} \tag{9.1}$$

with $\Delta p_{max} = 10^\circ/s$, $\Delta q_{max} = 10^\circ$/s, $\Delta r_{max} = 10^\circ/s$, $\Delta \alpha_{max} = 4^\circ$, $\Delta \beta_{max} = 4^\circ$, and $\Delta \sigma_{max} = 4^\circ$. The **R**-matrix as

$$\mathbf{R} = \text{diag}\left\{\frac{1}{\Delta \delta_{e_{max}}^{\ 2}}, \frac{1}{\Delta \delta_{a_{max}}^{\ 2}}, \frac{1}{\Delta \delta_{r_{max}}^{\ 2}}\right\} \tag{9.2}$$

with $\Delta \delta_{e_{max}} = \Delta \delta_{a_{max}} = \Delta \delta_{r_{max}} = 30^\circ$. The values were chosen after several attempts to find the optimal gains.

The gains **K** are calculated in Matlab using the states of the reference trajectory from Section 8.3. No easily implementable packages were found for C++. The schur method was used to solve the Ricatti equation (Bunse-Gerstner, 1996). However, Eigen gave different results when inverting matrices larger than three, therefore, Matlab was used to calculate the gains of the nominal trajectory. THe different gains are created for each second in the flight, the last available gains are used for prolonged flights. The gains are shown in Figure 9.1 and 9.2. The gains are relatively constant, except for the transonic phase in some cases.

The gains are used to calculate the deflections of the aerodynamic surfaces in the following fashion

$$\Delta \mathbf{u} = -\mathbf{K}\Delta \mathbf{x} \tag{9.3}$$

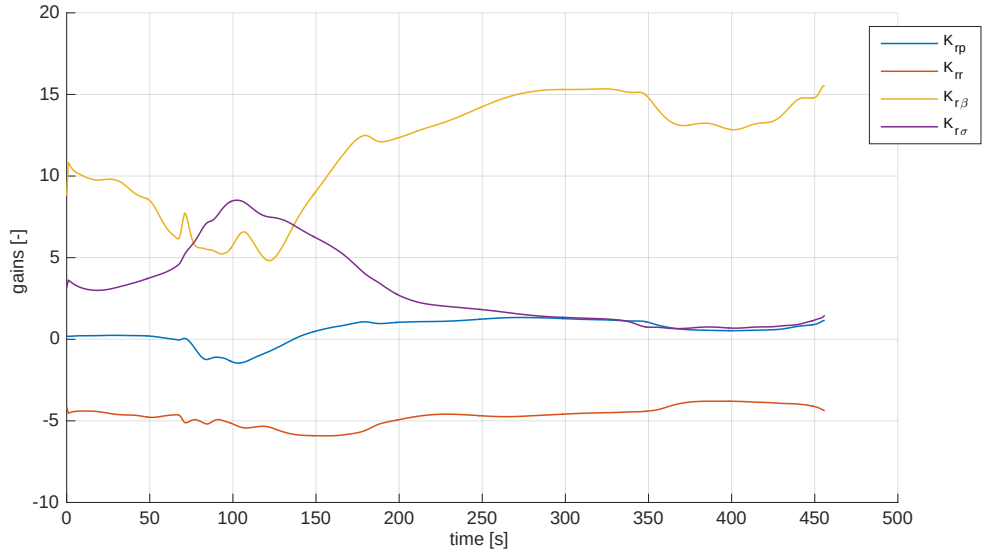


Figure 9.1: The longitudinal gains calculated for the reference trajectory.

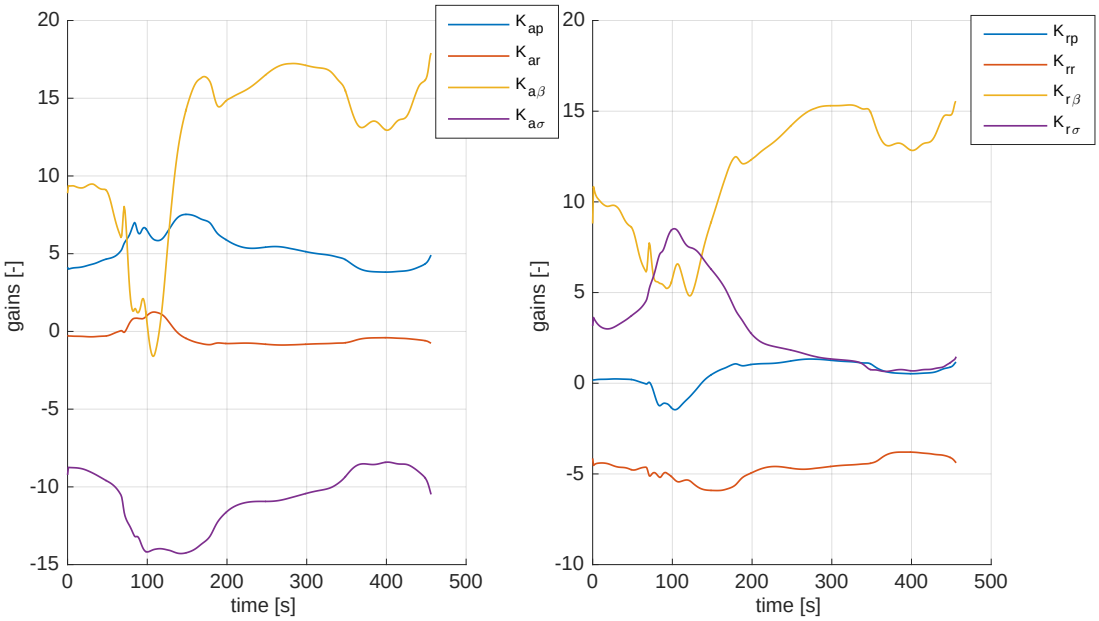


Figure 9.2: The lateral gains calculated for the reference trajectory.

where $\Delta\mathbf{u}$ is

$$\Delta\mathbf{u} = \begin{bmatrix} \Delta\delta_e \\ \Delta\delta_a \\ \Delta\delta_r \end{bmatrix} = \begin{bmatrix} \delta_e \\ \delta_a \\ \delta_r \end{bmatrix} - \begin{bmatrix} \delta_{e,trim} \\ \delta_{a,trim} \\ \delta_{r,trim} \end{bmatrix} \tag{9.4}$$

$\Delta\mathbf{x}$ is

$$\Delta\mathbf{x} = \begin{bmatrix} \Delta\alpha \\ \Delta\beta \\ \Delta\sigma \\ \Delta p \\ \Delta q \\ \Delta r \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \\ \sigma \\ p \\ q \\ r \end{bmatrix} - \begin{bmatrix} \alpha_c \\ 0 \\ \sigma_c \\ p_0 \\ q_0 \\ r_0 \end{bmatrix} \tag{9.5}$$

The commanded angle of attack and bank angle are given by the guidance system. The nominal rates are calculated with Equations 3.66-3.68. The new control deflections are passed by a bandwidth and derivative filter before it is used. The bandwidth filter makes sure that the deflections are within the operational limits, and the derivative filter guarantees that the rate stays at $30°/s$. The deflection is limited to The initial state is equal to the nominal rates, the aerodynamic angles as used in Chapter 8, and the trimmed deflections.

To calculate the offset of the deflection angles it is important to know what the trimmed deflection angles are. This is done by finding a stable solution such that all the moments generated cancel each other out. Relevant aerodynamic properties for attitude determination are explained in this context. The simplified rotational dynamics were already given in Section 3.6.2, these were

$$\dot{p} = \frac{M_x}{I_{xx}} + \frac{(I_{yy} - I_{zz})}{I_{xx}} qr \tag{9.6}$$

$$\dot{q} = \frac{M_y}{I_{yy}} + \frac{(I_{zz} - I_{xx})}{I_{yy}} pr \tag{9.7}$$

$$\dot{r} = \frac{M_z}{I_{zz}} + \frac{(I_{xx} - I_{yy})}{I_{zz}} pq \tag{9.8}$$

The moments **M** were not, these are as follows

$$M_x = M_{x,cm} + \frac{1}{2}\rho V^2 C_l S_r e f L_{ref} \tag{9.9}$$

$$M_y = M_{y,cm} + \frac{1}{2}\rho V^2 C_m S_r e f L_{ref} \tag{9.10}$$

$$M_z = M_{z,cm} + \frac{1}{2}\rho V^2 C_n S_r e f L_{ref} \tag{9.11}$$

The first part is the moment around the centre of mass generated by the discrepancy of a different centre of pressure. In general, this is mainly an issue in the XZ-plane, generating a $M_{y_{cm}}$ which can easily be countered with the elevator. However, there is always a small difference in the y-axis, thus generating both a rolling and yawing momen, which needs to be countered with the rudder and aileron. This moment is calculated with

$$\mathbf{M}_{cm} = \begin{pmatrix} x_{ref} \\ y_{ref} \\ z_{ref} \end{pmatrix} \times \begin{pmatrix} -X \\ -Y \\ -Z \end{pmatrix} \tag{9.12}$$

The vehicle needs to be trimmed at all times to stay close to let the derivatives of p, q, and r stay close to zero. A simple Newton-Rhapson method is used to achieve this, which is both quick and accurate, as long as the initial guess is close to the real answer (Ben-Israel, 1966). The basic form looks like

$$\mathbf{X_{n+1}} = \mathbf{X_n} - \mathbf{J}(\mathbf{X_n})^{-1}\mathbf{f}(\mathbf{x}_n) \tag{9.13}$$

