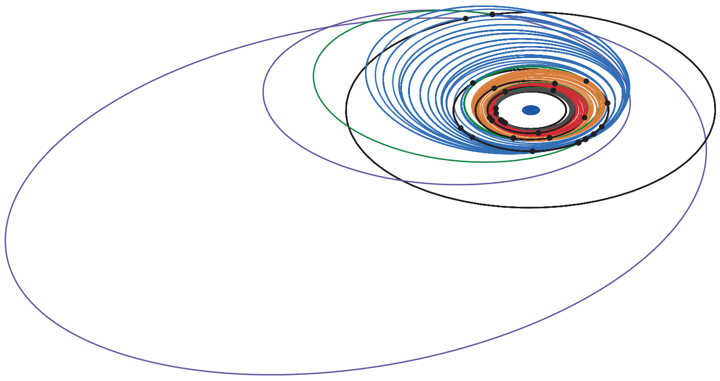


Preliminary Trajectory Design of a Mission to Enceladus

David Falcato Fialho Palma

Thesis to obtain the Master of Science Degree in

Aerospace Engineering

Supervisor(s): Prof. Paulo Jorge Soares Gil
Eng. Nuno Tiago Salavessa Cardoso Hormigo Vicente

December 2016

“So Long, and Thanks for All the Fish”

— Douglas Adams

Acknowledgments

I would like to start by expressing my gratitude to my supervisor, Prof. Paulo Gil, who taught me many valuable things, technical and beyond, but above all for his relentless support, encouragement and experience which helped shape this work. I also extend my best regards to Spin.Works, especially my co-supervisor Engineer Tiago Hormigo and Engineer João Seabra, for the interesting challenge that was proposed and for their guidance in the process.

Secondly, I'd like to acknowledge my friends, especially my brothers in arms — Nuno Martins, Jorge Araújo and Daniel Gonçalves — whose support, encouragement, fellowship and company for the long nights have been indispensable to my success in this work and programme.

To Mónica, at the Mechanical Engineering bar, a big thank you for brightening up so many days of hard work with a coffee and a smile!

I am also deeply grateful to my best friend, Catarina Farinha, for her wisdom and never-ending support and patience.

Finally, I would like to extend a special word to my family, to those who are with us and to those who are not, who have been the instruments that guided me all the way here.

Resumo

As agências espaciais despertaram recentemente para a importância da exploração de mundos oceânicos. Alguns destes corpos celestes, como Enceladus e Europa, orbitam dentro dos sistemas lunares de planetas gigantes, como Jupiter e Saturno. A vontade de investigar estas luas em detalhe motivou a concepção de uma nova classe de trajectórias — complexos “moon tours” que têm como objectivo uma redução drástica do custo de inserção orbital naquelas luas. Este trabalho ataca o desafio específico de desenvolver uma trajectória preliminar que possibilite o envio de uma sonda, com a maior massa possível, para órbita de Enceladus. Para começar, a estrutura da missão é definida. De seguida, o cálculo da trajectória começa com a optimização de diversas sequências interplanetárias, obtidas através de um processo baseado em diagramas de Tisserand. Para permitir isto, um processo de optimização global que envolve cooperação entre algoritmos estocásticos é delineado e implementado. Para obter soluções para o “moon tour”, uma técnica moderna denomeada V_∞*-Leveraging* é estudada e usada, demonstrando ser essencial neste tipo de missões. Finalmente, num passo além da literatura actual, o efeito da inclusão de transferências inter-lua na solução do “moon tour” é discutido e considerado não negligenciável. O resultado deste trabalho é um exemplo de uma trajectória preliminar que permite a colocação de um satélite com, no máximo, 2 toneladas, em órbita de Enceladus num prazo de 13.9 anos.

Palavras-chave: Enceladus, circuito inter-luas, alavancamento de V_∞, optimização de trajectórias, mundos oceânicos, diagramas de Tisserand

Abstract

Space agencies have recently awoken to the importance of exploring ocean worlds. Some of these celestial bodies, such as Enceladus and Europa, exist within the dynamic moon systems of giant planets like Jupiter and Saturn. The will to investigate these moons in detail motivated the study of a new class of trajectories — complex moon tours with the objective of drastically cutting down on the cost of orbital capture at these moons. This work tackles the specific challenge of designing a preliminary spacecraft trajectory capable of delivering as much mass as possible to Enceladus' orbit. To begin with, a framework for the mission is established. Then, the trajectory design starts with the optimisation of several interplanetary flight sequences, obtained through a process based on Tisserand plots. To achieve this, a global optimisation procedure with a cooperative stochastic topology is outlined and implemented. To obtain solutions for the moon tour, a modern V_∞-Leveraging technique, demonstrated to be essential for this class of missions, is studied and used. Finally, in a step beyond current literature, the effect of the addition of inter-moon transfer orbits to the moon tour's solution is discussed and found to be non-negligible. The outcome of this work is an example preliminary trajectory that enables an Enceladus orbiter spacecraft of up to two tonnes, within a mission time of 13.9 years.

Keywords: Enceladus,moon tour, V_∞-Leveraging, trajectory optimisation, ocean worlds, Tisserand plot

Contents

List of Tables

List of Figures

Nomenclature

Physics Constants

AU	Astronomical Unit.	$149,597,870,700\,m$
G	Universal Gravitational Constant.	$6.67408\times10^{-11}\,m^3kg^{-1}s^{-2}$

Greek Symbols

α	Pump angle.	[rad]
β	Plane-changle angle during a 3D gravity-assist.	[rad]
ΔV	Variation in velocity.	[rad]
δ	Gravity-assist turn angle.	[rad]
δ_∞	Declination of the arrival asymptote measured in the planet-centered reference frame.	[rad]
δ_{max}	Maximum turn angle due to minimum flyby height constraint.	[rad]
η	Parameter that describes the position of the DSM along an arc.	[nd]
μ	Standard gravitational parameter.	$[m^3/s^2]$
ϕ	Objective function.	
θ	True anomaly (angle from periapsis).	[rad]
θ_∞	True anomaly of the hyperbolic asymptote.	[rad]
θ_B	B-Plane angle.	[rad]

Roman Symbols

a	Orbital semi-major axis.	$[m]$
a_{sc}	Non-dimensional spacecraft orbit semi-major axis.	[nd]
$\mathbf{b}$	B-plane vector.	$[m]$
b	Impact parameter.	$[m]$
C_3	Launch energy.	$[m^2/s^2]$

C_{Tiss}	Tisserand's Invariant.	[nd]
CR	Crossover parameter of the Differential Evolution algorithm.	[nd]
D	Optimisation problem dimension.	[nd]
E	Eccentric anomaly.	[rad]
e	Orbital eccentricity.	[nd]
F	Mutation parameter of the Differential Evolution algorithm.	[nd]
$\hat{\mathbf{i}}, \hat{\mathbf{j}}, \hat{\mathbf{k}}$	Cartesian unit vectors.	[nd]
i	Orbital inclination.	[rad]
I_{sp}	Specific Impulse.	$[s]$
m	Mass.	$[kg]$
N_g	Number of generations.	[nd]
N_p	Population size.	[nd]
$\mathbf{p}$	Decision vector or chromosome.	
$\mathbf{r}$	Position vector.	$[m]$
$\mathbf{r_{s/c,\infty}}$	Position vector of the spacecraft at the edge of the sphere of influence.	$[m]$
r_p, r_a	Periapsis and Apoapsis radius.	$[m,m]$
R_{body}	Radius of body.	$[m]$
$R_{circ}, V_{circ}, T_{circ}$	Non-dimensionalization factors.	$[m,m/s,s]$
r_{fb}	Flyby radius.	$[m]$
r_{la}	Non-dimensional radius of the leveraging apse.	[nd]
r_{sphere}	Sphere of influence radius.	$[m]$
RA_{∞}	Right Ascension of the arrival asymptote measured in the planet-centred reference frame.	[rad]
rbf	Radial-basis function.	
$\hat{\mathbf{S}}, \hat{\mathbf{T}}, \hat{\mathbf{R}}$	B-Plane unit vectors.	[nd]
tof	Time of flight.	$[s]$
T	Orbital period.	
T_i	Simulated Annealing temperature parameter.	[nd]
$\hat{\mathbf{u_r}}$	Unit vector u in the direction of vector $\mathbf{r}$	[nd]

u, v	Non-dimensional constants that define departure geometry in the MGA-1DSM formulation.	[nd]
$\mathbf{V}$	Velocity vector.	$[m/s]$
$\mathbf{V}_{\infty}$	Hyperbolic excess velocity vector.	$[m/s]$
v_{∞}	Non-dimensional hyperbolic excess velocity with respect to a gravity-assist body.	[nd]
v_{∞}	Non-dimensional hyperbolic excess velocity.	[nd]
$\mathbf{x_{I\hat{A}U}}, \mathbf{y_{I\hat{A}U}}, \mathbf{z_{I\hat{A}U}}$	Unit vectors of the body-centred IAU reference frame.	

Subscripts

0	Initial.
EOI	Refers to Enceladus orbit insertion.
f	Final.
ga	Gravity Assist.
i	Initial or referent to the i^{th} object.
sc	Spacecraft.
SOI	Refers to Saturn orbit insertion.
$VILT$	Refers to a V_{∞}-leveraging manoeuvre.

VILT

L	Revolution number where the DSM occurs.	[nd]
M	Integer number of spacecraft revolutions.	[nd]
N	Integer number of moon revolutions.	[nd]
OO/OI/II/IO	Designation of the trajectory geometry at each flyby of a VILT. O: Outbound, I: Inbound	
k_{ei}	Integer parameter that indicates if a VILT is exterior (1) or interior (-1).	[nd]
k_{io}	Integer parameter that indicates if a VILT is outbound (1) or inbound (-1).	[nd]

Glossary

ΔV-EGA Earth launch followed by DSM and Earth Flyby.

ASRG Advanced Stirling Radioisotope Generator.

BFS Breadth-First Search.

CS Compass Search.

DE Differential Evolution.

DFS Depth-First Search.

DLR German Aerospace Center.

DSM Deep Space Manoeuvre.

EA Evolutionary Algorithm.

ELF Enceladus Life Finder.

EnEx Enceladus Explorer.

ESA European Space Agency.

GA Gravity Assist.

GTOC Global Trajectory Optimisation Competition.

GTOP Global Trajectory Optimisation Problem.

IAU International Astronomical Union.

J2000 Earth mean equator and equinox of J2000 reference frame.

JET Journey to Enceladus and Titan.

JPL Jet Propulsion Laboratory.

JUICE Jupiter Icy Moon Explorer.

LiFE Life Investigation for Enceladus.

MBH	Monotonic Basin Hopping.
MCS	Multilevel Coordinate Search.
MGA	Multi-Gravity Assist.
MGA-1DSM	Trajectory with multiple gravity assists and one DSM in each arc.
MS	Multi-Start.
NAIF	Navigation and Ancillary Information Facility.
NASA	National Aeronautics and Space Administration.
NLP	Non-Linear Programming.
PRM	Periapsis Raising Manoeuvre.
PSO	Particle Swarm Optimisation.
PyGMO	Python Parallel Global Multi-objective Optimiser.
PyKEP	Python Keplerian Toolbox.
RTG	Radioisotope Thermoelectric Generator.
SAAN	Simulated Annealing with Adaptive Neighbourhood.
SEP	Solar electric propulsion.
SGA	Simple Genetic Algorithm.
SLSQP	Sequential Least Squares Programming.
SOI	Saturn Orbit Insertion.
SPICE	Spacecraft trajectory development toolkit.
TCM	Trajectory Correction Manoeuvre.
TSSM	Titan Saturn System Mission.
VILT	V_∞-Leveraging Transfer.

Chapter 1

Introduction

Ocean worlds — celestial bodies that may harbour liquid oceans and lakes — are top candidates to satisfy one of humanity's greatest quests: to find extraterrestrial life in the vast emptiness of space. Many of these candidates exist in our own Solar System and several mission concepts are being designed to acquire scientific data to answer questions about the formation of these worlds and their astrobiological potential.

The objective of this work is to design an optimised trajectory with the final goal of inserting a satellite in orbit around one of those potential candidates — Enceladus — carrying the largest possible payload. This icy moon orbits Saturn, a harsh environment, where multiple-body gravitational effects dominate and particle rings must be avoided to mitigate damage to spacecraft.

1.1 Elements of Interplanetary Trajectories

Before dwelling into the world of spacecraft trajectory design, let's start by establishing a few key concepts that will be frequently mentioned in this work.

Interplanetary trajectories can be defined as the orbital path that a spacecraft follows between an initial and a final celestial body, in a certain "time of flight". The path may be divided into several segments, referred to here as *arcs* or *legs*, depending on the necessity to include additional manoeuvres between the start and end bodies. These celestial bodies are moving relative to each other and everything else, unlike what happens when plotting paths between cities on Earth, adding complexity. The determination of such trajectories is strongly dependent on the type of propulsion employed by the spacecraft:

- **Chemical Propulsion** — often referred to as "Impulsive". An impulsive trajectory manoeuvre is modelled as an instantaneous change in the spacecraft's velocity vector. This can be done because the burn time required to obtain the necessary ΔV (term that refers to the magnitude of the velocity change) is considerably lower than the time of flight of the respective trajectory *leg* [1]. Generally, the trajectory can be obtained through classical orbital mechanics methods (e.g. Lambert solvers).
- **Electric Propulsion** — often referred to as "Finite Thrust" or "Low-Thrust". The propulsion is modelled as the application of a small and continuous thrust force on the spacecraft, usually pro-

vided by ion, plasma thrusters or solar sails which use the solar wind's momentum to propel the spacecraft. To obtain the trajectory, numerical integration of the forces acting on the spacecraft is required, resulting in a procedure that is more complex to implement and compute than the impulsive case. Optimisation of this type of trajectories is more complex than in the impulsive case due to the larger number of optimisation variables in the problem and lack of exact solutions.

Furthermore, the trajectory can be of the direct type, when the spacecraft goes from the start body to the final body directly, or of the indirect type, when the spacecraft takes advantage of the angular momentum of other celestial bodies in order to speed up or slow down on the way to the final body. The latter manoeuvre is described as a *Gravity Assist* (GA) Manoeuvre or, more commonly, a *flyby*, and provides a change in the direction and magnitude of the velocity vector [1]. A set of bodies on the way to a final target will be referred to as a *flight sequence* or *flight path*. Additionally, due to the movement of the celestial bodies, in order to align the spacecraft to pass by the next planet in the sequence, another type of manoeuvre may be required — a *Deep Space Manoeuvre* (DSM). This manoeuvre is characterised by an impulse somewhere along the leg that links two bodies in a sequence.

By combining several celestial bodies in sets of Gravity Assists (GAs) and Deep Space Manoeuvres (DSMs), it may be possible to achieve savings in mission time or fuel mass — thus enabling higher payload capability — as the GAs are effectively providing "free" ΔV. This introduces computational complexity when determining the interplanetary trajectory, as using GAs vastly increases the number of possible combinations of travel paths that lead from body A to body B. If this is added to the diverse launch date possibilities, then exhaustively analysing all the paths becomes computationally demanding and unfeasible, from a practical perspective, due to high computation times.

Spacecraft trajectory optimisation allows the designer to obtain a *globally* optimum, or near optimum, solution for the problem in feasible time by using advanced optimisation algorithms to avoid analysing the entire search space (the space of feasible trajectories) [2].

Finally, if the goal is to be captured by the body at the end of an interplanetary mission, an *Orbit Insertion Manoeuvre* must be performed. This manoeuvre adjusts the velocity of the spacecraft so that it can be captured in the gravitational well of the destination body. Orbit insertion greatly depends on the spacecraft's *hyperbolic excess velocity* (V_∞), given by the velocity of the spacecraft relative to the target body.

1.2 Missions to Ocean Worlds

Besides Earth, there are now other objects in our own cosmic backyard — the solar system — that are thought to have oceans, some dynamic and others not, mostly located around the giant planets — Jupiter, Saturn and Neptune. The known objects vary in location and range and harbour two of the main ingredients for known life to thrive — liquid water and heat sources.

The two most promising ocean worlds, with strong evidences of dynamic water oceans and hydrothermal activity, are Jupiter's moon Europa [3] and Saturn's moon Enceladus [4]. These are the primary candidates for current mission concepts, as seen in the upcoming sections where we review some past and foreseen missions to ocean worlds.

Ceres

Ceres is a dwarf planet and the largest object located in the asteroid belt, beyond Mars. While Ceres is thought to have no geological activity, there is evidence of a sub-surface ocean [5].

The DAWN [6] mission, launched in 2007 by the National Aeronautics and Space Administration (NASA), is currently in orbit of Ceres. The spacecraft is entirely propelled by a solar-powered electric thruster and was the first spacecraft to orbit two different extraterrestrial bodies, Ceres and Vesta [6].

Jovian Moons

Three moons of Jupiter are thought to have liquid water bodies: Ganymede, Callisto and Europa [3, 7, 8, 9]. The largest of the moons, Ganymede, is theorised to have a salty water ocean with more water than Earth and benefits from geothermal activity that is caused by tidal forces [8]. These forces are the result of a slightly elliptical orbit in which gravity acts by stretching and compressing the moon, heating it. Due to the small eccentricity of its orbit, Callisto is not noticeably affected by tidal forces but is also thought to have some sort of sub-surface water ocean due to salt and water ice in its composition [7]. Europa is the most promising world of the set, with *cryogeysers* — jets made up of water vapour, ice crystals, carbon dioxide and methane — having been spotted erupting from its surface by the Hubble Space Telescope [10]. This leads to the theory that Europa might have a dynamic sub-surface ocean, with a high potential for life.

Europa and Ganymede are main targets for space missions currently being designed. NASA's Europa Multi-Flyby Mission (commonly known as "Europa Clipper") [11] is due to launch in the 2020's and ESA's Jupiter Icy Moon Explorer (JUICE) [12] will be the first mission to orbit an extraterrestrial moon (Ganymede) later that decade. Both spacecraft are chemically propelled platforms.

JUICE is a mission of special interest as it presents many similarities with the one being proposed in this work: multiple planetary flybys, a Jupiter-centred moon tour and the final goal of inserting a satellite into orbit of a moon within a giant planet's sphere of influence. Several trajectory alternatives are discussed in the JUICE's Consolidated Report on Mission Analysis (CReMA) [12], all of which present a combination of Gravity Assist Manoeuvres (GA) and Deep Space Manoeuvres (DSM) in the inner solar system in order to reach Jupiter with the least possible fuel consumption. In the list of trajectory alternatives proposed by the report, the most visited planets in the flight sequence are Earth and Venus, with Mars being present only once.

JUICE's CreMA [12] discusses the fact that a direct capture manoeuvre at Ganymede, with no flybys of the other moons in Jupiter's system (a so-called moon tour), would require a high ΔV : around 2.4 km/s for insertion into a 200 x 10000 km elliptic capture orbit, caused by a high V_∞ with respect to the moon. JUICE's moon tour was designed with the objective of mininimising this V_∞, which accomplished a total reduction of 1.7 km/s in the orbit insertion manoeuvre's ΔV but resulted in a complex trajectory design problem.

Cronian Moons

Saturn's moons provide two interesting candidates with potential for life: Titan and Enceladus [9].

Titan is the second largest moon in the solar system and the only one that maintains a dense atmosphere [13]. It is populated with lakes of liquid methane that may harbour extreme types of lifeforms [13]. It was explored in detail by the Cassini-Huygens mission [14, 15], which was launched in 1997 and is still active. The Huygens lander made valuable *in situ* observations of the surface of Titan. Most of the information available today about Saturn's moon system comes from observations made by Cassini-Huygens which has accomplished over 100 flybys of Titan since its arrival at the Saturn system in 2004 [14, 15]. Cassini is a chemically propelled spacecraft, electrically powered by a set of radioisotope thermoelectric generators (RTGs).

Enceladus is one of the worlds in the solar system with the highest potential for life [4]. Its surface is covered by water ice and there is evidence of thermal activity found by Cassini, when it observed *cryogeysers* erupting out of the moon's south pole. This leads to the theory that Enceladus may have a dynamic sub-surface ocean, with a considerable astrobiological potential, making the moon a prime candidate for astrobiology missions, with several concepts being designed [16].

Mission concepts to Saturn are abundant but haven't seen important investments as Jupiter missions have been given priority. The Journey to Enceladus and Titan (JET) mission (NASA) [17] (cancelled), would have performed multiple flybys of Titan and Enceladus to study them closely. NASA's Enceladus Life Finder (ELF) [18] and Life Investigation for Enceladus (LiFE) [19] are both astrobiology missions that ended up being scraped from the Discovery funding program. LiFE's concept included an ambitious return of samples from Enceladus' *cryogeysers* back to Earth. The German Space Agency (DLR) is also preparing for a mission to Enceladus. The Enceladus Explorer (EnEx) [20] concept includes a subsurface, ice-drilling probe and an orbiter.

NASA has produced two other mission concepts that are important. The first was a study for an Enceladus Orbiter, as a part of NASA's 2010 Decadal Survey [21], which served as base for many of the other concepts mentioned here. This study laid down important groundwork for a real mission, such as the study of novel techniques for the design of moon tours. The second concept is the Titan Saturn System Mission (TSSM) [22], an ambitious cooperation between NASA and European Space Agency (ESA) that was born from a merging of two missions, TandEM [23] and Titan Explorer. TSSM was initially planned to launch in the 2020's, on NASA's Space Launch System (SLS), but is currently on standby. This mission would carry three robotic vehicles to Titan: an orbiter, an atmospheric probe and a lander. In another innovative approach, TSSM would be propelled by two different systems—a traditional chemical system for manoeuvring at Saturn and a solar-electric system for the inner solar system. The use of solar-electric propulsion allows for a more flexible launch window, lower cruise time to Saturn and for a higher payload capability at the expense of extra cost and system complexity [22].

Enceladus is considered a prime target for upcoming missions due to the valuable science to perform in orbit of the icy moon [21], with interest growing within space agencies and scientists around the globe. This is one of the main motivations behind the choice of that world as the destination of this work's trajectory design.

Beyond Saturn

The outer regions of the solar system are not well explored. Main contributions to the knowledge of Neptune, Uranus and the Kuiper Belt came from Pioneer 10 and 11, Voyagers 1 and 2 and the New Horizons [24] spacecraft. Neptune's moon Triton may have a sub-surface ocean of liquid nitrogen as indicated by the presence of cryovolcanoes and geological activity spotted by the Voyager 2 probe on its way out of the solar system, but not much more is known [9, 25].

New Horizons [24], a chemically propelled platform, flew by Pluto in 2015 and retrieved information on the dwarf planet that is now being analysed. The probe observed that Pluto is covered in water ice and the planet may be thermally active if it is suffering from tidal forces caused by its moon — Charon. New Horizons will continue travelling into the Kuiper Belt until it runs out of power.

Summary

Table 1.1: Ocean Worlds—A summary

Type	World	Status	Previous Missions	
Planet	Earth	Only world known to harbour life.	-	
Dwarf Planet (Asteroid Belt)	Ceres	Evidence of a sub-surface ocean with an unknown potential for life.	DAWN [NASA]	-
Moon (Jupiter)	Ganymede	May have trapped salty water ocean with a low potential for life.	Galileo [NASA] Voyagers 1,2 [NASA] Pioneers 10,11 [NASA]	Europa Clipper [NASA] JUICE [ESA]
	Callisto	May have a trapped water ocean with a low potential for life.		
	Europa	Strong evidence of a dynamic sub-surface ocean that could support life.		
Moon (Saturn)	Titan	May have a trapped ocean with a low potential for known life. Has surface lakes of hydrocarbon compounds that could harbour novel forms of life.	Cassini - Huygens [NASA/ESA] Pioneer 11 [NASA] Voyagers 1,2 [NASA]	JET [NASA] TSSM [NASA/ESA] LiFE [NASA] ELF [NASA] EnEx [DLR] Enceladus Orbiter [NASA]
	Enceladus	Strong evidence of a dynamic sub-surface water ocean, with cryogeysers having been spotted erupting from the surface of the moon, indicating the presence of heat sources. Potential for life: high.		
Moon (Neptune)	Triton	May have a liquid nitrogen ocean, as suggested by cryovolcanic eruptions. Unknown potential for life.	Voyager 2 [NASA]	-
Dwarf Planet	Pluto	May have a trapped ocean with an unknown potential for life.	New Horizons [NASA]	-

Missions to ocean worlds are being approached with great excitement by major space agencies around the globe, due to their potential for ground-breaking science and innovative technological challenges.

Most missions to the outer solar system use a combination of flybys and DSMs to reach their destinations, with Venus and Earth being the most used planets, in the inner system, to perform the gravity assist manoeuvres. If the final destination is a moon, complex sets of GAs, known as moon tours, are common practice to reduce the spacecraft's velocity with respect to the target object, lowering the mass of fuel required to be captured by the moon.

1.3 Spacecraft Trajectory Optimisation

Spacecraft Trajectory Optimisation is defined by the determination of the trajectory of a spacecraft while minimising, or maximising, a certain quantity of interest, within a set of constraints or boundaries [2]. Optimisation goals differ but, in the case of interplanetary travel, the most common objectives are to either reduce mission time or maximise payload capability (by minimising fuel mass) [2]. Since these two objectives are in opposition—trajectories that minimise flight time usually result in higher fuel consumptions—the problem can also be defined as a trade-off, or pareto problem. This type of optimisation problems are often subject to constraints, such as box ([minimum,maximum]) constraints on time of flight and maximum allowed launch energy.

