BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

BERESHEET LUNAR LANDING SIMULATION

SIMULACE PŘISTÁNÍ BERESHEET NA MĚSÍCI

BACHELOR’S THESIS
BAKALÁŘSKÁ PRÁCE

AUTHOR **JAKUB KARPÍŠEK**
AUTOR PRÁCE

SUPERVISOR **doc. Ing. PETER CHUDÝ, Ph.D. MBA**
VEDOUCÍ PRÁCE

BRNO 2020

Brno University of Technology
Faculty of Information Technology

Department of Computer Graphics and Multimedia (DCGM) Academic year 2019/2020

Bachelor's Thesis Specification

22727

Student: **Karpíšek Jakub**
Programme: Information Technology
Title: **Beresheet Lunar Landing Simulation**
Category: Modelling and Simulation

Assignment:

1. Research commercial lunar lander mission: Beresheet.
2. Research lunar landing problem and descent trajectory physics.
3. Perform computation of optimal descent trajectory.
4. Design and implement visualization environment, for which you create or download from relevant sources basic 3D models and 3D engine for descent maneuver interpretation.
5. Evaluate achieved results and discuss potential further improvements.

Recommended literature:

- According to supervisor's recommendations.

Requirements for the first semester:

- Items 1, 2, 3 and partially item 4.

Detailed formal requirements can be found at https://www.fit.vut.cz/study/theses/

Supervisor: **Chudý Peter, doc. Ing., Ph.D. MBA**
Head of Department: Černocký Jan, doc. Dr. Ing.
Beginning of work: November 1, 2019
Submission deadline: May 28, 2020
Approval date: May 15, 2020

Abstract

The goal of this bachelor's thesis was to compute the optimal lunar descent trajectory for the Beresheet spacecraft and develop a visualization environment for interpreting lunar descents. The optimal descent trajectory was computed using Bocop, an optimal control problem solver, and the environment was implemented in Java, using the LibGDX 3D engine. The optimal descent trajectory was found and it enables soft landing with 29.2 kg (6.8%) of fuel left. Created environment enables users to see lunar descent trajectories in an intuitive way and to interact with the visualization.

Abstrakt

Cílem této bakalářské práce bylo vypočítat optimální trajektorii sestupu na Měsíc pro kosmickou loď Beresheet a vyvinout prostředí pro vizualizaci sestupových trajektorií na Měsíc. Optimální trajektorie byla vypočtena použitím programu Bocop, který řeší problémy optimálního řízení, a prostředí bylo vytvořeno v Javě s použitím 3D enginu LibGDX. Byla nalezena optimální trajektorie, která splňuje všechna kritéria použitelnosti, umožňující hladké přistání s palivovou rezervou 29.2 kg (6.8%). Vytvořené prostředí umožňuje uživatelům vidět trajektorii sestupu na Měsíc v intuitivním zobrazení a také jednoduše zasahovat do probíhající vizualizace.

Keywords

Beresheet, lunar landing, simulation, optimal descent trajectory, optimal control, bocop, libgdx

Klíčová slova

Beresheet, přistání na Měsíci, simulace, optimální trajektorie sestupu, optimální řízení, bocop, libgdx

Reference

KARPÍŠEK, Jakub. *Beresheet Lunar Landing Simulation.* Brno, 2020. Bachelor's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor doc. Ing. Peter Chudý, Ph.D. MBA

Rozšířený abstrakt

Na začátku roku 2019 se Izraelská soukromá firma SpaceIL pokusila přistát na Měsíci s kosmickou lodí Beresheet. Byl to velmi sledovaný pokus o přistání, jelikož by se jednalo o první Izraelské přistání na Měsíci a první přistání soukromé firmy vůbec. Mise byla bohužel neúspěšná – při závěrečném sestupu k povrchu došlo k selhání a kosmická loď byla zničena.

Tato bakalářská práce se zabývá výpočtem optimální sestupové trajektorie pro kosmickou loď Beresheet z kruhové oběžné dráhy 210 km nad povrchem Měsíce a návrhem a implementací vizualizačního prostředí pro vizualizaci přistání na Měsíci.

V první části práce byla nastudována mise Beresheet a z dostupných zdrojů byly získány technické parametry kosmické lodi Beresheet a astronomických těles vyskytujících se ve výpočtech – Měsíce a Země. Přistávací manévr byl omezen na pohyb v jedné rovině se třemi stupni volnosti, čímž došlo ke značnému zjednodušení výpočtů. Pro výpočet byly definovány dvě vztažné polární souřadné soustavy – jedna inerciální se středem v těžišti Měsíce a druhá rotující se středem v těžišti kosmické lodi. Trajektorie byla rozdělena na čtyři fáze – deorbitace, plutí, brzdění a vertikální sestup. Toto rozdělení umožnilo definování milníků trajektorie a odlišných omezení jednotlivých stavových a řídicích veličin v každé fázi. Pohybové rovnice byly odvozeny s použitím literatury z Newtonovy klasické mechaniky.

Výpočet optimální sestupové trajektorie byl identifikován jako problém optimálního řízení. Řídicími prvky jsou tah motoru a úhlové zrychlení způsobené systémem řízení náklonu. Tento výpočet byl proveden nástrojem Bocop[1], který je k výpočtu takových problémů určený. V nástroji bylo nutné definovat stavový a řídicí vektor, meze jednotlivých veličin, počáteční a koncové podmínky, konstanty, pohybové rovnice a užitkovou funkci. Jako užitková funkce byla zvolena maximalizace finální hmotnosti kosmické lodi. Výsledná optimální trajektorie splňuje všechny podmínky proveditelnosti, minimalizuje rizika, která by mohla vést k neúspěšné misi a zajišťuje palivovou rezervu 29.2 kg (6.8%) po přistání. Pro kontrolu byly zkoumány alternativní trajektorie, ale žádná lepší nebyla nalezena, což zvyšuje pravděpodobnost, že nalezená trajektorie je optimální. Pro zajímavost byla optimální sestupová trajektorie porovnána se skutečnou trajektorií z mise Beresheet.

Vizualizační prostředí bylo implementováno v Javě s použitím frameworku LibGDX[2], který poskytuje abstrakce grafické knihovny OpenGL a umožňuje jednoduchý vývoj 2D i 3D aplikací. Vyvinutou aplikaci lze spustit na počítači s operačním systémem Linux nebo Windows®. Aplikace umožňuje uživateli nahrát trajektorii a následně vizualizovat let kosmické lodi Beresheet po této trajektorii ve 3D prostředí. Navíc je možné do vizualizace zasahovat a měnit parametry jako čas a rychlost animace, úhel pohledu a další.

Trajektorii lze nahrát jako sled souřadnic ve 3-rozměrném prostoru buď v polární nebo kartézské souřadné soustavě. Je nutné doplnit sled časů, ve kterých se kosmická loď v jednotlivých souřadnicích nachází. Rozšířením je poté možnost nahrání sledu hmotnosti, rychlosti a otáčení v průběhu trajektorie. Tato data jsou pak zobrazena v čase během vizualizace pro lepší přehled o trajektorii. Otáčení je navíc promítnuto i do vizualizace.

Během práce byly vypracovány všechny zadané úkoly a bylo dosaženo uspokojivých výsledků. Výsledná optimální trajektorie minimalizuje jak spotřebu paliva, tak i rizika spojená s misí. Vyvinuté prostředí pro vizualizaci umožňuje pochopení trajektorie intuitivní cestou a naplňuje záměr určený před započetím práce.

[1] http://www.bocop.org/

[2] https://libgdx.badlogicgames.com/index.html

Beresheet Lunar Landing Simulation

Declaration

I hereby declare that this term project was prepared as an original work by the author under the supervision of doc. Ing. Peter Chudý, Ph.D. MBA. I have listed all the literary sources, publications and other sources, which were used during the preparation of this project.

. . . .
Jakub Karpíšek
May 27, 2020

Acknowledgements

I would like to thank doc. Ing. Peter Chudý, Ph.D. MBA. for willingness and valuable advice during work on this assignment. I would also like to thank my wife and parents for their help and support.

Contents

List of Figures

List of tables

Nomenclature

α	Spacecraft angular acceleration
Δv	Change in velocity
$\hat{i}_\theta$	The θ component of rotating polar reference frame unit vector
$\hat{i}_r$	The r component of rotating polar reference frame unit vector
$\hat{i}_x$	The X component of inertial cartesian reference frame unit vector
$\hat{i}_y$	The Y component of inertial cartesian reference frame unit vector
μ_E	Earth's gravitational parameter
μ_M	Moon's gravitational parameter
ω	Spacecraft angular velocity
ψ	Spacecraft pitch angle
θ	Angular coordinate in polar coordinate system
a_θ	Spacecraft acceleration tangential component
a_r	Spacecraft acceleration radial component
I_{sp}	Engine specific impulse
k_E	Spacecraft engine throttle command
m	Spacecraft mass
r	Radial coordinate in polar coordinate system
R_E	Earth's equatorial radius
R_M	Moon's equatorial radius
T_{max}	Spacecraft engine maximum thrust
v_θ	Velocity tangential component
v_r	Velocity radial component
V_{ex}	Engine exhaust velocity

Chapter 1

Introduction

More than sixty years have passed since humans first started to send objects to the outer space. Beginning with the Sputnik 1, which became the first artificial satellite of the Earth in 1957 and Gagarin, who became the first human to orbit the Earth in 1961. In 1966, man managed to soft land a spacecraft on another celestial body for the first time, when the Luna 9 landed on the surface of the Moon. Then in 1969, the Apollo program set a great milestone with placing men on the surface of the Moon for the first time in history, in their Apollo 11 mission. To date, the last human mission to the Moon took place in 1972. After that, the human exploration fast growing trend came to a surprising halt. Since then, over forty successful and unsuccessful uncrewed missions to the Moon took place, including impactors, orbiters and landers, bringing China, Japan, India, ESA[1] and recently even Israel to the scene. Though only China has managed to join USA and USSR in having successfully soft landed on the Moon.

Today, the Moon is receiving more attention again. NASA[2], for example, is planning to land humans on the surface of the Moon again by 2024 in their Artemis program [16], for the first time in fifty-two years, and use the Moon as a springboard for exploring other astronomical bodies. We now find ourselves in an era, when space travel will likely become more usual than ever before. Until recently, the space has been a domain only for government institutions. Now, however, even private companies can be seen in this field, increasing the number of planned space missions even more.

Astrodynamic computations are essential for these missions. Trajectory is one of the main factors that influences spacecraft's proportions and components when it is being designed. Knowing the optimal trajectory minimizes the amount of fuel and life support supplies needed for the mission and enables optimal use of spacecraft's cargo space.

It can be seen, that simulating a lunar landing is a relevant topic. In this work, the optimal descent trajectory of the Israeli Beresheet mission is researched. In addition, a 3D environment is developed for visualization of the resulting optimal descent trajectory. The goal is to find an optimal solution and create an intuitive environment, that would ease the process of understanding the trajectory data.

In chapter number 2, the Beresheet mission is described, including the spacecraft's specifications and launch information. Chapter number 3 presents the Lunar landing problem, descent trajectory phases and orbital mechanics definitions, relevant to the thesis. In chapter number 4, the equations of motion, needed for computing the optimal trajectory,

[1] European Space Agency

[2] National Aeronautics and Space Administration

are derived. Chapter number 5 presents the documentation of optimal descent trajectory computation using Bocop optimal control problem solver, and presents computed results in form of graphs. In chapter number 6 the visualization environment, implemented with the LibGDX 3D engine, is presented and its implementation details are described. In chapter number 7 the reader can find evaluation of achieved results and comparison with real trajectory of the Beresheet mission. Conclusion can be found in chapter number 8. Appendix A shows specifications of the Earth, the Moon and the launch vehicle of the Beresheet mission – the Falcon 9 rocket. Appendix B lists tables with specifications, that would not fit the text continuity. In appendix C the contents of associated media drive are listed.

Chapter 2

Commercial Lunar Lander Mission: Beresheet

This chapter begins with the overview of the Google Lunar XPRIZE competition. Then the Beresheet mission overview is described, along with launch configuration and brief introduction of the launch vehicle. At the end the planned trajectory of the Beresheet mission is described in detail.

2.1 Google Lunar XPRIZE

The XPRIZE is a nonprofit foundation since 1994, which has, to this day, designed and operated seventeen competitions in the domain areas of Space, Oceans, Learning, Health, Energy, Environment, Transportation, Safety and Robotics. Their goal is to inspire and guide scientists, engineers, academics and other innovators to spur new ideas and breakthrough technology, which will create a positive future across a set of domains [29].

In 2007, XPRIZE announced the Google Lunar XPRIZE competition, sponsored by Google, with a mission to

> "spur affordable access to the Moon and give space entrepreneurs a legitimate platform to develop long-term business models around lunar transportation and to inspire the next generation of scientists, engineers, space explorers and adventurers to enter the STEM[1] fields" [30].

Offering a $30-million prize purse to private companies, that would take part and succeed in the program. The objective for the competitors is to:

- Soft-land a craft on the Moon.
- Move 500 m along a deliberate path on or above the surface.
- Transmit two "mooncasts".
- Complete a data uplink exercise as set by XPRIZE.

According to [30], 45 teams have registered to the competition, five of which have become finalists, including Israeli SpaceIL. Unfortunately, none of the competitors was able to make

[1] Science, Technology, Engineering and Mathematics

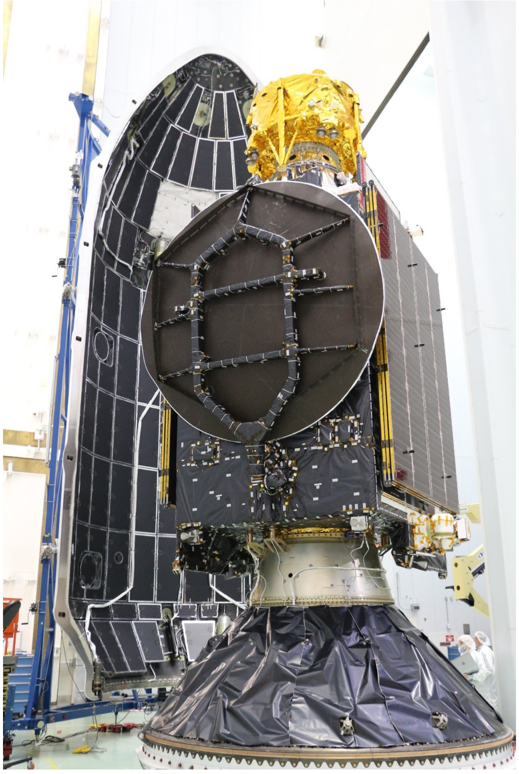


Figure 2.1: The Beresheet spacecraft and the Nusantara Satu launch configuration with a 13.1 meters tall Falcon 9 payload fairing in the background. Taken from [23].

a launch attempt to reach the Moon before the March 31, 2018 deadline and XPRIZE has terminated the program with the prize unclaimed. Regardless, many of the teams have continued to pursue the goal and planned to launch in following years. First of them to launch became SpaceIL on February 22, 2019 when they launched their Beresheet spacecraft on top of the SpaceX Falcon 9 rocket. Unfortunately, on April 11, 2019, the Beresheet crashed into the Moon's surface during a soft landing attempt. Although the Google Lunar XPRIZE was no longer taking place, XPRIZE has decided to recognize SpaceIL's achievement with a $1 million Moonshot award for becoming the first privately funded company to enter the lunar orbit and to attempt a lunar landing [30].

2.2 Mission launch

The Beresheet spacecraft was launched on top of the SpaceX Falcon 9 rocket. It shared the Falcon 9 with the Nusantara Satu Indonesian geostationary communications satellite and United States Air Force Research Laboratory S5 satellite, which was to be deployed by the Nusantara Satu later. The Beresheet was mounted directly on top of the Nusantara Satu, as it can be seen in Figure 2.1. Thanks to this configuration SpaceIL mission designers were able to choose the launch time, because it was possible to meet designated orbit for all three spacecrafts, if proper time of day was selected [22].