The goal is of course to minimise the following close to zero

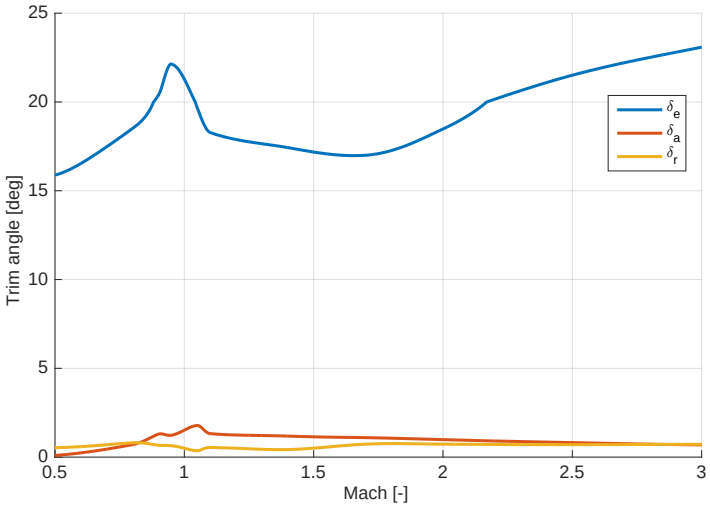


Figure 9.3: The trimmed deflections of the aerodynamic surfaces for $\alpha = 20°$.

$$\mathbf{f} = \begin{pmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{pmatrix} \tag{9.14}$$

This is done by altering the state vector **X**, which is the deflection of the control surfaces, thus

$$\mathbf{X} = \begin{pmatrix} \delta_a \\ \delta_e \\ \delta_r \end{pmatrix} \tag{9.15}$$

which results in the following Jacobian,

$$\mathbf{J} = \begin{bmatrix} \frac{\partial \dot{p}}{\partial \delta_a} & \frac{\partial \dot{p}}{\partial \delta_e} & \frac{\partial \dot{p}}{\partial \delta_r} \\ 0 & \frac{\partial \dot{q}}{\partial \delta_e} & 0 \\ \frac{\partial \dot{r}}{\partial \delta_a} & \frac{\partial \dot{r}}{\partial \delta_e} & \frac{\partial \dot{r}}{\partial \delta_r} \end{bmatrix} \tag{9.16}$$

The derivatives are easily calculated by taking of the derivative of the aerodynamic coefficient that is applicable, multiplied by the constants of the second component in Equations 9.9-9.11. The first attempt takes about five iterations until an answer where $\mathbf{f} \leq 10^{-10}$. The result is displayed in Figure 9.3 where the three control surfaces are displayed for when the spacecraft is in near-perfect trim for Mach 0.5 till Mach 3 for an angle of attack of 20 degrees. There is a sharp rise of the elevator in the transonic region, followed by a quick decrease and slow increase in the higher Mach regions. The aileron and rudder deflections are non-zero for non-banking flights because of the difference between the centre of gravity and centre of pressure in the y-axis of the body reference frame. Yet it is still relatively small, with a bump around the transonic region.

Two different situations are simulated. The first situation is an open-loop situation in which the reference trajectory from Section 8.3 is followed. The second situation simulates a closed-loop situation in which a new reference trajectory is calculated every second. The final state is given by Table 9.1. The closed-loop solution comes close to the designed final state. The open loop however, ends half a kilometer to the right of the runway. How the trajectories are different is given in Figure 9.4, 9.5, and 9.6. The total simulation time is around 211 s, well within real-time capabilities.

One of the first things that is noticed is that both solutions is poorly trimmed in the subsonic region. The difficulty with the current gain selection is that there is a narrow band of possible standard deviations in

	Open loop	Closed loop
x [m]	-626.25	-2854.02
y [m]	-583.57	4.09
z [m]	311.15	491.76
V [m/s]	91	96.88
γ [°]	-6.26	-6.14
χ [°]	100.2	91.7
α [°]	9.05	9.04
σ [°]	0.54	-14.9

Table 9.1: The final states of an open- and closed-loop situation with LQR.

the transonic region. The simulation would crash each time higher margins were implemented due to out-of-control angular rates. An advice is that there are different standard deviations per phase of the flight. This could alleviate this problem.

There is the same problem that was visible during the creation of the reference trajectories. Several states and all the controls spike at the final moment of the trajectory. This is once again due to the fact that PSM tries to reach the final target by seeking the limits of what the system can handle. All these spikes are within the allowable tolerances given, the maximum rate of deflection. This problem can be avoided by connecting the guidance of the terminal area with the approach and landing phase. The attempt to reach the centre of the FAC is then limited.

The guidance system does not seem to appreciate the responsiveness of the control system in general, and a spike of the controls are present at each new interval, trying to get the spacecraft back on course. The calculation of the new trajectory starts with the new state, with the base of the old trajectory as a former solution. The result is that the new trajectory tries to follow the old trajectory as soon as possible, instead of smoothing the differences over the remaining flight-time. This causes the bumps to appear, which are especially noticeable by the thick sections at the control states. The erratic movement of the control surfaces seem to have a minor impact on the angular velocities, and even smaller on the angle of attack and bank angle. The closed-loop simulation ends very near to the final point dictated by the guidance system. Therefore, even though the target is reached, it can be concluded that this method does not work due to the induced oscillations.

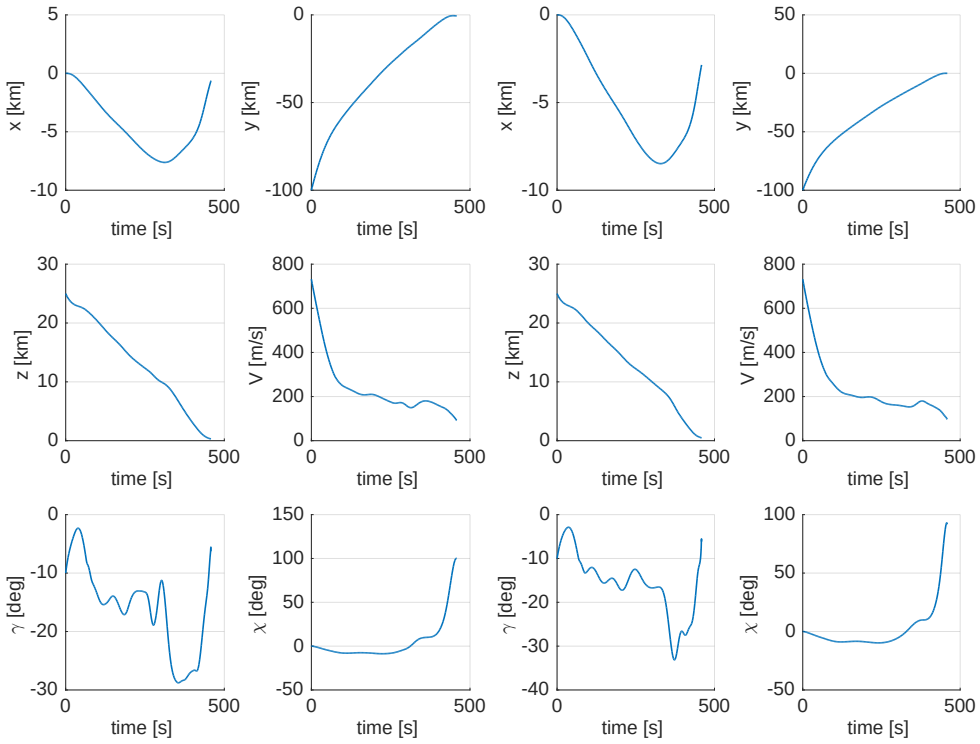


Figure 9.4: The translational states when using an LQR-controller during an open-loop simulation on the left, and closed loop on the right.

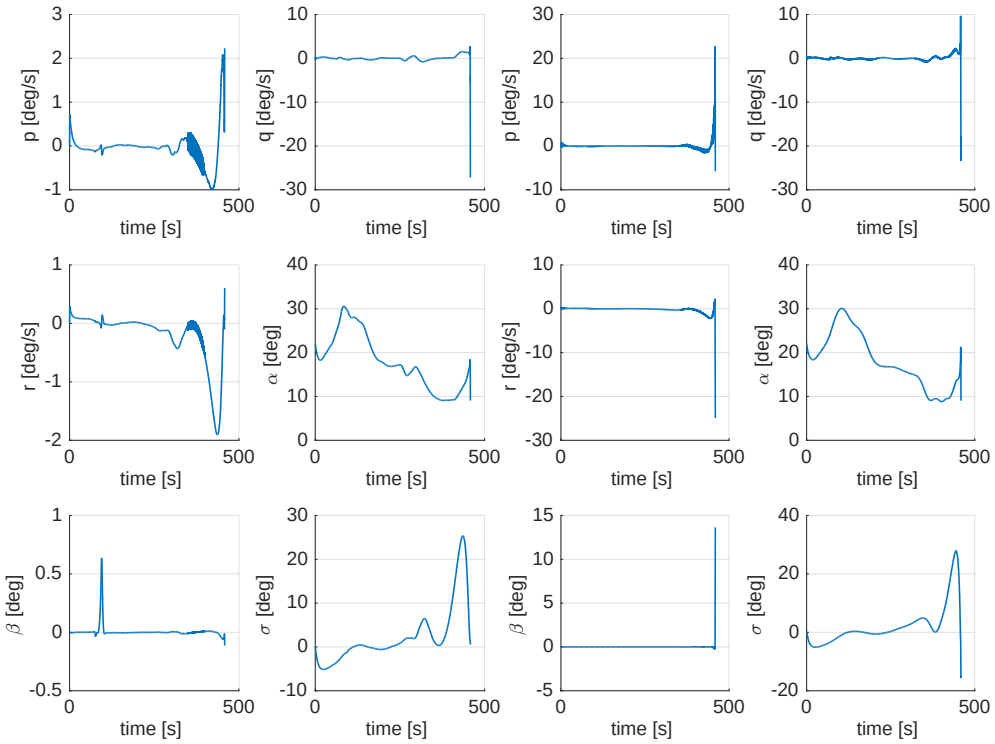


Figure 9.5: The rotational states when using an LQR-controller during an open-loop simulation on the left, and closed loop on the right.