Exhaustive search through the phase space of possible travel paths is often unfeasible due to high computational demands and long calculation times, prompting the need for global optimisation methods that help find a globally optimal, or near-optimal, solution in the least time possible. Some of these methods will be reviewed in Section 1.3.2.

Furthermore, these problems are characterised by a high degree of non-linearity and discontinuities in the spacecraft's state variables (as is the case with the instantaneous change in velocity in DSMs and GAs) [2]. When more complex models are required to adequately describe the physical problem, such as in the presence of time-dependent forces or orbital perturbations and multiple-body effects, the optimisation problem increases in difficulty. Finite thrust missions also add the direction, magnitude and thrust interval as additional optimisation variables [2].

1.3.1 GTOC: Global Trajectory Optimisation Competition

The Global Trajectory Optimisation Competition (GTOC) [26] is an important event where trajectory designers from around the world attempt to tackle "nearly-impossible" trajectory optimisation problems, with the goal of advancing the field with novel techniques and providing solutions for benchmarking new algorithms. Every year the contest is organised by the winner of the previous edition, with the first edition having been proposed by ESA's Advanced Concepts Team. This competition is introduced here because many of the methods discussed hereafter have been proposed in the framework of the competition's state-of-art problems. GTOC is currently in its 9th edition.

1.3.2 Methods

The spacecraft trajectory optimisation methods presented here will be divided in three main groups:

- Extensive Search;
- Deterministic Methods;

- Stochastic Methods.

Extensive Search

Extensive search algorithms are “brute-force” methods to solve an optimisation problem. The algorithm searches the entire space of possibilities, evaluating the objective function of each solution until it finds the global minimum.

For the problem of global spacecraft trajectory optimisation, this is not feasible due to the very large set of possible trajectories, resulting in prohibitive computational times.

Deterministic Methods

Deterministic methods are systematic procedures that take advantage of the mathematical description of problems in order to find minima. The most common are direct and indirect methods.

Indirect methods are based on variational calculus and make use of Pontryagin’s Minimum Principle [27]. A two-point boundary value problem is formulated and solved, obtaining a solution of the control variables with time. This method requires knowledge of the system’s dynamics equations and is very sensitive to the quality of the initial guess [27], introducing a lack of flexibility in its use.

Direct methods involve the parametrisation of time-dependent control variables and search for local minima by adjusting the state and control variables directly. Once the problem is formulated, it can be solved using Non-Linear Programming [2]. Direct methods find approximate solutions of a problem but are less sensitive to initial guesses and do not require an analytical differentiation of the constraint equations making them easier to use and more robust [27]. Direct and indirect methods are suited for local optimisation but can be applied to global optimisation problems, usually grouped with other algorithms that narrow the search space.

Betts [27] provides an overview of some traditional methods employed in spacecraft trajectory optimisation, discussing the advantages and disadvantages of each, with special focus on gradient-based methods. He argues that the most commonly used traditional methods are direct shooting, indirect multiple shooting and direct transcription. Direct shooting is widely employed and effective for launch vehicle and orbit transfer applications, excelling at problems with small sets of parametrised variables [27]. Indirect multiple shooting provides a boost in robustness from the simple shooting methods and allows for computational parallellisation, result of the division of the problem into arcs [27]. Direct transcription is robust, versatile and allows for parallellisation, making it preferable to be applied in global optimisation problems [27].

While the traditional methods reviewed by Betts [27] are still somewhat employed today, research in the field has provided engineers with novel algorithms and tools to perform trajectory optimisation.

A more recent review of global optimisation methods for trajectory design was performed by the European Space Agency [28], presenting novel deterministic algorithms such as DIRECT, a deterministic global optimiser for bound constrained optimisation and Multilevel Coordinate Search (MCS), an algorithm based on DIRECT that applies a flexible branching scheme to sub-partition the search space.

MCS performed successfully in solving impulsive interplanetary transfers and Multi-Gravity Assist (MGA) problems with a reduced number of gravity assists.

A direct transcription method for the optimisation of low-thrust trajectories with MGAs [29] has been presented, in which the trajectory is divided into legs where the propulsion is modelled as a series of small impulses. In comparison to other gradient-based methods, the “Sims-Flanagan” method was found to be more robust and handle more gravity assists. This method was successfully improved on [30] to include the full dynamics of low-thrust trajectories.

Casalino et al. [31] developed an indirect method for the optimisation of low-thrust trajectories, having applied it to participate in the two first GTOCs (a multi-gravity assist asteroid impact and a multiple asteroid rendezvous). The trajectory is divided into arcs that can be of three types — coast, thrust or flyby — which are then joined to provide an initial trajectory guess for optimisation. The algorithm allows for parallelisation but the downside is that sequencing has to be defined *a priori*. The indirect method is not fit to solve the first GTOC problem due to the great number of flybys while being well suited for the DAWN-type asteroid rendezvous mission, as it involves less ballistic and more thrust profiling, obtaining by far the winning solution. This method offers a very high accuracy on the numerical solutions obtained.

Branch, Bound and Prune [32] methods have been employed to attack the 5th GTOC problem — a low-thrust asteroid rendezvous mission. The algorithm first prunes the large search space by applying constraints to the energy required to go from one asteroid to another, eliminating high energy transfers. It then proceeds to use a traditional branch and bound method that systematically runs through the search space of candidate trajectories. This method can be seen as a tree that starts with the initial celestial object and each level is the next candidate in the flight sequence. The trajectory is locally optimised at each level and the unfeasible branches of the tree are pruned, leading to a smaller search space. The local optimisation is performed with an indirect method. This procedure achieved a third place in the 5th edition of the competition.

Shape-based direct methods [33, 34] have been used to solve finite-thrust problems with gravity assists by means of an exponential sinusoid shape approximation to obtain estimates for the trajectory, performing a global search and then optimising candidates with the use of direct methods. The sinusoid approach appears to be robust and provides the user with trajectories that can be implemented in practice.

Stochastic Methods

It is possible to distinguish Evolutionary Algorithms (EAs) from other optimisation methods for two distinctive features [35] — they are population based and there is communication and exchange of information between individuals in a population, through evolutionary operations such as mutation, crossover and selection.

Additionally, there are other stochastic methods that rely on behavioural analysis [36], such as Particle Swarm Optimisation (PSO) and Ant Colony optimisation, which derive their strategies from the behaviour of flocking birds and ant foraging by pheromone behaviour, respectively. Other, less used methods, include Monte-Carlo search, Invasive Weed Optimisation (based on weed growth and reproduction), Harmony Search, Gaussian adaptation (based on information theory), among several others.

EAs have various advantages over deterministic methods [36, 37]: they do not require an initial guess, as the initial conditions are randomly generated, they are easy to implement and converge faster to global optima, despite being computationally more demanding, and are overall better at avoiding local minima. The downside to EAs lies with their random nature — these algorithms are stochastic and therefore introduce a degree of non-systematic behaviour in the system with the addition of not having a convergence criteria to ensure the optimality of the solutions obtained. Furthermore, these strategies have difficulty in treating constraints [37], with penalty methods being applied to filter out unfeasible trajectories [38].

The application of EAs is more efficient when the search space is smaller and the problems can be represented by the least possible number of optimisation variables, therefore their application to spacecraft trajectories leans more towards impulsive trajectories, with research in the finite-thrust field expected to be stronger as electrical thrusters and solar sails become more advanced.

Betts [27] dismisses the usefulness of EAs due to their random, non-systematic, nature and main applicability to problems with discrete variables. Nevertheless, significant subsequent research and development has been put into evolutionary algorithms which are now being used to solve complicated trajectory optimisation problems with great success such as the ones proposed by the GTOC.

ESA's review [28] also investigates the usefulness of stochastic algorithms, specifically a Simple Genetic Algorithm (SGA), Differential Evolution (DE), PSO and Multiple PSO. The report concludes that DE performs better at solving MGA-DSM problems than every other deterministic and stochastic algorithm investigated, with MPSO ranking second in MGA-DSM problems and first in MGA problems without the inclusion of DSM's. This work also concludes that even the best algorithms find difficulties in optimising MGA-DSM trajectories efficiently and consistently. Therefore, the authors propose GASP [39] — a Gravity Assist Search Space Pruning algorithm, with the objective of narrowing the search space by removing unconvenient and unfeasible trajectories (such as those with a very high launch energy). The implementation of GASP is determined to be a success in helping the algorithms in converging to the global minimum.

Izzo and Yam [40] compare and apply three different evolutionary algorithms to obtain a final result that improves significantly on the winning solution of GTOC's 6th edition (which uses indirect methods [41] to solve a Jupiter moon tour). The procedures compared are: Differential Evolution with Self-Adaptation (jDE), Adaptive Neighbourhood Simulated Annealing (SA-AN) and Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES), in a new paradigm the authors codenamed asynchronous island model. They conclude that jDE performs better for this problem because of its good interacting with the island model and its low run time. A branch and prune scheme is then successfully implemented to search for the best trajectory sequences in the moon tour.

Hemoglop [36, 42] is a cooperative evolutionary strategy that joins three EAs — DE, SGA and PSO — to achieve a better efficiency and convergence than the algorithms would have when separated. The cooperative algorithm was tested on MGA and rendezvous problems, with results showing improvement over the independent algorithms. The benchmarks also pointed Differential Evolution as the strongest independent algorithm of the three tested. Following up, Hemoglop was merged with an indirect method developed by Casalino et al. [31], with the cooperative strategy providing initial estimates for the local optimiser. The results indicate that the algorithm converges to the global minimum and the new procedure diminishes the designer experience requirement of using indirect optimisers for complex problems.

Vasile and Pascale [43] introduced a hybrid evolutionary search algorithm. A Simple Genetic Algorithm is used to explore the solution space and then a branching scheme partitions the domain into subdomains, pruning sub-optimal branches. This approach worked well with Cassini and Rosetta-type problems, avoiding local minima.

In a similar approach [44], a hybrid SGA + Non-Linear Programming (NLP) algorithm is compared with a Branch + SGA algorithm, finding that SGA+NLP wields the best results when solving a multi-asteroid interception mission, due to the search benefitting from SGA's capability to be attracted to the basin of the global minimum, avoiding local minima in the process. In a second revision of this work, Wall and Conway [45] attack the multi-asteroid interception problem with focus on genetic algorithms. It finds that SGA+SGA schemes are more time efficient than the ones previously studied [44], yielding near-optimal results.

Vasile, Minisci and Locatelli [46] tested some methods for Evolutionary Algorithms in Spacecraft Trajectory Design, setting a baseline strategy for testing different algorithms and proceeding to compare several of them — specifically the Simple Genetic Algorithm (SGA), Monotonic Basin Hopping (MBH), Differential Evolution (DE), Particle Swarm optimisation (PSO) and Multi-Start (MS) — in various types of global optimisation problems. It is found that the method that works better for MGA problems is MBH. MBH is a meta-algorithm that samples the search space at random and then optimises each sub-domain locally by using a local optimiser such as Sequential Least Squares Programming (SLSQP).

IDEA [47] (an Inflationary Differential Evolution Algorithm) develops on the results from the benchmark of EAs [46], coupling DE with MBH and introducing a smart restart procedure that improves performance by reducing the probability of stagnation in a local basin. This procedure was tested against Cassini and Rosetta-type problems (MGA with DSMs and without), improving on the robustness and overall performance of the standard DE algorithm by converging repeatedly to objective function values close to the best known solutions.

Building on the same benchmark [46], a two-phase Multi-Start / MBH algorithm was designed [48]. This strategy inserts a MBH middle phase between Multi-Start (global search) and the local optimiser (local search) to refine the search space before proceeding with local optimisation, improving the algorithm's efficiency. The local optimiser used is SNOPT, a SLSQP implementation. The results of this strategy improved on the Cassini, Rosetta and Messenger solutions present in ESA's Global Trajectory Optimisation Problem (GTOP) database, a repository of the best known solutions for trajectory optimisation problems.

EAs are also capable of defining the sequence of bodies to visit as optimisation variables. An example of this is the variable chromosome length evolutionary algorithm [49], which is an adaptation of a Simple GA that allows for the simultaneous existence of trajectories with different sequences in the same population. To ease the computational demands, a first optimal set of trajectories is calculated in a wide search space with no DSMs allowed, which are then added in a second run of the algorithm when the sequence of events is already fixed. This strategy revealed good results when tested with MGA problems from GTOC competitions.

Evolutionary Algorithms have also found some applications in low-thrust trajectory optimisation. Genetic algorithms can be coupled with indirect methods [50], evolutionary neuro-control methods [51] have been applied to optimise simple trajectories in solar sail spacecraft (without gravity assists) and, more recently, genetic algorithms with shape-based approaches [52] have been successfully implemented to solve finite-thrust MGA problems.

General Considerations

Stochastic methods and variants are being primarily used for solving complex impulsive trajectory problems, involving multiple flybys and deep space manoeuvres, due to their ability to avoid local minima and their ease of use and implementation. Differential Evolution displays the highest robustness of all the standard algorithms with variants, such as IDEA [47] and cooperative algorithms [42], improving on its performance. Strategies based on Monotonic Basin Hopping are also showing very promising results.

Additional techniques, such as search space pruning also contribute to a better quality of the optimisation of MGA-DSM problems.

Deterministic methods are very capable of handling impulsive problems despite being less apt than EAs at avoiding local minima. Furthermore, deterministic methods have found a strong niche in low-thrust spacecraft trajectory optimisation, due to the high amount of optimisation variables involved and systematic nature of the methods, currently being preferred over evolutionary alternatives in this field. Indirect methods are very robust, as seen from the winning solutions of various GTOC editions, but require significant trajectory design experience and suffer from a lack of flexibility on implementation, making them harder to use for general purpose optimisation tools, with direct methods such as Direct Transcription with Shape Based methods, which allow for parallelisation, being preferred.

Chapter 2

A Mission to Enceladus

The objective of this mission's trajectory design is to deliver a space probe to Enceladus' orbit with the highest possible payload capability, in useful time, thus maximising the potential for collecting scientific data. Maximisation of the payload capability of the probe can be achieved through the minimisation of the spacecraft's fuel consumption, as this allows for an increase of the dry mass for the same allowable launch mass (equation 2.1), effectively improving the maximum mass of scientific instruments that the probe can carry to its destination,

$$\text{Launch Mass} = \text{Dry Mass} + \text{Fuel Mass}. \tag{2.1}$$

Furthermore, the launch mass is constrained by the performance of each specific launcher, that is fixed by the departure geometry, and must also be taken into account when maximising dry mass. Launcher performance will be approached in greater detail later in Section 3.4.

Fuel consumption can be directly related to the mission's ΔV budget (the sum of all the ΔVs), through the application of Tsiolkovsky's Rocket Equation (2.2) [1] to each manoeuvre,

$$\Delta V = I_{sp} g_0 \ln \frac{m_i}{m_f}, \tag{2.2}$$

where I_{sp} is the specific impulse of the propulsion system, g_0 is the standard gravity parameter, m_i is the spacecraft's mass before the manoeuvre and m_f is the mass after the manoeuvre has been completed. If there are no changes to the structural mass or payload of the spacecraft, then the difference between the initial and final mass is the spacecraft's fuel consumption for a given orbital manoeuvre. This indicates that minimising fuel consumption for a given spacecraft and trajectory is equivalent to minimising ΔV.

Provided that the mission can be divided into n phases, then the objective can be translated as finding a trajectory, or set of trajectories, that departs from Earth and reaches Enceladus within a certain useful time frame, while minimising the sum of the ΔV of all phases.

This chapter will provide an overview of the systems, procedures and other considerations involved in the design of the mission to Enceladus.

2.1 Mission Overview

The mission to Enceladus is divided in three main phases, each describing a different problem to be solved, as seen in Table 2.1.

Table 2.1: Mission Phases and their high-level objectives

i	Phase	Objective
1	Interplanetary Transfer to Saturn	Define Launch Geometry and trajectory that minimises interplanetary ΔV and optimises Saturn Orbit Insertion.
2	Saturn Orbit Insertion	Define and optimise capture geometry (orbit insertion and periapsis raising manoeuvres).
3	Saturn Moon Tour	Minimise Saturn-centred and Enceladus Orbit Insertion ΔV's.

The need for a moon tour arises from the necessity of lowering the cost of orbit insertion at Enceladus. A direct Hohmann transfer between Titan and the icy moon would result in an Enceladus Orbit Insertion ΔV of approximately 3.53 km/s (for a 200 km circular orbit), a prohibitive cost. For comparison purposes, TSSM's interplanetary cruise trajectory has a cost of around 2.75 km/s [22]. The moon tour works by gradually decreasing the periapsis of the spacecraft's orbit after insertion to the orbit of Enceladus. This is accomplished by stringing together transfers of the intermediate moons (Titan to Rhea, Rhea to Dione, Dione to Tethys and Tethys to Enceladus) [53]. The orbital insertion manoeuvre is proportional to the spacecraft's velocity with respect to Enceladus at the time of encounter. The variation in periapsis of the spacecraft's orbit is accompanied by a reduction of this relative velocity.

The overall design starts with fixing a suitable time frame for the mission to take place by setting bounds on the cruise duration and finding a realistic launch window (for specifics, see Chapter 5).

As the spacecraft launches from Earth, the mission starts. The first phase is defined by an interplanetary cruise with multiple gravity assists, in which the objective is to reach Saturn with the lowest ΔV possible, within the allotted time frame for the cruise's duration. At the end of this phase, the probe will perform the Saturn Orbit Insertion (SOI) manoeuvre so that it can be captured into a highly elliptic orbit around Saturn. Following the SOI, a Periapsis Raising Manoeuvre (PRM) is performed to move from the post-capture elliptic orbit into an orbit where an encounter with Titan will occur — then, the Moon Tour begins. Titan is chosen to begin the moon tour because it is the largest moon in Saturn's system (see Section 2.3) and, consequently, allows a higher degree of design options when performing gravity assists.

2.1.1 Mission Design Approach

The mission to Enceladus will be divided into several components that will be approached in greater detail throughout this work.

From a global optimisation perspective, it would be best to optimise the entire mission at once but that is computationally demanding and complicated to implement. As a consequence, each phase of the mission will be treated independently, with results from the previous phase being fed into the next, as described in figure 2.1, which specifies how the different components of the design are related to each other and how the information is exchanged between them.

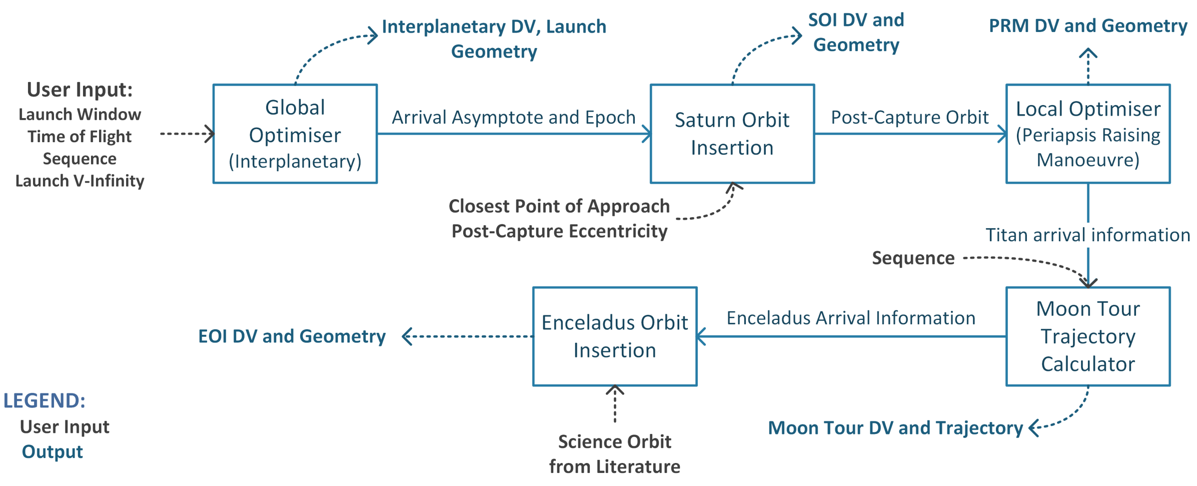

Interplanetary DV, Launch Geometry
SOI DV and Geometry
PRM DV and Geometry
User Input:
Launch Window
Time of Flight
Sequence
Launch V-Infinity
Global Optimiser (Interplanetary)
Arrival Asymptote and Epoch
Saturn Orbit Insertion
Post-Capture Orbit
Local Optimiser (Periapsis Raising Manoeuvre)
Closest Point of Approach
Post-Capture Eccentricity
Titan arrival information
Sequence
EOI DV and Geometry
Enceladus Orbit Insertion
Enceladus Arrival Information
Moon Tour Trajectory Calculator
LEGEND:
User Input
Output
Moon Tour DV and Trajectory
Science Orbit from Literature

Figure 2.1: Interconnections between the mission design's various components.

As stated in Section 2.1, the design starts with the user constraining the problem by defining a suitable time frame for the mission to take place. Additionally, the user also provides a maximum launch energy (associated with the launch hyperbolic excess velocity) and the sequence of planets that the spacecraft will visit during its path to Saturn. This is then placed into a global optimiser, described in detail in Section 5.2, that will return optimal trajectories within the constraints that were set by the user.

As a result of the global optimiser, the interplanetary trajectory is totally defined, including the ΔV, the launch conditions at Earth and the arrival conditions at Saturn. From here, it is possible to compute the geometry of the spacecraft's capture in an orbit around Saturn by using a B-Plane approach (see Section 5.3). At the apoapsis of the newly determined post-capture orbit, a Periapsis Raising Manoeuvre (PRM) is required to have the spacecraft meet Titan, starting the Moon Tour phase. This manoeuvre is locally optimised (see Section 5.3.2) with the objective of reducing the cost associated with it while maintaining a resulting trajectory that is suitable to begin the moon tour with.

The relevant information and user input is then transferred to the block that computes the moon tour (see Chapter 6). Here, a moon tour based on Tisserand graphics and a V_∞-leveraging technique (VILT), is computed, obtaining an approximate moon tour trajectory and an estimate of the ΔV required to perform it.

The mission ends with a capture manoeuvre around Enceladus that mirrors the procedure performed in the Saturn orbit insertion. The final orbit is described in Section 6.3.

This procedure is, ideally, iterative because a change in the interplanetary phase, such as arrival date, could have a significant impact on the quality of the solution found for the following phases.

2.2 The Solar System

The Solar System is the theatre for the mission's first phase. Table 2.2 displays the relevant planetary body information [54] that is used in this work.

Table 2.2: Planetary body information. The radii used are mean volumetric radii.

Body	Orbital Radius (AU)	μ (km^3/s^2)	Period (Years)	Orbital Inclination (°)	Radius (km)	Safe Flyby Radius
Sun	—	1.33E+11	—	—	—	—
Venus	0.723	324860	0.615	3.39	6051.8	1.05 R_{Venus}
Earth	1	398600	1	0	6371	1.03 R_{Earth}
Mars	1.524	42828	1.881	1.85	3389.5	1.05 R_{Mars}
Jupiter	5.204	1.27E+08	11.8	1.3	69911	5 $R_{Jupiter}$
Saturn	9.58	3.79E+07	29.4	2.49	58232	2.5 R_{Saturn}

Note that Jupiter has a higher safe flyby radius due to the presence of hazardous radiation (Jupiter's Van-Allen belts) and Saturn presents a higher safe flyby radius due to the presence of the rings. Furthermore, the planets are all approximately in the same orbital plane, meaning that a 2D approach could be applied to the problem, although this will not be the case in this work.

Also noteworthy is the fact that Mars' low gravity doesn't make it a very suitable planet to perform gravity-assist manoeuvres in the solar system. Venus, Earth and Jupiter will be preferred for flybys. While this would exclude an opportunistic Mars flyby, it also reduces the complexity of the problem.