Falcon 9 vehicle

The Falcon 9 is a two-stage rocket vehicle powered by liquid oxygen (LOX) and rocket-grade kerosene (RP-1). The Falcon 9 can be flown in two configurations: either with fairing, or with SpaceX Dragon spacecraft. The current version is called Block 5 and is operating since spring 2018. It is 70 meters tall and 3.7 meters wide at its diameter. It is capable to deliver a 22,800 kg payload to LEO[2], 8,300 kg to GTO[3] or 4,020 kg to Mars. The first stage is propulsed by nine Merlin engines with a total thrust of 7,686 kN at lift-off. It is unique by its ability to soft land on ground or on autonomous drone ship in ocean, after its primary mission is done, and be re-used. This ability greatly reduces cost of each launch. The second stage is propulsed by one Merlin Vacuum engine with a thrust of 981 kN in vacuum and this stage is not recovered after mission. Specifications taken from [7]. For picture of the Falcon 9 rocket and more specifications see Appendix A.

Nusantara Satu

The primary payload of the launch was the 4,100-kilogram communication satellite Nusantara Satu. Expected to operate for at least 15 years it will provide communications links to rural parts of Indonesia and expand broadband internet services to this regions. After the Falcon 9 drops the satellite off at a super-GTO, it will use its own engines to reach its final geostationary slot. Once it arrives there, it will deploy its subsatellite, S5, which is a 60-kilogram spacecraft, that will detect and track objects near and in the geostationary belt [10].

Beresheet spacecraft

The Beresheet is a Lunar Lander spacecraft designed to reach the Moon with its own propulsion system, once deployed at Geosynchronous Transfer Orbit (GTO). It is a four-legged circular craft, measuring 1.5 meters in height and 2 meters in diameter. Launch mass is 582 kg with dry vehicle mass of 150 kg. The main propulsion system is the LEROS$^{\mathrm{TM}}$2b engine with 456 N of thrust and 318 s specific impulse. For more specifications of the engine see Table B.1. Additional 8 small thrusters are used for attitude control. The spacecraft is equipped with solar panels on top deck, that provide power. It also carries imagers, magnetometer to study the Moon's magnetic field and a laser retroreflector – a small experimental tool, provided by NASA – intended for testing its navigation capabilities. Taken from [13, 15]. A photograph of the Beresheet spacecraft can be seen in Figure 2.2.

Launch

The Beresheet was launched on February 22, 2019 at 01:45 UTC from Cape Canaveral Air Force Station launch pad SLC 40 in Florida. After about 157 seconds, at about 67 km altitude, the first stage engines were cut off and stage separation occurred. The second stage engine was then ignited and continued to burn for another 319 seconds, which brought the vehicle into LEO, 165 km above the Earth. After coasting for almost 19 minutes, at t+27:03, the second stage engine ignited for the last time and burned for 65 seconds, injecting the satellites into a super-GTO with an apogee of about 60,000 km. The Beresheet deployment

[2] Low Earth Orbit
[3] Geostationary Transfer Orbit

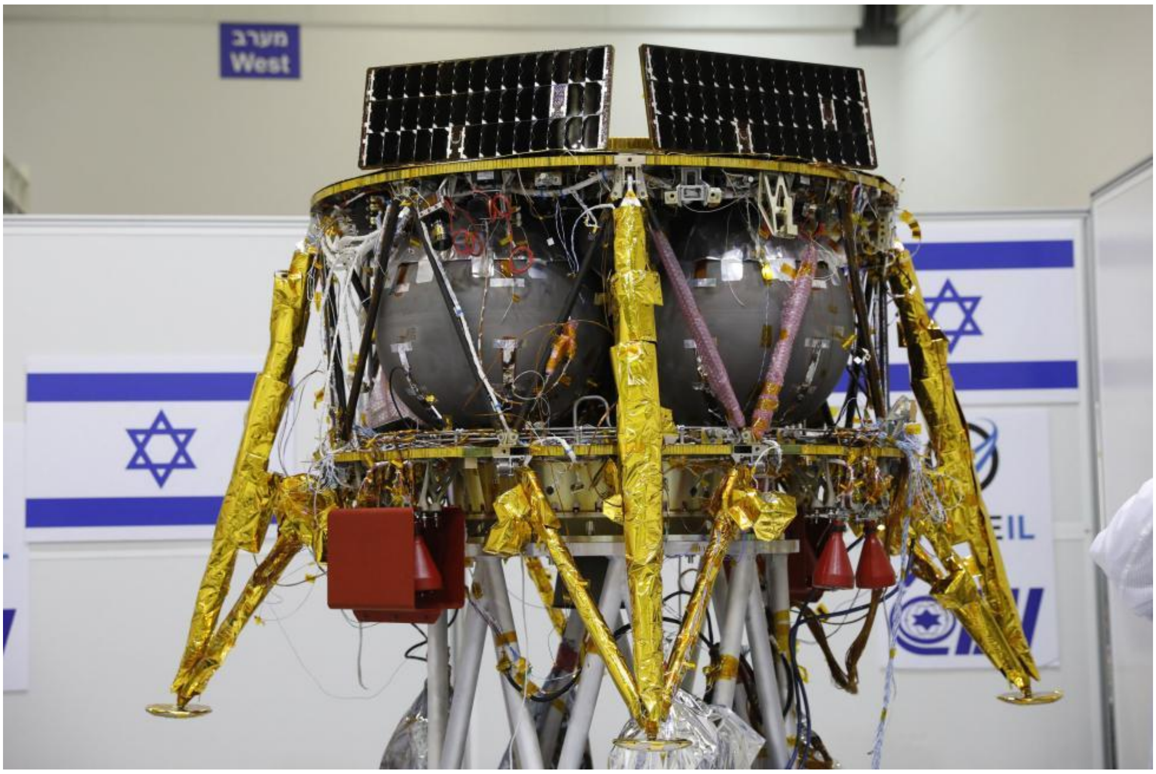


Figure 2.2: The Beresheet spacecraft at advanced stage of development. Taken from [22].

was executed about 6 minutes later at t+33:39 at about 720 km altitude. The Nusantara Satu deployment occurred 11 minutes later at t+44:38 at about 3,250 km altitude [6].

2.3 Mission trajectory

In this section the planned trajectory for the Beresheet spacecraft from super-GTO to lunar orbit is described and compared to the actual as-flown trajectory.

Planned trajectory

The Beresheet mission trajectory was designed to use a series of phasing orbits to accomplish an encounter with the Moon. It included four different apogee altitudes with multiple revolutions at each altitude to precisely time the encounter. The plan included slight changes in maneuvers and number of phasing loops for different launch days to arrive at the Moon on the same day. For launch on February 22, 2019 the spacecraft was planned to spend 4 loops in the initial super-GTO with ~60,000 km apogee, 5 loops in orbit with ~111,000 km apogee, 2 loops in orbit with ~275,000 km apogee and 1.5 loops in orbit with ~390 000 km apogee. Figure 2.3 (left) shows planned phasing loop trajectory.

When the lunar encounter, planned for April 4, 2019, would occur, the plan was to execute a lunar orbit insertion (LOI) maneuver (see Figure 2.4) and capture the spacecraft into a $290 \times 10,000$ km lunar orbit. Over the course of next seven days the Beresheet would perform several maneuvers to reach lunar parking orbit 210 × 210 km. Finally the periselene would be lowered to 15 km, where the terminal descent would begin. The landing was targeted for April 11, 2019 at about 20:00 UTC in Mare Serenitatis [22]. Table 2.1

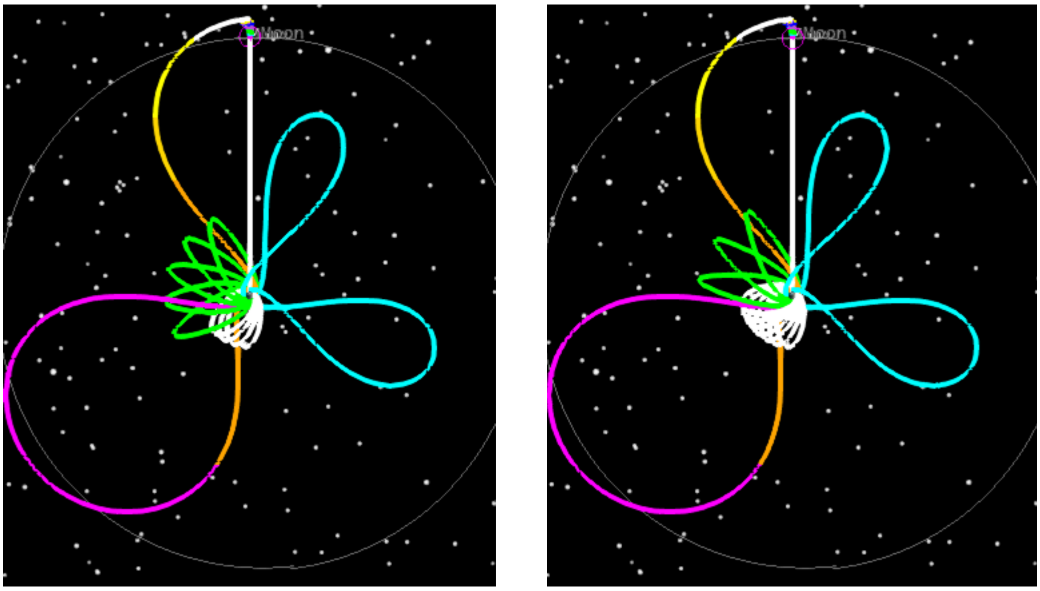


Figure 2.3: The Beresheet phasing loop trajectory planned (left) and actual (right) in the Earth-Moon rotating coordinate frame with the Moon always at the top. Taken from [22].

Table 2.1: Maneuvers as planned before launch. Taken from [22].

		Separation	2/22/2019	70,000	215
	#	**Maneuver**	**Start Time** [UTC]	**ApoAlt aft** [km]	**PeriAlt aft** [km]
Earth Orbit	1	**AM1**	2/22/2019	68,715	215
	2	**AM2**	2/24/2019	68,710	600
	3	**PM1**	2/25/2019	117,300	530
	4	**PM2**	3/07/2019	275,000	1,600
	5	**PM3**	3/20/2019	390,000	1,500
	6	**OPM**	3/21/2019	392,000	1,800
Lunar Orbit	7	**LOI1**	4/04/2019	10,000	290
	8	**LOI1A**	4/05/2019	10,000	285
	9	**LOI2**	4/07/2019	750	285
	10	**LOI2A**	4/08/2019	750	210
	11	**LOI2B**	4/09/2019	210	210
	12	**DM1**	4/10/2019	210	15

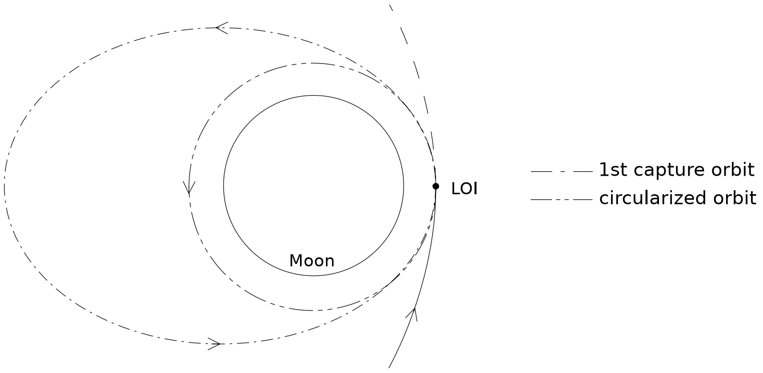


Figure 2.4: Lunar orbit insertion and circularization maneuvers.

shows an overview of planned Beresheet maneuvers. Apogee maneuvers AM1 and AM2 raise perigee. Perigee maneuvers PM1–PM3 raise apogee. Out-of-plane maneuver OPM adjusts the arrival time at LOI. LOI1–LOI2B refer to maneuvers in lunar orbit. DM1 is the descent maneuver.

As-flown trajectory

The actual followed trajectory varied from nominal. The phasing loop trajectory offers good amount of flexibility and can be re-planned when mission anomalies occur. The time and date of lunar encounter is fixed so in case of anomalies orbit periods of phasing loops are adjusted such that the spacecraft arrives at lunar encounter at correct time.

After launch, the AM1 maneuver was waived, because perigee altitude was satisfactory after insertion. The AM2 maneuver was executed as planned and raised perigee to about 650 km. The PM1 maneuver was delayed twice due to spacecraft performance issues, but it did not cause a threat to the mission. The trajectory was adjusted by adding loops in the super-GTO orbit and raising apogee values of subsequent loops, as it can be seen in Figure 2.3 (right).

LOI1 maneuver was executed nominally and the Beresheet was captured into a lunar orbit with 10,000 km apolune height. The rest of LOI maneuvers were executed nominally [22].

Landing

The landing started according to plan on April 11, 2019 and for 11 minutes was nominal. Then a failure was indicated by one of the two inertial measurement units, which led to a series of events including mission computer resets and main engine shutdown. Unfortunately the mission was lost, as the Beresheet impacted the surface at much greater velocity than it could withstand. The last telemetry from the Beresheet sent just before impact showed vertical velocity of 143.3 m/s, horizontal velocity of 946.7 m/s and altitude of 149 meters [22].

Chapter 3

Lunar Landing Problem

The lunar landing problem is defined by first stating the assumptions. Reference frames are then defined for the problem and the descent trajectory phases are introduced. At the end, orbital dynamics, needed for later computations, are presented.

Assumptions

As in [12], pages 41–42, the Moon will be assumed to be spherical, without an atmosphere. It rotates about its axis once in every 27.3217 days [17], which results in surface velocity of approximately 4.6 m/s. This rotation would be a factor if the goal was to target a specific landing site, because the site would move relative to the inertial frame. Since the target is not considered, it is reasonable to neglect the Moon's rotation, because the difference in fuel-usage would be no more than 0.3%.

A purely Newtonian gravity model is used, therefore gravity perturbations due to gravity of the Earth and the Sun are neglected, as well as the perturbations caused by oblateness of the Moon, which is assumed spherical.

Constant vehicle exhaust velocity will be assumed, as the properties of the engine do not change drastically during the maneuvers of lunar landing.

Constant values used for computations, such as the lunar equatorial radius and the lunar gravitational parameter, can be found in Table A.2.

3.1 Lunar Lander in coordinate system

In order to be able to determine the exact position of the Lunar Lander, it must be placed in a frame of reference. For purposes of this thesis, three frames of reference are used: an inertial polar frame of reference for determining the Lunar Lander's position, a rotating polar frame of reference for determining the Lunar Lander's attitude and an inertial cartesian frame of reference. The first two are used, when the optimal descent trajectory is being computed, because it provides mathematical simplifications for that task. The last one is used for displaying the descent trajectory in the developed visualization environment, as the LibGDX 3D engine uses cartesian coordinate system.

The Lunar Lander is simplified to a point mass. The motion is restricted to three degrees of motion (single plane), because the trajectory remains in one plane, when optimal solution is researched. This will simplify the computations. The x-y plane was selected as the plane of motion.