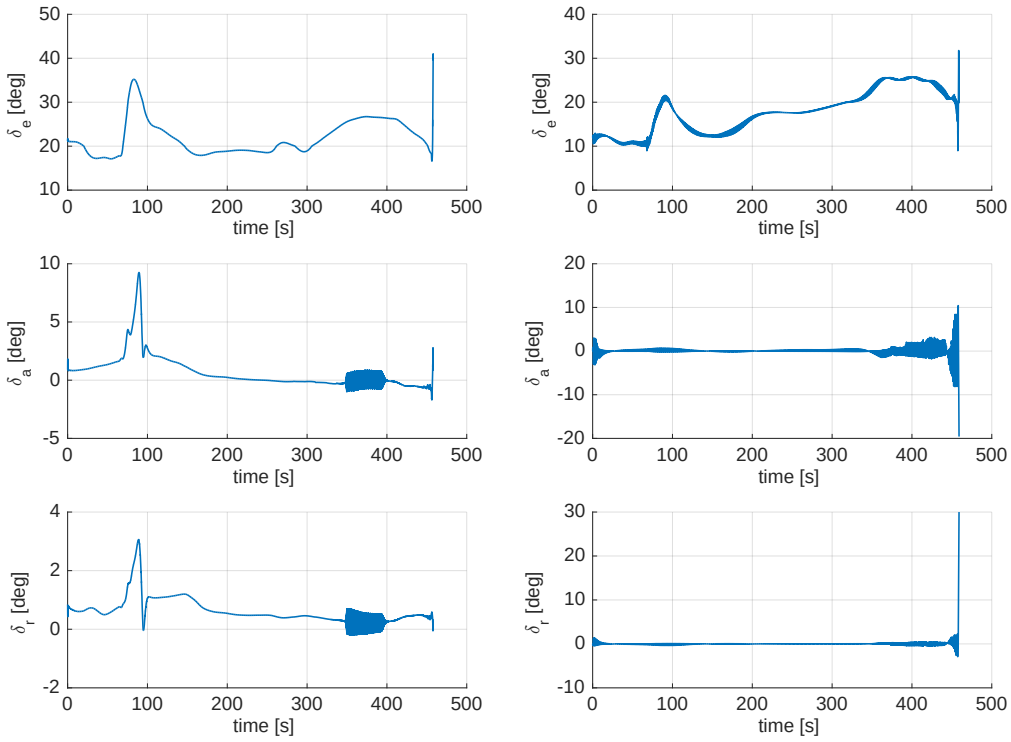


Figure 9.6: The translational states when using an LQR-controller during an open-loop simulation on the left, and closed loop on the right.

9.2. Pseudospectral Methods

It has been proven in the previous chapter that it is possible to create a 3-DOF system, and that the spacecraft can become a 6-DOF system when an LQR is added. The next step is too whether it is possible to use pseudospectral methods as a method to calculate the required deflections of the aerodynamic surfaces. First, a unit-step test is presented that is used to optimise a variety of variables. And second, an analysis of extending this to the full fledged trajectory.

The dynamics of the problem are already given in Section 3.6.2, the boundaries are determined to be

$$\begin{pmatrix} -10\,\text{deg} \\ -10\,\text{deg} \\ -10\,\text{deg} \\ -14\,\text{deg} \\ -8\text{deg} \\ -50\,\text{deg} \\ 0\,\text{deg} \\ -30\,\text{deg} \\ -30\,\text{deg} \end{pmatrix} \leq \begin{pmatrix} p \\ q \\ r \\ \alpha \\ \beta \\ \sigma \\ \delta_e \\ \delta_a \\ \delta_r \end{pmatrix} \leq \begin{pmatrix} 10\,\text{deg} \\ 10\,\text{deg} \\ 10\,\text{deg} \\ 40\,\text{deg} \\ 8\text{deg} \\ 50\,\text{deg} \\ 60\,\text{deg} \\ 30\,\text{deg} \\ 30\,\text{deg} \end{pmatrix} \tag{9.17}$$

and the derivatives of the controls are limited to stay within

$$\begin{pmatrix} -10\,\text{deg/s} \\ -10\,\text{deg/s} \\ -10\,\text{deg/s} \end{pmatrix} \leq \begin{pmatrix} \dot{\delta}_e \\ \dot{\delta}_a \\ \dot{\delta}_r \end{pmatrix} \leq \begin{pmatrix} 10\,\text{deg/s} \\ 10\,\text{deg/s} \\ 10\,\text{deg/s} \end{pmatrix} \tag{9.18}$$

The cost function has been determined to be

$$J = \int_{t_0}^{t_f} \left((\Delta\alpha)^2 + (\Delta\sigma)^2 + \beta^2 \right) dt \tag{9.19}$$

This cost function is designed to decrease all the differences of the mentioned states, and it is very similar to the cost function of LQR, but without setting the allowable deviations.
An initial attempt has been made by having a Mayer or a Lagrange cost function, both calculated a new solution for 0.05 seconds each time, similar to the control frequency. The end states of the controls were used as the input into the vehicle dynamics block. None of the solutions gave tolerable results. A successful method was found by using the cost function in combination with a feed-forward method. The result can be seen in Figures 6.7 and 6.8, where a change in angle of attack and bank angles is required, which is successfully attained within a second. The changes are close to what the maximum rate of these angle that the guidance system might require.

A time-step of 1 second has been selected for this trial for the feed forward method to ensure that good results are found before trying to slim this number down. The time it took to calculate this solution is 143 seconds, for a 20 nodes control system. This duration is unacceptable, especially since the frequency of the controller is 20 Hz. Trials indicated that the minimum time-step is 0.75 seconds, before faulty solutions are produced, this reduced the total time to 118 seconds. The solutions become wrong when the first few nodes are close to the starting point which creates a non-linear nature in the first fraction of the solution. This effect is reduced by spreading the nodes further apart.

On average, a solution took five seconds to calculate, even when using the previous solution as the basis. This is much longer than when calculating the reference trajectory, which takes roughly 120 seconds. The main reason is that the dynamics function is much more computationally expensive. Especially calculating the aerodynamics using Hermite cubic interpolation. The convergence of the solution starts out quick, but becomes slower, or even stagnates, when it is close. A possibility is the conflict between the Hermite interpolator and the Lagrange interpolation that PSM uses, but this needs to be tested by creating equations instead of tables. An advantage of doing this is that it speeds up the process since calculating the interpolation of four dimensional tables is very expensive. A part of the C_n-coefficient was removed that uses this, which reduced the required time by 16%. This difference cannot be explained solely by the reduced

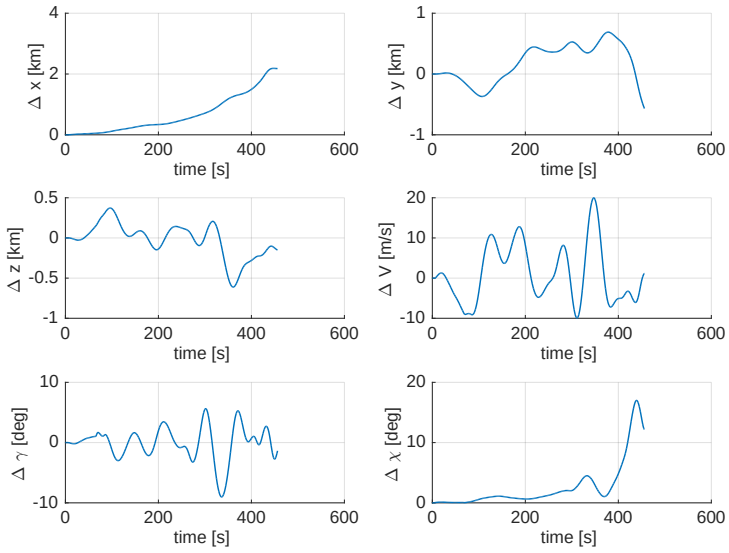


Figure 9.7: The translational states of an open-loop simulation with PSM as the control subsystem

computations, by far. Another point would be that the sparsity of the Jacobian is much smaller here, which means increased dependency between the different states.

An open-loop simulation is created similar to what was done in Section 9.1 to show the reaction of the controller. Figure 9.7-9.9 show these results. The first thing that can be noticed is the difference between the deflections required for the LQR method and PSM. The deflections for the latter method are smooth throughout the entire trajectory. This means in effect that the angular rates are also even. The downside is that the difference between the reference trajectory and this trajectory is very big, up to the deviation of the open-loop LQR. The time it took to calculate this solution is well above any real-time scenario. Therefore, this method is deemed infeasible. Producing the results for a closed-loop scenario does not make sense because of the computational duration. At least not until the computational issues are solved.

The next option that is available is to introduce a complete 6-DOF motion calculated by PSM. The idea is that PSM calculates the nominal translational and rotational states with the deflection angles as the control input over the entire trajectory, similar to what was shown in Chapter 8. The higher frequency control allocation would then be covered by LQR. The amount of states for this problem is 16 and there are 4 controls, the dependency matrix is therefore, large and dense. The controls propagate throughout the entire state vector. A major issue could be the amount of time it takes to calculate the solution, a simple problem is therefore tested first.