2.3 The Saturn System

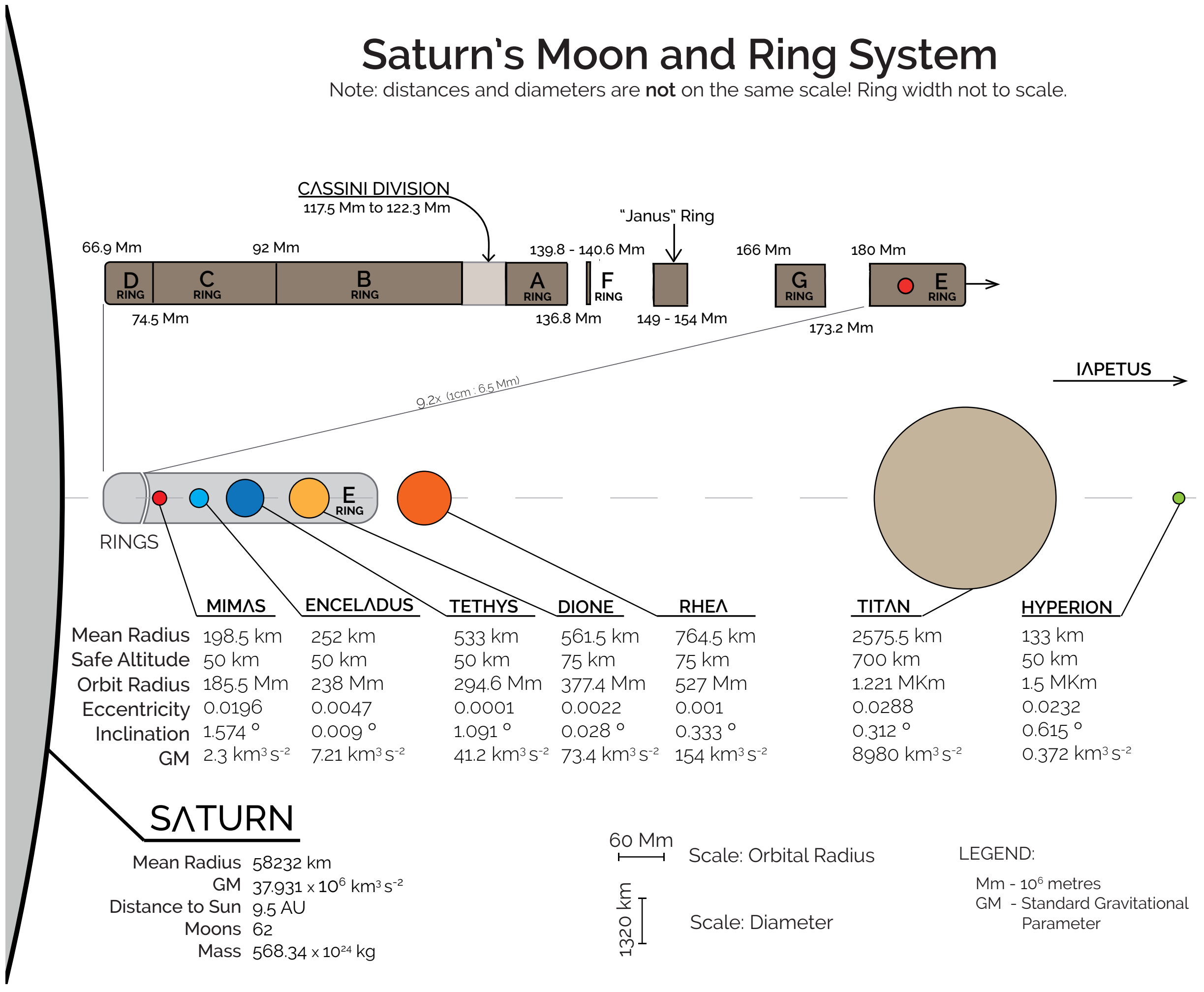
The Cronian system is comparable to the Solar System in the way that it has a central body (Saturn) and multiple smaller bodies (moons) orbiting around it. Saturn has 62 known satellites [54] of which 5 are of direct interest to the mission: Enceladus, Tethys, Dione, Rhea and Titan. In addition, the gas giant is populated with a dynamic ring system that poses a danger to spacecraft that cross it so collisions must be avoided whenever possible. Figure 2.2 displays a scaled image with important information about the celestial bodies of interest in Saturn's Moon System.

From Figure 2.2 it's visible that the rings exhibit gaps through which the spacecraft can travel with relative safety. Enceladus is located within the E ring, a diffuse structure mainly composed of faint dust [55, 56] that doesn't present a considerable risk to the safety of the probe — the Cassini spacecraft has performed multiple safe flybys of Enceladus and other moons in the E ring. Rings A, B, C and F are the densest of Saturn's rings [55, 56] and should be avoided for navigation. Cassini crossed the ring plane between the F and G rings, during its Saturn orbit insertion manoeuvre [15, 22].

The Cassini Division is a 4800 km-wide structure within the rings that is less dense than the rest and is riddled with small gaps, the greater of which is the Huygens gap, about 400 km wide [55, 56].

As is the case with the Solar system, all the moons in Saturn's system are approximately in the same orbital plane, making it possible to use a 2D approach to design preliminary trajectories in this system.

The estimate of the safe flyby height is based on literature [22, 53, 57] and accounts for navigation error margins, uncertainties in Titan's atmospheric model and in the gravity models of the moons. As other missions and studies improve on the existing knowledge of each body, these safety heights can be adjusted — Cassini has safely performed flybys of Enceladus at 25 km [15] of altitude.



	MIMAS	ENCELADUS	TETHYS	DIONE	RHEA	TITAN	HYPERION
Mean Radius	198.5 km	252 km	533 km	561.5 km	764.5 km	2575.5 km	133 km
Safe Altitude	50 km	50 km	50 km	75 km	75 km	700 km	50 km
Orbit Radius	185.5 Mm	238 Mm	294.6 Mm	377.4 Mm	527 Mm	1.221 MKm	1.5 MKm
Eccentricity	0.0196	0.0047	0.0001	0.0022	0.001	0.0288	0.0232
Inclination	1.574 °	0.009 °	1.091 °	0.028 °	0.333 °	0.312 °	0.615 °
GM	2.3 $km^3 s^{-2}$	7.21 $km^3 s^{-2}$	41.2 $km^3 s^{-2}$	73.4 $km^3 s^{-2}$	154 $km^3 s^{-2}$	8980 $km^3 s^{-2}$	0.372 $km^3 s^{-2}$

SATURN

Mean Radius	58232 km
GM	37.931 x 10^6 $km^3 s^{-2}$
Distance to Sun	9.5 AU
Moons	62
Mass	568.34 x 10^{24} kg

Figure 2.2: Saturn System: Celestial body information

2.4 Spacecraft

The spacecraft's function is to safely carry all the scientific instruments to Enceladus and then relay the data back to Earth. This section will give the reader an overview of the common structure of an interplanetary spacecraft, including typical mass, power systems, propulsion and others.

A typical spacecraft can be divided into the following components [58]:

- **Structure**: the structure where all the subsystems and instruments are positioned. Protects the probe during launch and against space environment.
- **Power**: Subsystem responsible for storing and distributing energy in the spacecraft.
- **Thermal**: Subsystem responsible for keeping the spacecraft within operational temperature range.
- **Attitude and Orbit Control**: Subsystem responsible for pointing and stabilizing the spacecraft.
- **Propulsion**: Spacecraft engines and fuel storage.
- **Electronics and Communications**: On-board computers, data transmission and sensors.
- **Payload**: Scientific instruments or other type of payload.

Table 2.3 shows data on some missions (or mission concepts) to the outer planets.

Table 2.3: Interplanetary spacecraft data.

Mission	Mission Duration	Dry Mass (kg)	Wet Mass (kg)	Fuel to Mass Ratio (%)	Power	Propulsion
Cassini-Huygens [15]	20 years	2445	5712	57.2	Nuclear	Chemical
TSSM [22]	14 years	3127	6203	49.6	Solar Electric + Nuclear	Ion + Chemical
Decadal Survey [21]	12.5 years	1150.4	3560.2	67.7	Nuclear	Chemical
JUICE [12]	11 years	1800	4362	58.7	Solar Electric	Chemical
Europa Clipper [11]	9 years	1349	2354	42.7	Solar Electric	Chemical
Juno [59]	9 years	1593	3625	56.1	Solar Electric	Chemical
New Horizons [60]	9.5 years	401	478	16.1	Nuclear	Chemical
Galileo [61]	22 years	2719	5712	52.4	Nuclear	Chemical

From Table 2.3 it is possible to verify that, while typical spacecraft masses vary according to mission specifications, the fuel to wet mass ratio is frequently around 40% to 60%, hence the importance of minimising fuel consumption: payload and spacecraft can then occupy a larger fraction of the launch mass. TSSM, for example, has a higher dry mass than Cassini and a lower fuel to launch mass ratio due to its use of low-thrust propulsion for the interplanetary trajectory, resulting in a lower fuel consumption thanks to the higher efficiency of this propulsive method [62].

Spacecraft that are designed to reach the outer solar system (beyond Jupiter) are electrically powered by nuclear devices such as Radioisotope Thermoelectric Generators (RTGs) (Cassini and New Horizons) [15, 60] or modern Advanced Stirling Radioisotope Generators (ASRGs) (TSSM) [22]. ASRGs have a higher recommended use time (17 years) than RTGs and provide the same power at a fraction of the plutonium required. Beyond Jupiter, the flux of solar energy is so low that employing solar electric power at that distance, with current technology, would result in large, heavy and hard to pack solar panels. The Juno spacecraft, which arrived recently at Jupiter, is at the limit of current technology and employs two 9m x 2.65m solar arrays [59]. Each array had to be folded four times to fit into the launcher, increasing the risk for malfunctions during deployment. All of this indicates that the use of solar electric power at Saturn would be highly inconvenient, with nuclear power being preferred for a mission to Enceladus, despite the obvious caveats associated with handling nuclear material.

2.4.1 Propulsion

The choice of the propulsion system is a critical step of the mission's design as it will affect the entire structure of the spacecraft and of the subsystems. More importantly, as far as this work is concerned, it affects the trajectory characteristics and thus the choice of the trajectory design methodology.

Rocket propulsion (chemical) can be modelled as an instantaneous change in velocity (ΔV). The trajectory before and after the "impulsive" manoeuvre is known and can be determined, in a relatively simple manner, by using Kepler's equations for conics (elliptic, parabolic and hyperbolic orbits). As for low-thrust propulsion, it is modelled as the application of a finite force acting on the spacecraft, during a certain period of time. This requires the trajectory to be numerically integrated to find the position of the spacecraft along its path, resulting in a more complex procedure. Additionally, low-thrust trajectories are inherently multi-revolution (more than one revolution around the central body) whereas impulsive trajectories are usually single revolution, further complicating the design methodology. Figure 2.3 displays a

Table 2.4: Comparison of propulsion systems [22, 62, 63]

System	Low-Thrust	Impulsive
Advantages	Greater launch window flexibility Shorter time of flight Very high fuel efficiency	Lower system cost and complexity Lower trajectory design complexity Lower Energetic Consumption

typical low-thrust trajectory and a typical impulsive trajectory (with deep space manoeuvres).

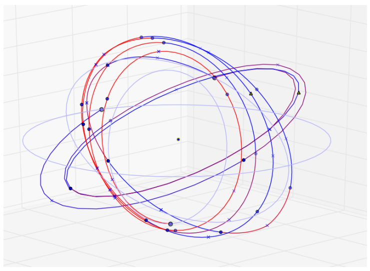
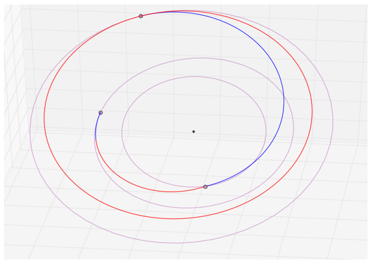



Figure 2.3: Comparison of a low-thrust path (left) and an impulsive path (right).

Each propulsion system has its own advantages, as shown in Table 2.4 [22, 63]. Thanks to their multi-revolution nature, low-thrust trajectories allow a greater launch window flexibility as it is easier to adjust the trajectory to account for launch delays. On the contrary, impulsive trajectories require mid-trajectory correction manoeuvres to correct these errors, which may result in a higher cost. Finite thrust trajectories also boast a generally shorter time of flight due to their ability to continuously and efficiently accelerate the spacecraft to higher velocities than their impulsive counterparts [22, 62].

Additionally, electric and plasma thrusters have a very high specific impulse (e.g. 4190 seconds for TSSM's NexT thruster [22]) when compared to traditional chemical engines (e.g. 312 s for Juno's [59] Leros-1b hydrazine thruster). Remembering equation 2.2, this means that the required fuel consumption, in terms of mass, is lower for the same requested ΔV.

Electric propulsion, however, has a critical downside. Because ion and plasma thrusters are mainly based on the principle of accelerating particles to high speeds, this means that the energetic consumption of these thrusters is somewhat higher than that of conventional chemical engines [62]. As seen in Section 2.4, a spacecraft on a mission to the outer planets, with the current technology level, will most likely employ nuclear power rather than solar electric power. Current RTGs and ASRGs have a lower power output that is not capable of powering both the spacecraft's subsystems and its thrusters [62], unless multiple are used which would drive the launch mass and risk factors up.

Due to the electrical considerations, **chemical** propulsion will be used for the entirety of this mission but the alternative of using a hybrid system — low thrust for the interplanetary cruise and impulsive for Saturn — remains open to be studied in a future work.

Besides lower power consumption, chemical propulsion systems are simpler and less complex and are backed by many years of development and proven use in space missions, lowering the risk associated with their use. The use of this propulsion type is also accompanied by an easier trajectory design methodology.

Chapter 3

Mission Design Models

The mission models are the building blocks that were implemented throughout this work, such as orbital mechanics, ephemerides, B-Plane geometry for orbit capture around Saturn and Enceladus, Tisserand Plots and a in-depth description of the implemented V_∞-Leveraging Technique.

It is important to state beforehand that all the models that were used in this work are valid under a patched conics approximation [1], meaning that the spacecraft is only under the influence of the gravitational pull of one dominant body at each instant and celestial bodies are modelled as point-masses, with no sphere of influence. As a consequence of this, any change in the path of the spacecraft (such as a flyby or deep space manoeuvre) is instantaneous and occurs at one point in space. This technique has been traditionally used with success in real-life preliminary spacecraft trajectory design (such as the previously mentioned GTOC) due to the, typically, large distances between bodies [1, 37]. All other perturbations such as solar pressure, oblateness effects and n-body perturbations were not considered.

The trajectories obtained via the patched conics method can then be inserted, as initial guesses, into full physics simulators, that account for all those external effects, to be numerically corrected to find the real trajectory of the spacecraft.

3.1 Rendezvous: Lambert's Problem

Lambert's problem, named after the physicist who established the theory, is defined as a boundary-value problem: given a start position, an end position and a time of flight, how can two points in space be connected? [63]

Lambert's theorem, in its original form, states that the time of flight $(t_2 - t_1)$ that the spacecraft takes to travel an orbit from P_1 to P_2 is a function of the orbit's semi-major axis (a), of the sum $\mathbf{r_1} + \mathbf{r_2}$ and of the length c which completes a triangle with r_1 and r_2 as sides [63]. The solution of this boundary-value problem is a set of velocities, fully defining the transfer orbit that connects the two points. All the relevant equations and procedures are very well described in literature [63, 64] and will not be presented here as they are not relevant to this work.

These solvers are especially useful to define transfers between planets and satellite orbits, within a two-body approach, and allow the designer to build useful preliminary design tools such as the "pork-chop" diagrams [63], that can be used to quickly analyse the space of direct orbits that connect two points in space. Due to the elevated interest in this type of situations, many computational formulations have been created to solve the rendezvous problem [65, 66, 67]. Izzo's formulation [65], an adaptation of Gooding's [66], is the one that corresponds to the algorithm that is already implemented and validated in the toolbox being used, PyKEP (see Section 4.1).

Izzo's implementation accepts hyperbolic, parabolic and elliptic orbits as solutions to Lambert's problem and also allows multi-revolution lambert problems to be solved (obtaining solutions with more than one complete revolution around the central body). Another advantage is that it is shown to be computationally less demanding than previous procedures [65], an important feature since Lambert solver routines are frequently called for to calculate orbits.

The Lambert solver implemented in PyKEP receives $\mathbf{r_1}$, $\mathbf{r_2}$, time of flight and maximum number of revolutions as input and then outputs the velocity vector of the calculated transfer orbit at the initial and final points, for each multi-revolution solution found.

3.2 Gravity Assists and Deep Space Manoeuvres

Gravity assist manoeuvres (GAs), commonly known as *flybys*, and Deep Space Manoeuvres (DSMs) are two of the most commonly used techniques in spacecraft trajectory design [1, 63]. Both concepts have already been briefly explained in section 1.1 and are well documented in astrodynamics literature [1, 63, 64, 68]. A combination of these two manoeuvres can be seen in figure 3.1, which shows a typical procedure in interplanetary missions: the ΔV - Earth Gravity Assist (ΔV-EGA).

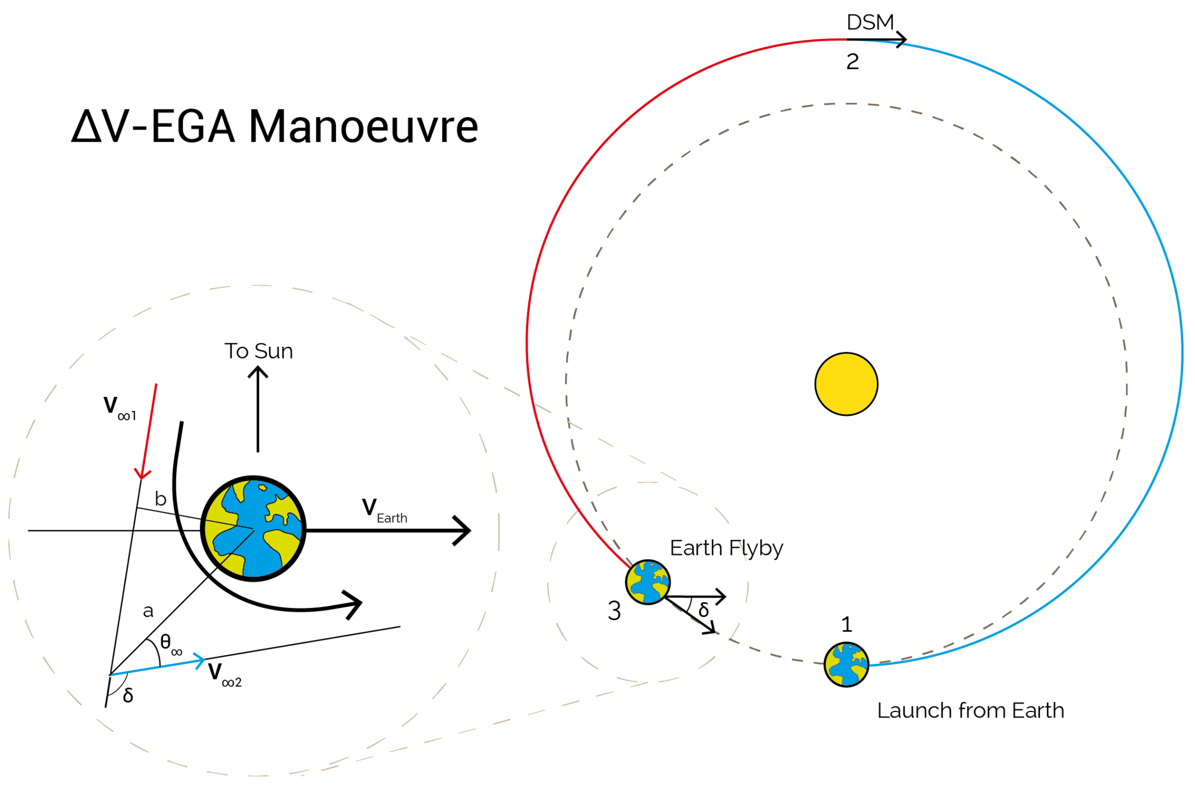


Figure 3.1: ΔV-EGA: an example of Flyby and DSM manoeuvres. To the left is a close-up of the flyby plane, in the planet's reference frame. To the right is the interplanetary trajectory with three key points.

In the ΔV-EGA example, the spacecraft is launched from Earth and the objective is to have it visit Earth again for a gravity-assist manoeuvre that will set the probe on a course to another planet. In this simple situation, a DSM is employed somewhere along the orbit with the purpose of correcting the trajectory so that the spacecraft will encounter Earth for a flyby at point 3. The DSM's usefulness here is that of a targeting manoeuvre and its magnitude and direction are obtained through the solution of a Lambert problem. The position of the DSM along the arc is an optimisable parameter.

At point 3, the spacecraft reaches Earth and performs a flyby. A gravity-assist operates through a transfer of momentum between the planet and the spacecraft, meaning that the trajectory of the spacecraft can be changed without using fuel, making it one of the most useful procedures in interplanetary mission design. This manoeuvre can be used for [64]:

- Orbit Pumping: Changing the orbit's energy relatively to the central body by changing the orbit's characteristics within the orbital plane (semi-major axis and eccentricity).

- Orbit Cranking: Changing the plane of the orbit ("free" inclination changes and other plane rotations). Only present if a 3D geometry is considered.

In the planet's reference frame, the gravity-assist orbit is hyperbolic. The characteristics of this orbit can be set by the user through the flyby periapsis radius, r_{fb}, which is an user-given parameter [63],

$$e = 1 - \frac{r_{fb}}{a} = 1 + \frac{r_{fb} V_\infty^2}{\mu_{Planet}}, \tag{3.1}$$

where $|\mathbf{V}_{\infty \mathbf{1}}| = |\mathbf{V}_{\infty \mathbf{2}}| = V_\infty$ is the hyperbolic excess velocity of the spacecraft with respect to the gravity-assist body at the time of encounter. This quantity remains constant during an unpowered flyby.

Note that, within safety considerations, any periapsis radius can be targeted through very small changes in the trajectory when the spacecraft is far from the gravity-assist body. These small changes are referred to as Trajectory Correction Manoeuvres (TCMs). Furthermore, using the TCMs, it is possible to target a flyby to occur behind or in front of the planet, respectively increasing or reducing the energy of the orbit with respect to the central body [1].

The closest point of approach between the undeflected incoming hyperbola and the planet is defined as the impact parameter, b [63],

$$b = r_{fb} \sqrt{1 + \frac{2\mu_{planet}}{r_{fb} V_\infty^2}}. \tag{3.2}$$

The rotation of the velocity vector during a flyby is given by the turn angle,

$$\delta = 2 \sin^{-1} \left(\frac{\mu_{Planet}}{\mu_{Planet} + r_{fb} V_\infty^2} \right). \tag{3.3}$$

Once the outgoing velocity is found, the resulting orbit around the primary body is fully defined as the orbital elements can be determined through the position and velocity vectors [64] or, alternatively, through the flight path angles [63].

3.3 B-Plane Geometry

The B-plane can be thought of as a target attached to a celestial body. Targeting different positions in this plane, through the small TCMs mentioned before, will yield different outgoing trajectories as the flyby's orbital plane is changed. The study of B-plane physics is of particular interest for orbit capture operations and 3D gravity assist manoeuvres.

The B-plane is defined, in a planet-centred reference frame, by its $\hat{R}$, $\hat{T}$ and $\hat{S}$ axes, as pictured in figure 3.2 [69].

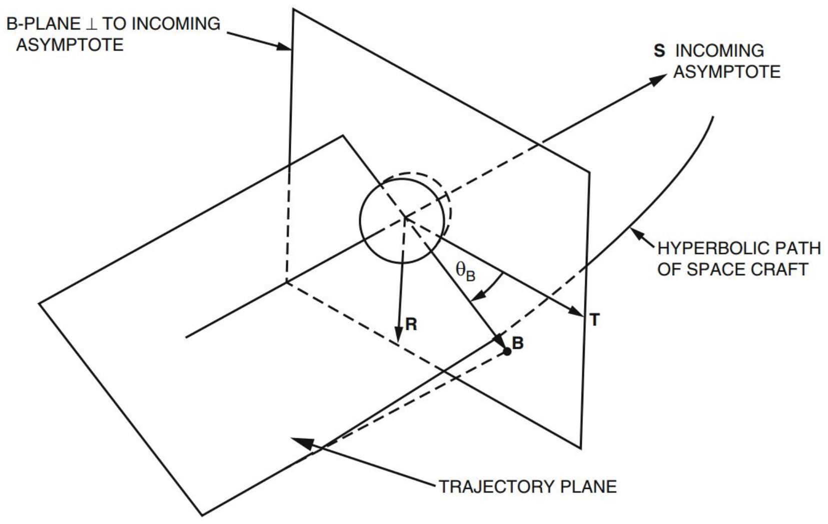

B-PLANE ⊥ TO INCOMING ASYMPTOTE
S INCOMING ASYMPTOTE
HYPERBOLIC PATH OF SPACE CRAFT
θ_B
R
T
B
TRAJECTORY PLANE

Figure 3.2: B-plane definition. Image taken from Hintz [69].

- $\hat{S}$: a unit vector that is centred on the planet and is parallel to the incoming hyperbolic excess velocity.
- $\hat{T}$: a unit vector that is parallel to a reference plane and perpendicular to $\hat{S}$. For the purposes of this work, the reference plane considered was the equatorial plane of the target planet, defined in the J2000 reference frame [70].
- $\hat{R}$: a unit vector that is the result of the $\hat{S} \times \hat{T}$ cross-product, completing the orthogonal reference frame.

Given the B-plane angle, θ_B, the B-plane vector, $\mathbf{B}$, can be decomposed into two components,

$$\mathbf{B} = B_T\hat{T} + B_R\hat{R} = b\cos\theta_B\hat{T} + b\sin\theta_B\hat{R}, \tag{3.4}$$

where b is the impact parameter defined in equation 3.2. Note that the r_{fb} in equation 3.2 is user-defined. The B-plane angle can be determined from the orbital inclination of the hyperbola, i, and the declination of the incoming asymptote, δ_∞, through [63]

$$\theta_B = \cos^{-1}\left(\frac{\cos i}{\cos\delta_\infty}\right). \tag{3.5}$$

While a 360-degree range of θ_B angles is allowed, the achievable inclination is limited by the arrival declination [63],

$$|\sin i| \geq |\sin \delta_\infty|. \tag{3.6}$$

The arrival asymptote's declination, δ_∞ and right ascension, RA_∞ are defined in the planet-centred rotating reference frame, designated by the International Astronomical Union (IAU) [70].

To compute δ_∞ and RA_∞, the arrival asymptote needs to be determined first. This asymptote is defined by the hyperbolic excess velocity vector, which is a result obtained from the interplanetary phase of the mission. After the arrival asymptote vector is transformed from the J2000 reference frame to the IAU reference frame, the declination and right ascension can be calculated as

$$\delta_\infty = \sin^{-1}\left(\frac{z_{\infty,IAU}}{||\mathbf{V}_\infty||}\right) \tag{3.7a}$$

and

$$RA_\infty = \cos^{-1}\left(\frac{x_{\infty,IAU}}{||\mathbf{V}_\infty||\cos\delta_\infty}\right), \tag{3.7b}$$

where $x_{\infty,IAU}$, $y_{\infty,IAU}$ and $z_{\infty,IAU}$ are the IAU components of the arrival asymptote. Note: quadrant fixing for right ascension is needed. If $y_{\infty,IAU} \leq 0$ then $RA'_\infty = 2\pi - RA_\infty$.