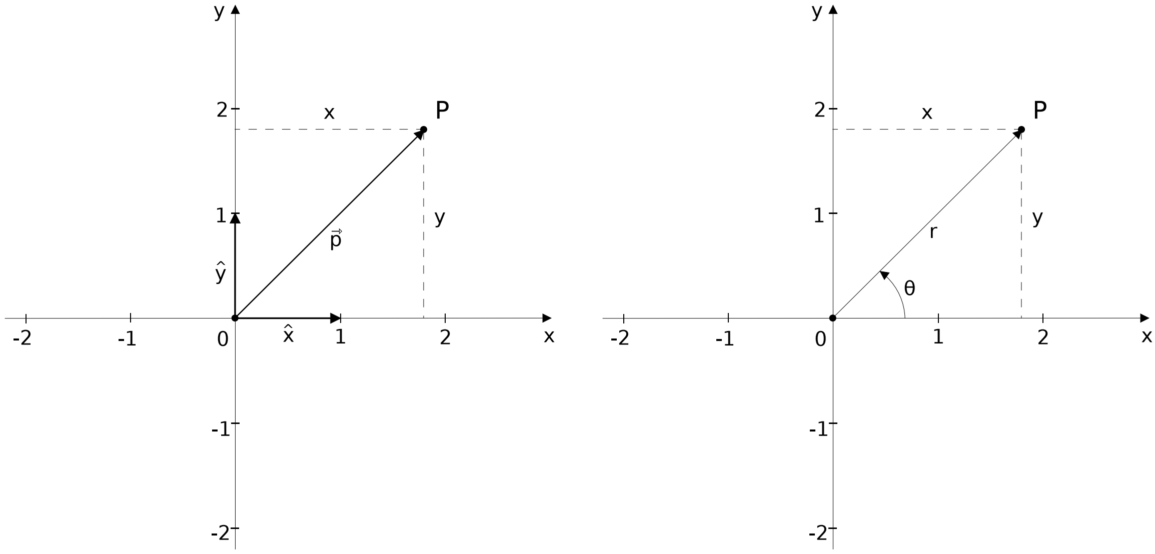


Figure 3.1: Cartesian coordinate system (left) and polar coordinate system (right).

Cartesian coordinate system

The cartesian coordinate system, (illustrated in Figure 3.1), is a system of n axes, which are mutually perpendicular. Two-dimensional coordinate system has two axis, usually denoted as x and y. The x and y coordinates may lie anywhere in the interval $(-\infty, \infty)$. Ordered pair of (x, y) is called a vector. The system is defined by an origin $(0, 0)$ and two unit vectors – $\hat{x}$ and $\hat{y}$. Position vector $\vec{p}$ is defined as:

$$\vec{p} = x\hat{x} + y\hat{y} \tag{3.1}$$

where x and y are the values of perpendicular projection from the point P to the axis [26].

Polar coordinate system

The polar coordinate system, (illustrated in Figure 3.1), is a system where a point P in space is represented as an ordered pair (r, θ), where r is the radial distance from origin and θ is the counterclockwise angle between positive part of x–axis and the vector going from the origin to the point P [27]. The r and θ lie within the intervals:

$$r \in \langle 0, \infty \rangle, \theta \in \langle 0, 2\pi \rangle \tag{3.2}$$

Conversion between cartesian and polar coordinate systems is made using following equations:

$$x = r \cdot \cos\theta \tag{3.3}$$

$$y = r \cdot \sin\theta \tag{3.4}$$

Frames of reference

The inertial polar frame of reference is defined by unit vectors $\hat{i}_x$ and $\hat{i}_y$, with origin in the Moon's center of gravity G_M. The rotating frame of reference shares origin with the inertial frame and is defined by unit vectors $\hat{i}_r$ and $\hat{i}_\theta$, while $\hat{i}_r$ always points at the direction of Lunar Lander's center of mass and $\hat{i}_\theta$ remains perpendicular to $\hat{i}_r$. It is referred to as "rotating polar frame", because it uses polar coordinate system and rotates. It rotates with the same rate the Lunar Lander LL revolves around the Moon. Both frames of reference are illustrated in Figure 3.2. The position vector $\vec{r}$ of the Lunar Lander is defined in inertial and rotating frames respectively:

$$\vec{r} = \sqrt{(x\hat{i}_x)^2 + (y\hat{i}_y)^2} \tag{3.5}$$

$$\vec{r} = r\hat{i}_r \tag{3.6}$$

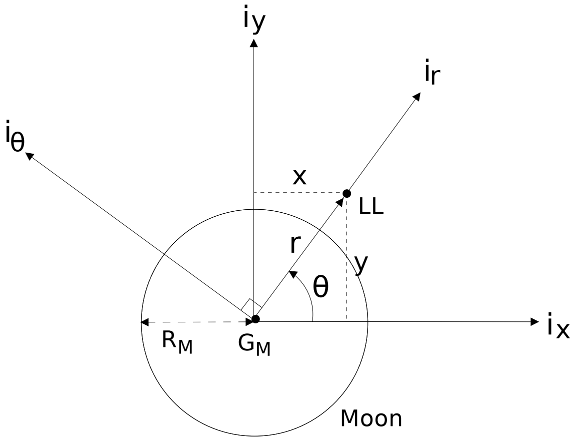


Figure 3.2: Two-dimensional inertial and rotating polar frames of reference.

Beresheet spacecraft attitude

The Beresheet spacecraft is free to rotate in three dimensions. According to aircraft principles these rotations are called *roll, pitch* and *yaw* and are rotations about x, y and z axes respectively. The x, y and z axes originate in spacecraft's center of mass. The Beresheet's rotation axes are demonstrated in Figure 3.3. For purposes of this thesis the Beresheet's rotation was restricted only to its y axis.

3.2 Descent trajectory definition

The descent trajectory computation in this thesis begins at a lunar parking orbit 210 × 210 km, in other words 210 km above the Moon's surface. It is convenient to partition the descent into four phases: *de-orbit, coast, brake* and *vertical descent phase*, as suggested in [12], page 124. Partitioning enables to propose different constraints for each phase and makes the descent profile safer with bigger margin for errors. Illustration of the descent phases can be seen in Figure 3.4.

De-orbit phase

The descent begins with a de-orbit burn. The objective of de-orbit phase is to place the Lunar Lander on an elliptical orbit with a perilune altitude 15 km above the Moon's surface. As researched in [12], the chosen value 15 km is reasonable for couple reasons: On one hand, the lower the targeted perilune is, the more fuel is saved, but on the other hand, the higher

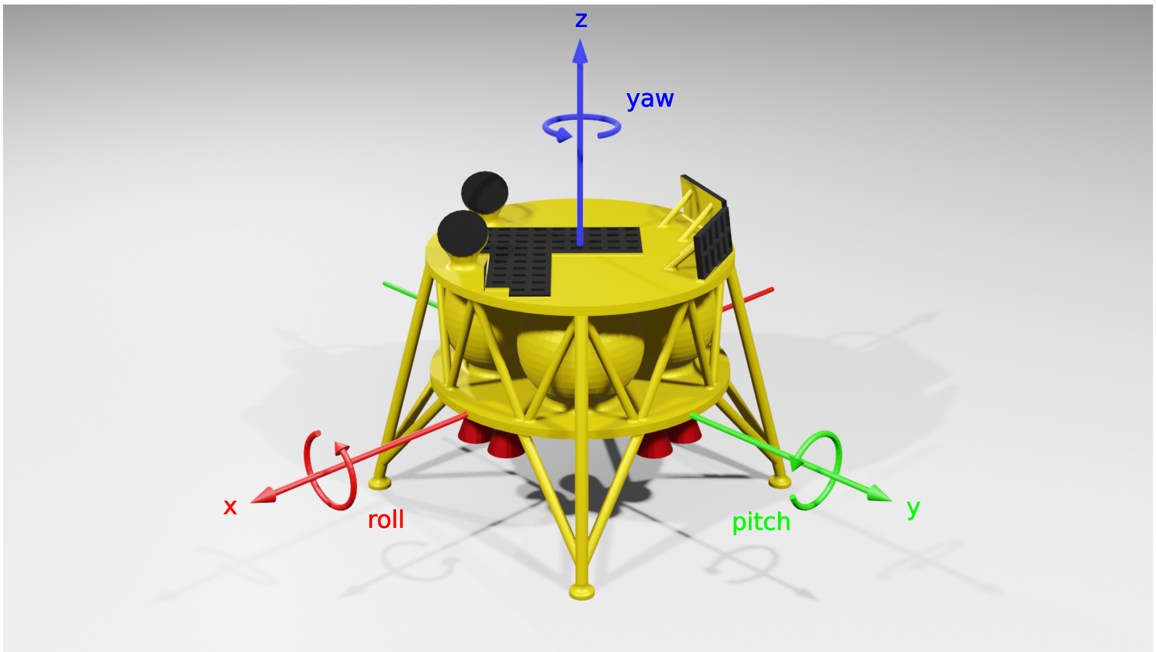


Figure 3.3: The Beresheet rotation axes demonstrated on the 3D model used in the developed visualization application. 3D model taken from [20].

the targeted perilune is, the safer the coast is. Perigee lower than 15 km could result in a collision with mountainous terrain. It is desired, that the Lunar Lander has an option to safely fly by, if, for any reason, an error occurs during the coast phase or braking phase initiation. Minimizing the time, the spacecraft is on a collision course with the Moon, increases mission success chances.

Coast phase

The coast phase begins with shutting the main engine down. During this phase, the altitude is lowered from 210 km to 15 km, while main engine remains shut down. For braking purposes, the spacecraft needs to arrive at the end of this phase in engine-first position.

Braking phase

The braking phase starts near perilune and its main task is to greatly decrease the Lunar Lander's velocity and bring it just 500 m above the Moon's surface in vertical legs-down position, so the Lunar Lander is prepared to soft land.

Vertical descent phase

The objective of vertical descent phase is to slowly descent, while measuring its altitude. At start of this phase the Lunar Lander is already moving slow enough to be able to land right away in case of wrong measurements. In this thesis it is assumed that the measurements are correct and the Beresheet will soft land at planned altitude.

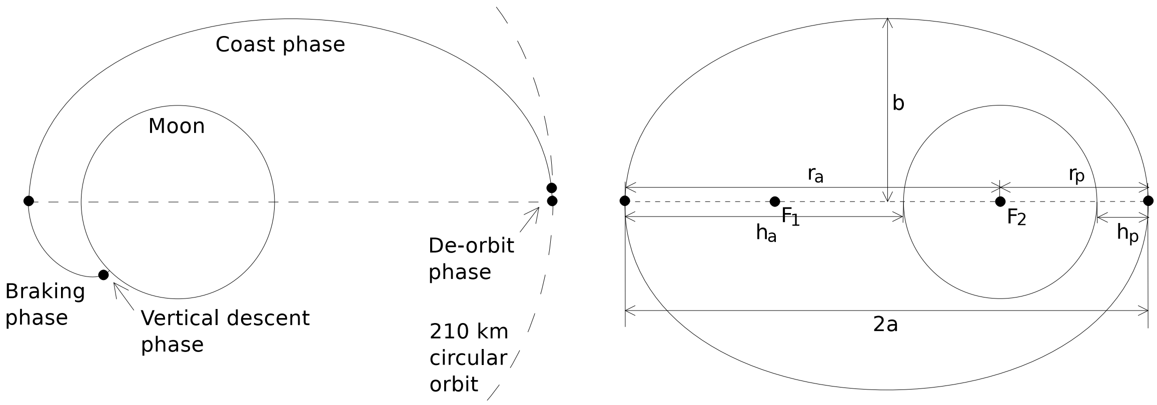


Figure 3.4: The four descent phases (left) and orbital dynamics definitions (right).

3.3 Orbital dynamics definitions

When a spacecraft is orbiting a celestial body in an elliptical orbit, the main body occupies one of the focus points of that ellipse, called occupied focus. The spacecraft's state can be described by a following set of variables: radius r, central angle θ, and velocity, defined by its radial v_r and tangential v_θ components. The maximum width of ellipse is major axis, and its length is $2a$. The maximum width in direction perpendicular to major axis is minor axis, with length $2b$. The closest orbit approach point the to the occupied focus is called periapsis. In case of the Moon this point is called *perilune*. The vector from the occupied focus to perilune is the perilune radius vector, r_p, and the distance from perilune to the Moon's surface will be referred to as perilune altitude, h_p. The farthest orbit point is apoapsis and for the Moon is called apolune. Similarly we define apolune radius vector, r_a, and apolune altitude, h_a. Defined variables can also be seen in Figure 3.4.

Orbital period

The orbital period is computed using the following equation from [4], page 16:

$$T = 2\pi\sqrt{\frac{a^3}{\mu}} \tag{3.7}$$

where a is the length of semi-major axis and μ is gravitational parameter of main body.

Orbital velocity

The orbital velocity on elliptical orbit is computed using following equation from [11], page 6:

$$v = \sqrt{\mu\left(\frac{2}{r} - \frac{1}{a}\right)} \tag{3.8}$$

which can be, for circular orbit, simplified to:

$$v = \sqrt{\frac{\mu}{r}} \tag{3.9}$$

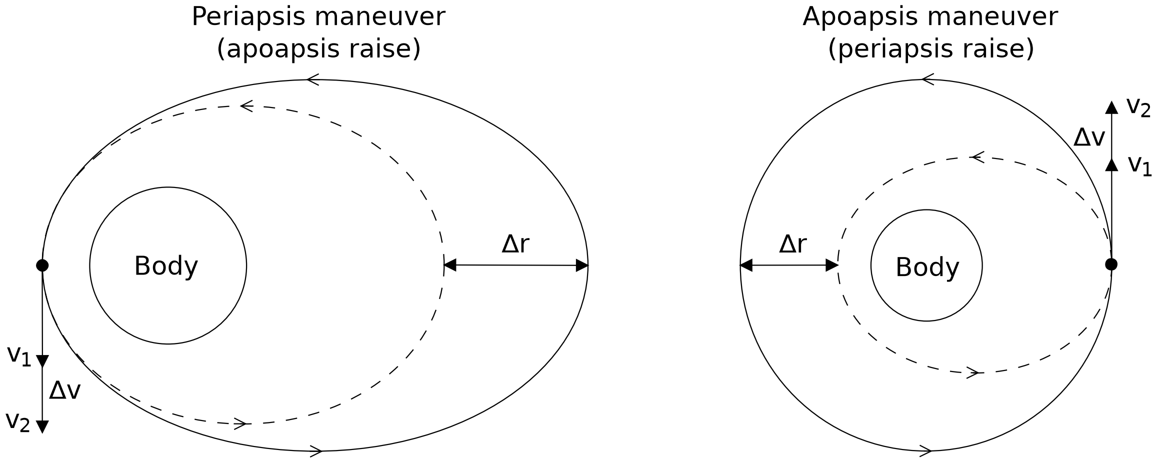


Figure 3.5: Examples of orbital maneuvers.

Orbital maneuvers

An efficient way of changing periapsis or apoapsis radius is to provide velocity increment at the opposite apsis. When the variation Δr of apsis radius is known, new length of semi–major axis can be computed as $a_2 = a_1 + \Delta r/2$ and subsequently the required Δv can be computed as:

$$\Delta v \quad = \quad v_2 - v_1 \tag{3.10}$$

where v_1 and v_2 represent the spacecraft's velocity before and after the burn respectively. Examples of orbital maneuvers are illustrated in Figure 3.5.

The relationship between velocity change, caused by thrust, and mass can be, according to [12], expressed as an integral of thrust acceleration magnitude:

$$\Delta v \quad = \quad \int_{t_0}^{t_f} \frac{T_{max} k_E}{m} dt \tag{3.11}$$

where m represents spacecraft's mass, T_{max} spacecraft's maximum thrust and and k_E spacecraft's thrust command. $k_E \in \langle 0, 1 \rangle \wedge k_E \in \mathbb{R}$.

Periapsis altitude

Being able to compute periapsis altitude is necessary for knowing when to terminate the de-orbit burn. To compute periapsis radius with knowledge about spacecraft's current state, a following equation from [12], page 104, is used:

$$r_p = \frac{r^2 v_\theta^2}{\mu \left(1 + \sqrt{\left(\frac{r v_\theta^2}{\mu} - 1 \right)^2 + \left(\frac{r v_r v_\theta}{\mu} \right)^2} \right)} \tag{3.12}$$

where v_r and v_θ are spacecraft's vertical and tangential velocity components respectively. Computing the periapsis altitude is then rather simple:

$$h_p = r_p - R_{eq} \tag{3.13}$$

where R_{eq} is the equatorial radius of the main body.