The maximum range trajectory is calculated just like what was done in Chapter 7. Figures 9.10, 9.11, and 9.12 show the results of the trajectory generation performed at the beginning of each simulation. This exact test has been performed before with the 3-DOF motion in Section 7.3, the results should therefore be similar. Unfortunately, this is not the case. This solution is shortened by 75% to 150 seconds. There are large variations in the lateral motion of the RV-W. The X-38 shakes around the Z-axis up to $10°/s$, and this continues throughout the entire simulation. The effect of this is that there is additional drag and that the flight path angle is lower than it should be. Both are counter to the cost function, which has the goal to maximise the travelled distance. A possible cause is the inclusion of the C_n-coefficient. The time to calculate the reference trajectory is reduced from 120.6 s to 47.7 s by removing this term. For comparison, the 3-DOF solution takes 0.86 s. This indicates that there are two possible problems. The first one is that the Hermite interpolator is not compatible with PSM, or, the second option, that there is a problem with the aerodynamic database. Both problems require more research before the answer can be given, but it seems that the limit of attitude control with PSM is reached with the current implementation.

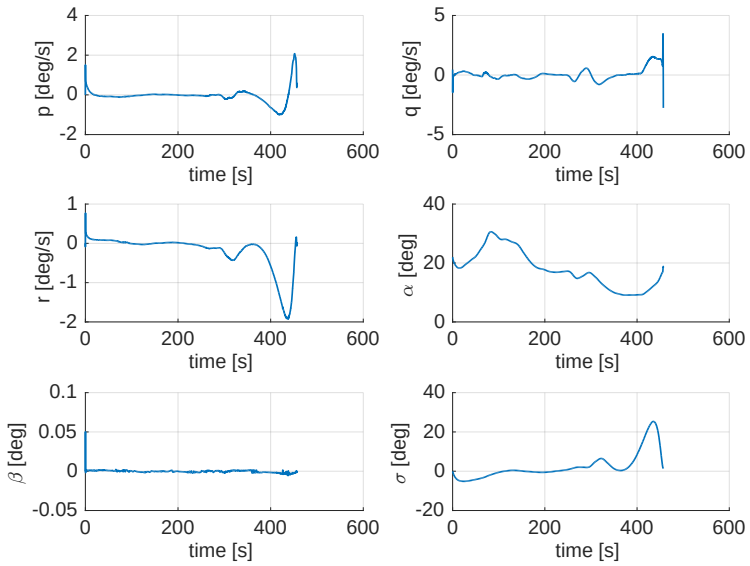


Figure 9.8: The rotational states of an open-loop simulation with PSM as the control subsystem.

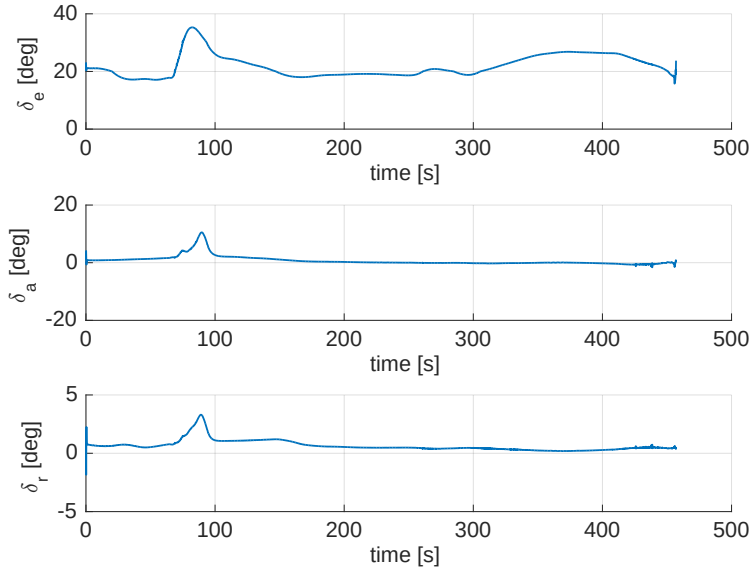


Figure 9.9: The controls of an open-loop simulation with PSM as the control subsystem.

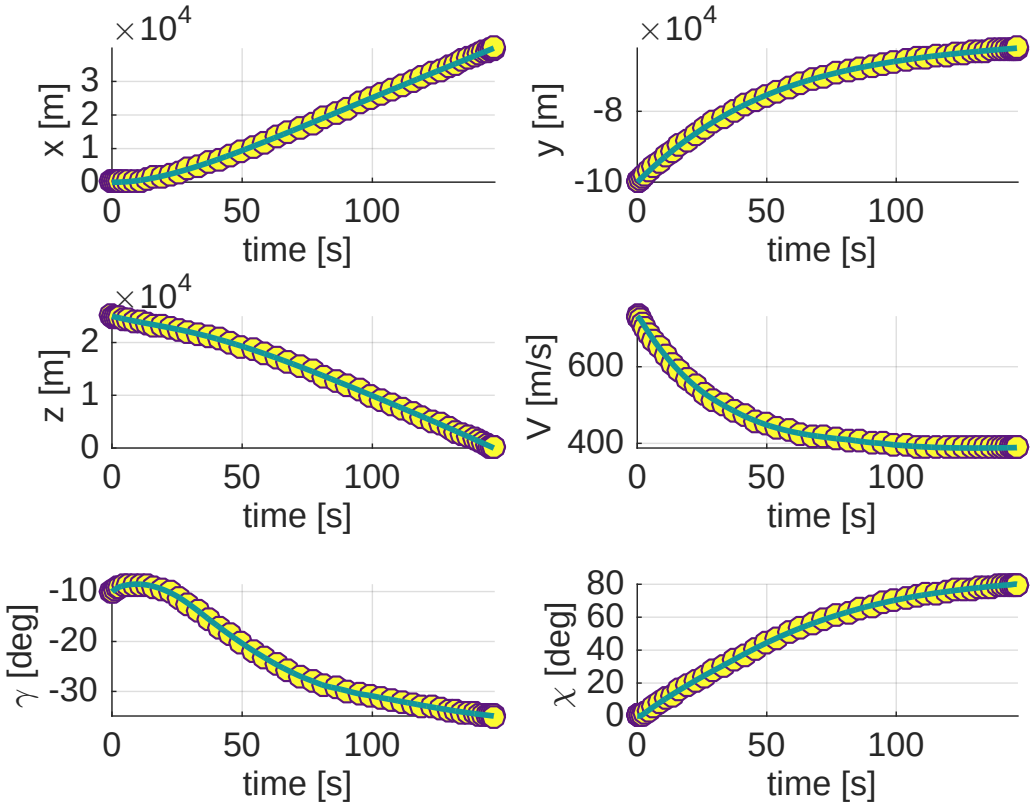


Figure 9.10: The translational states of a maximum distance simulation

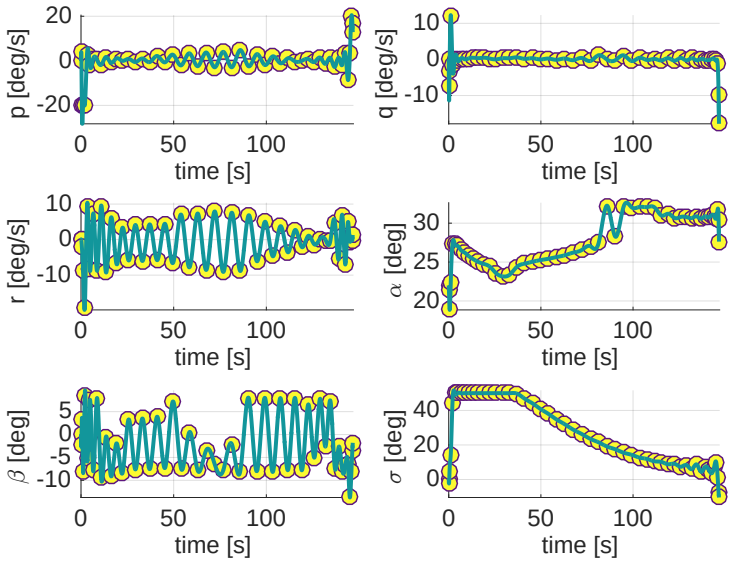


Figure 9.11: The rotational states of maximum distance simulation.

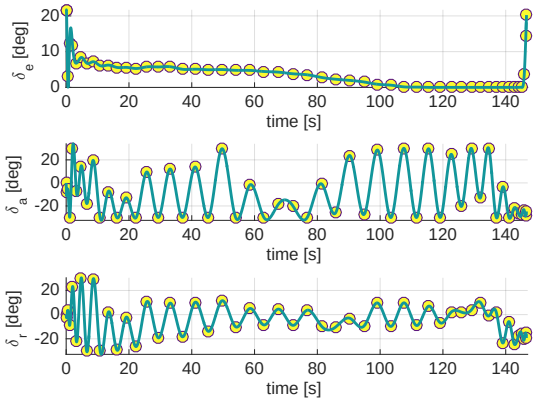


Figure 9.12: The controls of a maximum distance simulation.

10

Conclusions and Recommendations

This chapter closes this report with the conclusions that were seen in this report by answering the research questions posed in the introduction chapter. It is then followed by some heartfelt recommendations, to guide anyone who is willing to continue with this work.
The main question that was asked in the introduction is

> *Can pseudospectral methods be used as a robust, real-time guidance and control method for re-entry vehicles in the terminal area.*

It became apparent throughout this research that this question is not easily answered, and that there are many variables to be considered. However, it can be said that the implementation presented here favours guidance heavily over control in terms of their performance in this report. It is possible for the guidance algorithm to be implemented in other research, the control algorithm, however, requires much optimisation before it is mature enough. The text below shall give a more detailed answer.