Having determined the B-Plane components required for the desired hyperbolic trajectory, it is possible to differentially correct the interplanetary trajectory, making use of the trajectory correction manoeuvres, until the spacecraft's path passes through the desired point in the B-plane within a certain degree of confidence. While the required B-Plane components will be calculated as a result of this work, the trajectory correction manoeuvres are out of the scope of the preliminary design.

3.4 Launcher Performance

Launcher performance needs to be taken into account in the design of any space mission. The spacecraft mass that a launcher can deploy to an interplanetary departure orbit is dependent on the launch energy ($C_3 = V_\infty^2$), on the departure orbit declination and on the capabilities of the launcher itself [63]. In this work, launcher performance is not directly considered when optimising the trajectory. Instead, the launch V_∞ is minimised by the optimiser because a lower V_∞ at launch leads to an increase in the launcher's deployable mass. Once the trajectory design is complete and the required propellant mass is known, it will be possible to select a launcher vehicle.

The performance of specific launchers is usually given to clients in the form of curves, such as those displayed in figure 3.3 for the Falcon 9 (Block 2) and the Atlas 551.

More performance curves are shown in Biesbroek [68] and in the specific launcher manuals but the Falcon and Atlas were selected because they represent the lowest and highest payload capability of the available curves, respectively. The maximum declination for the use of the Falcon 9 is $\pm 28.5^\circ$ and the maximum declination for the use of the Atlas 551 is $\pm 28.6^\circ$.This limits the shape of the outgoing asymptote, as only certain inclinations will be available. Note that the performance of the launcher

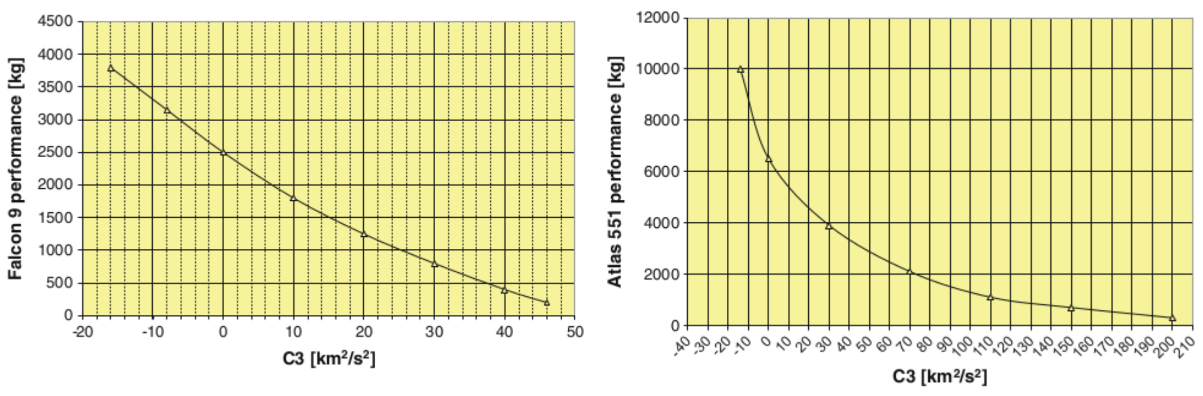


Figure 3.3: Left: Falcon 9 (Block 2) performance curve. Right: Atlas 551 performance curve. Both images are from Biesbroek [68].

is maximum when the departure inclination is equal to the departure declination [68]. The departure geometry is obtained by doing the reverse procedure of the arrival geometry, explained in section 3.3.

As an example, for an outgoing V_∞ of 4 km/s, or a C_3 of 16 km^2/s^2, the maximum payload the Falcon 9 can carry to an escape orbit is around 1400 kg, when the Atlas 551 can carry around 5000 kg for the same launch energy. Using the Atlas 551 will come at a higher price, however. Obviously, payload capability should not be the only metric used when choosing a launcher for a space mission. There is cost, risk, politics and so on to consider.

3.5 Ephemerides

The ephemerides define the state of celestial bodies (position and velocity) at a certain instant in time and are essential for the navigation of spacecraft through the Solar System. Modern day ephemerides are obtained through the compilation of astronomical observations of interest bodies, which are then fitted to a high-fidelity simulation of the solar system [71]. The result is an estimate of the position of the planets and moons that is valid only within a certain interval of time. Naturally, the farther the epoch of observations is from the epoch of the mission, the greater the error in using the ephemeris will be. An updated ephemerides system is critical if the goal is to design realistic space missions.

Besides accuracy, there is another factor to consider: as calls to ephemerides systems are frequent in modern trajectory design tools, the designer must also consider computation speed as an important metric to pay attention to.

3.5.1 Available Types

There are several ways to obtain the ephemerides of a celestial body, with varying degrees of precision and computation speed.

JPL Development Ephemerides (DE)

The need for high-precision ephemerides is satisfied by NASA Jet Propulsion Laboratory (JPL)'s Development Ephemerides (DE, for short) [71]. The data is provided in the form of an "SPK Kernel" built by JPL or on-line, via the HORIZONS system [72]. The positions of the planets are obtained through Chebyshev interpolation over a numerical integration of a dynamic model of the solar system [71]. This model is very precise as it takes into account multi-body effects, solar radiation pressure, oblateness effects and even the inner structure of some objects, amongst others [71]. The development ephemerides have become one of the standard general-purpose ephemerides systems in mission design.

The DE kernel files used for this work are:

- **DE432**: A minor upgrade to the general-purpose DE430 [71] to aid New-Horizons in its mission to Pluto. Contains planetary and lunar ephemerides in the range 21-Dec-1549 to 25-Jan-2650. Referred to the ICRF (International Celestial Reference Frame) 2.0;

- **SAT375**: Contains the ephemerides for the primary satellites of Saturn (integer IDs from 601 to 614), relative to the Saturn System. Ranges from 07-Jan-1800 to 16-Jan-2200.

Each celestial body is uniquely identified by its integer ID code. These can be viewed at `http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/FORTRAN/req/naif_ids.html`.

The DE's are implemented off-line through the SPICE [73] toolkit, provided by NASA's Navigation and Ancillary Information Facility (NAIF), to assist mission designers and scientists with orienting spacecraft and instruments in the solar system. The toolkit provides API's for relative positions of solar system bodies, spacecraft and instruments, as well as information about planet's sizes and shapes. It also provides useful reference frame conversions and is pre-loaded with information about the previously mentioned IAU planet-centred frames. Unfortunately, the toolkit is not thread-safe and therefore can not be used in multi-core environments, causing errors and inconsistent results.

This project is being coded in *Python* and SPICE does not offer a toolbox for this programming language, creating a need for the use of a *wrapper* to be able to access the toolkit's functionality. While PyKEP offers some SPICE functions, it is largely untested and undocumented, therefore the choice fell on a new SPICE wrapper that has seen frequent expansion updates and is well documented: SpiceyPy [74] by Andrew Annex.

JPL's Approximate Ephemerides (Low-precision)

Low precision ephemerides are useful to fulfill applications where speed is more important than accuracy, like fast trajectory searches and instrument pointing [75]. With this in mind, JPL created an approximation of the position of the major planets [75]. These ephemerides are the result of a best-fit adjustment to the accurate models (DE), over a certain period of time. The outcome is a set of orbital elements and their associated rates, allowing for the analytical determination of the state vector of the planet at any instant, within the period of validity of the adjustment.

This system has the fastest computation time but also displays a low-precision and is not suitable for the design of realistic space missions. All the steps and data necessary to implement these ephemerides, as well as information on the precision of this method, are well explained in JPL's own technical report [75].

3.5.2 Interpolated Multi-thread SPICE Ephemerides System

The lack of multi-threading support from the SPICE [73] library created the need to obtain a working ephemerides system that functions with the multi-core asynchronous island model of the PyGMO optimisation library, which greatly speeds up the global optimisation procedure and allows different instances to communicate information between themselves. It is also important to be able to run several simulations in parallel, with different parameters.

While approximate trajectories could be found for the Solar System through low-precision ephemerides or keplerian equations, with some loss in precision, the same does not apply to the Saturn System. In this environment, the moons influence each other in a complicated multi-body environment, in which SPICE-level precision is preferred to keplerian motion. Tests were performed to compare both approaches and the less precise ephemerides displayed errors that would make the trajectory design unrealistic (e.g. about 18^o of angular difference for Enceladus, after only one year).

To be able to use SPICE's data in a multi-thread environment, two options were considered:

1. Load the entire DE kernel into local variables and access it safely, passing it around the program.

2. Interpolate over the SPICE data with cubic spline interpolation and store only the interpolation coefficients in memory.

Option one has the obvious constraint of being memory heavy and would be harder to implement, but would allow very rapid access to ephemeris data. However, this option requires knowledge of the structure of SPICE kernel files, leading to a more complicated implementation.

Option two is accompanied by a reduction in the precision of the ephemerides, as it is an interpolation of an interpolation, but allows for a lighter program and fast access, as proved by the implementation of cubic spline interpolation in FIRE [76] — a "Fast, Accurate and Smooth" planetary ephemerides retrieval system. It is also easier to implement as it does not require intimate knowledge of the internal file structure of SPICE's kernels. Due to this ease of use, option two is chosen for the purposes work.

To accomplish this in an elegant way, without the need to interpolate every time the program is executed, it was decided that the interpolation data would be stored in a binary file format for later access. These formats are lighter and easier for the machine to process than plain-text formats.

A side program to build these files, codenamed *Ephemerides Generator*, was built. The pseudo-code can be seen in algorithm 1.

The cubic spline interpolation is performed by Scipy's *interpolate* package. Scipy [77] is a scientific library for *Python*. Each body's interpolation coefficients are stored in a file through the user-defined *keywords* and can be accessed through them.

For the purpose of this work, two ephemerides files were generated:

- **SolarSystem_2020_to_2050_J2000.eph**: Solar system ephemerides containing Venus, Earth, Mars, Jupiter and Saturn. The interpolation was performed at steps of 1 day. The time ranges from **01-Jan-2020** to **01-Jan-2050**. The reference frame is the inertial Earth mean equator and equinox of

Algorithm 1 Ephemerides Generator

```
1: kernel ← furnsh($(DE432, SAT375)$)                                   ▷ Load the Kernel through SPICE
2: $planets, time\_range, keywords$ ← Input
3: Initialize: Coefficients List
4: for planet in planets do
5:     Initialize: position list, velocity list
6:     for epoch in time_range do
7:         position,velocity ← spkezr($(epoch)$)                  ▷ Get planet ephemerides from SPICE
8:     knots, coefficients[0:2] ← interpolate($(time\_range, position)$)         ▷ Interpolate position
9:     knots, coefficients[3:5] ← interpolate($(time\_range, velocity)$)         ▷ Interpolate velocity
10:    Coefficients List ← coefficients
11: file ← $(knots, CoefficientsList, keywords)$                          ▷ Store in user-defined file
```

J2000. This file occupies 3MB in disk compared to JPL's 130MB DE432 file, as it has less bodies and a smaller time period.

- **SaturnSystem_2020_to_2050_J2000.eph**: Saturn's satellites of interest (Enceladus, Tethys, Dione, Rhea, Titan). The interpolation was performed at a constant time-step that yields a minimum resolution of 50 points per orbit at Enceladus (higher for the other moons). The time ranges from **01-Jan-2020** to **01-Jan-2050**. The reference frame is the inertial, Saturn-centred, J2000. This file occupies 53MB in disk and can be improved by using variable time-step for each object, at the cost of harder implementation.

The interpolation coefficients can now be loaded from the file, stored in memory and the cubic spline reconstructed with Scipy so that the state vector of the body can be acquired for any epoch, for multiple instances of code running simultaneously.

Finally, the quality of the interpolation in terms of accuracy and speed is assessed. A simple program was devised to obtain the value of the absolute error (position, velocity and central angle) for each object in the generated ephemerides files. The error is defined as shown in equation 3.8, where x is the quantity of interest.

$$\epsilon_{abs} = |x_{spice} - x_{interpolated}| \tag{3.8}$$

The state of the celestial bodies obtained with the multi-thread ephemerides is compared with the results from the SPICE toolbox over the entire period of the new ephemerides.

Two tests were conducted: one for the solar system and another for the Saturn system, with both showing that the interpolation is a good fit. Peak error magnitudes are shown in table 3.1. Only the worlds closest to their central body are shown because the faster the orbit, the lower the interpolation resolution and the higher the absolute error.

Table 3.1: Peak absolute errors of the cubic spline interpolation in the 2020 to 2050 period.

Object	Peak Position Error (m)	Peak Velocity Error (m/s)	Peak Angular Error (rad)
Venus	140	10^{-5}	10^{-7}
Enceladus	10^{-3}	10^{-7}	10^{-7}

While this approach does show good results in terms of accuracy, in a single-thread benchmark on an Intel core i7 processor @ 3.5 GHz, the cubic spline interpolation is slower than its alternatives, probably due to being scripted in a non-compiled language, which hinders its runtime performance. This test made 10000 calls to obtain the state vector of planet Earth, for each system, at random epochs between 2020 and 2050. Additionally, the cubic spline method was also tested for one of Saturn's moons and it showed that the run time was adversely affected by file size. Results are shown in table 3.2. It is important to keep in mind that, while the computational speed of the new system is lower, the new approach allows several instances of the same program to run at the same time. This opens up the possibility to use optimisation topologies and other useful functionalities of PyGMO that improve the quality of the optimisation (See Chapter 4).

Table 3.2: Benchmark of Ephemerides System Runtimes.

System	Mean run time [μs]
Low-Precision JPL	5
Keplerian	5
SPICE	26
Cubic Spline (Earth)	84
Cubic Spline (Titan)	227

3.6 Tisserand Plots

The Tisserand graph is based on a criterion derived by François Félix Tisserand, a 19^{th} century French astronomer. Tisserand's observation of comets led him to the interesting question of knowing whether he was observing a comet he had already seen before or a new one. Through observations of these bodies as they flew by Jupiter, he was able to develop a method, based on an invariant constant, to determine if a comet he was observing in a different part of the sky was indeed the same one he had spotted, before a suspected perturbation suffered by the gravity of a planet [63]. Tisserand had laid the foundation for an astrodynamics tool that is still used today in preliminary mission design — the Tisserand graph. Unknowingly, he had also observed something which would only be theorized in detail almost a century later: the gravity-assist manoeuvre.

Note: the analysis in this section will use non-dimensionalised equations, unless stated otherwise. The Tisserand Graph assumes that the orbits of the bodies are circular, a reasonable assumption for both systems being studied in this work (recall Chapter 2). Because of this, all distances are divided by R_{circ}, the distance between the planet and the central body. Velocities are divided by V_{circ}, the orbital velocity for that circular orbit. Time is non-dimensionalised by T_{circ}, the associated period.

Tisserand graphs come in many shapes, depending on their purpose, but the one that is most useful to us is the apoapsis vs periapsis (R_a vs R_p) formulation (see figure 3.4). It allows for a clearer vision of the spacecraft orbit, despite not being suitable for representing hyperbolic trajectories but these won't be studied in this work.

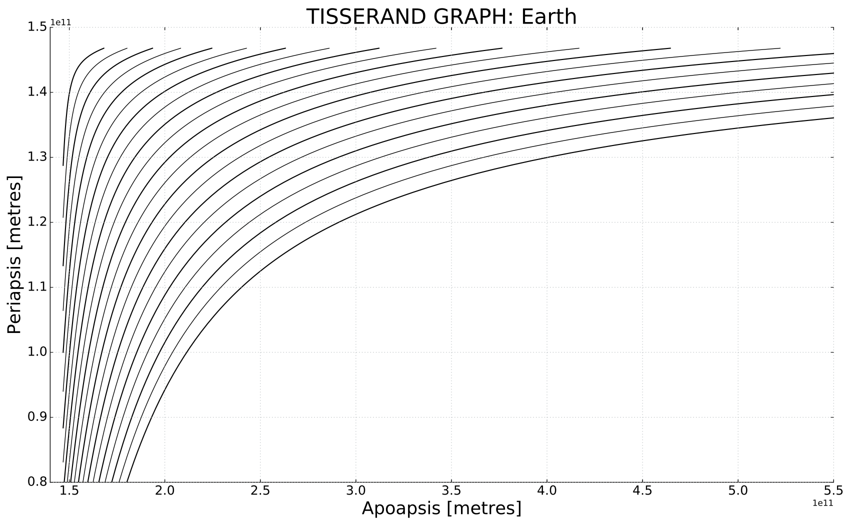


Figure 3.4: Example of a Tisserand Graph for Earth. V_∞ curves range from 1 km/s (top) to 9 km/s (bottom), with a step of 0.5 km/s.

Each **point** in the Tisserand plot is an orbit around the central body (e.g. Saturn) and every **contour** is a curve where the points have the same Tisserand Constant, which is intimately related to the hyperbolic excess velocity (refer to Strange et al. [53] for the procedure to obtain this expression),

$$C_{Tiss} = 3 - v_\infty^2, \tag{3.9}$$

where v_∞ is the **non-dimensional** hyperbolic excess velocity of the spacecraft's orbit with respect to the gravity-assist body (e.g. a moon). Moving along a contour is then equivalent to performing a non-impulsive flyby around a planet or moon: the spacecraft's orbit around the central body changes but the magnitude of $\mathbf{V}_\infty$, with respect to the gravity-assist body, remains approximately constant.

Tisserand graphs are powerful tools to help us understand how flybys let us access different orbits around the central body. Additionally, if the graphs are plotted for several gravity-assist bodies, they also become helpful in solving the sequencing problem: if I want to go from Earth to Saturn, what are possible sequences of flybys that could lead me there faster and with the least cost possible? With these purposes in mind, the plots have been used as part of the winning solution of the GTOC 6 [41], with the objective of designing a Jupiter moon tour. Last but not least, Tisserand graphs will also be used when attempting to visualise the V_∞-Leveraging manoeuvres, explained in detail in Section 3.7.

3.6.1 Constructing the Tisserand Graphs

The construction of a Tisserand plot is fairly simple. The idea is to find all possible orbits that can be obtained after a gravity-assist in a certain body, for a certain range of v_∞ levels. The procedure explained here follows the logic presented by Strange et al. [53].

First, let's introduce the concept of pump angle, α, through vector analysis of a planar gravity-assist (figure 3.5).

A very important remark is that the variation in α during a flyby, $|\alpha_1 - \alpha_2|$, equals the turn-angle, δ.

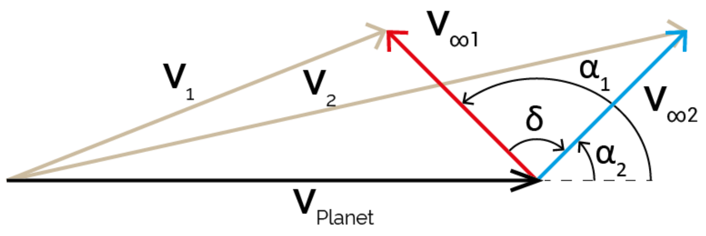


Figure 3.5: Gravity-Assist V_∞ Triangle. V_∞ is the dimensional hyperbolic excess velocity.

This can be used to calculate the flyby height from the definition of turn angle (see equation 3.3).

Applying the law of cosines to the outgoing velocity triangle (referent to v_2) we obtain

$$v_2^2 = 1 + v_{\infty_2}^2 + 2v_{\infty_2}\cos(\alpha_2). \tag{3.10}$$

Note that, with the non-dimensionalisation, $v_{planet} = 1$. Also, from the characteristics of an unpowered flyby, $v_{\infty_1} = v_{\infty_2} = v_\infty$. Equation 3.11 relates the spacecraft's velocity with its semi-major axis through the vis-viva equation, in its non-dimensional form,

$$v_2^2 = 2 - \frac{1}{a_{sc}}, \tag{3.11}$$

where a_{sc} is the post-flyby orbit non-dimensional semi-major axis. Substituting the spacecraft's post-flyby velocity in equation 3.10 by the vis-viva equation it's possible to find a_{sc},

$$a_{sc} = \frac{1}{1 - v_\infty^2 - 2v_\infty\cos(\alpha)}. \tag{3.12}$$

Immediately, equation 3.12 indicates that, for each fixed v_∞, there is a different post-flyby orbit for every α, with $0 \leq \alpha \leq \pi$.

The eccentricity of the outgoing orbit can be obtained from the Tisserand Constant [53] and is given by

$$e_{sc}^2 = 1 - \frac{1}{a_{sc}}\left(\frac{3 - \frac{1}{a_{sc}} - v_\infty^2}{2\cos(i_{sc})}\right)^2, \tag{3.13}$$

where i_{sc} is the spacecraft orbit's inclination with respect to the orbital plane of the gravity-assist body. Having found the semi-major axis and the eccentricity, the periapsis and apoapsis of the post-flyby orbit can be calculated through traditional orbital mechanics [1] as

$$r_a = a_{sc}(1 + e_{sc}) \qquad \text{and} \qquad r_p = a_{sc}(1 - e_{sc}). \tag{3.14}$$

To construct the plot for a given gravity-assist body, a program steps through a user-defined $[v_{\infty_{min}}, v_{\infty_{max}}]$ range and, for each v_∞ "level", finds the dimensional values of r_a and r_p for $0 \leq \alpha \leq \pi$. These points are stored and then plotted on a r_a vs r_p graphic.

3.6.2 Reading the Tisserand Graphs

Now that we have the ability to construct Tisserand plots for any body in a near-circular orbit, it is time to learn how to interpret them. Figure 3.6 displays several graphical elements showing what kind of information a basic Tisserand graph is capable of transmitting.

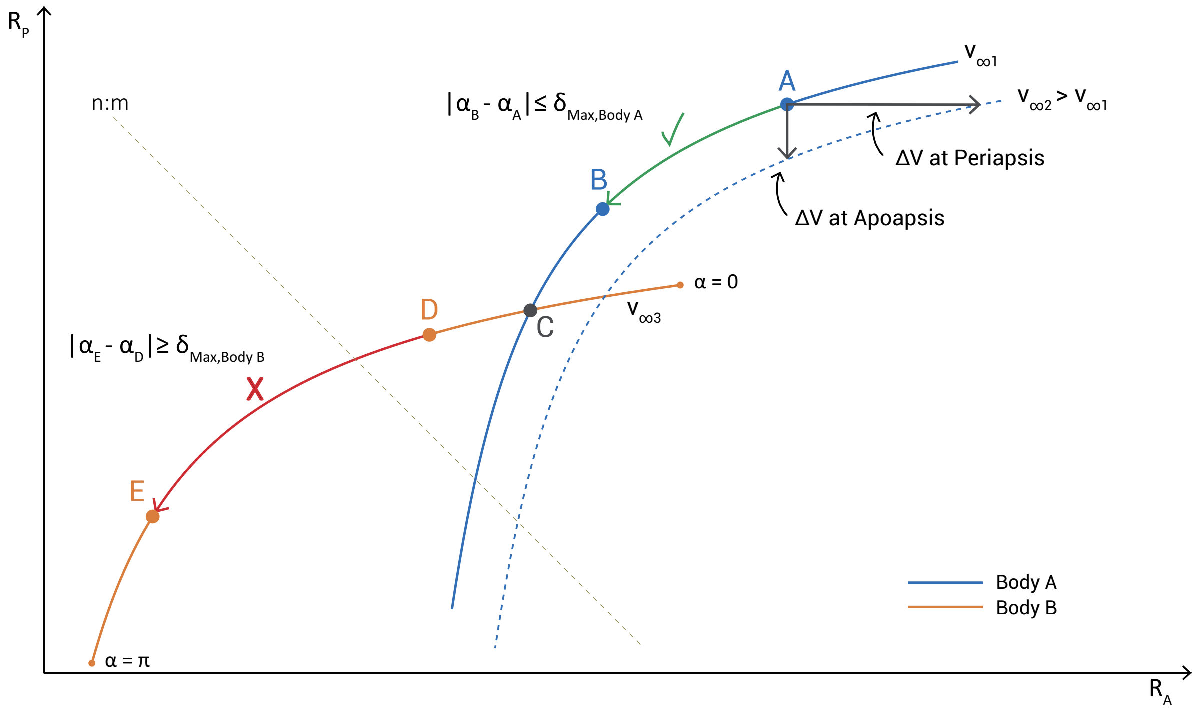


Figure 3.6: An example Tisserand plot showing several graphical elements. See text for the description of each element.

Remember that each point in a Tisserand plot is an orbit around the central body (Saturn or the Sun), defined by its periapsis and apoapsis. In this space, it is possible to find orbits that have the same v_∞ with respect to a given moon or planet. These are called the v_∞ curves, or levels. Orbits with increasing v_∞ yield curves that are translated downwards and rightwards in the r_a vs r_p Tisserand plot.

One of the most frequent uses of the Tisserand graph is to know what post-flyby orbits can be achieved, given a pre-flyby orbit. Performing a gravity assist manoeuvre is the Tisserand equivalent of moving along a fixed v_∞ curve, as the magnitude of the hyperbolic excess velocity, with respect to the gravity assist body, remains constant during a flyby (remember section 3.2).

If the spacecraft reaches body A at a certain α_A, with a certain $v_{\infty 1}$, it will be at point A in figure 3.6. If a flyby is performed, it is possible to move from point A to any point in the $v_{\infty 1}$ curve with only one restriction: the minimum flyby radius (also referred to as safe radius). This radius is user-defined and varies according to the navigational constraints of performing a gravity-assist on a specific celestial body. For example, Earth has an atmosphere but the environment is well known so a usual minimum flyby height (for preliminary design) would be around 200 km [22, 78]. The safe flyby radii used in this work are defined in sections 2.2 and 2.3.