Chapter 4

Descent Trajectory Physics

Physical description of the problem, needed for computing the optimal descent trajectory, is derived in this chapter. First, the Newton's laws of motion are presented, then the derivation steps are described, and in the end, the equations of motion are introduced.

4.1 Newton's laws of motion

The three Newton's laws of motion and the law of universal gravitation are used as a base for deriving the equations of motion. Rest of the section is taken from [4], page 3.

First law

> *Every object continues in its state of rest or of uniform motion in a straight line unless it is compelled to change that state by forces impressed upon it.*

The first law requires the identification of an inertial system with respect to which it is possible to define the absolute motion of the object.

Second law

> *The rate of change of momentum is proportional to the force impressed and is in the same direction as that force.*

It can be expressed as:

$$\vec{F} = \frac{d\vec{p}}{dt} \tag{4.1}$$

where $\vec{F}$ is the resultant of the forces acting on the object and $p = m\vec{v}$ is its momentum. For a constant-mass system, it is:

$$\vec{F} = m\vec{a} \tag{4.2}$$

and for a variable-mass system, it is:

$$\vec{F} = m\vec{a} + \dot{m}\vec{v} \tag{4.3}$$

where $\vec{a}$ is the acceleration of the mass measured in an inertial frame of reference.

Third law

> *To every action there is always opposed an equal reaction.*

Newton's law of universal gravitation

> *Two bodies, the masses of which are M and m, respectively, attract one another along the line joining them with a force proportional to the product of their masses and inversely proportional to the square of the distance between them.*

$$F = G\frac{Mm}{r^2} \tag{4.4}$$

where G is the universal gravitational constant and r mutual distance.

4.2 Derivation of the equations of motion

The Moon landing can be simplified to a two-body problem, where the two bodies are the Moon and the Lunar Lander. These bodies interact with each other and the equations of motion can be derived from Newton's laws of motion. Section inspired by [4], pages 5–13.

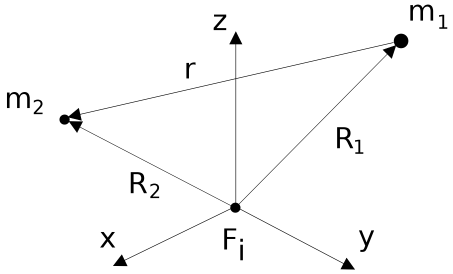


Figure 4.1: Two-body system in an inertial frame of reference. Taken from [4], page 4.

Having a system of two bodies of masses m_1 and m_2, $m_1 > m_2$, where the bodies are spherically symmetrical and no other forces than gravitational are present, the position of m_2 relative to m_1 can be described as:

$$\vec{r} = \vec{R_2} - \vec{R_1} \tag{4.5}$$

where R_1 and R_2 describe the positions of m_1 and m_2 respectively in an inertial frame $\mathbf{F}_i$ (Figure 4.1). Their mutual distance is $r = \|\vec{r}\|$ and following equations describe their mutual velocity and acceleration:

$$\dot{\vec{r}} = \dot{\vec{R_2}} - \dot{\vec{R_1}} \tag{4.6}$$

$$\ddot{\vec{r}} = \ddot{\vec{R_2}} - \ddot{\vec{R_1}} \tag{4.7}$$

Using Newton's second law of motion (eq. 4.2) and law of universal gravitation (eq. 4.4), and considering that $\vec{a} \Leftrightarrow \ddot{\vec{R}}$, the motion of each mass can be described as:

$$m_1\ddot{\vec{R_1}} = G\frac{m_1 m_2}{r^3}\vec{r} \tag{4.8}$$

$$m_2\ddot{\vec{R_2}} = -G\frac{m_1 m_2}{r^3}\vec{r} \tag{4.9}$$

By subtracting eq. 4.9 from eq. 4.8, one obtains:

$$\ddot{\vec{r}} \quad = \quad -G\frac{m_1 + m_2}{r^3}\vec{r} \tag{4.10}$$

As stated in [4], since one of the masses is several orders of magnitude larger than other, it is possible to let M be the mass of the Moon and m be the mass of the Lunar Lander, $m_1 = M$ and $m_2 = m$, with $M \gg m$. and simplify the equation of relative motion 4.10 to:

$$\ddot{\vec{r}} \quad = \quad -\frac{\mu}{r^3}\vec{r} \tag{4.11}$$

where $\mu = GM$ is the primary body gravitational parameter.

The unit vectors of rotating polar frame, relative to inertial frame, are defined as:

$$\hat{i}_r \quad = \quad \cos(\theta)\hat{i}_x + \sin(\theta)\hat{i}_y \tag{4.12}$$

$$\hat{i}_\theta \quad = \quad -\sin(\theta)\hat{i}_x + \cos(\theta)\hat{i}_y \tag{4.13}$$

taking the time derivative, one gets:

$$\dot{\hat{i}}_r \quad = \quad \dot{\theta}\hat{i}_\theta \tag{4.14}$$

$$\dot{\hat{i}}_\theta \quad = \quad -\dot{\theta}\hat{i}_r \tag{4.15}$$

The position vector is defined simply as:

$$\vec{r} \quad = \quad r\hat{i}_r \tag{4.16}$$

The velocity vector is defined as a sum of its radial and tangential components:

$$\vec{v} \quad = \quad v_r\hat{i}_r + v_\theta\hat{i}_\theta \tag{4.17}$$

By computing the first derivative of position vector, and applying equations 4.16, 4.14 and 4.17:

$$\vec{v} = \frac{d\vec{r}}{dt} = \frac{d}{dt}(r\hat{i}_r) = \dot{r}\hat{i}_r + r\dot{\hat{i}}_r = \dot{r}\hat{i}_r + r\dot{\theta}\hat{i}_\theta \tag{4.18}$$

the radial and tangential components of the velocity vector can be defined as:

$$v_r \quad = \quad \dot{r} \tag{4.19}$$

$$v_\theta \quad = \quad r\dot{\theta} \tag{4.20}$$

In a similar way by computing the first derivative of the velocity vector:

$$\vec{a} = \frac{d\vec{v}}{dt} = \frac{d^2\vec{r}}{dt} = (\ddot{r} - r\dot{\theta})\hat{i}_r + (r\ddot{\theta} + 2\dot{r}\dot{\theta})\hat{i}_\theta \tag{4.21}$$

the radial and tangential components of the acceleration vector can be defined as:

$$a_r \quad = \quad \ddot{r} - r\dot{\theta} \tag{4.22}$$

$$a_\theta \quad = \quad r\ddot{\theta} + 2\dot{r}\dot{\theta} \tag{4.23}$$

As stated in [4], page 12, when there are other forces acting upon the spacecraft besides the gravitational pull from the primary body, such as thrust of the engine in this case, and

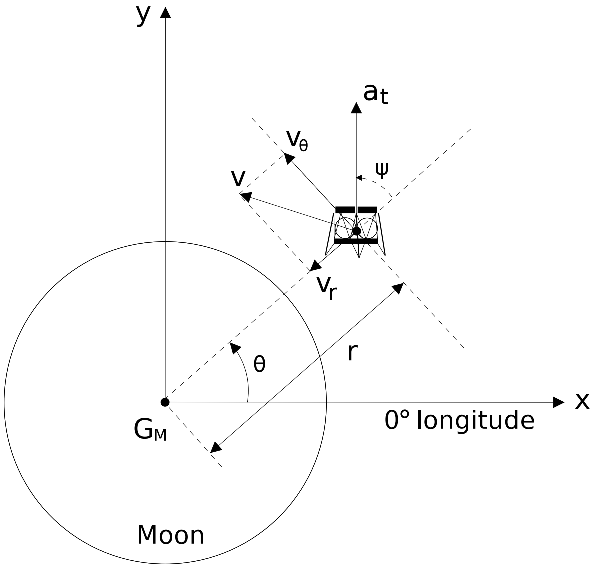


Figure 4.2: The Lunar Lander in the frame of reference.

gravitational perturbations due to other celestial bodies are neglected, the equation 4.11 can be extended to:

$$\ddot{\vec{r}} \quad = \quad -\frac{\mu}{r^3}\vec{r} + \vec{a}_t \tag{4.24}$$

where $\vec{a}_t$ is the acceleration caused by engine thrust (see Figure 4.2). The $\vec{a}_t$ can be split into its components using equations 4.22 and 4.23:

$$a_r \quad = \quad -\frac{\mu}{r^2} + a_t \cos\psi \tag{4.25}$$

$$a_\theta \quad = \quad a_t \sin\psi \tag{4.26}$$

where ψ is the spacecraft pitch angle, defined as the counterclockwise angle between axis i_r and acceleration vector $\vec{a}_t$ (see Figure 4.2).

As stated in [12], page 54, with the use of Newton's second law of motion (eq. 4.2), the acceleration vector can be described as:

$$\vec{a}_t \quad = \quad \frac{T_{max}k_E}{m} \tag{4.27}$$

where T_{max} is the maximum engine thrust, k is the engine throttle command and m is the Lunar Lander mass.

Remembering that:

$$v_r = \dot{r} \quad \Rightarrow \quad \dot{v}_r = \ddot{r} \tag{4.28}$$

$$v_\theta = r\dot{\theta} \quad \Rightarrow \quad \dot{v}_\theta = \dot{r}\dot{\theta} + r\ddot{\theta} \tag{4.29}$$

the system of first-order ordinary differential equations, describing the evolution of the state of the spacecraft, can be defined. It can be found in the following section.

4.3 The equations of motion

Equations presented in section below are used for the computation of the optimal descent trajectory.

Translational dynamics of motion in polar form

In polar form, the position of the spacecraft is defined by r, the distance from main body's center, and θ, the central angle (see Figure 4.2). The velocity of the spacecraft is defined by its radial and tangential components, v_r and v_θ. The translational dynamics, as presented in [12], page 57, are then governed by:

$$\dot{r} = v_r \tag{4.30}$$

$$\dot{\theta} = \frac{v_\theta}{r} \tag{4.31}$$

$$\dot{v}_r = \frac{v_\theta^2}{r} - \frac{\mu}{r^2} + \frac{T_{max}k_E}{m}\cos\psi \tag{4.32}$$

$$\dot{v}_\theta = -\frac{v_r v_\theta}{r} + \frac{T_{max}k_E}{m}\sin\psi \tag{4.33}$$

Last two equations were derived using equations 4.25 and 4.27. Terms $\frac{v_\theta^2}{r}$ and $\frac{\mu}{r^2}$ represent centrifugal and gravitational acceleration respectively. Fractions with goniometric functions represent acceleration caused by engine thrust.

Rotational kinematics

As stated in [12], page 58, in cases where the vehicle rotates about an axis, which is normal to the plane of translational motion, the rotational equations of motion can be described as:

$$\dot{\psi} = \omega - \frac{v_\theta}{r} \tag{4.34}$$

$$\dot{\omega} = \alpha \tag{4.35}$$

where ω is the inertial angular velocity and α the inertial angular acceleration, which can be regulated by firing the attitude control system (ACS) thrusters.

Loss of mass

Finally the mass flow needs to be accounted for. Mass flow due to main engine firing is significantly larger than mass flow due to ACS firing. According to [12], page 58, the ACS can thus be neglected in the mass flow equation and an indirect method of accounting for ACS firing is presented later during implementation of objective function (section 5.3). The mass flow equation is:

$$\dot{m} = -\frac{T_{max}k}{V_{ex}} \tag{4.36}$$

Chapter 5

Optimal Descent Trajectory Computation

The optimal descent trajectory is computed in this chapter. First, the optimal control solver Bocop is presented. Then the optimization problem theory and methods for solving these problems are discussed. Follows description of implementation of the problem in Bocop and presentation of the results.

5.1 Optimal control problem solver: Bocop

The Bocop[1] is an open-source toolbox for solving optimal control problems. It implements a local optimization method. The direct approach transforms optimal control problem into a finite dimensional optimization problem (NLP). This is achieved by discretization in time applied to the state and control variables and the dynamics equation. Taken from [2], page 6. An optimal control problem in Bocop is defined as:

$$(P)\left\{\begin{array}{lr} MinJ(t_0, y(t_0), t_f, y(t_f), \pi) & \text{Objective} \\ \dot{y}(t) = f(t, u(t), y(t), y(t), \pi) & \text{Dynamics} \\ \Phi_l \leq \Phi(t_0, y(t_0), t_f, y(t_f), \pi) \leq \Phi_u & \text{Boundary conditions} \\ y_l \leq y \leq y_u, u_l \leq u \leq u_u, z_l \leq z \leq z_u, \pi_l \leq \pi \leq \pi_u & \text{Bounds} \\ g_l \leq g(t, u(t), y(t), z(t), \pi) \leq g_u & \text{Path constraints} \end{array}\right.$$

where $y(\cdot)$ represents the state variables, $u(\cdot)$ the control, $z(\cdot)$ the optional algebraic variables and π the optional variables to be optimized. Taken from [2], page 10.

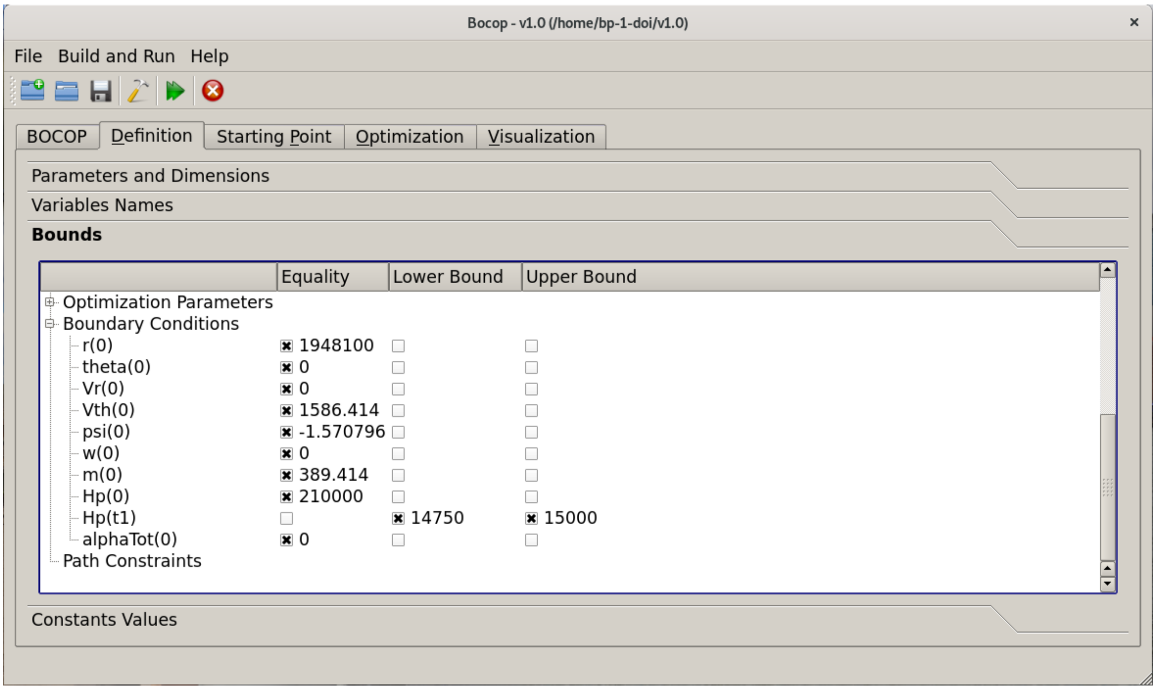
Bocop GUI

The Bocop features a user-friendly GUI (see Figure 5.1) for easier manipulation. There are four tabs, which lead the user through the process of defining and solving the problem.