Literature about re-entry is abundantly available, yet most of the guidance techniques used, even nowadays, are based on the Space Shuttle. The techniques can be divided in two sections, longitudinal and lateral. The longitudinal one is usually based on controlling the angle of attack based on the amount of energy that is available, and a bank angle is calculated using longitudinal techniques to aim either towards the edge of a pre-calculated heading alignment cylinder or the runway. The downsides of these techniques is that it can be hard to predict what the exact circumstances of the external environment are during the approach, and that each different entry requires its own pre-made trajectory. The beauty of pseudospectral methods, theoretically, is that both issues are no longer present, and that the trajectory is calculated onboard during the flight, to end directly in front of the runway without using a heading alignment cylinder. This thesis makes use of a flight approach corridor which must be entered. It is even possible to combine the longitudinal and lateral movement, by taking advantage of the non-linear dynamics, to achieve a smoother flight.

There are multiple types of PSM that are available, each with their own distinctive mathematical structure. The type that was chosen is called flipped Legendre-Gauss-Radau. The advantage of this one is that it allows an end-point and time to be chosen, or limited close to, and that is numerically stable. The SNOPT solver was selected to solve the non-linear optimisation problem. Not much research has been performed with this type, let alone re-entry, which made it extra interesting. The first demonstration of this method was the calculation of maximum range trajectories of both the HORUS and the X-38. Both spacecraft were easily manoeuvred to their maximum range in the subsonic phase without complications. Not so much for the supersonic phase, in which it was impossible for Horus to find a stable solution, yet the X-38 could. A possible reason for this is that there is a small angle of attack corridor which the optimisation program cannot handle, or that the aerodynamic database is lacking. The X-38 was therefore chosen to be the reference vehicle. The downside of this is that the X-38 is not a winged re-entry vehicle, but a lifting body vehicle. Variables were therefore adjusted to mimic the behaviour of winged re-entry vehicles, which in-

creased the operational range greatly.

It is now possible to go more in-depth by answering the sub-questions. The first one was

1. Is it possible to use pseudospectral methods as a guidance and control method.

This question can further be divided by separating the guidance and control. First, the guidance subsystem. The first step was to determine the variables related to solving the optimal problem. The initial state and boundaries were selected with literature and simple trial and error techniques. Multiple cost functions were tested and it was determined that a simple Mayer term which had the goal to, get the flight path angle to 9 degrees was the most efficient one, both cost- and result-wise. The final result is limited to be within the flight-approach corridor, it is therefore not required to be a part of this cost function. The result is a smooth trajectory that is 456 seconds long with a smooth change of the angle of attack and banking angle. The only downside is a bumpy flight path angle and load factor. The reason for this is the change in aerodynamic behaviour that was performed. The load factor stays well within the set boundaries. Heat loads are never a problem in this phase, so these are not taken into account. The guidance system proved to be excellent to calculate the reference trajectory. Second, the attitude control was added after the guidance subsystem. The LQR-controller was created first as a baseline, followed by a PSM-controller. The LQR gave satisfactory results during the open-loop simulation, the reference trajectory was not perfectly followed, but the general trend was that the aerodynamic surfaces had a smooth deflection rate. The closed-loop simulation that followed resulted in a non-smooth deflection graph, there were vibrations in the order of 1° of deflection. The PSM controller was implemented and it proved that a feed-forward method was required with a Mayer cost function to create the correct deflection angles. The downside of this method was that it is too slow to be used as a controller. A complete 6-DOF simulation was finally attempted to combine both the guidance and control methods. The current implementation of PSM proved to be unable to handle the optimisation process, and no viable solution could be found. It can be concluded that it is possible to use PSM as a guidance method, but not as a control method. The implemented guidance method even showed similarities to the heritage trajectories that use HAC, including the approach of the HAC that is performed by a S-turn and a constant bank angle around a non-existent cylinder.

The second sub-question was,

2. Is it possible for a range of initial conditions to create a guidance and control solution.

An analysis was performed by moving the starting point across a region around the runway. It is possible to reach the runway from a variety of positions, but there are also some gaps in this field, that cannot be easily explained. The gaps exist between viable points. Different initial heading angles were attempted at both a successful and failed position. A range of viable trajectories was found, as long as it was pointing roughly towards the runway. The reliability is high, higher than 90%, but that is not reliable enough to be flight-certified. The third sub-question involved the reliability once a trajectory has been found

3. How robust is the designed guidance and control method.

Several analyses were performed to assess how robust the guidance method is. The control method was not tested, since the results of that component were inadequate. The first test was a variation of the lift- and drag-coefficient that is unknown by the PSM guidance method. The system was able to handle these differences very well. The second test was to introduce wind and gusts. It was required that the general magnitude and direction of the wind is known beforehand. Variations of the magnitude and direction were possible. The problem was that the energy loss was smoothed out over the trajectory, and therefore, that a predictive method was required. A large wind-gust at hurricane levels was applied as well, and the RV/W could still reach the FAC. The conclusion of this sub-question is that the guidance method is robust.

4. Can the designed guidance and control method be executed in real-time.

The execution in real-time needs to be split into two parts to be able to answer this question. The required amount of time for the guidance subsystem is roughly a third of the simulation time, which is feasible for real-life applications. The current implementation of the control method requires a computation time that far exceeds the available time. The answer to this subquestion is therefore, partially.

The sub-questions have now been answered and that leaves us to the following statement. The final conclusion is thus that PSM can be used as a robust and real-time guidance method for re-entry vehicles in the terminal area. The control method is currently not viable.

During the course of this thesis there were many branches of work that could have been executed but are considered interesting.

The original aim was that during the research that the HORUS was used the reference vehicle. Later it proved to impossible due to the lack of a proper aerodynamic database that could handle non-linear optimisation programs. The database was generated back in 1988 during the conceptual design phase and only the coefficients for the supersonic and subsonic regime was created (Müller, 1988). An improved database has to be created with current day CFD methods and/or wind tunnel tests, for all the relevant Mach regions, before it can be used with Pseudospectral methods. A further improvement would be if the tables are converted into analytical functions. The improvement would be twofold. First, Pseudospectral methods is based on the Lagrange polynomial, however, Hermite interpolation had to be used to reduce computation times, creating an oscillation which becomes larger if the amount of collocation points is increased. Second, the reduce in computation times would reduce the overall time required to compute the correct solution. Adding this is especially helpful for attitude control. It would be a fun but challenging task to reach a real-time 6-DOF simulation by doing this. It could help to introduce trajectories from other sources to reach this goal.

The program right now as it stands is able to perform a single run, with single dynamics, and uniform boundaries. The program can be enhanced by creating a multistep program, one that can handle multiple phases of the flight (Rao, 2009). This could mean that the entire re-entry of a spacecraft could be simulated by segmenting each phase and matching the final and first state, from the early re-entry until the landing. This also opens the possibility of using Horus by setting the phase at the edges of the transonic region and using a constant angle of attack in this regime, similar as done by De Ridder (2009), and test if this corridor is the limiting factor for PSM.

A very interesting scenario that can be tested by combining both systems is that of vehicle failures. For example, there is now no direct feedback to the guidance system if an elevon is damaged. The guidance system could then take this into account when calculating a new trajectory. Another scenarios that is possible is using improved wind models. The current setup is only good for first-order analyses.

In short, the following recommendations are given:

- Create analytical aerodynamic coefficients
- Create a multi-step method to simulate the complete re-entry.
- Simulate failure scenarios
- Improve the wind model
- Create a full 6-DOF guidance method
- Use existing trajectories from other methods as the reference trajectory

A

State Space System of the Linearised Rotational Motion

This part of the appendix is an addendum to Section 3.6.2. It gives the linearised rotational motion in matrix form. It can be written down in a state-space form that has the following form (Mooij, 2013)

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \tag{A.1}$$

This matrix equation has **x** as the $n \times 1$ state vector, **u** the $m \times 1$ control vector, whereas **A** is a $n \times n$ and **B** is a $m \times n$ coefficient matrix. The output equation shall not be presented here, but it suffices to say that all the states and controls are required as output. The states are directly chosen from Equations 3.107-3.112, they are

$$\dot{\mathbf{x}} = \begin{bmatrix} \Delta\dot{p} & \Delta\dot{q} & \Delta\dot{r} & \Delta\dot{\alpha} & \Delta\dot{\beta} & \Delta\dot{\sigma} \end{bmatrix}^T \tag{A.2}$$

$$\mathbf{x} = \begin{bmatrix} \Delta p & \Delta q & \Delta r & \Delta\alpha & \Delta\beta & \Delta\sigma \end{bmatrix}^T \tag{A.3}$$

The control vector **u** consists of both the aerosurfaces and the reaction control system. It comprises the following

$$\Delta\mathbf{u} = \begin{bmatrix} \Delta\delta_{\mathrm{e}} & \Delta\delta_{\mathrm{a}} & \Delta\delta_{\mathrm{r}} & \Delta\mathrm{M}_{\mathrm{T,x}} & \Delta\mathrm{M}_{\mathrm{T,y}} & \Delta\mathrm{M}_{\mathrm{T,z}} \end{bmatrix} \tag{A.4}$$