The minimum flyby height restricts the maximum turn angle, δ_{max}, that the velocity vector can suffer when performing a gravity assist, as determined by equation 3.3. Since moving along a v_∞ curve is equivalent to a change in the pump angle, α, then the maximum movement within a curve that is achievable with a single flyby is given by δ_{max} because $\Delta\alpha = \delta$. In figure 3.6, which is merely an illustrative example, a flyby from A to B would be allowed because it satisfies the minimum flyby radius constraint but a flyby from D to E does not (due to the difference between pump angles being greater than the maximum turn angle).

Another useful element shown is that the intersection between the v_∞ curves of two bodies, as indicated by point C, is a potential transfer orbit from body A to body B (or vice-versa), because the post-flyby orbit, coming from body A, crosses the orbit of body B. Tisserand graphs are not attached to an epoch and, as such, don't consider the phasing between bodies. If there was a perfect alignment between bodies A and B, the orbit indicated by C would be a *ballistic transfer orbit*, meaning that a consecutive flyby of these two bodies would be free, in terms of ΔV. Intersections between curves allow us to find *potential* sequences of interplanetary trajectories that are cheap and depart, or arrive, at a target body with a certain V_∞.

Additionally, the dashed line indicated by n:m identifies post-flyby orbits at body B that show a n:m resonance, meaning the spacecraft will return to the gravity assist body after performing m revolutions for n revolutions performed by the planet or moon. A resonant orbit is characterised by having an orbital period that is an integer multiple of the body's period [53]. Orbits of this type are identified in the Tisserand R_a vs R_p graph by constant energy lines with a -45^o slope [53] (constant energy is equivalent to a constant semi-major axis).

Finally, the last element shown are two grey lines, one horizontal and another vertical, that show the effect that different ΔV's have on an orbit, according to the Tisserand graph. A ΔV tangent to the velocity vector, at apoapsis, will result in a change of the periapsis and vice-versa. This will be useful for the analysis of V_∞-Leveraging Manoeuvres, explained in section 3.7, for which we now possess all the tools to understand.

3.7 V_∞-Leveraging Technique

The V_∞-Leveraging Transfer (VILT) was introduced by Jon Sims and James Longuski, in 1997 [79]. This technique couples the use of a small deep-space manoeuvre to modify V_∞ with respect to a body (e.g. a moon) with the use of gravity-assist manoeuvres to target resonant, or near-resonant, orbits that allow for a cheap return to that body. This idea has laid somewhat dormant since 1997 but, with the upcoming missions to icy moons, it was revived as a way to design complex moon tours with the goal of lowering the final capture ΔV [53, 57, 80].

This analysis will maintain the approximations that have been made in section 3.6. The trajectories will be computed in a patched-conics, zero sphere-of-influence gravity assist model. The bodies' orbits are considered to be circular, a reasonable approximation for this work (see Section 2.3 to confirm that the moons of interest have a low orbital eccentricity). Furthermore, the VILTs will be assumed to be in the orbital plane of the gravity assist body (a 2D approach), as the inclusion of the crank angle to modify inclination during flybys would result in an overly complicated analysis. All these approximations are in line with those made in the available literature [53, 79, 57, 80]. Finally, this model allows the trajectories to have a multi-revolution behaviour (more than one revolution around the central body), which can be useful when targeting moons with low periods (such as in the Saturn System).

3.7.1 Anatomy of a VILT

The notation used in this work to represent VILTs is defined in the literature [53]:

[interior/exterior]-[OO/OI/II/IO] N:M(L)

- **Interior / Exterior**: If the DSM is performed at the apoapsis of the transfer orbit, the VILT is said to be *exterior*. On the other hand, an *interior* VILT has a DSM at the periapsis of the transfer orbit.
- **OO/OI/II/IO**: Related to the location of the flybys on either end of a VILT (whether they are inbound or outbound relatively to the periapsis of the transfer).
- **N**: Integer number of moon revolutions.
- **M**: Integer number of spacecraft revolutions.
- **L**: Revolution number where the DSM occurs.

Between interior and exterior leveraging and each type of encounter (II,OO,OI,IO) there are 8 types of possible VILTs for a given N:M (L) combination.

VILTs can be of the *tangent* or *non-tangent* kind. Tangent VILTs restrict the geometry of the manoeuvre so that the $\mathbf{V}_\infty$ vector is tangent to the velocity of the gravity assist body at one end of the VILT. While tangent VILTs have been shown to provide efficient V_∞ change for large moons, they have also been found to limit, or decrease, the efficiency of gravity-assist manoeuvres in small moons, such as Enceladus, Tethys or Dione [53, 57]. The method implemented in this work will include both tangent and non-tangent transfers.

The DSM is tangent to the orbital velocity and is placed at apoapsis or periapsis because it simplifies the model and was deemed to be an efficient location by Sims et al. [79].

Figure 3.7 displays a visual example of an interior and an exterior VILT. Notice the similarities with figure 3.1. In fact, the commonly used ΔV-EGA manoeuvre is an exterior V_∞-Leveraging Transfer with the purpose of increasing, or decreasing, V_∞ with respect to Earth in order to access new orbits going to other planets in the solar system.

3.7.2 Computing VILTs

The process, described here, to determine a VILT that brings the spacecraft's orbit from a specified $V_{\infty 1}$ to another specified $V_{\infty 2}$, with respect to a gravity-assist moon, is based on the analysis performed in Strange et al. [53].

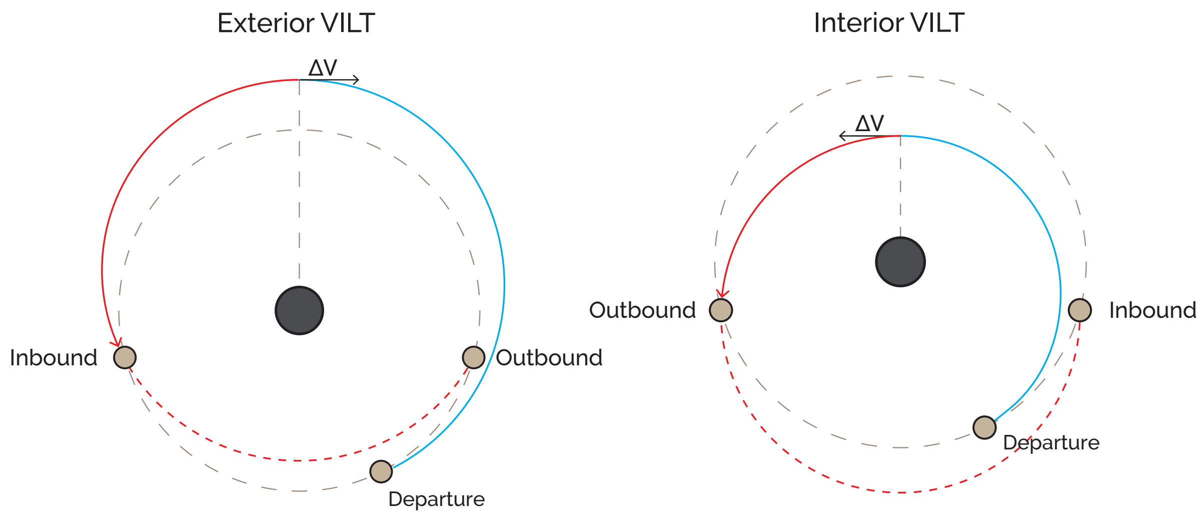


Figure 3.7: Types of VILTs. In this case, departure is outbound for exterior and inbound for interior VILT.

Quantities in this section will be non-dimensionalised. As in section 3.6, distances will be divided by R_{circ}, the distance between the planet/moon and the central body. Velocities are divided by V_{circ}, the orbital velocity for that circular orbit. Time is non-dimensionalised by T_{circ}, the associated period.

The point at which the DSM occurs is the *leveraging apse* and its distance from the central body (in our case, Saturn) can be written as

$$r_{la} = a_{sc}(1 + k_{ei}e_{sc}), \tag{3.15}$$

where k_{ei} is an integer parameter that distinguishes whether the VILT is exterior (+1) or interior(-1). Solving for the eccentricity and replacing into equation 3.13, we get a quadratic equation,

$$4a_{sc}r_{la}^2 - 8a_{sc}^2 r_{la} + a_{sc}^2(3 - v_\infty^2)^2 - 2a_{sc}(3 - v_\infty^2) + 1 = 0. \tag{3.16}$$

Solving equation 3.16 for r_{la},

$$r_{la} = a_{sc} + k_{ei}\sqrt{a_{sc}^2 - \frac{1}{4}(3 - v_\infty^2)^2 + \frac{1}{2}(3 - v_\infty^2) - \frac{1}{4a_{sc}}}. \tag{3.17}$$

From a given $v_{\infty 1}$ and α_1, which is used to obtain a_{sc_1} through equation 3.12, it is possible to calculate r_{la} for the first arc of the VILT by solving equation 3.17. Then, from equation 3.16 we can compute a_{sc2} since $r_{la2} = r_{la1}$ and $v_{\infty 2}$ is a user input. Attention is required to choose the root of the quadratic equation that is correct: $a_{sc2} > r_{la}$ (r_{la} at periapsis) for interior leveraging and vice-versa for exterior.

Once a_{sc2}, a_{sc1} and r_{la} have been determined, it is possible to calculate the eccentricities of the post and pre-DSM orbits through equation 3.15 and every other orbital characteristic from traditional orbital mechanics. The **non-dimensional** ΔV is given by the difference in velocities of both arcs at the leveraging apse,

$$\Delta V = \left| \sqrt{\frac{2}{r_{la}} - \frac{1}{a_{sc1}}} - \sqrt{\frac{2}{r_{la}} - \frac{1}{a_{sc2}}} \right|. \tag{3.18}$$

The next step is to constrain the VILT so that it returns to the same gravity-assist body, in order to continue the tour. This is done by matching the spacecraft's flight time with the gravity assist body's flight time. This is called the phasing constraint.

$$\text{tof}_{sc} = \text{tof}_{ga}. \tag{3.19}$$

Determining Spacecraft Flight Time

The determination of the spacecraft's flight time is done through Kepler's equation,

$$\tau_{sc} = \frac{T_{sc}}{2\pi}(E_{sc} - e_{sc}\sin(E_{sc})), \tag{3.20}$$

where E_{sc} is the eccentric anomaly of the encounter, given by [53],

$$E_{sc} = k_{io}\cos^{-1}\left(\frac{a_{sc}-1}{e_{sc}a_{sc}}\right). \tag{3.21}$$

The integer parameter k_{io} describes the direction of the encounter: +1 for an outbound encounter, -1 for an inbound encounter. Finally, tof_{sc} can be obtained through equation 3.22 [53],

$$\text{tof}_{sc} = \tau_{sc2} - \tau_{sc1} + T_{sc1}\left(L + \frac{1+k_{ei}}{4}\right) + T_{sc2}\left(M_a - L - \frac{1+k_{ei}}{4}\right). \tag{3.22}$$

Equation 3.22 unifies the flight times of the 8 different types of VILTs. M_a is an integer parameter that is equal to the number of spacecraft revolutions counted by apoapsis crossings and is necessary to correct the time of flight for the Outbound-Inbound (OI) VILTs,

$$\begin{cases} M_a = M & \text{if type is IO,II or OO;} \\ M_a = M + 1 & \text{if type is OI.} \end{cases} \tag{3.23}$$

Determining Gravity Assist Body Flight Time

The gravity assist body's time of flight is also obtained through equation 3.20 (replacing with the gravity assist body's information). The eccentric anomaly is given by

$$E_{ga} = 2\tan^{-1}\left(\tan\left(\frac{\theta_i}{2}\right)\sqrt{\frac{1-e_{ga}}{1+e_{ga}}}\right), \tag{3.24}$$

where θ_i is the true anomaly of each encounter and is given by the orbit equation in non-dimensional form,

$$\theta_i = k_{io}\cos^{-1}\left(\frac{1}{e_{sc}}(a_{sc}(1-e_{sc}^2)-1)\right). \tag{3.25}$$

If we apply the assumption that the gravity assist body has a near-circular orbit, then $e_{ga} = 0$, $E_{ga} = \theta_i$ and the time of flight of the gravity-assist body, in non-dimensional form, is

$$\text{tof}_{ga} = N_a + \frac{1}{2\pi}(\theta_2 - \theta_1), \tag{3.26}$$

where N_a, in similarity with M_a, is meant to correct for the OI case,

$$\begin{cases} N_a = N & \text{if type is IO,II or OO;} \\ N_a = N+1 & \text{if type is OI.} \end{cases} \tag{3.27}$$

Algorithm to determine phase-fixed VILTs

To find the phase-fixed VILT, an iterative method is required that will vary a parameter until the phasing constraint is met. Algorithm 2 displays the pseudo-code for the procedure implemented in this work.

Algorithm 2 VILT Calculator

```
1: planet, VILT Type, N, M, L← User Input
2: for $0 \leq$ Pump Angle $(\alpha_1) \leq \pi$ do
3:     Find: $a_{sc1}, r_{la}, e_{sc1}$                    ▷ Equations 3.12,3.17 and then 3.15
4:     Find: $a_{sc2}$                        ▷ Solve Equation 3.16 and select correct root
5:     Find: $e_{sc2}$                                              ▷ Equation 3.15
6:     Find: $E_{sc1}, E_{sc2}$                                     ▷ Equation 3.21
7:     Find: $tof_{sc}$                                             ▷ Equation 3.20
8:     Find: $\theta_1, \theta_2$                                   ▷ Equation 3.25
9:     Find: $tof_{ga}$                                             ▷ Equation 3.26
10:    if $|tof_{sc} - tof_{ga}| \leq \epsilon$ then
11:        Store: $a_{sc1}, a_{sc2}, e_{sc1}, e_{sc2}, r_{la}, tof_{sc}, \theta_1, \theta_2, \alpha_1$   ▷ dimensionalise and store VILT information
```

The iterative method implemented is provided by *Scipy* [77], a scientific library for *Python*, through its *optimize* toolbox. The tolerance used is $\epsilon = 10^{-5}$.

When sequencing VILTs, or resonant flybys, the bounds on the pump angle need to be adjusted to meet the minimum flyby radius constraint. These bounds become

$$\alpha_{previous} - \delta_{max} \leq \alpha_1 \leq \alpha_{previous} + \delta_{max}, \tag{3.28}$$

where $\alpha_{previous}$ is the pump angle before the first encounter of the VILT.

Finally, when required, it is possible to obtain the velocity vectors at each encounter by calculating them in the orbital plane of the gravity assist body first (using the true anomalies of the encounters) and then converting the vectors to the J2000 inertial reference frame.

The procedure outlined in this section was successfully validated by comparing individual phase-fixed VILT solutions with those obtained in Strange et al. [53].

Chapter 4

Trajectory Optimisation: Problem Formulation and Methodology

4.1 Software: PyGMO and PyKEP

PyGMO and PyKEP [81, 82, 83] are open-source programming libraries coded in C++, and exposed to Python, by the Advanced Concepts Team at the European Space Agency. Both libraries are meant to be used together to solve astrodynamics problems and are coded in a generalised form that allows the user to easily add new optimisation problems and algorithms, taking advantage of Python's ease of use and C++'s higher computational speed.

PyKEP [81] provides simple astrodynamics tools with a focus on computational efficiency [83]. At the core of this toolbox is a powerful multi-revolution Lambert solver (mentioned in Section 3.1) but also other useful tools such as a keplerian propagator, a 3D flyby propagator, taylor integrators for low-thrust trajectories, orbit plotting utilities, ephemerides implementations and more (see the online documentation [81] for a full description of PyKEP's features).

PyGMO [82], a scientific library that is part of ESA's Pagmo project, is a generalised optimisation toolbox with focus on parallelisation through the implementation of a novel paradigm, the asynchronous island model [83]. This new model allows for easy parallelisation and exchange of information between different algorithm instances.

In PyGMO, each solution candidate is codenamed an *individual*. A set, or population, of individuals is named an *island*, over which an optimisation algorithm is executed. It is also possible to cluster islands together into an *archipelago*, allowing each island to be optimised in parallel and information (such as the best individual) to be exchanged between them, following a user-defined topology. Each island in an archipelago may have a different optimisation algorithm associated with it. This is the so-called the *Asynchronous Island Model*.

The optimisation problem to be solved by each island is user-defined and easy to implement. In Python, the user defines a *Problem* class that implements the desired objective function and, optionally, a constraint evaluation function. These problems may be single or multi-objective, depending on the algorithm that is used to solve them.

PyGMO has several pre-implemented optimisation algorithms, shown in table 4.1. The toolbox also allows for the use of powerful third-party libraries (NLOPT, IPOPT, SNOPT) and the implementation of user-defined algorithms, through a procedure similar to the implementation of problems. More information on this can be found in PyGMO's online documentation [82].

Algorithms in PyGMO can be divided according to the following criteria [82]:

- **Decision Vector**: Continuous (C) / Integer (I) / Mixed (MI)
- **Constraints**: Constrained (C) / Unconstrained (U)
- **Number of Objectives**: Single-objective (S) / Multi-objective (M)

Table 4.1: Main algorithms available in PyGMO. [82]

Algorithm	**Type**
Heuristic Optimisation	
Differential Evolution (DE)	C-U-S
Self-Adaptive DE (jDE)	C-U-S
Particle Swarm optimisation (PSO)	C-U-S
Simple Genetic Algorithm	C-U-S and MI-U-S
Vector Evaluated Genetic Algorithm (VEGA)	MI-U-M
Non-Dominated Sorting GA (NSGA-II)	C-U-M
Corana's Simulated Annealing (SA)	C-U-S
Parallel Decomposition (PADE)	C-U-M
Strength Pareto Evolutionary Algorithm (SPEA2)	C-U-M
Artificial Bee Colony (ABC)	C-U-S
Improved Harmony Search (IHS)	MI-U-M
Covariance Matrix Adaptation	C-U-S
Monte Carlo Search	MI-C-S
Meta - Algorithms	
Monotonic Basin Hopping (MBH)	Adaptive
Multistart (MS)	Adaptive
Augmented Lagrangian (AL)	C-C-S
Local Optimisers	
Compass Search (CS)	C-U-S
Nelder-Mead simplex	C-U-S
BFGS 2	C-U-S
Sequential Least Squares Programming (SLSQP)	C-C-S
Truncated Newton Method	C-U-S
Method of Moving Asymptotes	C-C-S

Note: adaptive meta-algorithms take on the characteristics of the secondary algorithm that they use.

Both PyGMO and PyKEP have been extensively validated by the Advanced Concept Team's use of these toolboxes in the GTOC [83].

4.2 MGA-1DSM Formulation

The MGA-1DSM is a typical problem formulation for optimising Multi Gravity Assist trajectories, allowing one Deep Space Manoeuvre in each leg [37]. The MGA-1DSM formulation presented here was defined by Vinkó et al. [84] and is implemented in both PyGMO and PyKEP. It will be used to optimise the interplanetary phase of the mission to Enceladus.

4.2.1 General Form of the Problem

The most general form that a spacecraft trajectory optimisation problem can take is the following [37]:

$$\text{Optimise: } \phi(t_s, t_f, \mathbf{x_s}, \mathbf{x_f}) + \int_{t_s}^{t_f} \mathcal{L}(\mathbf{x}(\mathbf{t}), \mathbf{u}(\mathbf{t}), t)dt, \tag{4.1a}$$

$$\text{Subject to: } \dot{\mathbf{x}} = \mathbf{a}(\mathbf{x}, \mathbf{t}) + \mathbf{u}(\mathbf{t}), \qquad \text{dynamic constraints} \tag{4.1b}$$

$$\mathcal{G}(t_s, t_f, \mathbf{x_s}, \mathbf{x_f}) \leq 0, \qquad \text{boundary constraints} \tag{4.1c}$$

$$\mathbf{u} \in \mathcal{U}(\mathbf{x}, t), \qquad \text{propulsion constraints} \tag{4.1d}$$

where $\mathbf{x}(t) = (\mathbf{r}, \mathbf{v}, \mathbf{m})$ is the spacecraft state vector, $\mathbf{a}(\mathbf{x}, t)$ are the external forces acting on the spacecraft (such as gravity, solar pressure, drag, and so on), and $\mathbf{u}(t)$ represents the control variables such as thrust. The objective function may be only dependent on the initial and final conditions, indicated by ϕ, or may also depend on events during a specific path of the spacecraft, indicated by $\mathcal{L}$.

However general this formulation may be, in its current form it is very hard to handle, due to the complicated nature of the dynamic constraints. As a result, there aren't many suitable methods that are capable of solving the general problem in useful time [37]. This motivates the introduction of a series of simplifications in order to obtain a formulation that is easier to handle computationally. This is often done in spacecraft trajectory optimisation and is called *problem transcription* [37].

The following simplifications were made to obtain the simplified form of the problem [37]:

1. The only external force acting on the spacecraft is the point-mass gravitational attraction force from the central body, resulting in a two-body problem.

2. The sequence of gravity-assist bodies is imposed *a priori* by the user and is not left for the optimisation algorithm to find.

3. The intermediate gravity assist manoeuvres assume the bodies have no sphere of influence. This means that the change in the spacecraft state is instantaneous, therefore not continuous, and happens at one point in space. Furthermore, the GAs are constrained by the minimum flyby radius, which is a user-defined value.

4. The propulsion is considered to be high-trust (impulsive) and thrust can only happen during departure, arrival and once for every trajectory leg. There are no powered flybys.

All these assumptions are in line with those made when describing the physical models (Chapter 3).

4.2.2 Simplified Form of the Problem

The simplified and final form of the MGA-1DSM problem is:

$$\text{Optimise: } \phi(\mathbf{p}), \tag{4.2a}$$

$$\text{Subject to: } \mathcal{G}(\mathbf{p}) \leq 0, \qquad \text{boundary constraints} \tag{4.2b}$$

where ϕ is the objective function and $\mathbf{p}$ is the decision vector. No phase constraints are left as they have all been explicitly determined from the information in the decision vector. The boundary constraints, $\mathcal{G}$, are optional and can be defined by the user to account for specific mission requirements (e.g. maximum distance to the Sun). Without these, the result is a continuous, box-bound, optimisation problem.

Objective Function

The objective of the MGA-1DSM formulation is to maximise the ratio between payload mass and spacecraft total mass. As discussed in Chapter 2, because the propulsion is impulsive and no payload is dropped in intermediate phases of the mission, minimising fuel consumption is equivalent to minimising ΔV. Therefore, the objective function of this optimisation problem is the ΔV budget,

$$\phi(\mathbf{p}) = \Delta V_{departure} + \sum_{i=1}^{N-1} \Delta V_i + \Delta V_{arrival}, \tag{4.3}$$

where N is the number of planets in the sequence. There are N-1 legs. If the departure and arrival geometries are not yet known, the ΔV's can be replaced by the corresponding V_∞.

Decision Vector

The decision vector, also referred to as chromosome, is the vector that contains all of the problem's optimisation variables that define an individual [37]. This vector contains all the parameters that describe the departure geometry, the epoch at which the spacecraft passes by planets, and more. For the MGA-1DSM problem it is defined as [37]

$$\mathbf{p} = [t_s, V_{\infty 0}, u_0, v_0] + [\eta_1, T_1] + [r_{p2}, \beta_2, \eta_2, T_2] + \dots \tag{4.4}$$

The first term is referent to departure: t_s is the launch epoch, $V_{\infty 0}$ is the launch hyperbolic excess velocity and u_0, v_0 are non-dimensional variables that define the departure geometry (explained later). The remaining terms are referent to each trajectory leg. η_i is a non-dimensional parameter that establishes the location of the DSM along the i_{th} arc, T_i is the duration of the i_{th} trajectory leg, r_{pi} is the flyby radius at the i_{th} planet of the sequence and β_i is the plane change angle of the 3D gravity assist at the i_{th} body of the sequence. The problem is box-bound which means the user must insert reasonable lower and upper bounds for every parameter in the decision vector. The dimension of an optimisation problem is given by the length of its decision vector. In this case, the problem has a dimension of $6 + 4(N - 2)$.

Modelling Departure Geometry

In order to allow for a DSM in the first leg, it is necessary to have the initial velocity and position vectors of the spacecraft [84]. The initial position is simply given by the position of the planet at departure, due to the patched conics assumption. The velocity vector at launch can be defined as [84]

$$\mathbf{V}_{\infty\mathbf{0}} = V_{\infty 0}[\cos(\theta)\cos(\psi)\mathbf{i} + \sin(\theta)\cos(\psi)\mathbf{j} + \sin(\psi)\mathbf{k}], \tag{4.5}$$

where the reference frame unit vectors $\mathbf{i}$, $\mathbf{j}$, $\mathbf{k}$ are

$$\mathbf{i} = \frac{\mathbf{v}_{\mathbf{planet}}(t_s)}{|\mathbf{v}_{\mathbf{planet}}(t_s)|}, \tag{4.6a}$$

$$\mathbf{k} = \frac{\mathbf{r}_{\mathbf{planet}}(t_s) \times \mathbf{v}_{\mathbf{planet}}(t_s)}{||\mathbf{r}_{\mathbf{planet}}(t_s) \times \mathbf{v}_{\mathbf{planet}}(t_s)||}, \tag{4.6b}$$

$$\mathbf{j} = \mathbf{k} \times \mathbf{i}, \tag{4.6c}$$

Using spherical coordinates to define the geometric domain of $\mathbf{V}_{\infty\mathbf{0}}$ and defining two constants, $u, v \in [0, 1]$ then [84]

$$\theta = 2\pi u \qquad \text{and} \qquad \psi = \cos^{-1}(2v - 1) - \frac{\pi}{2}. \tag{4.7}$$

Modelling DSMs and GAs

As far as optimisation is concerned, GA manoeuvres and DSMs are modelled in the exact same way as explained in Section 3.2. For the GAs, the decision vector inputs the required information to propagate the manoeuvre: the flyby radius, r_p, and the plane-change angle (or B-Plane angle), β.