Definition tab enables user to define the optimal control problem by providing dimensions and names for variables and constants and setting bounds, initial and final conditions to variables. It also includes a choice of discretization method, where in this thesis the midpoint method was selected.

Starting Point enables user to set a starting point for the discretized NLP problem, which may strongly influence the result, since the solver uses a local method.

[1] http://www.bocop.org/



	Equality	Lower Bound	Upper Bound
Optimization Parameters			
Boundary Conditions			
r(0)	☒ 1948100	□	□
theta(0)	☒ 0	□	□
Vr(0)	☒ 0	□	□
Vth(0)	☒ 1586.414	□	□
psi(0)	☒ -1.570796	□	□
w(0)	☒ 0	□	□
m(0)	☒ 389.414	□	□
Hp(0)	☒ 210000	□	□
Hp(t1)	□	☒ 14750	☒ 15000
alphaTot(0)	☒ 0	□	□
Path Constraints			

Figure 5.1: The Bocop GUI in a definition tab and boundary conditions settings.

Optimization has various options settings for the NLP solver, for example batch optimization or choosing existing solution as a starting point.

Visualization finally enables user to preview the contents of a solution file, generated after optimization, in graphical representation, such as state and control variables in time. Taken from [2], pages 7–8.

As stated in [2], page 10, the problem P in Bocop is defined by:

- **4 functions (C/C++)** corresponding to J, f, Φ, g:
  `criterion`, `dynamics`, `boundarycond` and `pathcond`.

- **3 definition files in plain text**, that are automatically filled when using the GUI:
  `problem.def` for general definition and settings
  `problem.bounds` for setting bounds
  `problem.constants` contains optional constant variables and values.

At first, in *Definition*, the state vector from equations 4.30–4.36 is defined:

$$y(t) = [r(t), \theta(t), v_r(t), v_\theta(t), \psi^r(t), \omega(t), m(t)] \tag{5.1}$$

and then the control vector is defined:

$$u(t) = [k_E(t), \alpha(t)] \tag{5.2}$$

5.2 Optimization problem theory

Optimization is the act of achieving the best possible result under given circumstances. Decisions are made in order to either minimize effort or maximize benefit. The effort or the benefit can usually be described as a function of certain variables. Optimization is the process of finding the maximum or minimum value of that function. If a point x^* corresponds to the minimum value of a function $f(x)$, it also corresponds to the maximum value of a function $-f(x)$. Optimization can be thus taken as minimization [1], page 2.

Optimization problem statement

An optimization problem is in [1], page 3, stated as:

Find

$$x = \left(x^1, x^2, \ldots, x^n\right) \tag{5.3}$$

which minimizes

$$f(x) \tag{5.4}$$

subject to the constraints

$$g_j(x) \leq 0 \tag{5.5}$$

for $j = 1, \ldots, m$, and

$$l_j(x) = 0 \tag{5.6}$$

for $j = 1, \ldots, p$.
The vector x is variable, $f(x)$ is the objective function, $g_j(x)$ are the inequality constraints and $l_j(x)$ are the equality constraints. The number of variables n and the number of constraints $p+m$ need not to be related. If $p+m = 0$ the problem is called an unconstrained optimization problem.

Objective function

If the objective function is globally convex, then found local minimum is also global minimum. If the function is non convex, found local minimum is not necessarily the global minimum. The objective function for the optimal descent trajectory problem is nonconvex. Thus more optimal solutions exist, one of which is global optimum, other are local optimum solutions [12], page 22. A graphical representation of this phenomenon is shown in Figure 5.2.

There is a **local minimum** at point x^* if $\varepsilon > 0$ exists such that $f(x^*) \leq f(x)$ for all x within interval $\langle x^* - \varepsilon, x^* + \varepsilon\rangle$. There is a **global minimum** at point x^* if $f(x^*) \leq f(x)$ for all x within X, where X is the domain of x.

Finding a minimum

A local minimum of a single-variable function can be found by computing its first and second derivative. If a function has local minimum at x^*, then the first derivative at that point $f'(x^*)$ must either be equal to 0 or be non–existing. That point would be called a *critical point*. Critical point can correspond to any local extremum, therefore the *Second Derivative Test* needs to be performed. The test states that if $f''(x^*) > 0$, there is local minimum at point x^*.

In multivariable case a point x^* is *critical*, if the gradient is equal to zero: $\nabla f(x^*) = 0$. The *Second Derivative Test* is now generalized: first, the *Hessian* is formed, which is a matrix of second partial derivatives at x^*. For a function of n variables, the *Hessian* is a $n \times n$ matrix H, and the entry in i-th column and j-th row of H is

$$H_{ij} = \frac{\delta^2 f}{\delta x_i \delta x_j}(x^*) \tag{5.7}$$

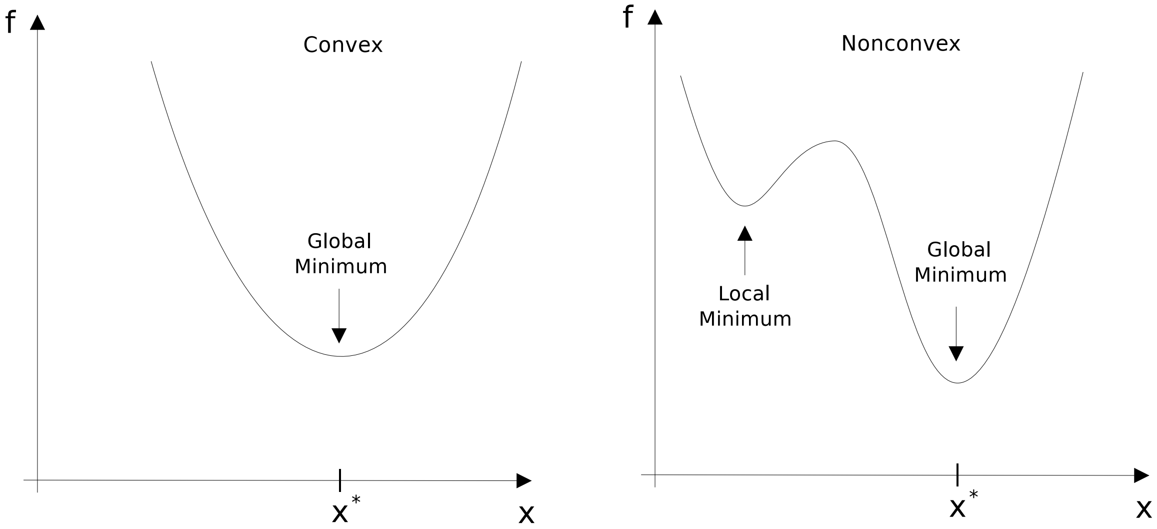


Figure 5.2: Global convexity of function. Taken from [12], page 22.

The multi-variable *Second Derivatives Test* is now stated as: If x^* is a critical point of f and the *Hessian* H is *positive definite*, then x^* is local minimum of f. A *positive definite* matrix is symmetrical and its determinant and eigenvalues are positive. From [14].

Optimal control problem

Optimization problems, that include input or control, fall into a subset called optimal control problems. Therefore finding the optimal trajectory is an optimal control problem with the control being thrust and attitude control system.

Let $x(t) \in \mathbb{R}^n$ be the state of continuous system and $u(t) \in \mathbb{R}^m$ be the control. The system is governed in time by a following differential equation:

$$\dot{x}(t) = f(x(t), u(t), t) \tag{5.8}$$

To solve the optimal control problem, we are trying to find such history of control $u(t)$, that minimizes the objective function, which can be defined as

$$J = \int_{t_0}^{t_f} f(x(t), u(t), t)dt \tag{5.9}$$

where t_0 and t_f are beginning and end of the problem respectively [12], pages 25–26.

Solving optimal control problem

Optimal control problems can be solved either analytically or numerically. The analytical solution is most of the times very difficult or impossible. Therefore the latter is used commonly. Numerical methods are divided into two categories: direct and indirect. *Indirect* methods use estimate of the costate which may be also very difficult or impossible to make. *Direct* methods optimize the objective function directly, and are used most often. These methods start from an initial guess of the state and control and proceed to find a local minimum of the objective function. If the problem is proven to be convex, then the found local minimum is also global minimum. It has to be kept in mind, that if the problem is

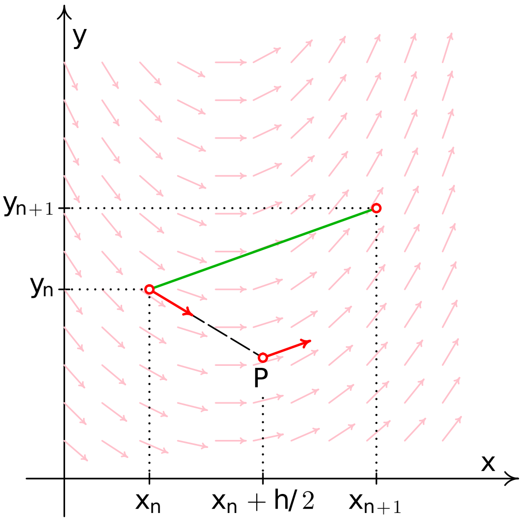


Figure 5.3: Illustration of midpoint method. Taken from [8].

highly nonconvex, a good initial guess is crucial in finding good local optimum solution. A direct method is also used in this thesis [12], pages 28–29.

Midpoint method

The used optimal control problem solver offers a selection between multiple methods for solving differential equations. In this thesis the midpoint method was selected.

It is a modification of the *Euler's method*, that is one-step, which means the new state is determined only by the state in previous step. Its objective is to solve following differential equation:

$$\dot{y} = f(x, y), \quad y(x_0) = y_0 \tag{5.10}$$

where y is function of x. Following equations are used in midpoint method:

$$k_1 = f(x_i, y_i) \tag{5.11}$$

$$k_2 = f(x_i + \frac{1}{2}h, y_i + \frac{1}{2}hk_1) \tag{5.12}$$

$$y_{i+1} = y_i + hk_2 \tag{5.13}$$

where h is size of the step and $i = 0, \ldots, n - 1$.

Midpoint method is geometrically represented in Figure 5.3. First, the point P (*midpoint*) is found by computing the derivative at $[x_n, y_n]$ and finding a point in this direction at $x_n + h/2$. Approximate value at x_{n+1} can be then obtained by using the derivative at P as a direction to head from $[x_n, y_n]$. Red arrows represent directional field. Taken from [8], page 149.

5.3 Optimal control problem implementation

The problem implementation in Bocop is described in this section.

Trajectory segmentation

The trajectory is segmented into four phases as mentioned before: *de-orbit, coast, brake* and *vertical descent*. To compute the optimal descent trajectory with such segmentation, each phase will be computed separately with different constraints for each phase and 3 knots linking the phases. The variables t_1, t_2 and t_3 represent the time locations of the first, second and third knot respectively. Variables t_0 and t_f represent start and end.

Control Variable Constraints

$$0 \leq k_E(t) \leq 1 \qquad for \quad t_0 \leq t \leq t_1 \qquad \textit{(De-orbit phase)} \tag{5.14}$$

$$k_E(t) = 0 \qquad for \quad t_1 \leq t \leq t_2 \qquad \textit{(Coast phase)} \tag{5.15}$$

$$0.4 \leq k_E(t) \leq 1 \qquad for \quad t_2 \leq t \leq t_3 \qquad \textit{(Braking phase)} \tag{5.16}$$

$$0.4 \leq k_E(t) \leq 1 \qquad for \quad t_3 \leq t \leq t_f \qquad \textit{(Vertical descent phase)} \tag{5.17}$$

$$-0.5^\circ \leq \alpha(t) \leq 0.5^\circ \qquad for \quad t_0 \leq t \leq t_f \qquad \textit{(All phases)} \tag{5.18}$$

The throttle bound during coast phase is set to 0, to prevent changes of periapsis during this phase. Lower throttle bound for last two phases of descent was selected to be non–zero to prevent a case when the engine is shut down and fails to restart. This would be fatal when the spacecraft is on a collision course with the Moon. Therefore it is safer to keep the engine from shutting down. The bounds for angular acceleration were chosen the same as in [12], page 100, where the reasonable estimate, taking into account the vehicle specifications, was made.

Event Constraints

$$y^-(t_1) = y^+(t_1) \tag{5.19}$$

$$y^-(t_2) = y^+(t_2) \tag{5.20}$$

$$y^-(t_3) = y^+(t_3) \tag{5.21}$$

$$t_0 < t_1 < t_2 < t_3 < t_f \tag{5.22}$$

The state y remains unchanged at the knots. The y^- and y^+ denote the values just prior to and after the knot respectively. The last constraint ensures that events are sequential.

Constants

The constants used for the optimal trajectory computation are presented in Table 5.1. Engine exhaust velocity V_{ex} was computed by using specific impulse I_{sp} of the Beresheet engine (see Table B.1 for engine specifications) and a following equation:

$$V_{ex} = I_{sp} \cdot g_E \tag{5.23}$$

where g_E is gravitational acceleration of the Earth and is equal to 9.8 m/s^2.

Initial values

According to mission plan, the descent is initiated at a 210×210 km circular lunar orbit. The initial radius $r(t_0)$ can be simply computed as $R_M + h$. From radius, the spacecraft's initial speed $v(t_0)$ is computed using equation 3.9. The radial component of velocity $v(t_0)$ is equal to 0 since the spacecraft is moving about a circular orbit, therefore the tangential component $v_\theta(t_0)$ is equal to $v(t_0)$. Initial central angle $\theta(t_0)$ and initial pitch angle $\psi(t_0)$ were selected.

Table 5.1: Constant values used in computations. Taken from [9, 17].

T_{max}	456 N
V_{ex}	$3,116.4\ m/s$
R_M	$1.7381 \cdot 10^6\ m$
μ_M	$4.902800076 \cdot 10^{12}\ m^3/s^2$
R_E	$6.3781 \cdot 10^6\ m$
μ_E	$3.98600436233 \cdot 10^{14}\ m^3/s^2$

Initial mass

Finally it is important to know initial mass $m(t_0)$ of the spacecraft, which, unfortunately, can not be found in any mission description documents. Therefore at least an approximate value was computed by computing the values of necessary Δv of the spacecraft (eq. 3.8 and 3.10), for executing all maneuvers prior to initiating the lunar descent, and converting that Δv to amount of depleted mass (eq. 3.11).

As seen in Table 2.1, the spacecraft was deployed on an insertion super-GTO of $68,715 \times 215$ km and it arrived at a circular 210×210 km lunar orbit, before initiating the lunar descent. The change of mass between these two events needs to be computed. Computing the Δv and Δm for each of the periapsis and apoapsis maneuvers, both in the Earth's orbit and the Moon's orbit, is quite straightforward with the use of equations 3.8, 3.10 and 3.11. The maneuvers are considered impulsive burns. The lunar orbit insertion (LOI) burn needs to be included. In article [24], a LOI Δv was researched for a spacecraft with engine thrust of 450 N, which is very close to the Beresheet spacecraft thrust – 456 N. Therefore it is safe to use the value from [24] as an approximation for the Beresheet LOI burn. It can be seen in [24], page 211, that in best case it takes about 355.55 m/s of Δv to capture the spacecraft into a lunar orbit with orbital period of about 12.235 hours, which corresponds to about $8,681 \times 290$ km orbit. From this point the spacecraft executes remaining burns necessary to reach the target orbit of 210×210 km. The mass of burned fuel up to this point is 192.586 kg, the initial mass $m(t_0)$ is then computed to be 389.414 kg. Table B.2 shows Δv and corresponding amount of depleted mass for each maneuver. All initial values for the computation can be seen in Table 5.2.

State Variable Bounds

The state variable bounds are presented to maintain feasibility of the mission, so the variables do not reach invalid values at any time. The boundary conditions need to be associated with specific state variables in *boundarycond.tpp* file, using few simple C commands. State variable bounds can be seen in Table 5.3.