Matrices **A** and **B** have the following shape

$$\mathbf{A} = \begin{bmatrix} a_{pp} & a_{pq} & a_{pr} & a_{p\alpha} & a_{p\beta} & a_{p\sigma} \\ a_{qp} & a_{qq} & a_{qr} & a_{q\alpha} & a_{q\beta} & a_{q\sigma} \\ a_{rp} & a_{rq} & a_{rr} & a_{r\alpha} & a_{r\beta} & a_{r\sigma} \\ a_{\alpha p} & a_{\alpha q} & a_{\alpha r} & a_{\alpha\alpha} & a_{\alpha\beta} & a_{\alpha\sigma} \\ a_{\beta p} & a_{\beta q} & a_{\beta r} & a_{\beta\alpha} & a_{\beta\beta} & a_{\beta\sigma} \\ a_{\sigma p} & a_{\sigma q} & a_{\sigma r} & a_{\sigma\alpha} & a_{\sigma\beta} & a_{\sigma\sigma} \end{bmatrix} \tag{A.5}$$

$$\mathbf{B} = \begin{bmatrix} b_{pe} & b_{pa} & b_{pr} & b_{px} & b_{py} & b_{pz} \\ b_{qe} & b_{qa} & b_{qr} & b_{qx} & b_{qy} & b_{qz} \\ b_{re} & b_{ra} & b_{rr} & b_{rx} & b_{ry} & b_{rz} \\ b_{\alpha e} & b_{\alpha a} & b_{\alpha r} & b_{\alpha x} & b_{\alpha y} & b_{\alpha z} \\ b_{\beta e} & b_{\beta a} & b_{\beta r} & b_{\beta x} & b_{\beta y} & b_{\beta z} \\ b_{\sigma e} & b_{\sigma r} & b_{\sigma r} & b_{\sigma x} & b_{\sigma y} & b_{\sigma z} \end{bmatrix} \tag{A.6}$$

The first matrix **A** consists of the following entries.

$$
\begin{aligned}
a_{pp} &= a_{pq} = a_{pr} = a_{p\alpha} = a_{p\sigma} = 0 \\
a_{p\beta} &= \frac{1}{I_{xx}} \frac{\partial C_l}{\partial \beta} \bar{q}_0 S_{ref} b_{ref}
\end{aligned}
$$

$$
\begin{aligned}
a_{qp} &= a_{qq} = a_{qr} = a_{q\beta} = a_{q\sigma} = 0 \\
a_{q\alpha} &= \frac{1}{I_{yy}} \frac{\partial C_m}{\partial \alpha} \bar{q}_0 S_{ref} b_{ref}
\end{aligned}
$$

$$
\begin{aligned}
a_{rp} &= a_{rq} = a_{rr} = a_{r\alpha} = a_{r\sigma} = 0 \\
a_{r\beta} &= \frac{1}{I_{zz}} \frac{\partial C_n}{\partial \beta} \bar{q}_0 S_{ref} b_{ref}
\end{aligned}
$$

$$
\begin{aligned}
a_{\alpha p} &= a_{\alpha r} = a_{\alpha\beta} = 0 \\
a_{\alpha q} &= 1 \\
a_{\alpha\alpha} &= -\frac{1}{mV_0} \frac{\partial C_L}{\partial \alpha} \bar{q}_0 S_{ref} \\
a_{\alpha\sigma} &= -\frac{g_0}{V_0} \cos\gamma_0 \sin\sigma_0
\end{aligned}
$$

$$
\begin{aligned}
a_{\beta q} &= a_{\beta\sigma} = 0 \\
a_{\beta p} &= \sin\alpha_0 \\
a_{\beta r} &= \cos\alpha_0 \\
a_{\beta\beta} &= -\frac{1}{mV_0} \frac{\partial C_S}{\partial \beta} \bar{q}_0 S_{ref} \\
a_{\beta\sigma} &= -\frac{g_0}{V_0} \cos\gamma_0 \cos\sigma_0
\end{aligned}
$$

$$
\begin{aligned}
a_{\sigma q} &= 0 \\
a_{\sigma p} &= -cos\alpha_0 \\
a_{\sigma r} &= -\sin\alpha_0 \\
a_{\sigma\alpha} &= \frac{\tan\gamma_0}{mV_0} \sin\sigma_0 \frac{\partial C_L}{\partial \alpha} \bar{q}_0 S_{ref} \\
a_{\sigma\beta} &= \frac{g_0}{V_0} \cos\gamma_0 \cos\sigma_0 - \frac{L_0}{mV_0} + \frac{\tan\gamma_0}{mV_0} \cos\sigma_0 \frac{\partial C_S}{\partial \beta} \bar{q}_0 S_{ref} \\
a_{\sigma\sigma} &= \frac{\tan\gamma_0}{mV_0} + \cos\sigma_0 L_0
\end{aligned}
$$

And matrix **B** has the following entries

$$
\begin{aligned}
b_{pe} =& b_{pr} = b_{py} = b_{pz} = 0 \\
b_{pa} =& \frac{1}{I_{xx}} \frac{\partial C_l}{\partial \delta_a} \bar{q}_0 S_{ref} b_{ref} \\
b_{px} =& \frac{1}{I_{xx}}
\end{aligned}
$$

$$
\begin{aligned}
b_{qa} =& b_{qr} = b_{qx} = b_{qz} = 0 \\
b_{qe} =& \frac{1}{I_{yy}} \frac{\partial C_m}{\partial \delta_e} \bar{q}_0 S_{ref} b_{ref} \\
b_{qy} =& \frac{1}{I_{yy}}
\end{aligned}
$$

$$
\begin{aligned}
b_{ra} =& b_{rx} = b_{ry} = 0 \\
b_{re} =& \frac{1}{I_{zz}} \frac{\partial C_n}{\partial \delta_e} \bar{q}_0 S_{ref} b_{ref} \\
b_{rr} =& \frac{1}{I_{zz}} \frac{\partial C_n}{\partial \delta_r} \bar{q}_0 S_{ref} b_{ref} \\
b_{rz} =& \frac{1}{I_{zz}}
\end{aligned}
$$

$$
\begin{aligned}
b_{\alpha e} =& b_{\alpha a} = b_{\alpha r} = b_{\alpha x} = b_{\alpha y} = b_{\alpha z} = 0 \\
b_{\beta e} =& b_{\beta a} = b_{\beta r} = b_{\beta x} = b_{\beta y} = b_{\beta z} = 0 \\
b_{\sigma e} =& b_{\sigma r} = b_{\sigma r} = b_{\sigma x} = b_{\sigma y} = b_{\sigma z} = 0
\end{aligned}
$$

The final state-space form is then created by combining the previous equations

$$
\begin{bmatrix} \Delta\dot{p} \\ \Delta\dot{q} \\ \Delta\dot{r} \\ \Delta\dot{\alpha} \\ \Delta\dot{\beta} \\ \Delta\dot{\sigma} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & a_{p\beta} & 0 \\ 0 & 0 & 0 & a_{q\alpha} & 0 & 0 \\ 0 & 0 & 0 & 0 & a_{r\beta} & 0 \\ 0 & a_{\alpha q} & 0 & a_{\alpha\alpha} & 0 & a_{\alpha\sigma} \\ a_{\beta p} & 0 & a_{\beta r} & a_{\beta\alpha} & 0 & 0 \\ a_{\sigma p} & 0 & a_{\sigma r} & a_{\sigma\alpha} & a_{\sigma\beta} & a_{\sigma\sigma} \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta q \\ \Delta r \\ \Delta\alpha \\ \Delta\beta \\ \Delta\sigma \end{bmatrix} + \begin{bmatrix} 0 & b_{pa} & 0 & b_{px} & 0 & 0 \\ b_{qe} & 0 & 0 & 0 & b_{qy} & 0 \\ b_{re} & 0 & b_{rr} & 0 & 0 & b_{rz} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta\delta_e \\ \Delta\delta_a \\ \Delta\delta_r \\ \Delta M_{T,x} \\ \Delta M_{T,y} \\ \Delta M_{T,z} \end{bmatrix} \tag{A.7}
$$

The resulting matrix of Section A, Equation A.7, contains both the lateral and longitudinal motion. It can however be said that the effect of both environments on each other is limited or even negligible. This statement can be verified by analysing the difference in eigenvalues of both the coupled and the uncoupled system. For that, there is also a need to present the uncoupled system. The longitudinal part is up first, followed by the lateral components.

The longitudinal component is the smaller one of the two, since it consists merely of two states and controls. The states are the angle of attack α and the pitch angle q, and the controls are the elevator δ_e and the thruster in y-direction $M_{T,y}$.

$$
\Delta\dot{q} = \frac{1}{I_{yy}} \left(\frac{\partial C_m}{\partial \alpha} \Delta\alpha + \frac{\partial C_m}{\partial \delta_e} \Delta\delta_e \right) \bar{q}_e S_{ref} c_{ref} + \frac{\Delta M_{T,y}}{I_{yy}} \tag{A.8}
$$

$$
\Delta\dot{\alpha} = \Delta q - \frac{1}{mV_e} \frac{\partial C_L}{\partial \alpha} \bar{q}_e S_{ref} \Delta\alpha \tag{A.9}
$$

Or, in matrix form

$$
\begin{bmatrix} \Delta\dot{q} \\ \Delta\dot{\alpha} \end{bmatrix} = \begin{bmatrix} a_{q\alpha} & 0 \\ a_{\alpha q} & a_{\alpha\alpha} \end{bmatrix} \begin{bmatrix} \Delta q \\ \Delta\alpha \end{bmatrix} + \begin{bmatrix} b_{qe} & b_{qy} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta\delta_e \\ \Delta M_{T,y} \end{bmatrix} \tag{A.10}
$$

The lateral component consists of the four states that are present in Equation A.7 and are not in the longitudinal component. This part focuses on the roll and yaw movement and not on the up and down motion. The equations are

$$\Delta\dot{p} = \frac{1}{I_{xx}}\left(\frac{\partial C_l}{\partial \beta}\Delta\beta + \frac{\partial C_l}{\partial \delta_a}\Delta\delta_a\right)\bar{q}_e S_{ref} b_{ref} + \frac{\Delta M_{T,x}}{I_{xx}} \tag{A.11}$$

$$\Delta\dot{r} = \frac{1}{I_{zz}}\left(\frac{\partial C_n}{\partial \beta}\Delta\beta + \frac{\partial C_n}{\partial \delta_a}\Delta\delta_a + \frac{\partial C_n}{\partial \delta_r}\Delta\delta_r\right)\bar{q}_e S_{ref} b_{ref} + \frac{\Delta M_{T,z}}{I_{zz}} \tag{A.12}$$

$$\Delta\dot{\beta} = \sin\alpha_e p - \cos\alpha_e \Delta r - \frac{g_e}{V_e}\cos\gamma_e \cos\sigma_e \Delta\sigma \tag{A.13}$$

$$\Delta\dot{\sigma} = -\cos\alpha_e \Delta p - \sin\alpha_e \Delta r + \left(\frac{g_e}{V_e}\cos\gamma_e \cos\sigma_e - \frac{L_e}{mV_e}\right)\Delta\beta + \frac{\tan\gamma_e}{mV_e}\cos\sigma L_e \Delta\sigma \tag{A.14}$$