The DSMs are also modelled as described before. To account for the position of the DSM, the decision vector introduces a new parameter, $\eta \in [0, 1]$. For a DSM occurring between planet A and planet B,

$$t_{DSM} = t_A + \eta T_{AB}, \tag{4.8}$$

where T_{AB} is the transfer time between A and B and t_A is the epoch of passage at planet A.

4.3 Optimisation Algorithms

Different optimisation algorithms were used in the various phases of the mission to Enceladus. This section describes the logic behind the choice of each algorithm and provides insight into their functionality.

4.3.1 Interplanetary Phase

As mentioned in Section 4.2, the widely used MGA-1DSM problem formulation will be employed for the optimisation of the interplanetary phase of our mission. Many methods exist to globally optimise a trajectory, as seen in Chapter 1.

As discussed in Section 1.3.2, evolutionary algorithms are better at avoiding local minima, despite being slower and lacking an optimality criterion, meaning it is not possible to be sure that a certain solution is indeed the global minimum or not. On the other hand, calculus-based deterministic methods rely on the mathematical formulation of the MGA-1DSM problem, which is quite complex, and the quality of their convergence is dependent on initial guesses that may be hard to provide.

The MGA-1DSM problem is not easy to solve as it features areas where many local minima are clustered together (valleys) and where there may be a global optimum or not [2]. Because EAs have the upper hand in this type of problems, they will be used to perform global optimisation in the first phase.

Nevertheless, there is a possibility to use local optimisation to further refine the trajectory obtained through the global optimiser, ensuring a better result quality. Local optimisers are deterministic methods that use initial guesses to find an optimum solution within a given subset of the larger search space, which may or may not contain the global optimum.

From the literature review performed in Section 1.3.2, three optimisation methods are selected as *potential* candidates to perform the global optimisation: Differential Evolution (DE), Monotonic Basin Hopping (MBH) and Simulated Annealing with Adaptive Neighbourhood (SAAN). These three methods have already been shown to perform well in solving MGA-1DSM problems [37, 36, 40, 42, 46, 47]. In fact, MBH and DE have some of the best known results for several MGA-1DSM problems present in the Global Trajectory Optimisation Problems database (GTOP) [85]. This is a service provided by ESA that stores the best known solutions for spacecraft trajectory optimisation problems with the intent of allowing benchmark of new optimisation methods. DE, MBH and SAAN are already implemented in PyGMO (see table 4.1).

Differential Evolution (DE)

Differential Evolution (DE) was designed by Storn and Price [86] as a general optimisation method with ease of use, robustness and speed in mind. This optimisation method has few control parameters, is capable of handling continuous nonlinear optimisation problems and is easily implemented in a multi-core environment, such as PyGMO.

DE is an evolutionary algorithm and is, therefore, based in the principles of evolution. The procedure starts with the generation of a population of N_i individuals from random sampling across the search-space defined by the user. Then, for N_g generations, the algorithm performs the following operations on the population [86]:

- **Mutation**: Generation of a mutated chromosome by performing a certain operation on two randomly selected individuals and adding this change to a third individual. Mutation is controlled by the mutation parameter, $F \in [0, 1]$.

- **Crossover**: Mixes the mutated chromosome with a pre-determined individual to create a new trial chromosome. Each of the individual's chromosome entries is exchanged with the corresponding entry on the mutated chromosome based on a user-defined crossover probability, $CR \in [0, 1]$.
- **Selection**: If the generated trial chromosome is better than the pre-determined individual that originated it, then the last is replaced by the first in the next generation. The new chromosome's entries must be within the bounds set by the user for the optimisation problem.

The mutation operation has several possible variants. The variants in table 4.2 are already implemented in PyGMO [82]. The parameter *best/rand/rand-to-best* describes how the chromosome that will be the recipient of the mutation operation is chosen: randomly, population champion or a mix of the two. Secondly, *1/2* define the number of chromosomes that will take part in the mutation operation. Using 2 chromosomes improves the diversity of the population but requires more individuals to be effective. Lastly, *exp/bin* is related to the crossover scheme (exponential vs binomial) [86]. The mathematical description of each variant and of the Differential Evolution algorithm is very well covered in Brest et al. [87] and will not be presented here as it is out of the scope of this work.

Table 4.2: Mutation variants of the Differential Evolution algorithm. Each variant is assigned an ID that will be used for the remainder of the work, for the sake of shortness.

IDs	Variants
1,6	DE/best/1/exp and DE/best/1/bin
2,7	DE/rand/1/exp and DE/rand/1/bin
3,8	DE/rand-to-best/1/exp and DE/rand-to-best/1/bin
4,9	DE/best/2/exp and DE/best/2/bin
5,10	DE/rand/2/exp and DE/rand/2/bin

DE requires only five control parameters: F (mutation parameter), CR (crossover probability), N_g (number of generations), N_p (population size) and the variant to be used, making it a very easy algorithm to configure. Storn and Price [86] suggest reasonable parameters for DE. N_p should be between $5D$ and $10D$, where D is the problem's dimension, F should be higher than 0.5 as lower values slow convergence and are not more effective. CR has an effect on convergence speed and should be tested with different values.

Simulated Annealing with Adaptive Neighbourhood (SAAN)

Simulated Annealing with Adaptive Neighbourhood [88] is an optimisation method to solve continuous, non-linear optimisation problems with large numbers of local minima. The advantage of this method is that, unlike DE, it has a self-adaptive nature: its original parameters change to suit the variations that the optimiser feels in the search space. In other words, it is less likely to get stuck in local minima as it possesses strategies that allow the optimiser to avoid those situations.

Given an initial chromosome, the optimiser starts by performing a broad search in the search space to identify areas where the global minima could be. Then, by adjusting a temperature parameter, the optimiser starts iteratively producing finer searches until it finds a minimum. Progress is made by accepting or rejecting solutions based on the Metropolis Criterion [88] which states that the solution candidate is automatically accepted if its objective function value is better than the best known objective function value. If this is not verified, then the solution can still be accepted with an acceptance probability of

$$p = \exp \frac{f_i - f'}{T_k}, \tag{4.9}$$

where T_k is the variable temperature parameter, f_i is the current cost and f' is the best cost. As in DE, the new chromosome is only accepted if it is within the bounds of the problem, otherwise another one is generated until that happens.

The algorithm evaluates the progress it makes at each step. If too many new solutions are being rejected or accepted, the step is adjusted to maintain a healthy progression. If the algorithm gets stuck in a local minimum, uphill moves are allowed under control of the temperature parameter. If there is no progress in the objective function value or if the maximum number of iterations has been achieved, the algorithm returns the current best solution.

Higher temperatures influence the optimiser to only consider the gross fluctuations of the objective function as relevant. This aids the optimiser in searching zones where local minima may be clustered (*valleys*). On the contrary, lower temperatures are very sensitive to the fluctuation of the objective function and are better to help the optimiser distinguish between local minima.

The SAAN algorithm implemented in PyGMO has 6 important control parameters:

- **Start Temperature (T_s)**: Should be high enough to let the algorithm detect broad variations of the search space in the beginning of the optimisation.
- **Final Temperature (T_f)**: Should be low enough that the optimiser can distinguish between local minima but not too low as that will reduce performance.
- **Number of Step Adjustments**: Number of step adjustments that can occur within one iteration.
- **Bin Size**: Size of the pool of solutions used to evaluate algorithm progression.
- **Range**: Initial size of the neighbourhood that the algorithm can access to test new solutions.
- **Maximum Number of Iterations**: Should allow enough runs for the temperature to decrease to the final temperature before the optimisation ends. A good starting value would be Iter $=$ $D \times N \times$ Nr. of steps $\times$ Bin Size [82], where D is the dimension of the problem and N is the number of desired temperature adjustments to take place between T_s and T_f.

A detailed mathematical and computational description of the algorithm and all its intricacies is very well detailed in Corana et al. [88].

Monotonic Basin Hopping (MBH)

Monotonic Basin Hopping started as a global optimiser for molecular chemistry problems but was later generalised so that it could be applied to other types of problems [46].

MBH is based on a very simple concept [46]. The optimiser starts by picking an individual at random from the initial population. Then, a secondary optimisation algorithm acts on the individual to produce another solution. If this new solution is better than the current best, its stored. The best solution's chromosome is then perturbed within a user-defined neighbourhood parameter (Δ) and then the secondary algorithm acts again. This procedure is repeated until the best solution hasn't changed for a user-defined number of iterations, N_{stop}. After N_{stop} is reached, MBH either returns the best candidate or global resampling happens and the search space is sampled again until a set number of maximum re-samplings is achieved.

The selection of neighbourhood parameter is important to keep MBH from missing other local optima in the same basin of attraction (low perturbation) or to prevent MBH from jumping all over the search space without criteria (high perturbation).

MBH is implemented in PyGMO as a generalised algorithm, meaning it will accept any kind of secondary optimiser and will take on its characteristics (whether it is constrained, unconstrained, . . .). When coupled with another global optimiser such as DE, MBH acts as a smart restart method that doesn't allow the secondary algorithm to get stuck in local minima, improving its consistency. MBH can also run with a local optimiser, such as Compass Search (CS) or Sequential Least Squares Programming (SLSQP). In this case, it will perform deterministic optimisation while helping the local optimiser to come out of valleys.

PyGMO's implementation of MBH requires only two control parameters: Δ and N_{stop}.

Sequential Least Squares Programming (SLSQP)

Sequential Least Squares Programming, introduced by Dieter Kraft in 1988 [89], is based on the concept of Sequential Quadratic Programming, a set of algorithms design to tackle non-linear problems. Sequential Quadratic Programming is a powerful optimisation method as it allows non-linearity to be present in the objective function or in the constraints of the optimisation problem [90].

This algorithm is a derivative-based method and works by breaking the parent non-linear problem into simpler quadratic sub-problems. The objective function is evaluated by varying optimisation parameters while making sure constraints are observed [90]. This procedure is iterated until convergence is obtained and a feasible solution is produced, or a maximum number of iterations is reached. Usually, Newton's method is used to estimate the variation in the optimisation parameters through analysis of the behaviour of the objective function. If the optimiser senses that the objective function is increasing consistently (going "uphill"), then the direction of variation of the optimisation variables is changed so that the objective function decreases instead.

Because of this gradient-based characteristic, the algorithm's performance suffers for large numbers of optimisation variables, as numerical differentiation (e.g. finite differencing) applied to a problem with a higher number of variables becomes computationally heavier [90].

SLSQP is implemented as a part of the PyGMO and Scipy toolboxes [77, 82] and will be used in this work to solve constrained optimisation problems, such as the Periapsis Raising Manoeuvre.

The control parameters available in PyGMO are:

- **Maximum Number of Iterations**

- **Accuracy**
- ϵ: Step size for finite-difference derivative estimates

Compass Search (CS)

Compass Search [91] is a local optimiser that falls under the category of *direct search* method. This means that it does not use derivatives to find the minimum of the function, which is desired for highly non-linear problems where derivatives may not be well defined. It is presented here because it will be used for the optimisation of the interplanetary trajectory.

The principle behind CS is very simple. The algorithm begins by receiving a starting individual, x_i, and an initial step size, δ. Then, for each entry of the x_i's chromosome, the optimiser adds or subtracts δ. If the new perturbed chromosome has a better objective function value than x_i, it becomes the new x_i. Otherwise, $\delta = \dfrac{\delta}{2}$ and the process is repeated. The optimisation is stopped when δ reaches a minimum value. The simplicity of this algorithm means that it is very fast but not very robust and will be easily attracted to a local minimum [91].

The control parameters available in PyGMO are [82]:

- **Maximum number of evaluations**: Maximum number of overall repetitions.
- **Start Range**: Initial δ.
- **Stop Range**: Minimum δ, after which optimisation stops.
- **Reduction Coefficient**: Allows for a reduction in δ that is different from $\frac{1}{2}$.

Because the intention will be to refine a known solution, start and stop range should be small.

Optimisation Topology

PyGMO allows us to combine different algorithms and run them in parallel, while letting them exchange information, thanks to its asynchronous island model. The literature review in Chapter 1 suggests that employing optimisation algorithms in cooperation with each other will have a positive effect, as the optimisation will benefit from the stronger characteristics of each method. Successful examples of this are IDEA [47], an MBH+DE method, and Hemoglop [42], an evolutionary strategy that joins DE, Genetic Algorithm and Particle Swarm Optimisation.

Based on this tendency, two candidate topologies were designed for testing. The first topology, shown in figure 4.1, couples MBH with DE. The second one, in figure 4.2, builds on another idea [92] that joins SAAN and DE together to take advantage of SAAN's ability to avoid local minima and DE's fast convergence to the global optimum. Note that it doesn't make sense to pair MBH with SA because SA already possesses a strategy that encompasses sampling the search space to avoid local minima.

As each island corresponds to a computer core and there are only 8 available in the Intel Core-i7 being used, the topology is built with 7 islands so that one core remains available for daily PC operations.

Both alternatives exhibit a *rim* topology because this allows us to place a local optimiser in the center. This local optimiser will receive the solutions found by the global optimisers, refine them, and transmit the information back to the global level, improving the quality of the optimisation [92].

In both topologies, the local optimiser used is Compass Search. This algorithm is faster and simpler than Sequential Least Squares Programming and is already implemented in PyGMO.

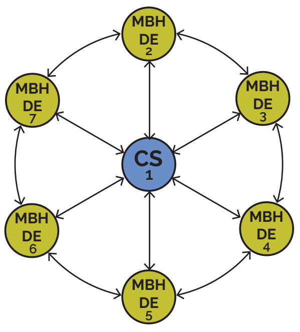


Figure 4.1: MBH+DE topology.

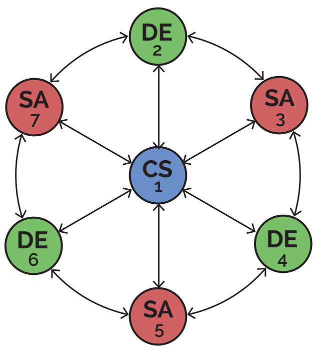


Figure 4.2: DE + SAAN topology.

Algorithm Racing and Topology Selection

In order to choose the parameters of the algorithms that will be used in each topology, the quality of each parameter set needs to be tested. This can be done via a procedure named Algorithm Racing, available through PyGMO [82]. Racing statistically compares several different algorithm configurations, by evolving populations and then comparing the best solutions, also referred to as champions, of each. When the racing algorithm arrives at the conclusion that a certain optimisation method is statistically inferior than the others, it is dropped from the race. This is an iterative procedure that continues until the best methods are found or until the maximum number of iterations has been exceeded.

The global optimisation problem that the methods were asked to solve, Cassini2, is available in the GTOP database [85] and is of the MGA-1DSM type. This problem mirrors the flight sequence of the Cassini probe through the solar system.

Tables 4.3 and 4.4 show the combinations of parameters tested for each method. For all algorithms $N_p = 100$. The winners are described in tables 4.5 and 4.6.

Table 4.3: DE Test Parameters. $N_g = 500$.

Parameter	F	CR	Variants
Values	0.1,0.3,0.5,0.75,0.8,0.95	0.1,0.5,0.8,0.9,0.95	1,2,3,4,5,6,7,8,9,10

Table 4.4: SAAN Test Parameters. Some parameters are based on suggestions by Izzo [37].

Parameter	Iterations	T_s	T_f	Nr. of Steps	Bin Size	Range
Values	10^4, 10^5, 10^6	10,10^5	0.036,0.073,1,138	1,10	10,20,40	1

From table 4.5 it is possible to conclude that, for MGA1DSM problems, the crossover probability has a higher effect on the convergence of the algorithm to the best known solution, with high crossover

Table 4.5: Winners of the DE racing.

Position	F	CR	Variant
1^{st}	0.75	0.95	1
2^{nd}	0.8	0.95	6
3^{rd}	0.8	0.95	1
4^{th}	0.5	0.95	9
5^{th}	0.8	0.9	6

Table 4.6: Winners of the SA racing.

Position	Iterations	T_s	T_f	Nr. of Steps	Bin Size
1^{st}	10^6	10	0.073	1	20
2^{nd}	10^5	10	0.036	1	20
3^{rd}	10^6	10	0.036	1	20
4^{th}	10^6	10	0.036	1	10
5^{th}	10^5	10	0.073	1	20

probabilities being preferred. Variants 1 and 6 also show better results than their competition. As for the SA trials, naturally a higher number of maximum iterations will converge better, in average, than test runs with a lower value. Also, all the winners found have the same start temperature and number of steps, with T_f spread between 0.036 and 0.073. Note that the test run that came in second place is similar to the one in third, but has a lower number of iterations. This is a reminder of the stochastic nature of these algorithms that, sometimes, return results that may induce the designer in error.

Finally, the candidate topologies are pitted against each other to find the best solutions for the Cassini2 problem [85]. Different configurations were ran for 50 trials and, in the end, both methods showed similar test results in terms of convergence. MBH+DE consistently achieved better mean fitness results than DE+SAAN but failed to obtain the global minimum, unlike the latter. MBH+DE, however, showed a faster runtime in general. The selected configuration is displayed in table 4.7.

Table 4.7: Global Optimisation Topology for the Interplanetary Phase

Algorithm	Islands	Configuration
Compass Search	1	Max. Evaluations = 1000 Start Range = 10^{-2} Stop Range = 10^{-5} Reduction Coeff = $\frac{1}{2}$
MBH+DE	2,6	DE-5 and MBH with $N_{stop} = 5$, $\Delta = 0.25$
	3,5	DE-1 and MBH with $N_{stop} = 5$, $\Delta = 0.25$
	4,7	DE-2 and MBH with $N_{stop} = 5$, $\Delta = 0.25$

4.3.2 Moon Tour

The procedure implemented to determine the moon tour, detailed later in Chapter 6, requires testing a vast number of possible combinations of VILT manoeuvres. This results in a difficult combinatorial optimisation problem. According to the literature, Branch and Bound (BB) [32, 40, 43, 44], a deterministic global optimisation scheme, has been the most frequently used algorithm to solve this type of problems in spacecraft trajectory optimisation.

Let us think of the entire search space as a rooted tree, where every branch is a possible solution for the problem. Branch and Bound starts at the root and explores the branches of the tree whilst applying pruning criteria that progressively trim sub-optimal branches, cutting down on the complexity of exhaustively searching through the entire search space. Branches are pruned by comparing their current fitness value to the best solution found by the algorithm at that point. A symbolic example of the branch and bound strategy is shown in figure 4.3.

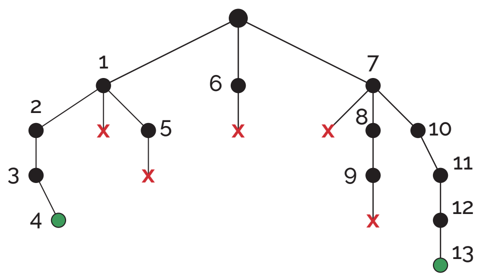


Figure 4.3: Example branch and bound scheme. Green circles mark solutions found by the algorithm. Red crosses mark branches that were pruned and whose children algorithm does not explore any further.

Common pruning criteria in spacecraft trajectory optimisation are the cost (ΔV) of the path and the time of flight associated with it [32, 40]. A Depth-First strategy is recommended for use with Branch and Bound because it produces full solutions faster, enabling the pruning criteria to be refined early on.

Chapter 5

Trajectory Design: Interplanetary Phase

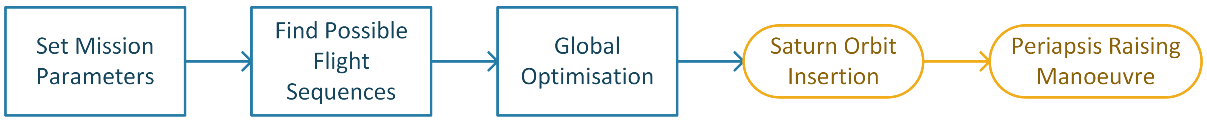


Figure 5.1: Design approach for the interplanetary and Saturn orbit insertion phases

The design process, pictured in figure 5.1, starts with fixing a suitable time frame for the mission to Enceladus to take place. By taking into account the scope of other missions already built, or being designed, it was considered that the parameters established in Table 5.1 are realistic. The Cassini-Huygens [15], TSSM [22] and Decadal Survey [21] missions are also displayed for comparison, with particular interest in the two latter which are foreseen to perform moon tours. Note that, unlike the mission to Enceladus, the TSSM mission is propelled by electric thrusters during the interplanetary phase. Cassini has a faster cruise time because it took advantage of a Jupiter gravity assist on the way to Saturn. Total mission time is usually constrained by the nominal lifetime of current nuclear power cores, around 14 years for typical RTGs [93] or 17 years for more recent ASRGs [21, 94].

Table 5.1: Mission start and duration parameters for the mission to Enceladus and others.

Parameter	Mission to Enceladus	Cassini-Huygens [14, 15]	TSSM [22]	Decadal Survey [21]
Mission Start	Jan 1, 2022 to Jan 1, 2030	Launched 15 Oct 1997	10 Sep 2020 to 30 Sep 2020	January 28 2023
Interplanetary Flight Time	(6 to 10) years	7 years	9 years	8.5 years
Moon Tour Duration	(0 to 4) years	12+ years performing science	2 years	3.5 years
Total Mission Duration	$\leq$ 17 years	20 years (extended mission)	14 years (extendable)	13.5 years

The next step is to define a set of possible sequences that the spacecraft can use to get to Saturn. Once this is done, all the information necessary to globally optimise the interplanetary trajectory is ready. The trajectory is optimised using the topology defined in the previous chapter to obtain sets of possible options. Finally, the spacecraft is inserted in Saturn's orbit and a periapsis raising manoeuvre is performed to reach Titan, starting the moon tour.

5.1 Finding Sequences

To find a multi gravity-assist trajectory between two planets one must first define the sequence of gravity assist bodies in the spacecraft's path. At a first glance, there would be an infinite number of paths that a spacecraft could follow to go from Earth to Saturn but, in practice, some combinations lead to a higher mission cost or unfeasible times of flight. Naturally, choosing an adequate sequence will have positive effects on fuel consumption, flight time and launch energy required.

To determine these flight paths, two methods were employed. The first method consists in finding trajectories to Saturn in literature and retrieve their flight paths. The second, more exhaustive, method makes use of the Tisserand graph to find *potential* flight sequences within an allotted time frame. This will be referred to as Tisserand sequencing from now on.

Alternatively, automatic sequencing [49] could have been performed. This method consists in allowing the optimiser to automatically define the sequence of planets from a pool of candidates. The upside to this approach is that it is fully automatic and requires no user knowledge or input. This method, however, is very inefficient as the combinations of planets generated may be far from optimal, vastly increasing the search space and the problem's complexity.

5.1.1 Tisserand Sequencing

The main idea behind Tisserand sequencing is to create an automated traversal algorithm that will find transfer sequences from Tisserand graphs. Unlike automatic sequencing, this method does not increase the complexity of the optimiser as it only needs to be executed once to find possible sequences. The downside to Tisserand sequencing is that, while it tells us if a trajectory is possible from an energy point of view, it may provide unfeasible solutions due to not taking planetary phasing into account (as discussed in Section 3.6) [78].

Flight sequences are found by linking the intersections of V_∞ curves from different planets, which correspond to *potential* transfer orbits between the intersecting planets. Ballistic resonant orbits ($T_{sc} = NT_{planet}$, where N is integer) are also added to the graph to allow potential resonant flybys.

The Tisserand Sequencer presented here will be divided into three parts:

1. **Construct the plot**: compute the Tisserand Graph for all the planets of interest. This was already explained in section 3.6. An example plot for the Solar System can be consulted in Appendix A (figure A.1).

2. **Build the search graph**: compute the intersection nodes, the resonant nodes and link them together in an adjacency matrix.

3. **Find paths**: run a path-finding algorithm on the search graph to find flight sequences.

Building the Search Graph

Building the search graph is a crucial part of the program. The structure implemented here has a multi-directional graph topology, where nodes are connected to each other in a bi-directional way. The basic data structure of the graph is the node. In this instance, a node is either a transfer orbit - given by the intersection of two V_∞ curves - or a resonant orbit. Nodes are connected by edges that will be represented computationally by using an adjacency dictionary – a fast-access structure that tells each node its neighbours.

The intersection nodes are found from the discrete V_∞ curve data by approximating each curve with a radial-basis function (RBF) and then using a root-finding bisection method to solve equation 5.1 for each set of intersecting curves,

$$rbf_{curve_1}(x) - rbf_{curve_2}(x) = 0. \tag{5.1}$$

The radial-basis function was chosen because it is fast and readily available through the Scipy scientific library for Python but other methods may be used. Because of the monotonic nature of the V_∞ curves, if two curves intersect they do so at one point only.

Resonant nodes are computed by using the radial-basis approximation of each V_∞ curve to find points in which the period of the spacecraft's orbit is an integer multiple of the period of the gravity-assist body. For the purposes of the Solar System, we are not interested in high resonances as these would have impractical times of flight. The resonances considered, for each planet in the inner solar system, were 1:1, 2:1 and 3:1.