Table 5.2: Initial values of the problem.

$r(t_0)$	$1,948,100\ m$
$\theta(t_0)$	$0°$
$v_r(t_0)$	$0\ m/s$
$v_\theta(t_0)$	$1,586.414\ m/s$
$\psi(t_0)$	$-90°$
$m(t_0)$	389.414 kg

Table 5.3: State variable bounds of the problem.

Variable	Lower bound	Upper bound	Note
$r(t)$	$1,738,100\ m$	-	*Above surface,* $r(t) \geq R_M$
$v_r(t)$	-	$0\ m/s$	*Descend*
$v_\theta(t)$	$0\ m/s$	-	*Move forward*
$\psi(t)$	$-90°$	$0°$	
$\omega(t)$	$-10°/s$	$10°/s$	
$m(t)$	150 kg	-	$m(t) \geq m_{dry}$

Final conditions

Final conditions are set for several variables at time t_f. The radius condition is set, so the spacecraft lands on surface. To ensure soft landing, spacecraft needs to land with zero horizontal velocity and near zero vertical velocity. The attitude of the spacecraft needs to be near vertical, in legs down position.

$$r(t_f) = 1,738,100\ m \tag{5.24}$$

$$v_\theta(t_f) = 0\ m/s \tag{5.25}$$

$$0 \leq v_r(t_f) \leq 0.5\ m/s \tag{5.26}$$

$$-0.5 \leq \psi(t_f) \leq 0.5° \tag{5.27}$$

Objective function

The objective function needs to be specified in *criterion.tpp* file. The objective is to minimize fuel depletion, in other words maximize mass. In preliminary analysis it was found that the angular acceleration was fluctuating rapidly, which would unnecessarily deplete fuel of attitude control thrusters. The cost of angular acceleration command was therefore included in the cost to be minimized, as suggested in [12], page 100. This is included by adding an integral of a constant C, times the square of angular acceleration. The value of C was given a small value, such as 10^{-3}, to minimize the impact on the cost. The objective function is following:

$$J_{min} = -m(t_f) + \int_{t_0}^{t_f} C\,[\alpha(t)]^2\,dt \tag{5.28}$$

Finally the dynamics need to be specified in *dynamics.tpp* file. That is done by coding the equations 4.30–4.36 in C language.

De-orbit phase

The objective of de-orbit burn is to lower the perilune height h_p to a desired value – in this case 15 km above surface. To meet this objective, an additional state variable h_p is temporarily added to the state vector and its state is computed by equations 3.12 and 3.13. The final condition for this phase is:

$$h_p(t_1) \leq 15,000\ m \tag{5.29}$$

Coast phase

The objective of coast phase is to bring the spacecraft near perilune. The vehicle follows the descent coast trajectory until its altitude is below a specified value, then the braking phase begins. The target altitude is selected slightly above perilune height, to leave some margin for error of the optimization software. Pitch is constrained so the spacecraft arrives in engine-first position to the braking phase. Thrust is constrained to 0 in this phase and final conditions for this phase are:

$$r(t_2) \leq R_M + 15,100\ m \tag{5.30}$$

$$\theta(t_2) = -90^\circ \tag{5.31}$$

Breaking phase

The objective of breaking phase is to null most of the velocity and prepare the vehicle for landing. At the end of this phase, the vehicle should be able to land immediately, in case of incorrect measurements, therefore near-zero vertical attitude in legs-down position is also required. Multiple final conditions are introduced for this phase:

$r(t_3) = 1\ 738\ 600$ m	*Altitude 500 m*
$v_r(t_3) \geq -2$ m/s	*Near-zero vertical speed*
$v_\theta(t_3) \leq 0.5$ m/s	*Near-zero horizontal speed*
$-0.5^\circ \leq \psi \leq 0.5^\circ$	*Vertical attitude*

Vertical descent phase

The objective of vertical descent phase is to finish the descent and ensure soft landing. The horizontal velocity is constrained to 0 during this phase. Final conditions for this phase equal to final conditions for the whole descent, at time t_f, which can be found earlier in this section 5.3.

5.4 Results

The optimal descent trajectory for the Beresheet Lunar Lander was computed to take 5,025.3 s (a little under 84 minutes) during which the spacecraft burns 210.2 kg of fuel, and soft lands with a safe margin 29.2 kg of unburnt fuel (about 6.8%). The spacecraft gradually performs the four planned phases, during which it reduces its velocity and soft lands with vertical attitude in legs down position, which is considered a successful landing. Detailed information about each phase can be seen in Table 5.4.

Table 5.4: Phase breakdown of the computed optimal descent trajectory.

Phase	**Duration** $[s]$	Δv $[m/s]$	Δm $[kg]$	$\Delta\theta$ $[deg]$	**Downrange distance** $[km]$	**Vertical distance** $[km]$
De-orbit	36	41.2	5.26	1.61	50.28	0.018
Coast	3,499	0	0	176.1	5,341.73	194.91
Braking	1,426.3	2,268.7	198.69	43.2	1,309.05	14.57
Vertical descent	64.3	104.8	6.19	0	0	0.51
Total	5,025.3	2,391.1	210.15	220.93	6,702.11	210

Results shown in Figures 5.4–5.11 were plotted using the open source tool *Gnuplot*[2]. The descent begins by *de-orbit phase*, *coast phase* starts at about $t = 36\ s$, *braking phase* begins at about $t = 3,535\ s$ and *vertical descent phase* starts at about $t = 4,961\ s$.

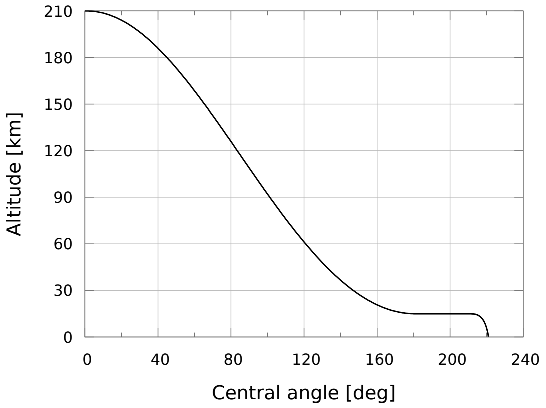


Figure 5.4: The optimal descent trajectory profile.

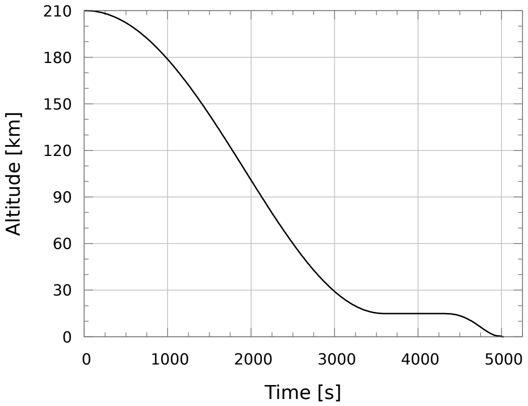


Figure 5.5: Time history of altitude during the optimal descent trajectory.

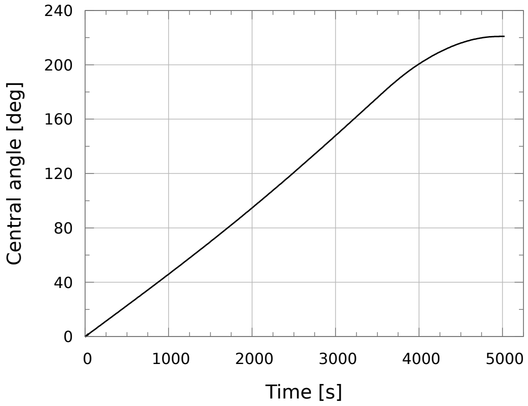


Figure 5.6: Time history of central angle during the optimal descent trajectory.

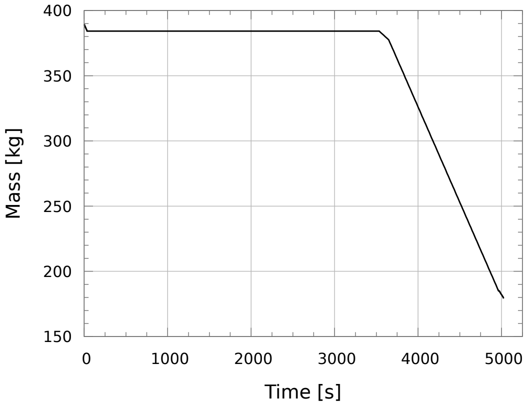


Figure 5.7: Time history of mass during the optimal descent trajectory.

[2] http://gnuplot.sourceforge.net/

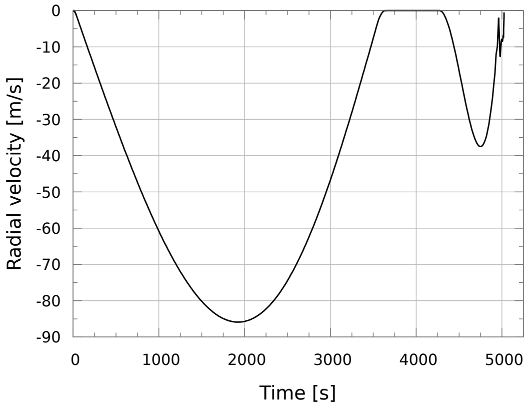


Figure 5.8: Time history of radial velocity during the optimal descent trajectory.

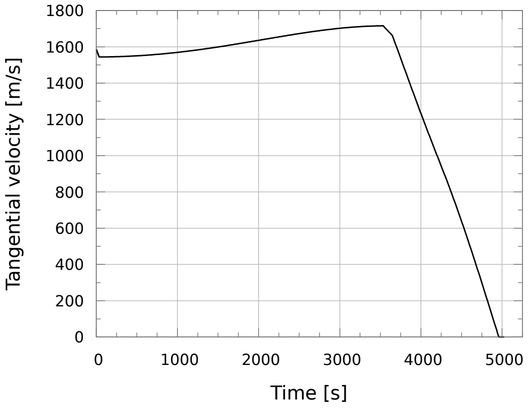


Figure 5.9: Time history of tangential velocity during the optimal descent trajectory.

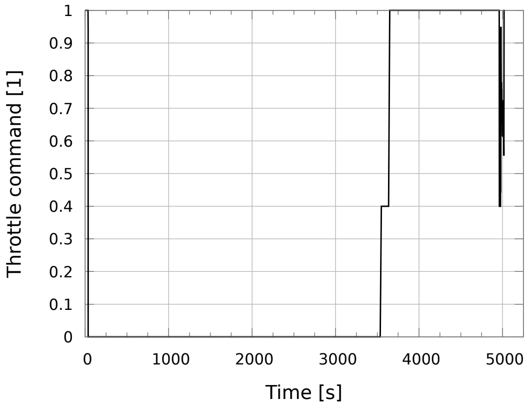


Figure 5.10: Time history of throttle command during the optimal descent trajectory.

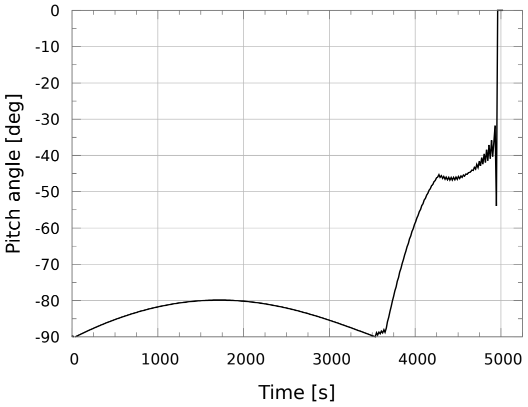


Figure 5.11: Time history of pitch angle during the optimal descent trajectory.

Chapter 6

Design and Implementation of Visualization Environment for Descent Trajectory Interpretation

For purposes of visualizing the computed optimal descent trajectory a visualization application was developed. In this chapter the design and implementation details of the application are presented.

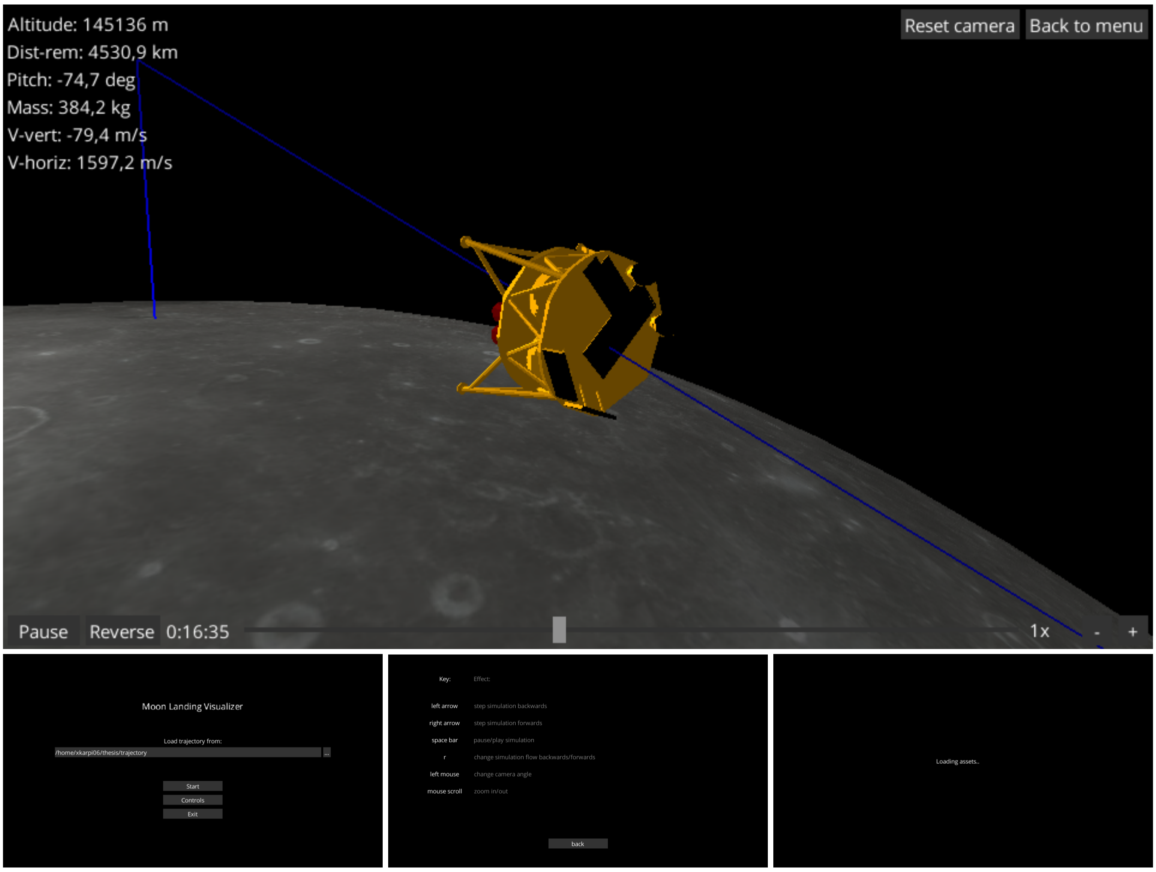
6.1 Application overview

Motivation

The general idea of the application is to provide the user with a tool, which would help him/her to easily understand the trajectory. The application would display the Moon and the Beresheet spacecraft in 3D space, where all the objects and their mutual distances are on the same scale. Once the trajectory would be loaded from dataset, the user would be able to see an animation of the spacecraft following the trajectory from start to end, and he would also be able to interact with the animation to for example change its pace or jump into different point in time.