And also in matrix form

$$\begin{bmatrix}\Delta\dot{p}\\ \Delta\dot{r}\\ \Delta\dot{\beta}\\ \Delta\dot{\sigma}\end{bmatrix} = \begin{bmatrix}0 & 0 & a_{p\beta} & 0\\ 0 & 0 & a_{r\beta} & 0\\ a_{\beta p} & a_{\beta r} & 0 & 0\\ a_{\sigma p} & a_{\sigma r} & a_{\sigma\beta} & a_{\sigma\sigma}\end{bmatrix}\begin{bmatrix}\Delta p\\ \Delta r\\ \Delta\beta\\ \Delta\sigma\end{bmatrix} + \begin{bmatrix}b_{pa} & 0 & b_{px} & 0\\ 0 & b_{rr} & 0 & b_{rz}\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\end{bmatrix}\begin{bmatrix}\Delta\delta_a\\ \Delta\delta_r\\ \Delta M_{T,x}\\ \Delta M_{T,z}\end{bmatrix} \tag{A.15}$$

Analysing the difference in eigenvalues should give a clear understanding about how much the dynamic are decoupled in different (atmospheric) conditions.

B

Aerodynamic Database

This chapter presents the aerodynamic database of Horus used during this thesis. The X-38 database is classified. The lift and drag coefficients are directly from the Horus database (Mooij, 1995). The original values are extracted from (Rockwell, 1965). The lower region of the supersonic regime is interpolated from the other data, this causes a asymptote which is physically not possible. The region of $\alpha < 5°$ for $M > 0.9$ should be avoided. The values in between the points are calculated using cubic Hermite interpolation.

The lift coefficient is calculated with

$$C_L = C_{L_{trim}} \tag{B.1}$$

The tables used to calculate $C_{L_{trim}}$ are

Table B.1: Trimmed lift coefficient during the subsonic phase

M α	0.20	0.60	0.80	0.90	0.95
0.0	0.02	0.02	0.02	0.02	0.02
2.5	0.10	0.10	0.10	0.10	0.10
5.0	0.18	0.18	0.18	0.18	0.18
7.5	0.27	0.27	0.27	0.27	0.27
10.0	0.35	0.35	0.35	0.34	0.34
12.5	0.44	0.44	0.44	0.42	0.41
15.0	0.50	0.50	0.50	0.48	0.45
20.0	0.63	0.63	0.61	0.56	0.53
25.0	0.72	0.71	0.68	0.63	0.56
30.0	0.80	0.76	0.72	0.65	0.58

Table B.2: Trimmed lift coefficient during the transonic phase

M α	0.95	1.05	1.1	1.2
0	0.02	0.02	0.02	−0.075
2.5	0.10	0.12	0.12	0.0225
5.0	0.18	0.21	0.21	0.1200
7.5	0.27	0.30	0.30	0.2350
10.0	0.34	0.37	0.37	0.3500

Table B.3: Trimmed lift coefficient during the supersonic phase

M α	1.20	1.50	2.00	3.00	5.00
0.0	−0.075	−0.075	−0.075	−0.075	−0.075
5.0	0.1200	0.0800	0.0600	0.0500	0.0400
10.0	0.3500	0.2650	0.2150	0.1800	0.1600
15.0	0.6300	0.4800	0.3900	0.3400	0.2950
20.0	0.9550	0.6900	0.5750	0.4950	0.4400
25.0	1.2750	0.9100	0.7600	0.6500	0.5800
30.0	1.2750	1.1250	0.9200	0.7900	0.7150
35.0	1.2750	1.3100	1.0700	0.9200	0.8300
40.0	1.2750	1.3100	1.1800	1.0200	0.9300
45.0	1.2750	1.3100	1.2600	1.0900	1.0000

The drag coefficient is calculated with

$$C_D = C_{D_{trim}} - C_{D_h} + C_{D_{SPB}} \tag{B.2}$$

The values of $C_{D_{trim}}$ are calculated using L/D-tables.

Table B.4: Trimmed lift over drag during the subsonic phase

M α	0.20	0.60	0.80	0.90	0.95
0.0	0.35	0.40	0.40	0.20	0.15
2.5	2.35	2.60	2.40	1.49	0.80
5.0	3.40	3.75	3.60	2.30	1.35
7.5	3.75	4.00	3.90	2.70	1.70
10.0	3.70	3.85	3.82	2.73	1.82
12.5	3.45	3.60	3.60	2.60	1.79
15.0	3.20	3.28	3.28	2.35	1.60
20.0	2.65	2.75	2.63	1.80	1.30
25.0	2.20	2.25	2.05	1.40	1.10
30.0	1.70	1.70	1.60	1.20	0.90

Table B.5: Trimmed lift over drag during the transonic phase

M α	0.95	1.05	1.1	1.2
0.0	0.15	0.05	0.05	−0.25
2.5	0.80	0.45	0.40	0.350
5.0	1.35	0.90	0.85	0.790
7.5	1.70	1.40	1.35	1.265
10.0	1.82	1.76	1.74	1.740

Table B.6: Trimmed lift over drag during the supersonic phase

M α	1.20	1.50	2.00	3.00	5.00
0.0	−0.25	−0.35	−0.50	−0.60	−0.07
5.0	0.790	0.750	0.700	0.600	0.550
10.0	1.740	1.850	1.900	1.850	1.850
15.0	1.920	2.200	2.270	2.270	2.130
20.0	2.000	2.100	2.150	2.090	1.980
25.0	1.880	1.880	1.880	1.820	1.740
30.0	1.650	1.650	1.590	1.570	1.500
35.0	1.390	1.400	1.325	1.325	1.290
40.0	1.200	1.220	1.150	1.150	1.100
45.0	1.020	1.050	0.950	0.950	0.930

Table B.7: Drag height coefficient

M km	0.20	0.60	0.80	0.90	0.95	1.2	1.5
0.0	0.0072	0.0028	0.0026	0.0023	0.0018	0.0013	0.0010
5.0	0.0046	0.0015	0.0012	0.0010	0.0007	0.0005	0.0001
10.0	0.0023	0.0007	0.0004	0.0002	0.0001	0.0000	0.0000
15.0	0.0008	0.0001	0.0001	0.0000	0.0000	0.0000	0.0000
20.0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

Bibliography

Ackermann, J., Breteau, J., Kauffmann, J., Ramusat, G., and Tumino, G. (2005). The Road to the Next-Generation European Launcher – An overview of the FLPP. Bulletin 123.

Bayer, M. (2010). Description of the FESTIP VTHL-TSTO System Concept Studies.

Ben-Israel, A. (1966). A Newton-Raphson method for the solution of systems of equations. DOI: 10.1016/0022-247X(66)90115-6, *Journal of Mathematical Analysis and Applications*, 15(2):243–252.

Bertin, J. J. and Cummings, R. M. (2003). Fifty years of hypersonics: where we've been, where we're going. DOI: 10.1016/S0376-0421(03)00079-4, *Progress in Aerospace Sciences*, 39(6-7):511–536.

Betts, J. (2010). Practical Methods for Optimal Control and Estimation Using Nonlinear Programming. Society for Industrial and Applied Mathematics (SIAM). ISBN 9780898716887.

Bollino, K. (2006). High-fidelity Real-Time Trajectory Optimization For Reusable Launch Vehicles. PhD thesis, Naval Postgraduate School of Monterey, California.

Bollino, K., Ross, M. I., and Doman, D. (2006). Optimal Nonlinear Feedback Guidance for Reentry Vehicles. In Guidance, Navigation, and Control and Co-located Conferences, pages —–. American Institute of Aeronautics and Astronautics. DOI: 10.2514/6.2006-6074.

Bunse-Gerstner, A. (1996). Computational Solution of the Algebraic Riccati Equation. DOI: 10.1.1.51.5104, *The Society of Instrument and Control Engineers*, 35(8):632–639.

Cooke, D. (1982). Space Shuttle stability and control test plan, In 9th Atmospheric Flight Mechanics Conference, Reston, Virigina. AIAA, American Institute of Aeronautics and Astronautics. DOI: 10.2514/6.1982-1315.

Corless, R. M. and Fillion, N. (2013). A graduate introduction to numerical methods, volume 10. Springer, 10th edition. ISBN 978-1-4614-8452-3.

Cox, K. J. (1971). Space Shuttle Guidance, Navigation and Control Design Equations: Volume 4 De-orbit and Atmospheric Operations. Technical Report NASA-TM-X-67709, NASA.

da Costa, R. (2003). Studies for Terminal Area GNC of Reusable Launch Vehicles, In AIAA Guidance, Navigation, and Control Conference and Exhibit, Reston, Virigina. American Institute of Aeronautics and Astronautics. DOI: 10.2514/6.2003-5438.

Dassault (2001). Formulation of the X-38 (V-201 & V-131r) Aerodynamic Data Base.

De Ridder, S. (2009). Study on optimal trajectories and energy mangement capabilities of a winged re-entry vehicle during the terminal area. Technical report, Delft University of Technology.