Finally, to determine the edges of the graph, nodes must be linked together. Due to the bi-directional nature of the search graph, if node A is in range of node B, then it is possible to travel between node A and node B in two directions. A node will be in range of another if it matches two conditions:

1. It lies within the same V_∞ curve.
2. A flyby from A to B obeys the minimum flyby radius constraint.

Condition two is verified if

$$|\alpha_A - \alpha_B| \leq \delta_{max}, \tag{5.2}$$

where δ_{max} is given by equation 3.3, substituting r_{fb} with the planet's safe flyby radius. The pump angle for each node can be determined via equation 3.12. The edges are built by adding each node to the other's entry in the adjacency dictionary.

Finding Paths

The objective of the path-searching algorithm is to string together sequences of intersection and resonant nodes within the constraints provided by the user. These are:

- Start Planet and End Planet
- Launch V_∞
- Maximum Number of Flybys
- Maximum Time of Flight
- Maximum Number of Sequences

The program starts by finding a list of start and destination nodes. The start vertexes must be in the launch V_∞ curve and be intersection nodes. The destination vertexes must belong to the destination planet and must also be intersection nodes. This means that a path is determined when a transfer orbit to the final planet is found.

Next, the path-finding algorithm is executed with each start node as the root of the search tree. Branches of the search tree are pruned if their current fastest possible time-of-flight, or length (flyby count), are above the user-defined constraints. The path-finding stops when all paths from the selected start node to the destination planet are found or when the maximum number of sequences is reached.

Two alternatives for the path-finding algorithm were considered:

- Depth-First Search (DFS) - A node is selected as root and the algorithm proceeds to completely explore along a branch of the tree ("vertically") before proceeding to the next branch. Returns all paths between two nodes in no specific order.
- Breadth-First Search (BFS) - The algorithm explores the search tree level by level ("horizontally"). Returns all paths between two nodes with the path with shortest length being found first.

The first comment about these algorithms is that their complexity scales with the number of nodes and edges in a graph. Moreover, because the graph is multi-directional, if the maximum flyby number constraint is not implemented, then cyclic infinite paths may exist between two nodes. For the purpose of the Tisserand Sequencer, BFS was chosen. When compared to DFS it has the advantage of returning the paths by order of length, which allows for a better feel of how the algorithm is progressing. It also lets us implement a maximum number of sequences with certainty that the shorter ones, in terms of flyby number, will be found.

The fastest flight time can be found by testing all possible combinations of flight times for a trajectory, as detailed in table 5.2 where t_i is the solution of Kepler's equation at each encounter and T_{sc} is the period of the transfer orbit. When joining path nodes together to calculate the fastest time of flight, one must be careful because when the spacecraft arrives at a planet inbound or outbound, it must also leave inbound or outbound, respectively.

Table 5.2: Flight times. I: Inbound; O: Outbound. "Up" signifies a transfer from an inner to an outer planet and "Down" is the opposite.

"Up"	**"Down"**	**Flight Time**
I-I	O-O	$T_{sc} + t_1 - t_2$
O-I	O-I	$T_{sc} - t_1 - t_2$
I-O	I-O	$t_1 + t_2$
O-O	I-I	$t_2 - t_1$
Resonant Orbit		T_{sc}

5.2 Global Optimisation

Now that the time frame for the mission and the list of possible sequences is formulated, it is possible to start the global optimisation procedure.

The global optimiser implements the MGA-1DSM formulation as a PyGMO optimisation problem, receives the required inputs from the user and then optimises the problem by using the topology defined in Section 4.3.1.

The computational implementation of MGA-1DSM is already present in PyGMO and PyKEP because it is so frequently used as a baseline model for the study of preliminary spacecraft trajectories in the GTOC competition. This formulation is implemented in Python as a *PyGMO.problem* class. For box-bound problems, PyGMO requires the specification of two methods: *__init__* and *_objfun_impl* [82]:

- *__init__* is the method that initialises the problem, stores all relevant information for the optimisation and computes the chromosome's bounds from the user's inputs. The bounds set the minimum and maximum value that each entry of the chromosome can take during the optimisation procedure, to make sure that resulting trajectories are not outside the user's specifications. For this purpose, it requires the following information: the flight sequence, the launch window, the minimum and maximum allowed time of flight and the maximum allowed launch V_∞ magnitude.
- *_objfun_impl*, is the PyGMO problem's method that computes the objective function value of a given candidate chromosome. It does so by closely following the formulation described in Section 4.2.2 and the procedure described in Section 3.2. Algorithm 3 describes this implementation.

Algorithm 3 MGA-1DSM Objective Function Calculator

```
Input: Candidate chromosome
 1: Decode the chromosome to obtain each leg's TOF and launch $V_\infty$ components
 2: for planet in sequence do
 3:     Store: Launch / Arrival / Flyby Epoch
 4:     Store: Position and Velocity at Epoch $\leftarrow$ Ephemeris
 5: Calculate spacecraft's initial heliocentric velocity from $V_\infty$ and $V_{Earth}$
 6: Propagate spacecraft's trajectory until first DSM
 7: Solve a Lambert problem from location of first DSM to the next planet in sequence
 8: Compute the First DSM's $\Delta V$
 9: for remaining planets in sequence do
10:     Propagate flyby
11:     Propagate outgoing spacecraft trajectory until DSM
12:     Solve a Lambert problem from DSM to the next planet in the sequence
13:     Compute DSM $\Delta V$
14: Compute Arrival $V_\infty$ vector
15: Return: $\sum \Delta V_{DSM} + ||\mathbf{V}_{\infty_{\mathbf{Arrival}}}||$
```

Note: the two-body propagator used in this work is already implemented in PyKEP through the *propagate_lagrangian* function. More documentation is available online [81].

Everything is ready to start the global optimisation procedure. The implementation is described in algorithm 4. To begin with, the ephemeris data is loaded into special PyKEP classes that define planets. These hold information such as planetary data (radius, μ, ...) and the *eph* method which calls the ephemerides system and returns the position and velocity of the planet at a given epoch. Then, the optimiser receives the inputs required by the MGA-1DSM problem to initialise it and does so. Subsequently, each algorithm is initialised and configured through PyGMO, according to what was established in Section 4.3.1. Finally, the PyGMO archipelago can now be built, with each island, or separate algorithm, connected via a rim topology.

Each run of the optimiser yields a trajectory that *may be* the optimal trajectory for the defined user

Algorithm 4 Global Optimisation Procedure

```
1: Load interpolated ephemerides data
2: Initialise planets                                          ▷ PyKEP.planet data structures
3: Flight Sequence, Launch Window, Time of Flight range, Launch $V_\infty$ range $\leftarrow$ User Input
4: Initialise MGA-1DSM optimisation problem                   ▷ PyGMO.problem data structure
5: Configure optimisation algorithms
6: Build optimisation archipelago                             ▷ Topology defined in Section 4.3.1
7: Optimise
8: Retrieve champions from each optimisation island
9: Store: best trajectory data and plot, best solution's chromosome
```

parameters. In practice, because of the stochastic nature of the algorithms, the optimisation is ran several times for the same user configuration, to increase the probability of finding the global minimum.

The outputs are stored in a plain-text file with the following information:

1. **User Inputs**: Launch window, flight sequence, launch V_∞ range and time-of-flight range.

2. **Algorithm Configuration**: The configuration of the algorithms used to produce the result.

3. **Main Results**: A summary of the main results. Objective function value, optimisation chromosome, time of flight for each leg, total time of flight, interplanetary ΔV and arrival V_∞.

4. **Launch data**: Launch V_∞ and epoch.

5. **Leg data**: Information about each leg of the trajectory. Duration, flyby epoch and radius, DSM epoch and magnitude.

6. **Arrival data**: Arrival epoch and $\mathbf{V}_\infty$. This information will be transmitted to the Saturn orbit insertion phase.

Different trajectory candidates can now be easily evaluated and categorised in order to choose several options for the interplanetary phase. The choice of interplanetary trajectory options is intimately related to the nature of the mission and its operational constraints (e.g. engineer and ground station availability). Because this work lacks most of this information, the choice will be based on four main parameters: launch V_∞, interplanetary ΔV, launch date and arrival V_∞. The results of the interplanetary phase are detailed in Chapter 7.

5.3 Saturn Orbit Insertion

At the end of the interplanetary trajectory, it is necessary to perform a Saturn Orbit Insertion (SOI) manoeuvre to have the spacecraft enter into an orbit around Saturn and be captured by the planet's gravity — phase two of the mission to Enceladus begins.

Orbital operations during phase two are to be divided into two parts: orbital capture, which deals with the determination of the arrival and capture geometries and produces the capture ΔV as a main result, and a Periapsis Raising Manoeuvre (PRM), with the objective of targeting a flyby at Titan, resulting in the determination of the PRM's ΔV and Titan arrival geometry. Titan was chosen to start the moon tour because it is the largest moon in the Saturn system and, therefore, it allows a higher design flexibility as it is easier to change orbital characteristics, such as inclination, with Titan than other lower gravity moons. With Titan, it is possible to reduce the apoapsis of the spacecraft's orbit down from the long-period PRM orbit faster than with a smaller moon.

The models implemented in this phase continue to assume a two-body approach and do not take into account the effect of Saturn's oblateness (J_2 effects). Additionally, in order to link the interplanetary phase with the capture phase, it is assumed that the epoch of arrival at Saturn, given by the optimiser, is coincident with the epoch at which the orbit insertion manoeuvre is performed.

The most challenging and unique aspect of entering Saturn's orbit are its massive rings that pose a considerable threat to the spacecraft's structural integrity. Two gaps in the rings exhibit the best chance for the spacecraft to cross the ring plane safely with the least cost associated: the Huygens Gap, in the Cassini Division, and the F-G gap. Additionally, TSSM's mission designers also put forth the possibility of having a spacecraft cross the ring plane between the planet and the D ring, but this region is not well mapped and crossing so close to the planet involves considerable navigation challenges, such as dealing with possible atmospheric drag [22].

The Huygens gap is a 400 km-wide strip at the inner edge of the Cassini Division [56]. While 400 km seems rather large at an earthly scale, it is a narrow distance that poses considerable navigation challenges to a spacecraft that is almost 10 astronomical units away and must be completely autonomous. The F-G gap has two regions where the spacecraft can cross: an 8500 km strip between the outer edge of the F ring and the inner edge of the Janus / Epimetheus ringlet and another, larger, 12000 km strip between the outer edge of the Janus ringlet and the inner edge of the G ring, where the Cassini-Huygens probe crossed when it arrived at Saturn in 2004 [22, 55].

Because it is safer to cross in the F-G gap, that will be option A for the mission to Enceladus but the possibility to cross in the Cassini Division remains an alternative that is more fuel efficient because r_p will be lower (as will be explained in Section 5.3.1).

5.3.1 Capture

The design of the arrival geometry is based on B-Plane geometry (remember Section 3.3) and starts with the user specifying the arrival B-plane components, obtained from the desired orbital inclination and periapsis radius. Then, taking advantage of the definition of the B-plane, the arrival hyperbolic orbit is determined. Finally, the SOI manoeuvre is performed at the periapsis of the arrival hyperbola, where it has a lower cost, and the probe is inserted into a highly elliptical, co-planar orbit around Saturn. For the duration of the capture procedure, the designer can not be oblivious of Saturn's most obvious orbital constraint, the massive rings.

To understand the orbit capture procedure in more detail let's start by studying its high-level computational implementation, shown in algorithm 5, breaking it down in small steps.

To begin with, the SPICE [73] kernel containing useful reference frame conversions is loaded. This allows easy conversion between the J2000 inertial reference frame and the IAU_SATURN planet-centred,

Algorithm 5 Saturn Orbit Capture

```
Input: $\mathbf{V}_\infty$, Arrival Epoch, $r_{periapsis}$, $e_{capture}$
1: Load: SPICE Kernel
2: Reference Frame conversions ← SPICE                                    ▷ J2000 to IAU_SATURN
3: Convert arrival asymptote from J2000 to IAU_SATURN and get $\delta_\infty$
4: Compute Saturn's rings in the IAU reference frame            ▷ Will be needed to test collisions
5: Calculate B-Plane axes and B-Plane components
6: Compute the arrival hyperbolic orbit
7: Propagate backwards from periapsis and detect ring crossings / impacts
8: Compute insertion $\mathbf{\Delta V}$
9: Propagate forward from periapsis to apoapsis and detect ring crossings / impacts
Output: $\mathbf{r_{apoapsis}}$,$\mathbf{v_{apoapsis}}$ and epoch at apoapsis.
```

rotating, reference frame at a given epoch. IAU_SATURN, hereafter referred to as IAU only, is important because it defines Saturn's equatorial plane, which is approximately coincident with that of the rings and moons. For the duration of the design of this phase, orbital inclination will be measured with reference to Saturn's equatorial plane and not to J2000's reference frame. The IAU body-fixed reference frame is defined as follows:

- $\mathbf{\hat{x}_{IAU}}$: Unit vector in the direction of the prime meridian of the planet, as defined by IAU convention [70].
- $\mathbf{\hat{z}_{IAU}}$: Unit vector normal to the equator, pointing towards the planet's north pole.
- $\mathbf{\hat{y}_{IAU}}$: Unit vector that completes the right-handed reference frame.

Then, the arrival asymptote, which is the symmetric of the arrival V_∞ (because it points towards Saturn and not from Saturn), is converted from the J2000 frame, used in the interplanetary phase, to the IAU reference frame so that declination can be calculated from equation 3.7. Note that, because the planet rotates, the IAU frame will not be in the same positions when the spacecraft is at periapsis or when it is at "infinity", crossing Saturn's sphere of influence. This, however, does not change the declination because it is measured relatively to the Z-axis, which is the axis of rotation of the reference frame and is approximately fixed (if the precession and nutation of the axis are not taken into account).

B-Plane

B-plane components, θ_B and b, are derived from the user-defined inclination and periapsis radius, through equations 3.5 and 3.2, respectively. Because it is desirable to enter Saturn with the lowest possible inclination, as the moon tour phase will require orbits to be in the orbital plane of the moons, the inclination is set to match the arrival declination, the lowest value it can take due to the constraint set by equation 3.6. This is equivalent to $\theta_B = 0$.

The B-plane is defined in the J2000 inertial frame and axes are as described in Section 3.3:

- $\mathbf{\hat{S}} = -\dfrac{\mathbf{V}_\infty}{||\mathbf{V}_\infty||}$ — A unit vector parallel to the incoming asymptote.
- $\mathbf{\hat{T}} = \hat{S} \times [0,0,1]_{IAU\to J2000}$ — A unit vector parallel to the reference plane (in this case the equator) and perpendicular to the incoming asymptote.

- $\hat{\mathbf{R}} = \hat{S} \times \hat{T}$ — A unit vector completing the right-handed reference frame.

Arrival Hyperbolic Orbit

Once the B-Plane and the B-plane vector $\mathbf{B}$ (see equation 3.4) are defined, the point of entry of the spacecraft in Saturn's sphere of influence can be determined. The sphere of influence is the region of space where Saturn's gravity attraction is dominant over that of the Sun and is approximately given by

$$r_{sphere} \approx a_{saturn} \left(\frac{m_{Saturn}}{m_{Sun}} \right)^{2/5} \approx 54.6 \times 10^9 m, \tag{5.3}$$

where a_{saturn} is the semi-major axis of Saturn's orbit around the Sun and m is mass. The point of entry in the sphere of influence is found by drawing a line, starting at the edge of the b-plane vector, that is perpendicular to the B-plane, until it intersects the sphere of influence,

$$\mathbf{r_{s/c,\infty}} \approx \mathbf{B} + r_{sphere}.(-\hat{\mathbf{S}}). \tag{5.4}$$

From here on, the arrival hyperbola is fully defined and the periapsis vector and periapsis velocity vector can be found by propagating the hyperbola forward from the initial conditions and detecting where the distance to Saturn is minimum.

Orbit Insertion

Orbit insertion is performed at the common periapsis of the arrival hyperbola and the post-insertion ellipse as it is the point where the manoeuvre has the lowest ΔV cost. The burn is performed anti-parallel to the periapsis velocity vector because no changes are desired in the line of apsides or orbital inclination, at this point.

The insertion ΔV is given by the subtraction of the hyperbola's velocity at periapsis, $V_{h|p}$, and the desired ellipse's velocity at periapsis, $V_{e|p}$,

$$\Delta V_{SOI} = V_{h|p} - V_{e|p} = V_{h|p} - \sqrt{2\mu_{Saturn} \left(\frac{1}{r_p} - \frac{1}{2a_{capture}} \right)}, \tag{5.5}$$

where $V_{h|p}$ is given by the forward propagation of the arrival hyperbola until periapsis, r_p is the magnitude of the periapsis vector and $a_{capture}$ is the semi-major axis of the user-defined post-capture ellipse,

$$a_{capture} = \frac{r_p}{1 - e_{capture}}. \tag{5.6}$$

Since r_p and $e_{capture}$ are user-defined values, it is important to understand how their choice affects the outcome of the problem. From inspection of equation 5.5 it is clear that a high eccentricity and low r_p lead to a lower ΔV_{SOI}. However, it is not as simple as choosing the values for these quantities that minimise ΔV: varying r_p and $e_{capture}$ greatly affects the point of passage of the spacecraft through the ring plane and, consequently, whether the spacecraft collides with the rings or makes it through safely!

Ring Crossing Detection

Ring collision detection is performed by propagating the trajectory of the spacecraft backward and forward from periapsis, at a fine step. At each instant, the position vector of the spacecraft is converted to the IAU frame. When the sign of r_z changes it means the spacecraft crossed the equator plane, which is also the ring plane. Then, the magnitude of the position vector at crossing is cross-checked with the position of the rings. If impact is detected, r_p and $e_{capture}$ need to be adjusted by the user until the spacecraft clears the rings.

Usual values for $e_{capture}$ are very close to 1, such as 0.99 or 0.985 (TSSM's and Cassini's, respectively) [15, 22] and this tends to stay fixed, while r_p, which has a smaller effect on the variation of ΔV_{SOI}, is adjusted to ensure passage through the rings.

5.3.2 Periapsis Raising Manoeuvre

After the orbital capture manoeuvre, it is necessary to perform a Periapsis Raising Manoeuvre (PRM) to have the spacecraft's orbit clear the rings and, more importantly, to target the first flyby of Titan, starting the moon tour. The PRM burn changes periapsis more efficiently when placed at apoapsis of the post-insertion ellipse, although this may not be an optimal location if the goal is to reach Titan. The ΔV associated with the PRM manoeuvre will vary with the period of the post-capture orbit. Typically, longer orbits with a high period and eccentricity produce cheaper PRM manoeuvres.

The method that was chosen to design the PRM is rather straightforward: solve Lambert problems that start in the vicinity of the apoapsis of the post-capture orbit and end at Titan, with variable time of flight. This is complicated by the need to impose two constraints:

1. **Inclination**: The models implemented for the moon tour assume a planar geometry, which means the inclination of the post-PRM orbit should be as low as possible to permit a first flyby of Titan that will perform an orbital plane change to a near-equatorial orbit, reducing errors associated with the use of a 2D approach.

2. $\mathbf{V_{\infty,Titan}}$: It is desirable, for the moon tour, that the arrival V_∞ with respect to Titan is as low as possible. This reduces the cost of V_∞-leveraging manoeuvres required to perform the Titan phase of the moon tour. More details in Chapter 6.

This results in a constrained optimisation problem with the following specific definition:

$$\text{Minimise: } \Delta V_{PRM}(\mathbf{p}), \tag{5.7a}$$

$$\text{Subject to: } i(\mathbf{p}) \leq i_{max}, \qquad \text{boundary constraints} \tag{5.7b}$$

$$V_{\infty,Titan}(\mathbf{p}) \leq V_{\infty_{max}}, \tag{5.7c}$$

where **p** is the decision vector, or chromosome, given by

$$\mathbf{p} = [\Delta t_{apo}, \text{tof}_{PRM}], \tag{5.8}$$

with tof$_{PRM}$ being the post-PRM orbit time of flight and Δt_{apo} is the time from apoapsis of the post-insertion orbit at which the Lambert problem is launched.

In PyGMO, constrained problems require an additional method, *_compute_constraints_impl*. This method computes the violation of each constraint for a given chromosome and defines if a trajectory is feasible or infeasible. The objective function, defined by method *_objfun_impl* is calculated as shown in algorithm 6. $\mathbf{r_{apo}}$ and $\mathbf{v_{apo}}$ indicate the state of the spacecraft at the apoapsis of the post-insertion orbit (instant t_{apo}). These quantities are all transmitted from the capture phase.

Algorithm 6 PRM Objective Function

```
Input: $\Delta t_{apo}$, $tof_{PRM}$, $\mathbf{r_{apo}}$, $\mathbf{v_{apo}}$, $t_{apo}$
1: $\mathbf{r_i}$, $\mathbf{v_i}$ ←Propagate spacecraft's trajectory for $\Delta t_{apo}$ seconds
2: $\mathbf{r_{titan}}$, $\mathbf{v_{titan}}$ ← Ephemeris at $t_{apo} + \Delta t_{apo} + $ tof$_{PRM}$
3: Solve Lambert problem                                  ▷ From $\mathbf{r_i}$ to $\mathbf{r_{titan}}$ within tof$_{PRM}$
4: Compute $\Delta V_{PRM}$                                 ▷ From $\mathbf{v_i}$ and $\mathbf{v_{lambert,i}}$
Output: $\Delta V_{PRM}$
```

The determination of the inclination and $V_{\infty,Titan}$ for the *_compute_constraints_impl* method follows the same strategy. Inclination is computed by transforming the spacecraft state vector, evaluated at any point of the post-PRM orbit, to the IAU frame and then obtaining the orbital elements from the new state vector [64].

The post-PRM maximum inclination, i_{max}, and Titan $V_{\infty_{max}}$ are problem specific and need to be tested to see what yields feasible trajectories. This $V_{\infty_{max}}$ is an important parameter with direct effect in the quality of the moon tour solution. It should be as low as possible so that the cost and number of VILTs for the Titan and Rhea phase of the moon tour are reduced, as it will be seen in Chapter 7.

The constrained optimisation problem is solved by means of a local optimiser. There is no need for a global optimiser due to the narrow search space and small number of optimisation variables. For the purposes of this work, Sequential Least Squares Programming (SLSQP) was chosen because it is already implemented in PyGMO and solves constrained optimisation problems. In PyGMO's implementation of SLSQP [82], the initial guesses are generated at random from user-defined bounds for each entry of the chromosome. Due to this stochastic component, optimisation is ran several times to build a set of feasible trajectories from which the best is chosen and returned to the user.

The results of the Saturn orbit insertion and PRM are stored in a plain-text file that sums up all relevant information from arrival at Saturn to arrival at Titan.

Chapter 6

Trajectory Design: Moon Tour

The moon tour is one of the key aspects of the mission to Enceladus. If, after Saturn insertion and PRM, a direct Hohmann transfer from Titan's orbit to Enceladus were to be performed, the cost of Enceladus Orbit Insertion (EOI) would be around 3.6 km/s [57] for a 100 km circular orbit, an unacceptably high value for a deep space mission. The moon tour's main purpose is to drastically reduce this value by stringing together sets of transfers that progressively lower the periapsis of the spacecraft's orbit to the orbit of Enceladus (a procedure known as *pump down*).

Pump down can be achieved by linking together resonant orbits, at each moon, that will rotate the V_∞ vector by the 180º required to perform, ideally, Hohmann transfers from one moon to another, as the spacecraft orbit's periapsis becomes its new apoapsis [53]. Figure 6.1 illustrates this. Resonant orbits are useful because they allow a "free" return to the moon. A purely ballistic resonant moon tour would require no ΔV.

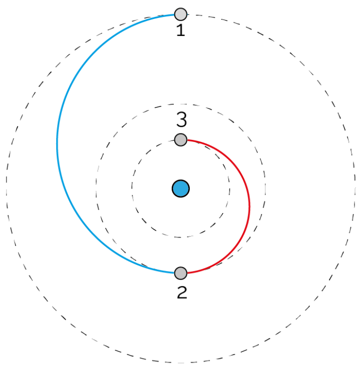


Figure 6.1: Hohmann transfers between moons.

However, because the moons involved have low mass, the amount of bending they can provide to the velocity vector is limited (remember Tisserand graphs). Therefore, a ballistic resonant tour as described would include a lot of resonant transfers and, where low-period resonances are not available, lengthy resonances would need to be used, resulting in a very long moon tour.

But what if a manoeuvre could help by providing additional bending? This is where the V_∞-Leveraging technique is useful. Through the use of a small ΔV, a VILT changes the spacecraft orbit's V_∞ with respect to the gravity-assist body by modifying the point of encounter between the spacecraft and the moon. Having lower V_∞ serves two purposes: the amount of bending that is available from the moon is increased (see equation 3.3) and orbit insertion is cheaper. The use of this technique is rather novel and opens up the possibility to visit exciting low mass moons such as Enceladus. It was first introduced for this purpose by NASA, in 2010's Decadal Survey [21] and has since seen further study [53, 57].

The strategy implemented in this work will make use of a deterministic Branch and Bound method to string together a tour of VILTs and resonant orbits. It will be assumed here that all manoeuvres are performed in the orbital plane of the moons and the moons' orbits will initially be considered circular, agreeing with the approximations laid down in sections 3.6 and 3.7. This is a reasonable assumption since the orbital eccentricity of the moons of interest is indeed small (see section 2.3).

The Periapsis Raising Manoeuvre results in the first encounter of the spacecraft with Titan. At this point, the incoming orbit has a certain inclination and V_∞ with respect to Titan, given by the outputs of the PRM optimiser. Before starting the VILT tour, it is necessary to change the plane of the spacecraft's orbit from whatever inclination it had before to the orbital plane of Titan. To save ΔV, this is performed through a first gravity assist at Titan that changes the plane of the spacecraft's orbit. In addition to being able to accomplish the plane change, the post-flyby orbit needs to be resonant so that the spacecraft returns to Titan to start the VILT tour.