Design

On application startup, a main menu is loaded. Here the user is first asked to provide a path to directory with trajectory dataset. Then the user can choose to either display the list of available user interface controls, or to proceed to the visualization screen. If the latter one is chosen, a screen with a 3D scene is displayed. The scene shows 3D models of the Moon, the Beresheet spacecraft, and the trajectory on the same scale. For better illustration, the Moon is covered with a texture, created from real Moon imagery. By default, an animation of the lunar descent is played in real time, and state variable information is displayed in top left corner, for better knowledge of current state of the descent. A user interface is provided in form of buttons and a slider, that enables the user to pause/resume the visualization, change the time flow to forward/backward, jump to different time using slider, change the visualization speed and adjust the camera view angle, or return to the main menu. Application screens can be seen in Figure 6.1.



Altitude: 145136 m
Dist-rem: 4530,9 km
Pitch: -74,7 deg
Mass: 384,2 kg
V-vert: -79,4 m/s
V-horiz: 1597,2 m/s

0:16:35

Moon Landing Visualizer

Load trajectory from:
/home/xkarpi06/thesis/trajectory

Key:	Effect:
left arrow	step simulation backwards
right arrow	step simulation forwards
space bar	pause/play simulation
r	change simulation flow backwards/forwards
left mouse	change camera angle
mouse scroll	zoom in/out

Loading assets..

Figure 6.1: Application's visualization, main menu, controls and loading screens.

6.2 Application implementation

The application was developed using the LibGDX framework built on top of Java SE 8[1] environment. The LibGDX was chosen because it supports creating 3D graphical output for Java applications. The trajectory is loaded from dataset previously computed by the optimization and smoothed out using Bézier curve algorithm before being displayed as a line in space. The Moon and the Beresheet spacecraft are loaded as 3D objects.

LibGDX

The LibGDX[2] is a free open source game development framework providing an API that works on all supported platforms. When the application is targeting multiple platforms at the same time, including desktop, mobile, and web, the API is abstract enough that the code and assets can be mostly shared. The LibGDX is using OpenGL (graphics library) for rendering on all platforms and offers users class abstractions as *Mesh*, *Texture*, *Shader*, *PerspectiveCamera* and more.

The main entry point of a LibGDX application is *Application* interface, which sets up a window and manages graphics, audio, input and files of the application. *Application* can be instantiated as a desktop, android, HTML5 or iOS application. The application logic is

[1] https://www.oracle.com/java/technologies/javase-jdk8-downloads.html

[2] https://libgdx.badlogicgames.com/index.html

implemented in a class implementing *ApplicationListener*, which provides methods *create, render, pause, resume, dispose* and *resize*, that are called by *Application* in corresponding events. For further information please refer to the LibGDX documentation [31].

Discussed application was developed for desktop use, but could easily be extended for use on other platforms by adding a launcher that uses different *Application* implementation. It uses *Game* abstract implementation of *ApplicationListener*, which in addition enables to easily switch between multiple application screens.

Implementation details

The application consists of four screens: main menu screen – which loads on startup, controls screen – which informs user about ways of controlling the visualization, loading screen – which is only displayed while assets are being loaded and 3D models created, and visualization screen – which is the main screen, displaying the visualization.

The visualization screen consists of 3D scene, representing the bodies and trajectory in space, which uses *PerspectiveCamera* to be displayed, and a 2D layer, rendered on top of the 3D scene, which provides a user interface and displays additional information about current state of the descent. Perspective cameras are commonly used in 3D graphics, because they display the world in an intuitive way to the user, as more distant objects appear smaller. Opposed to that orthographic cameras display more distant objects the same size as the closer ones, which can be used for example for creating 2D games, but would feel very unintuitive in 3D world.

The Moon is represented by a sphere covered with color texture, downloaded from [28], which was created by NASA's Scientific Visualization Studio from real images taken by LRO[3]. The Beresheet spacecraft is represented by a 3D model, downloaded from Thingiverse [20], created by royyehu, licenced under CC BY 4.0. The model was painted gold to represent golden layers of insulation, the spacecraft was covered in before launch. Both of the 3D models are loaded into the application using the LibGDX *AssetManager*, which enables asynchronous loading.

The trajectory is loaded from a file in form of three-dimensional points in polar or cartesian coordinate system. If the input is in polar coordinate system, the points are converted to cartesian coordinate system using equations 3.3 and 3.4. Then the trajectory is smoothed out using the Bézier curve algorithm (discussed later). For this purpose the LibGDX offers class *Bezier*, which supports linear, quadratic and cubic curves. Obtained points are then used to create a mesh and the trajectory is displayed in form of line.

A time history (list of times) needs to be provided to the application in order for it to know how fast to clear individual sections of the trajectory. Additional state information about the spacecraft during lunar descent can be loaded, such as mass, pitch and velocity in time, and displayed in top left corner during the visualization.

The user interface in whole application was created with the use of VisUi[4] user interface toolkit for the LibGDX, which provides skin for typical ui elements such as buttons, labels, etc. and also *FileChooser* class – a window in which the user provides the path to trajectory source files. The application is using model-view-controller (MVC) architecture, which ensures that the user input changes the model first, and the change effects the view displayed to the user.

[3] Lunar Reconnaissance Orbiter is a NASA spacecraft mapping the Moon's surface.

[4] https://github.com/kotcrab/vis-ui

Bézier curve algorithm

The input trajectory may not always have sufficient amount of points and if such trajectory would be displayed, sharp changes of direction could be observed at certain places. To improve user experience, the input trajectory is approximated by a smooth curve. For this purpose a Bézier curve algorithm was used in the application.

From [3]: Bézier curve (also called Bézier spline) is a line, that passes through a set of points, or knots. It is a collection of polygonal segments, which can be linear, quadratic, cubic or higher order polynomials. The cubic Bézier curve is in [3] defined as

$$\mathbf{B}(t) = (1-t)^3\mathbf{P}_0 + 3(1-t)^2t\mathbf{P}_1 + 3(1-t)t^2\mathbf{P}_2 + t^3\mathbf{P}_3, \quad t \in [0,1] \tag{6.1}$$

where points $\mathbf{P}_0$ and $\mathbf{P}_3$ correspond to the knots and points $\mathbf{P}_1$ and $\mathbf{P}_2$ are control points, which determine the curve's shape. To create a smooth curve, the first and second derivatives must be continuous across the spline boundaries, which for the left boundary of i–th segment can be written as:

$$\begin{aligned} \mathbf{B}_i'(0) &= \mathbf{B}_{i-1}'(1) & (6.2)\\ \mathbf{B}_i''(0) &= \mathbf{B}_{i-1}''(1) & (6.3) \end{aligned}$$

The first and second derivatives are defined as:

$$\begin{aligned} \mathbf{B}'(t) &= -3(1-t)^2\mathbf{P}_0 + 3(1-4t+3t^2)\mathbf{P}_1 + 3(2t-3t^2)\mathbf{P}_2 + 3t^2\mathbf{P}_3 & (6.4)\\ \mathbf{B}''(t) &= 6(1-t)\mathbf{P}_0 + 3(-4+6t)\mathbf{P}_1 + 3(2-6t)\mathbf{P}_2 + 6t\mathbf{P}_3 & (6.5) \end{aligned}$$

After simplifying these equations and taking into account that $\mathbf{P}_{0,i} = \mathbf{P}_{3,i-1} = \mathbf{K}_i$, the following pair of equations is obtained:

$$\begin{aligned} 2\mathbf{K}_i &= \mathbf{P}_{1,i} + \mathbf{P}_{2,i-1} & (6.6)\\ -2\mathbf{P}_{1,i} + \mathbf{P}_{2,i} &= \mathbf{P}_{1,i-1} - 2\mathbf{P}_{2,i-1} & (6.7) \end{aligned}$$

These equations are only defined at the internal knots, therefore two following boundary conditions are defined for the end knots: $\mathbf{B}_0''(0) = 0$ and $\mathbf{B}_{n-1}''(1) = 0$ (the spline is linear at the end knots). The equations for the endpoints are:

$$\begin{aligned} \mathbf{K}_0 - 2\mathbf{P}_{1,0} + \mathbf{P}_{2,0} &= 0 & (6.8)\\ \mathbf{P}_{1,n-1} - 2\mathbf{P}_{2,n-1} + K_n &= 0 & (6.9) \end{aligned}$$

The set of equations 6.6–6.9 can be used to compute the individual segments of Bézier curve. For illustration of Bézier curve see Figure 6.2.

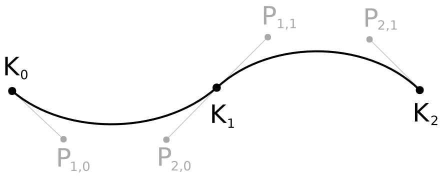


Figure 6.2: Bezier curve with knots and control points.

Chapter 7

Evaluation

In this chapter the optimal descent trajectory is evaluated first, then it is compared to the real trajectory, followed during the Beresheet mission, and in the end short evaluation of visualization environment is given.

Optimal descent trajectory

The computed optimal descent trajectory meets all proposed criterions and is considered feasible, thus can be evaluated as successful. To verify the correctness of computational model, the best way would be to compare the trajectory to the Beresheet mission planned trajectory, or to real flight data. Unfortunately the former is not available and the latter can not be used for verification, since the descent resulted in a crash. Follows a best effort verification. The optimal descent trajectory consists of four phases, each of which will be evaluated separately.

The *de-orbit* phase was expected to lower the perilune height to about 15 km and last only few dozens of seconds. The perilune was lowered successfully and the maneuver took 36 seconds to complete, therefore the result shows no signs of errors.

The *coast* phase has brought the vehicle near the perilune as expected. Since the spacecraft travels during this phase almost 180° in longitude, roughly from apolune to perilune, without applying any thrust, the duration of the phase can be expected to be slightly less than a half of orbit period. The spacecraft is orbiting a 210×15 km ellipse, the period is thus 7,143 s (eq. 3.7). The coast phase takes a little less than half of that time – 3,499 s – which suggests the model is correct. The difference is caused by phase start 36 seconds after apolune and phase ending sooner than reaching the perilune.

The *braking* phase of the optimal descent trajectory begins with a horizontal section, lasting for about 600 seconds, during which the vertical velocity remains zero. This solution appeared to be unintuitive and sub-optimal, therefore some alternatives were examined to verify the solution. The flatness of the section was caused by setting the upper bound for vertical velocity to 0. When the bound was removed, a 10 km raise in altitude appeared instead of the flat section, but it did not result in additional fuel savings. Next a variant with initiating the braking phase up to 800 seconds earlier was examined, but this also did not result in additional fuel savings. Neither of alternatives resulted in better solution, therefore the presented solution appears to be optimal.

To see whether the flat section would disappear if higher thrust was available, two fictional scenarios were examined: first with twice the Beresheet's thrust – 912 N – and second with four times the Beresheet's thrust – 1824 N. The optimal braking phase for

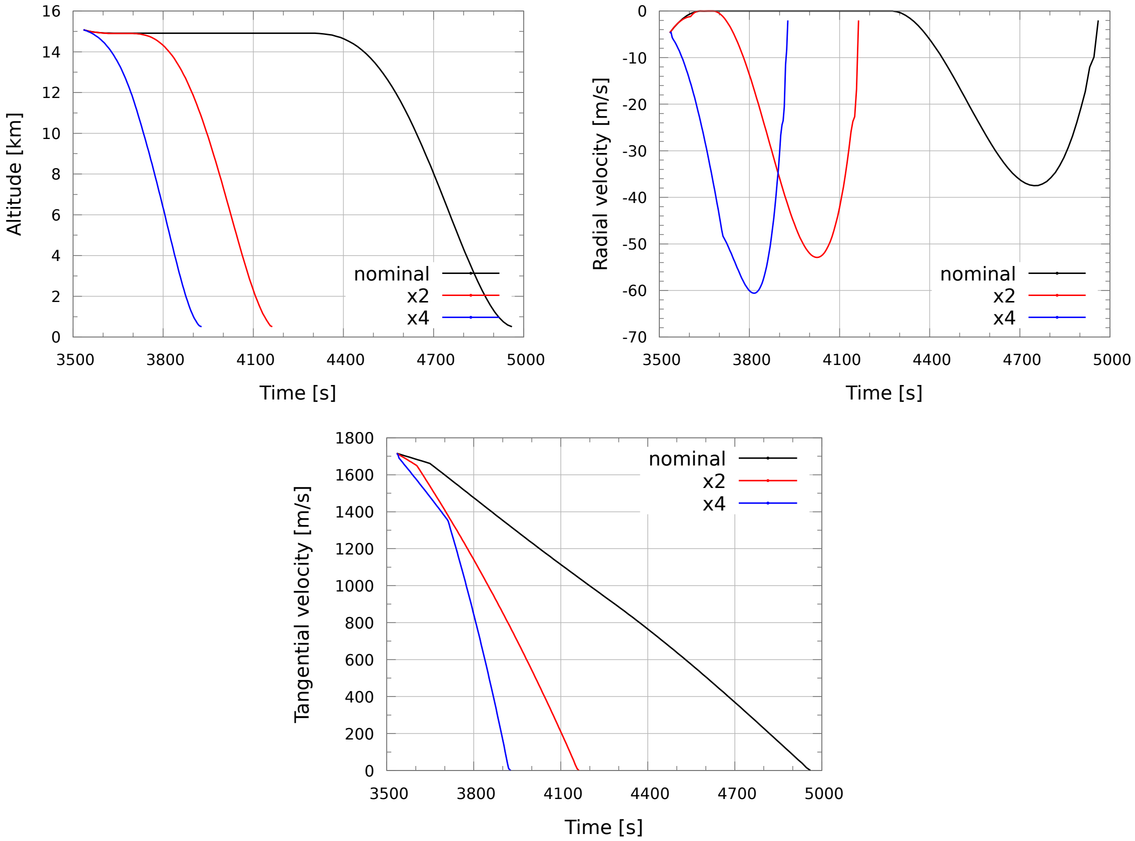


Figure 7.1: Braking phase comparison with nominal, double and quadruple thrust.

both of the scenarios was computed and compared to the original braking phase of the optimal descent trajectory. As it can be seen in Figure 7.1, with higher engine capabilities the trajectory would actually look more intuitive, which suggests that the long horizontal section is only a coincidence, but not a fault in model.

The *vertical descent* phase remains vertical and takes about 64.3 seconds, which seems adequate.

The best effort to evaluate the computed optimal descent trajectory was made and no irregularities were found that would suggest the model is incorrect.

Comparison with the real trajectory followed during the Beresheet mission

The real descent trajectory was obtained by sampling the telemetry displayed in live broadcast from the Beresheet lunar landing [25] on youtube. The sampling begins about 33.5 km above surface, because at that moment the telemetry first appeared in the live broadcast. It is imperfect, because the telemetry was unavailable several times in the broadcast, but it is good enough for visual comparison. The comparison of trajectories starts at a point, where the optimal descent trajectory reached 33.5 km altitude. As Figure 7.2 shows, compared trajectories differ significantly, but that was expected since the real one led to a crash.