De Ridder, S. and Mooij, E. (2009). Optimal Terminal Area Strategies and Energy Tube Concept for a Winged Re-Entry Vehicle. In AIAA Guidance, Navigation, and Control Conference, pages —–. American Institute of Aeronautics and Astronautics, Reston, Virigina. DOI: 10.2514/6.2009-5769.

De Ridder, S. and Mooij, E. (2011). Terminal area trajectory planning using the energy-tube concept for reusable launch vehicles. DOI: 10.1016/j.actaastro.2010.08.032, *Acta Astronautica*, 68(7-8):915–930.

Delft University of Technology (2012). The TU Delft Astrodynamics Toolbox. url: http://tudat.tudelft.nl/projects/tudat/wiki.

D'Onofrio, V., Sagliano, M., and Arslantas, Y. E. (2016). Exact Hybrid Jacobian Computation for Optimal Trajectories via Dual Number Theory, In AIAA Guidance, Navigation, and Control Conference, number January, page 23, Reston, Virginia. American Institute of Aeronautics and Astronautics. DOI: 10.2514/6.2016-0867.

Dujarric, C. (1999). Possible future European launchers- A process of convergence. ESA Bulletin.

Duke, E. L. (1994). Derivation and Definition of a Linear Aircraft Model. Technical report, NASA.

Ehlers, H. and Kraemer, J. (1977). Shuttle orbiter guidance system for the terminal flight phase. DOI: 10.1016/0005-1098(77)90005-X, *Automatica*, 13(1):11–21.

Elnagar, G., Kazemi, M., and Razzaghi, M. (1995). The pseudospectral Legendre method for discretizing optimal control problems. DOI: 10.1109/9.467672, *IEEE Transactions on Automatic Control*, 40(10):1793–1796.

Fumenti, F., Circi, C., and Romagnoli, D. (2013). Collocation points distributions for optimal spacecraft trajectories. DOI: 10.1016/j.cnsns.2012.07.023, *Communications in Nonlinear Science and Numerical Simulation*, 18(3):710–727.

Garg, D. (2011). Advances in global pseudospectral methods for optimal control. PhD thesis, University of Florida.

Georgie, J., Valasek, J., and Ward, D. T. (2002). Reentry vehicle flight control design guidelines: Dynamic inversion. Technical Report NASA/TP - 2002-210771, NASA.

Ghaffari, H. R. and Hribar, M. B. (2007). Non-Linear Optimization Software KNITRO.

Gill, P. E., Murray, W., and Saunders, M. a. (2008). User's Guide for SNOPT Version 7 : Software for Large Scale Non-Linear Programming.

Hanson, J. M. (2002). A plan for advanced guidance and control technology for 2nd generation reusable launch vehicles. *AIAA paper*, 4557:9.

Helmersson, A. (1988). Terminal area guidance strategies. Technical Report TN-ESA 6718/85-8, Saab Space.

Hendrickx, B. and Vis, B. (2007). Energiya—Buran. Springer Praxis Books. Praxis, New York, NY. ISBN 978-0-387-69848-9.

Herman, A. L. and Conway, B. A. (1996). Direct optimization using collocation based on high-order Gauss-Lobatto quadrature rules. DOI: 10.2514/3.21662, *Journal of Guidance, Control, and Dynamics*, 19(3):592–599.

Hirschel, E. H. and Weiland, C. (2009). Selected Aerothermodynamic Design Problems of Hypersonic Flight Vehicles. Springer Berlin Heidelberg, Berlin, Heidelberg, first edition. ISBN 978-3-540-89973-0.

Jenkins, D., Landis, T., and Miller, J. (2003). AMERICAN X-VEHICLES An Inventory—X-1 to X-50. NASA, SP-2003-4531.

Johnson, D. L. (2008). Terrestrial Environment (Climatic) Criteria Guidelines for Use in Aerospace Vehicle Development, 2008 Revision. TM 2008-215633, NASA.

Juliana, S. (2003). Work Package 2140 Flight Control Design Support - Sphynx Project. Technical report, Faculty of Aerospace Engineering Delft University of Technology.

Kafer, G. C. (1982). Space Shuttle Entry/Landing Flight Control Design Description, In Culp, R., Bauman, E., and Dorroh Jr., W., editors, Guidance and Control Conference.

Karpenko, M., Bhatt, S., Bedrossian, N., Fleming, A., and Ross, I. M. (2012). First Flight Results on Time-Optimal Spacecraft Slews. DOI: 10.2514/1.54937, *Journal of Guidance, Control, and Dynamics*, 35(2):367–376.

Kawajir, Y., Laird, C. D., and Waechter, A. (2015). Introduction to IPOPT: A tutorial for downloading, installing, and using IPOPT. url: https://projects.coin-or.org/Ipopt.

Kluever, C. A. and Horneman, K. R. (2005). Terminal trajectory planning and optimization for an unpowered reusable launch vehicle, In Guidance, Navigation, and Control Conference and Exhibit, AIAA, San Francisco, California.

Kreyszig, E. (2011). Advanced engineering mathematics. Wiley, 10th edition. ISBN 978-0-470-45836-5.

Lammens, S. (2015). Suborbital optimization for point-to-point travel. Master's thesis, Delft University of Technology.

Lu, P. and Hanson, J. M. (1998). Entry Guidance for the X-33 Vehicle. DOI: 10.2514/2.3332, *Journal of Spacecraft and Rockets*, 35(3):342–349.

Luo, J. and Lan, C. E. (1995). Determination of weighting matrices of a linear quadratic regulator. DOI: 10.2514/3.21569, *Journal of Guidance, Control, and Dynamics*, 18(6):1462–1463.

Martins, J. R. R. A., Sturdza, P., and Alonso, J. J. (2003). The complex-step derivative approximation. DOI: 10.1145/838250.838251, *ACM Trans. Math. Softw.*, 29(3):245–262.

Montenbruck, O. and Gill, E. (2012). Satellite Orbits: Models, Methods and Applications. Springer Berlin Heidelberg. ISBN 978-3-642-58351-3.

Mooij, E. (1995). The HORUS-2B Reference Vehicle. Technical report, Delft University of Technology.

Mooij, E. (1998). Aerospace-plane flight dynamics: analysis of guidance and control concepts. Dissertation, TU Delft.

Mooij, E. (2013). Re-entry systems. Delft University of Technology.

Müller, W. (1988). Study on re-entry guidance and control: final report; Vol. 3,2: Appendix to TN-ESA 6718/85-3, In ESA CR;2652.

NOAA, NASA, and USAF (1976). U.S. Standard Atmosphere, 1976 (NOAA Document S/T 76-1562). NOAA and NASA and USAF, 1st edition.

Oberkampf, W. L. and Trucano, T. G. (2002). Verification and validation in computational fluid dynamics. *Progress in Aerospace Sciences*, 38(3):209–272.

Ogata, K. (2010). Modern Control Engineering. Instrumentation and controls series. Prentice Hall. ISBN 9780136156734.

Papp, Z. (2014). Mission Planner for Heating-Optimal Re-Entry Trajectories with Extended Range Capability. Master's thesis, Delft University of Technology.

Qi Gong, Wei Kang, Bedrossian, N. S., Fahroo, F., Pooya Sekhavat, and Bollino, K. (2007). Pseudospectral Optimal Control for Military and Industrial Applications.

Rao, A. (2009). A Survey of Numerical Methods for Optimal Control, In 2009 AAS/AIAA Astrodynamics Specialist Conference, page 32, AAS Paper 09-334, Pittsburgh, PA, August 10-13, 2009. DOI: AA.

Regan, F. J. (1984). Re-entry vehicle dynamics. New York, N.Y. : American Institute of Aeronautics and Astronautics. ISBN 0915928787.

Rockwell (1965). Aerodynamic Design Data Book: Orbiter vehicle. vol. 1. Aerodynamic Design Data Book: Orbiter Vehicle. National Technical Information Service, U.S. Department of Commerce.

Ruijgrok, G. (2009). Elements of airplane performance. VSSD. ISBN 978-9065622327.

Sagliano, M. (2014). Performance analysis of linear and nonlinear techniques for automatic scaling of discretized control problems. DOI: http://dx.doi.org/10.1016/j.orl.2014.03.003, *Operations Research Letters*, 42(3):213–216.

Sagliano, M. and Theil, S. (2013). Hybrid Jacobian Computation for Fast Optimal Trajectories Generation. In Guidance, Navigation, and Control and Co-located Conferences, pages —–. American Institute of Aeronautics and Astronautics. DOI: 10.2514/6.2013-4554.

Schierman, J. D., Ward, D. G., Hull, J. R., Gandhi, N., Oppenheimer, M., and Doman, D. B. (2004). Integrated adaptive guidance and control for re-entry vehicles with flight test results. *Journal of Guidance, Control, and Dynamics*, 27(6):975–988.

Singla, P., Mortari, D., and Junkins, J. L. (2005). How to avoid singularity when using Euler Angles. *Advances in astronautical sciences*, 119:1409–1426.

Treadwell, T. C. (2010). Stepping stones to the stars, the story of manned spaceflight. The history press.

Vinh, N. X. (1981). Optimal trajectories in atmospheric flight. Elsevier scientific publishing company.

Wacker, R. and Munday, S. and Merkle, S. (2001). X-38 Application of Dynamic Inversion Flight Control. Technical report, Colorado.

Wakker, K. (2015). Fundamentals of Astrodynamics. ISBN: 9789461864192.

Wassel, D. (2013). Exploring novel designs of NLP solvers. PhD thesis, University of Bremen.

Weiland, C. (2014). Aerodynamic Data of Space Vehicles. Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN 978-3-642-54167-4.

Wie, B. (2008). Space Vehicle Dynamics and Control. AIAA education series, second edition.