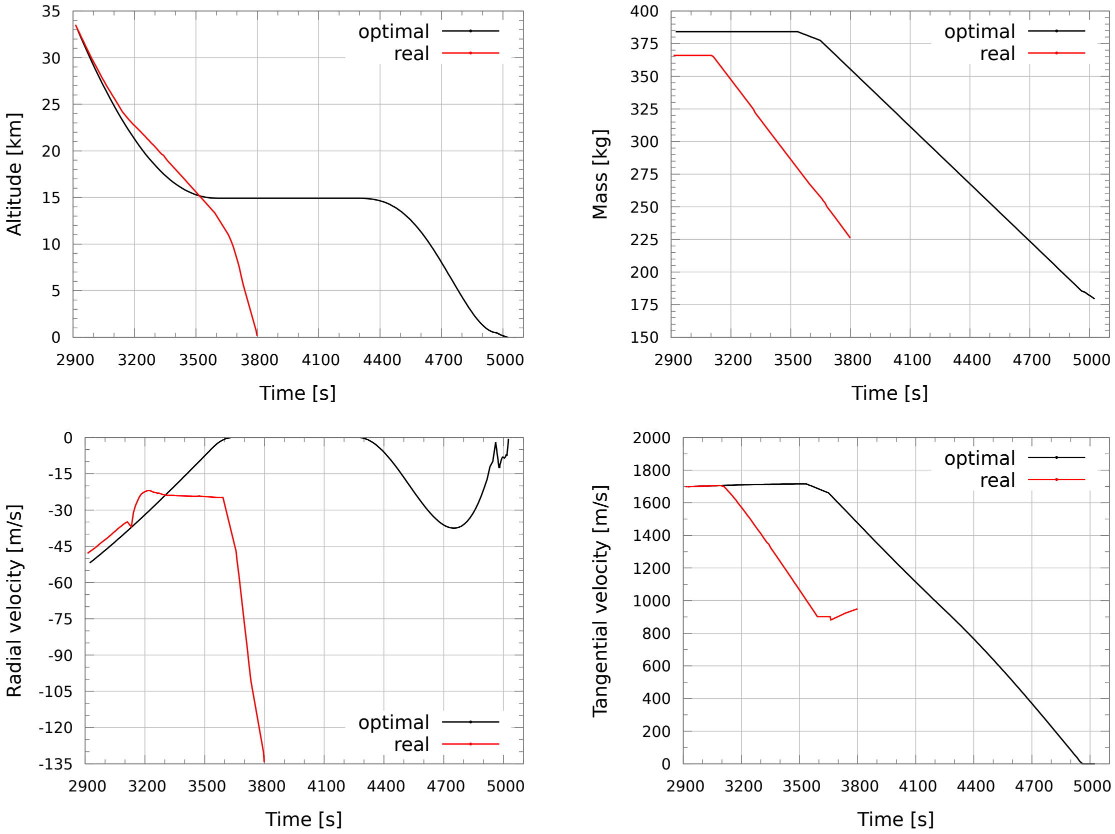


Figure 7.2: Comparison of the optimal descent and the real Beresheet mission trajectories.

Visualization environment

The goal was to create an environment, that would enable to visualize a lunar descent trajectory while enabling the user to interact with the visualization. The environment was tested by loading the computed optimal descent trajectory. It was observed, that the trajectory, relative to the Moon, was displayed as expected, confirming correct scale of the Moon in the scene. A user interface enables interactions with the visualization, as planned. The environment is ready to be used for intended purpose and therefore the goal is considered as fulfilled.

Chapter 8

Conclusion

The goal of this bachelor's thesis was to compute the optimal descent trajectory for the commercial Lunar Lander Beresheet and develop an environment for visualizing the Beresheet lunar landing. Both of these goals were fulfilled. The assignment also included research of the Beresheet mission, research of orbital mechanics, derivation of equations of motion and evaluation of achieved results. All of the assignments were also fulfilled.

The computed optimal descent trajectory meets all feasibility criterions, reduces risk that could endanger mission success and brings the Beresheet spacecraft to a safe landing with 29.2 kg (6.8%) of fuel left. The descent maneuver takes about 84 minutes to complete. Alternative trajectories were examined, but no better was found. In the end it was compared with the real Beresheet mission trajectory.

Developed visualization environment is a desktop application that can be launched on Linux or Windows® operating systems. It provides an intuitive trajectory visualization enabling users to easily understand computed lunar descent trajectories. Relevant 3D model of the Beresheet spacecraft and a Moon texture created from real LRO[1] images were downloaded and used to provide the best experience.

This assignment has taught me many things from astrodynamics, optimization and implementation of 3D application, which I am grateful for and I would like to use the gained knowledge to pursue even greater tasks in my master's thesis.

Future extensions to this thesis may include computing the optimal descent trajectory in six degrees of motion, taking into account a specific landing site and comparing the optimal trajectory with the mission plan, if it is released in the future. The visualization environment could be improved by adding a displacement map to the model of the Moon, to distinguish mountains and craters, and by adding a flame animation to the spacecraft, to see when it is applying thrust.

[1] Lunar Reconnaissance Orbiter is a NASA spacecraft mapping Lunar surface.

Bibliography

[1] Astolfi, A. *Optimization* [online]. 2006 [cit. 2020-03-18]. Available at: http://www3.imperial.ac.uk/pls/portallive/docs/1/7288263.PDF.

[2] Bonnans, F., Martinon, P., Giorgi, D., Grélard, V., Maindrault, S. et al. *BOCOP 2.2.0 - User Guide* [online]. 2019 [cit. 2020-01-25]. Available at: https://files.inria.fr/bocop/UserGuide-BOCOP.pdf.

[3] Brieda, L. *Smooth Bézier Spline Through Prescribed Points* [online]. 2012 [cit. 2020-05-07]. Available at: https://www.particleincell.com/2012/bezier-splines/.

[4] Colasurdo, G. *Astrodynamics* [online]. Polytechnic University of Turin, 2006 [cit. 2020-02-26]. Available at: http://dma.ing.uniroma1.it/users/lss_mo/MATERIALE/AvanziniColasurdoAstrodynamics.pdf.

[5] Corp., S. E. T. *Falcon User's Guide* [online]. 2019 [cit. 2020-04-11]. Available at: https://www.spacex.com/sites/spacex/files/falcon_users_guide_10_2019.pdf.

[6] Corp., S. E. T. *Nusantara Satu Mission Press Kit* [online]. 2019 [cit. 2020-04-11]. Available at: https://www.spacex.com/sites/spacex/files/nusantara_satu_press_kit.pdf.

[7] Corp., S. E. T. *Falcon 9* [online]. 2020 [cit. 2020-04-11]. Available at: https://www.spacex.com/falcon9.

[8] Fajmon, B., Hlavičková, I., Novák, M. and Vítovec, J. *Numerická matematika a pravděpodobnost*. Vysoké učení technické v Brně, 2014 [cit. 2020-03-20].

[9] Folkner, W. M., Williams, J. G. and Boggs, D. H. *The Planetary and Lunar Ephemeris DE 421* [online]. 2009 [cit. 2020-02-25]. Available at: https://ipnpr.jpl.nasa.gov/progress_report/42-178/178C.pdf.

[10] Graham, W. *SpaceX launches Indonesian satellite launch and Israeli moon mission* [online]. [cit. 2020-04-11]. Available at: https://www.nasaspaceflight.com/2019/02/spacex-indonesian-launch-israeli-moon-mission/.

[11] Hamme, W. V. V. *Orbital Mechanics* [online]. 2020 [cit. 2020-03-17]. Available at: http://faculty.fiu.edu/~vanhamme/ast3213/orbits.pdf.

[12] Hawkins, A. M. *Constrained Trajectory Optimization of a Soft Lunar Landing from a Parking Orbit*. Cambridge, 2005. Masters Thesis. Massachusetts Institute of Technology. Dept. of Aeronautics and Astronautics. Available at: https://dspace.mit.edu/handle/1721.1/32431.

[13] Krebs, G. D. *Beresheet (SpaceIL Lunar Lander)* [online]. [cit. 2020-04-10]. Available at: https://space.skyrocket.de/doc_sdat/beresheet.htm.

[14] Lambers, J. *Maximum and Minimum Values* [online]. 2009 [cit. 2020-03-19]. Available at: http://www.math.sci.hokudai.ac.jp/~s.settepanella/teachingfile/Calculus/Calculus1/pagine/lecture8.pdf.

[15] NASA. *Beresheet* [online]. [cit. 2020-04-10]. Available at: https://nssdc.gsfc.nasa.gov/nmc/spacecraft/display.action?id=2019-009B.

[16] NASA. *Humanity's return to the moon* [online]. [cit. 2020-05-09]. Available at: https://www.nasa.gov/specials/artemis/.

[17] NASA. *Moon Fact Sheet* [online]. [cit. 2020-02-25]. Available at: https://nssdc.gsfc.nasa.gov/planetary/factsheet/moonfact.html.

[18] NASA. *Earth* [online]. 2012 [cit. 2020-05-08]. Available at: https://images.nasa.gov/details-PIA18033.

[19] NASA. *Full Moon* [online]. 2017 [cit. 2020-05-08]. Available at: https://images.nasa.gov/details-GSFC_20171208_Archive_e001861.

[20] royyehu. *Beresheet SpaceIL 3D model* [online]. April 2019 [cit. 2020-05-06]. Available at: https://www.thingiverse.com/thing:3570931.

[21] satsearch. *LEROS 2b* [online]. [cit. 2020-03-03]. Available at: https://satsearch.co/products/moog-inc-leros-2b.

[22] Shyldkrot, H., Shmidt, E., Geron, D., Kronenfeld, J., Loucks, M. et al. THE FIRST COMMERCIAL LUNAR LANDER MISSION: BERESHEET. In:. August 2019.

[23] Society, T. P. *Beresheet, S5 and Nusantara Satu with payload fairing* [online]. [cit. 2020-04-11]. Available at: https://www.planetary.org/multimedia/space-images/spacecraft/beresheet-s5-and-nusantara-satu.html.

[24] Song, Y.-J., Park, S.-Y., Kim, H.-D., Lee, J.-H. and Sim, E.-S. Analysis of Delta-V Losses During Lunar Capture Sequence Using Finite Thrust. *Journal of Astronomy and Space Sciences*. september 2011, vol. 28, p. 203–216, [cit. 2020-04-05]. DOI: 10.5140/JASS.2011.28.3.203. Available at: https://www.researchgate.net/publication/258495488_Analysis_of_Delta-V_Losses_During_Lunar_Capture_Sequence_Using_Finite_Thrust.

[25] SpaceIL. *LIVE broadcast - Beresheet lands on the Moon Fasten your seatbelts, we are about to land.* [online]. YouTube, April 2019 [cit. 2020-05-21]. Available at: https://www.youtube.com/watch?v=HMdUcchBYRA&t=1318s.

[26] Stover, C. and Weisstein, E. W. *Cartesian Coordinates* [online]. [cit. 2020-02-26]. From MathWorld–A Wolfram Web Resource. Available at: https://mathworld.wolfram.com/CartesianCoordinates.html.

[27] Stover, C. and Weisstein, E. W. *Polar Coordinates* [online]. [cit. 2020-02-26]. From MathWorld–A Wolfram Web Resource. Available at: http://mathworld.wolfram.com/PolarCoordinates.html.

[28] Wright, E. and Petro, N. *CGI Moon Kit* [online]. 2019 [cit. 2020-05-06]. Available at: https://svs.gsfc.nasa.gov/4720.

[29] XPRIZE, F. *About Us* [online]. [cit. 2020-04-09]. Available at: https://xprize.org/about/mission.

[30] XPRIZE, F. *The new space race* [online]. [cit. 2020-04-09]. Available at: https://lunar.xprize.org/prizes/google-lunar.

[31] Zechner, M. and Sweet, N. *Libgdx API* [online]. 2010-2020 [cit. 2020-05-06]. Available at: https://libgdx.badlogicgames.com/ci/nightlies/docs/api/.

Appendix A

Astronomical Bodies and Launch Vehicle in Beresheet Mission

The Earth

Figure A.1: The Earth. Source: [18].

Table A.1: Earth specifications. Taken from [9, 17].

Equatorial radius	$6,378.1\ km$
Gravitational parameter	$3.98600436233 \cdot 10^{14}\ m^3/s^2$
Surface gravity	$9.8\ m/s^2$
Escape velocity	$11.2\ km/s$
Atmospheric pressure	$1\ bar$

The Moon

Figure A.2: The Moon. Source: [19].

Table A.2: Moon specifications. Taken from [9, 17].

Equatorial radius	$1,738.1\ km$
Gravitational parameter	$4.902800076 \cdot 10^{12}\ m^3/s^2$
Surface gravity	$1.62\ m/s^2$
Escape velocity	$2.38\ km/s$
Atmospheric pressure	$3 \cdot 10^{-15}\ bar$ (vacuum)
Mean orbital radius	$378,000\ km$
Mean orbital velocity	$1.022\ km/s$

The Falcon 9

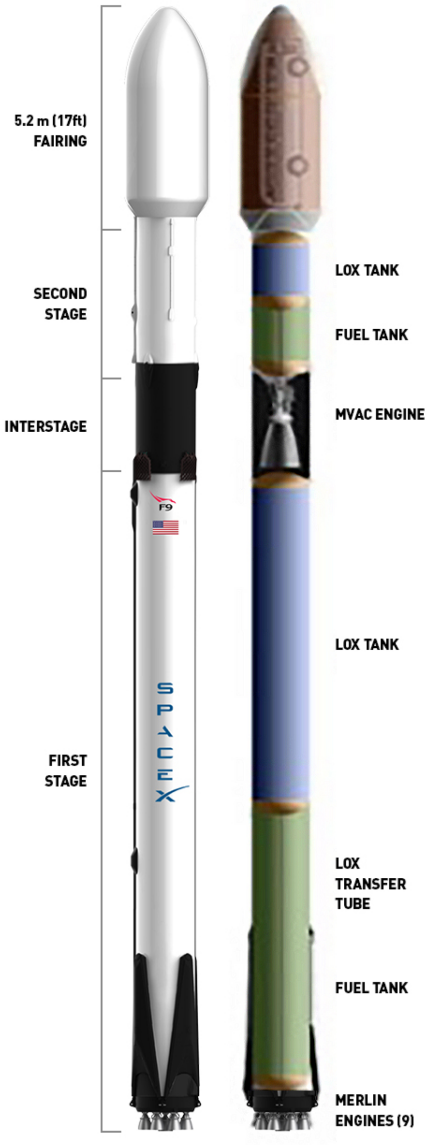


Figure A.3: The Falcon 9 rocket in block 5 version. Taken from [5].

Table A.3: The Falcon 9 specifications. Taken from [7].

Manufacturer	SpaceX
Height	$70\ m$
Diameter	$3.66\ m$
Weight (fuelled)	$594,054\ kg$
Stages	2
Propellant	Liquid oxygen & RP-1
Propulsion ($1^{st} stage$)	9×Merlin 1D engine
Propulsion ($2^{nd} stage$)	MVac engine
Thrust ($1^{st} stage$)	$7,686\ kN$ (sea level)
Thrust ($2^{nd} stage$)	$981\ kN$ (vacuum)
Payload to LEO	$22,800\ kg$
Payload to GTO	$8,300\ kg$
Payload to Mars	$4,020\ kg$

Appendix B

Additional Specifications

Table B.1: LEROS$^{\text{TM}}$2b technical specifications. Taken from [21].

Manufacturer	Moog Inc.
Propellant	MMH/MON
T	367–456 N
I_{sp}	318 s
Mass	5.0 kg
Diameter	334 mm

Table B.2: The Beresheet maneuvers with computed velocity and mass differences [22].

	Separation	70,000	215	-	-
		ApoAlt aft	**PeriAlt aft**	Δv	Δm
	Maneuver	[km]	[km]	[m/s]	[kg]
Earth Orbit	**AM1**	68,715	215	0	0
	AM2	68,710	600	24.05	4.48
	PM1	117,300	530	167.82	30.28
	PM2	275,000	1,600	176.87	30.20
	PM3	390,000	1,500	39.67	6.54
Lunar Orbit	**LOI1**	8,681	290	355.55	55.00
	LOI2	750	285	379.67	52.24
	LOI2A	750	210	14.80	1.91
	LOI2B	210	210	93.78	11.90
Total:				1,251.21	192.586

Appendix C

Contents of Associated Media Drive

- `/xkarpi06-BP.pdf` - pdf version of the thesis.
- `/thesis.zip` - compressed source files of the thesis.
- `/vis_env/` - directory containing source files of the visualization environment
- `/vis_env/odt/` - directory containing the optimal descent trajectory dataset
- `/bocop/` - directory containing the source files of the optimization problem
- `/bin/` - directory containing the executables for running the visualization environment