6.1 Building Block

The building block of the moon tour is the V_∞-leveraging transfer (see Section 3.7.2). The Tisserand plot is useful to understand the idea behind linking these manoeuvres, as pictured in Figure 6.2.

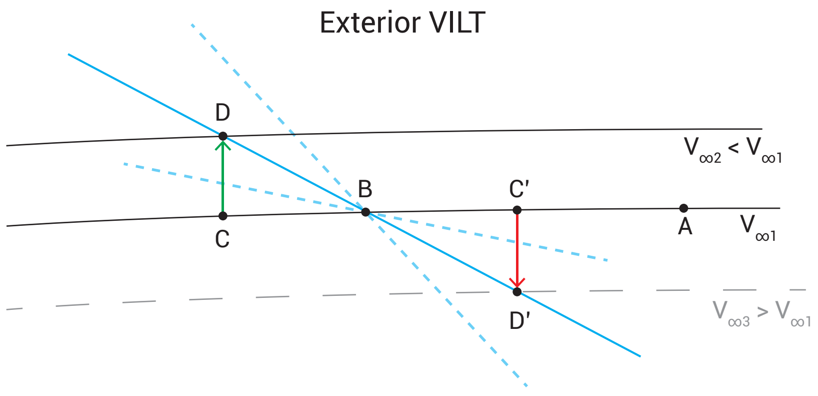


Figure 6.2: Example of an exterior VILT overlaid on a r_a-r_p Tisserand plot.

The blue line in Figure 6.2 is the result of computing several phase-fixed exterior N:M VILTs starting at $V_{\infty1}$, where each point in the line is associated with a different post-VILT V_∞. Changing VILT parameters, such as L, k_{ei} or the geometry, yields different curves (like the blue dashed lines). For a fixed set of (EXT/INT) N:M VILTs starting at $V_{\infty1}$ there are $4M$ curves like the blue one — one for each combination of geometry and L.

Point B, where $\Delta V_{VILT} = 0$ and V_∞ is the same as in point A, is either a resonant ballistic transfer orbit (II and OO geometries) or a non-resonant ballistic transfer orbit (OI and IO geometries). Non-resonant ballistic transfers are orbits that, despite not having a period that is an integer multiple of T_{Moon}, still encounter the moon at one of the intersection points of the two orbits.

If the incoming orbit is at point **A**, there are three alternatives to move across the Tisserand plot. If point **C** is within the maximum possible rotation, then it is possible do a gravity-assist to move along the $V_{\infty 1}$ curve to reach that point and then perform an exterior VILT (constant r_a) to move to **D**. At point **D**, the spacecraft re-encounters the moon and can now repeat this process in curve $V_{\infty 2}$. Alternatively, the spacecraft can perform a flyby to move to point **B** and enter a resonant orbit that will end back at the gravity-assist body, with no change in V_∞ and no ΔV cost.

If neither **B** nor **C** are reachable due to the minimum flyby constraint, it is still possible to perform a gravity assist to point **C'** and hop to a higher V_∞ curve, allowing the spacecraft to possibly reach new resonances to continue the VILT tour. While this is non-intuitive and goes against the objective to lower V_∞, if this particular VILT has a low resonance, associated with an attractively low flight time, it may justify the increase.

This procedure is iteratively repeated for each combination of VILT parameters, at each resonance, to form tours like the one shown in figure 6.3 which is, in no way, optimal. The procedure outlined here is compatible with a search tree algorithm such as Branch and Bound.

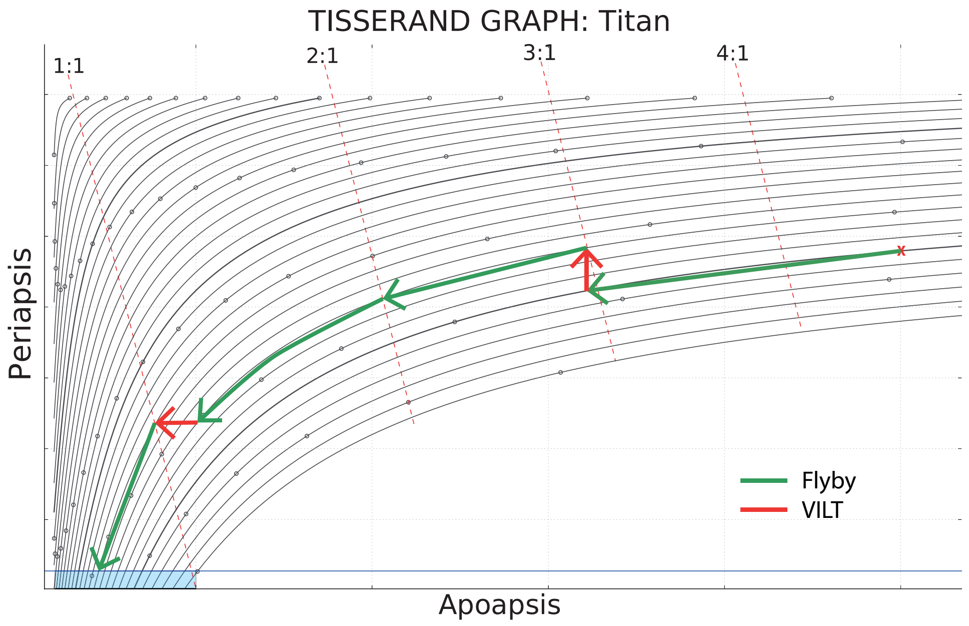


Figure 6.3: Example of a non-optimal VILT tour at Titan. V_∞ curves range from 0.1 km/s (top) to 3.5 km/s (bottom) at 0.1 km/s steps. Red dashed lines represent ballistic resonant transfers. Red lines are VILTs and green lines are flybys. The blue line represents the orbital radius of the next moon in the sequence (Rhea). Orbits in, or below, the blue rectangle are potential transfer orbits.

Figure 6.3 shows three types of possible manoeuvres: an exterior VILT (ending in a 3:1 resonance), a ballistic resonant transfer (2:1) and an interior VILT (ending in a 1:1 resonance). Orbits in the blue rectangle are Saturn-centred orbits with an apoapsis at, or above, Titan's orbit and a periapsis at, or below, Rhea's orbit, meaning that these are orbits that intercept Rhea's orbit. Assuming there is a perfect phasing between moons at the time of the last flyby, it would be possible to encounter Rhea without the need to perform an impulsive manoeuvre.

6.2 Finding Tours

The moon tour is divided into several phases. Each phase is characterised by a VILT tour around one moon. Therefore, there are 5 phases in this moon tour: Titan, Rhea, Dione, Tethys and Enceladus. The first four are similar as the objective is to reach a transfer orbit to the next moon. The Enceladus phase has the different goal of lowering V_∞ as much as possible for orbit insertion. This is sometimes referred to as an endgame [80].

Each phase of the moon tour is divided in three main components. The phase starts with a branch and bound search, where a library of solutions is built. Then, one or more solutions are selected according to criteria, such as ΔV and time of flight. Finally, the phase is connected to the next one through a constrained optimisation problem based in a MGA-1DSM formulation.

6.2.1 Preparing the Search

The first step in each phase is to construct a list of allowed resonances that the branch and bound algorithm will test. In general, the N number of moon revolutions is limited so that the maximum time of flight for a resonant orbit is around 30 days (except for Titan). Short of testing all the possibilities, there is no *a priori* way of telling what are the best resonances. With some design experience, it is possible to detect higher resonances in the list, that are not required to build the moon tour, and remove them, improving the performance of the algorithm.

The resonances allowed for each phase are shown in table 6.1.

Table 6.1: Resonances allowed for the branch and bound search.

Moon	Resonances Allowed
Titan	4:1, 3:1, 2:1, 1:1
Rhea	3:1, 8:3, 5:2, 7:3, 9:4, 11:5, 2:1, 11:6, 7:4, 8:5, 3:2, 10:7, 7:5, 9:7, 6:5, 9:8, 1:1, 5:6
Dione	4:3, 9:7, 5:4, 6:5, 7:6, 8:7, 9:8, 10:9, 11:10, 12:11, 1:1, 11:12, 10:11, 9:10, 8:9, 7:8, 6:7
Tethys	11:9, 6:5 ,7:6 ,8:7 ,9:8 ,10:9 ,11:10 ,12:11 ,13:12 ,14:13 ,15:14 ,1:1 ,13:14 ,10:11 ,9:10 ,7:8
Enceladus	7:6, 20:17, 15:13, 8:7, 17:15, 9:8, 19:17, 10:9, 21:19, 11:10, 12:11, 13:12, 14:13, 15:14, 16:15, 19:18, 24:23

Next, the search space is partitioned in discrete V_∞ values, like in a Tisserand graph. Naturally, the more V_∞ levels present in the search, the slower the algorithm is. For the purposes of this work, all solutions were obtained at 50 m/s steps (except Titan, which allowed a reduction in step).

For each of these levels, the resonant and non-resonant ballistic orbits, corresponding to the allowed resonances, are pre-calculated and stored as pump angles. The R_a and R_p of these resonant and non-resonant ballistic orbits can be obtained by solving a phase-fixed VILT with $V_{\infty_1} = V_{\infty_2}$. The pump angle can then be obtained from the R_a and R_p of the orbit through equation 3.12.

The last thing left to prepare before starting the Branch and Bound search are the start and goal nodes. Goal nodes are found from the intersection of the V_∞ levels with the horizontal line that marks the periapsis of the orbit of the next moon in the sequence (which would be the points in the top edge of the blue rectangle in figure 6.3). For Enceladus, result detection is a little different. The search algorithm detects whether the node being tested has a V_∞ that equals a user-defined V_∞ level. If it does, it is considered a result hit and stored.

Start nodes are obtained from the previous phase of the moon tour. In Titan's case, the start node is given by the R_a and R_p of the post-PRM orbit. For other moons, the start node is given by the R_a and R_p of the transfer orbit that connects the two phases together (see section 6.2.3). Start and goal node coordinates are also stored as pump angles via the node data structure.

A node is the basic data structure of the Branch and Bound search. It represents an orbit in the Tisserand plot and contains the following information:

- **Pump Angle (α)**: Pump angle in the V_∞ curve where the node is located.
- **Geometry**: OO/OI/IO/II
- **Type**: Start, Goal, VILT or Resonant
- **Cost**: ΔV
- **Time of Flight**: Only stored for nodes of type VILT and resonant.
- **Orbit Info**: Flyby radius, resonance N:M, VILT parameters (L, k_{ei}, post-flyby α) and V_∞ level

6.2.2 Branch and Bound

All the information to start the branch and bound search is now available. The rules that define the search algorithm are the following:

- **Pruning Criteria**: a branch is pruned when the path's time of flight has exceeded a user-given TOF_{max} or when the path's cost is higher than a user-given ΔV_{max} or than the actual best cost.
- **Stop Conditions**: the search stops when a maximum number of results has been reached or when the entire tree has been explored.
- **Result Criteria**: A path is a result when it includes the goal node and its cost is equal or lower than the actual best.
- Flybys have to obey the minimum flyby radius constraint.
- VILTs are constrained to a user-given $\Delta V_{VILT_{max}}$. This is done to avoid needlessly increasing the search space with costly manoeuvres.
- Flybys can not be performed in an "uphill" direction, meaning $\alpha_{post-GA} > \alpha_{pre-GA}$ has to be verified.
- The velocity vector can not be turned by 180^o in a single flyby. This means that if the spacecraft arrives outbound / inbound, it leaves outbound / inbound. This prunes out infeasible flybys and reduces the number of VILTs that have to be computed at every step.

Algorithm 7 VILT Tour Branch and Bound Search

```
Input: Start node, goal nodes, pruning and stop criteria, resonant transfers, $V_\infty$ levels
Initialise best cost                                                    ▷ Can be $\infty$ or a user-given value
Initialise the stack with the start node
while stack is not empty or nr. results $\leq$ max nr. results do
    Vertex, Path $\leftarrow$ Last node in stack
    if Path Cost $>$ best cost or Path TOF $>$ Max. TOF then
        Prune branch and go to 2
    Initialise an empty adjacency list
    Allowed geometries $\leftarrow$ OI,OO if vertex geometry is IO or OO else IO,II
    for N:M resonance in the resonance list of the current $V_\infty$ level do
        if $\alpha_{resonance} < \alpha_{vertex}$ then                          ▷ Prevents uphill flybys
            Go to 7
        if $|\alpha_{resonance} - \alpha_{vertex}| \leq \delta_{max}$ then          ▷ Minimum flyby radius constraint
            Add two resonant nodes to vertex's adjacency list (one for each allowed geometry)
        for $V_\infty \in V_\infty$ levels do                             ▷ Add possible VILTs to the adjacency list
            for exterior, interior do
                for geometry in allowed geometries do
                    for L from 0 to M-1 do
                        Compute phase-fixed VILT starting at $V_{\infty_{vertex}}$
                        if $\Delta V_{VILT} < \Delta V_{VILT_{max}}$ then
                            if $|\alpha_{pre-GA} - \alpha_{vertex}| \leq \delta_{max}$ then    ▷ Minimum flyby radius constraint
                                Add VILT node to adjacency list
        for node in adjacency list do
            if $|\alpha_{node} - \alpha_{goal}| \leq \delta_{max}$ then              ▷ If node is in range of a goal node
                if cost of path + node $<$ best cost then
                    best cost $\leftarrow$ cost of path + node                   ▷ Set new best cost
                Store: path + node                                       ▷ Store the result
                Increment result counter and go to 22
            else
                Add the node and path information to the stack and go to 3
Output: Stored tour paths
```

With all the rules and conditions established, it is now possible to proceed to the implementation of the branch and bound scheme. The pseudo-code is described in algorithm 7.

Branch and bound assumes a depth-first scheme, where the last nodes to enter the stack are the first to be analysed (last in, first out). The depth-first search allows the search algorithm to refine the best cost early on. This helps to narrow down the remaining search space as pruning will be more aggressive. The downside to this approach is that the complexity of the algorithm is proportional to the dimension of the tree. If too many resonances are added to the list, it may take a long time for the algorithm to find results and will make it hard to exhaustively search the entire space because few branches are pruned. If computational time is a problem, the number of computed VILTs can be drastically reduced by limiting the L parameter of the VILTs to L = M - 1. This value was proven to generally produce the most efficient VILTs [80].

The algorithm stores its results in a plain-text file with all the relevant information such as VILT information, cost and time of flight of each manoeuvre and flyby radius.

Potential solutions for a given phase of the moon tour are selected based on ΔV and time of flight. Sometimes, because phasing has to be taken into account, a tour that exhibits the lowest ΔV and time of flight might not be the best if the transfer between consecutive moons has a high cost. This means that choosing a trajectory for each phase of the moon tour is an iterative procedure as the user has to select trajectories and compute the inter-moon transfers until a satisfactory solution is found. Ideally, the phasing problem should be solved for all solutions obtained by the search algorithm but this could lead to high computation times. This automation remains a possibility to be investigated.

6.2.3 Inter-moon Transfers

Now that we have a potential tour for a selected phase, it is time to attach the solution to an epoch and deal with the transfer orbits between one moon and the next. This procedure begins at the last flyby in the tour where, to determine a transfer orbit to the next moon in the sequence, a constrained optimisation problem is launched. The decision vector of this problem is given by

$$\mathbf{p} = [T, \eta, \beta, r_{fb}], \tag{6.1}$$

where T is the transfer time of flight, η is the non-dimensional parameter that defines where in the arc the DSM is executed, β is the 3D flyby plane change angle and r_{fb} is the flyby radius of the last flyby.

The problem is similar to MGA-1DSM in formulation. The user inputs the pre-flyby velocity, time of flight bounds and desired arrival V_∞ and inclination with respect to the next moon. The objective function then works by calculating the post-flyby velocity vector, propagating the post-flyby trajectory until the time of the DSM (given by ηT) and then launching a Lambert problem to have the spacecraft meet the next moon after $t = (1 - \eta)T$. If T is big enough, then the spacecraft can perform multiple revolutions of the central body before the DSM is executed. This is convenient in the Saturn system because moon periods are small, when compared to the Solar system. A multi revolution geometry allows the spacecraft to “wait” until the alignment with the moon is most convenient to perform the impulsive manoeuvre.

Furthermore, it is important for the optimiser to be able to configure the flyby’s properties, such as flyby height and plane change angle, allowing it to target different V_∞ curves in the next phase of the moon tour. Figure 6.4 illustrates this. For different target V_∞ levels, different flyby heights are required to move from the green circle (the last flyby in the phase) to each of the red crosses.

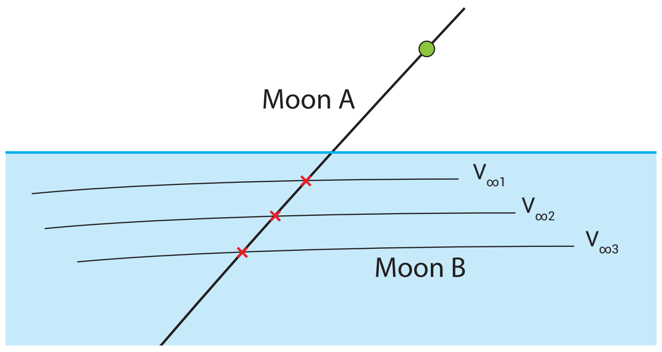


Figure 6.4: Example tisserand graph portraying potential transfer orbits between moons. The blue rectangle represents orbits with a periapsis lower than that of the next moon. The green circle is the last flyby in moon A’s tour.

This problem is subject to two equality constraints,

$$V_\infty = V_{\infty_{next}} \qquad \text{and} \qquad i_{transfer} = i_{next}, \tag{6.2}$$

where i is the inclination. The arrival V_∞ at the next moon must equal a user-defined value and the inclination of the arrival orbit must be in the orbital plane of the next moon in the sequence, to maintain the reasonability of the assumption that all flybys are planar during a moon tour phase. The complete Tisserand plot of the moon system (an example is shown in figure A.2) is useful when choosing $V_{\infty_{next}}$, as it allows an overview of the paths that lead to the lowest possible V_∞ level when arriving at Enceladus.

Unlike the interplanetary trajectory, this optimisation problem is solved with the SLSQP algorithm implemented in PyGMO (see section 4.3.1). This problem is suited for local optimisers due to its small number of optimisation variables. Stochastic algorithms were not chosen here due to the difficulty that this type of algorithms has in handling constraints (see Chapter 4).

The coordinates of the starting point for the next phase in the moon tour are fully defined by the arrival V_∞, and the R_a,R_p of the transfer orbit. The branch and bound search can now be used to find solutions for the next phase and the process is repeated.

6.3 Enceladus Orbit Insertion

The capture procedure around Enceladus mirrors the method explained in Section 5.3. The key differences are that the post-insertion orbit is different and there is no need for a PRM.

Enceladus' proximity to Saturn, coupled with its low mass and orbital perturbations induced by other moons, leads to instabilities in the orbit of spacecraft around the moon [95]. Traditional planetary science orbits aim for a near-polar inclination that allows the sensors to acquire data with near-global coverage and under varied lighting circumstances. At Enceladus, these traditional orbits are not long-term stable, either requiring constant maintenance (expensive) or impacting the surface of the moon within 1 to 3 days [95]. Through detailed stability analysis using unaveraged models, researchers [95] have found long-term stable orbits ($>$180 days) around Enceladus with relatively high inclinations (up to 65^o) at average altitudes, around 200 km, that are stated to be suitable for scientific missions. An example orbit, determined by Russell and Lara [95], is shown in table 6.2. The orbital elements are defined in the IAU-defined Enceladus body fixed frame [70].

Table 6.2: Enceladus science orbit. Subscripted quantities are initial values.

Element	Units	
h_0	km	295
a_0	km	498.763
e_0	—	0.075947
i_0	^{o}	58.725
Period	days	10.26
Average Inclination	^{o}	60.3

The spacecraft will be inserted into this long-term stable orbit upon reaching Enceladus.

Chapter 7

Results

The methods outlined in the previous chapters are applied to design a mission to the icy moon Enceladus. Results are shown for each main phase of the mission. Several options are provided for the interplanetary cruise but only one is chosen to perform the insertion and moon tour phases. The moon tour's five phases are individually presented and then the final trajectory obtained is analysed in-depth and compared to other missions and mission concepts from literature.

7.1 Phase I — Earth to Saturn

The interplanetary phase of the mission is fixed by the constraints established in Chapter 5: the mission should take place from January 1, 2022 to January 1, 2030 and the cruise duration can not exceed 10 years. The trajectory design starts with establishing possible sequences through use of the Tisserand sequencer (see Section 5.1.1) and then globally optimising each of these sequences to find the best trajectory candidates.

7.1.1 Flight Sequences

Table 7.1 shows the flight sequences that will be tested in the optimisation phase. The Tisserand Sequencer parameters used to obtain these results were:

- **Planets**: Venus, Earth, Jupiter, Saturn
- **Tisserand Plot Resolution**: V_∞ ranging from 1 km/s to 12 km/s at 1 km/s steps.
- **Launch** V_∞: 3.0 km/s, 4.0 km/s and 9.0 km/s
- **Max. Flyby Number**: 4
- **Max. Time of Flight**: 10 years

Table 7.1 shows that the Tisserand sequencer finds some of the sequences seen in literature, specifically the sequences of the Cassini and Decadal Survey missions. TSSM's sequence is not found because it makes use of a ΔV-EGA manoeuvre that is not compatible with the Tisserand graph unless VILT manoeuvres are added. The V_∞ required to enter this sequence is low because TSSM departed the Earth under its own low-thrust propulsion. The reader can "connect the dots" in figure A.1 to verify

Table 7.1: Sequences found in the literature and using the Tisserand sequencer. Note that the minimum launch V_∞ may not be accurate due to the grid size used with the Tisserand Sequencer (V_∞ can be smaller than stated).

Sequence	Min. Launch V_∞ [km/s]	Source
EVEVES	3.0	Sequencer, Decadal Survey [21]
EVEJS	3.0	Sequencer
EVVEJS	3.0	Sequencer, Cassini [96]
EVEEJS	3.0	Sequencer
EVES	4.0	Sequencer
EVVEES	4.0	Sequencer
EVEES	4.0	Sequencer
EVVES	4.0	Sequencer
EJS	9.0	Sequencer
EEVEES	0.75	TSSM (Low-Thrust) [22]

that each sequence can indeed be traced in the Tisserand Graph. Mars was not considered to be a part of the flight sequence of this mission because its lower gravity attraction is not as beneficial to perform gravity assists as Earth or Venus. Not considering Mars may rule out an opportunistic flyby but, on the other hand, does not increase the search space.

7.1.2 Interplanetary Trajectories

The sequences obtained in the previous section were globally optimised according to the methods laid out in Chapter 5. The results obtained were compiled and the best from each sequence were selected and are shown in table 7.2. The table is sorted by total estimated ΔV. The Saturn orbit insertion values shown are merely an estimate for insertion into an elliptic orbit with $r_p = 1.7R_s$ and $e = 0.99$. This estimate does not account for the full characteristics of the b-plane approach.

As table 7.2 shows, some sequences perform clearly better than others. Sequences using Jupiter as a gravity-assist body underperformed, leading to the conclusion that the giant planet is not in the proper alignment to be used for flybys, during the selected launch window. On the other hand, EVEES and EVVES sequences consistently performed better than other sequence candidates and established their superiority for the defined launch window. In general, the optimiser tends towards the end of the launch window, indicating that a late 2020's launch may offer a better planetary alignment and, therefore, a higher number of trajectory options to choose from. Especially interesting trajectories are trajectory 1, with the lowest overall ΔV and interplanetary ΔV, trajectory 3, with the earliest arrival, trajectory 14, with the fastest interplanetary cruise and trajectory 16, with the lowest arrival V_∞ (due to taking advantage of a Jupiter gravity assist).

Due to the stochastic nature of the algorithms employed, it is not possible to guarantee that the solutions presented are the globally optimum solutions. Moreover, a solution that presents the lowest ΔV cost may not be a suitable trajectory due to the Saturn insertion characteristics. In fact, this is what happens to some of the solutions presented in table 7.2. Trajectory 1 (T1) presents the lowest cost but the asymptote declination for arrival at Saturn is 15º (in the IAU_SATURN reference frame) when compared to the 4º declination at arrival for trajectory 2 (T2). When T1 is patched through to the Saturn

Table 7.2: Interplanetary phase results. The best results for each sequence are shown.

ID	Sequence	Launch V_∞ [m/s]	Interplanetary ΔV [m/s]	Arrival V_∞ [m/s]	Time of Flight [Years]	Launch Date	Arrival Date	Est. ΔV_{SOI} [m/s]	Total ΔV [m/s]
1	EVEES	3.95	136	5826.95	9.98	19-11-2024	15-11-2034	676.68	812.68
2	EVEES	3.99	455.13	5761.4	9.97	12-12-2022	03-12-2032	663.24	1118.30
3	EVEES	3.99	462.49	5760.9	9.56	14-12-2022	30-11-2032	663.12	1125.6
4	EVEES	3.49	769.45	5237.3	9.8	19-04-2028	13-02-2038	560.97	1330.42
5	EVVES	3.5	652.64	5943.3	9.54	16-03-2026	02-10-2035	700.91	1353.55
6	EVVES	3.49	893.70	5908.35	9.35	15-10-2026	23-02-2036	693.58	1587.28
7	EVVES	3.49	894.80	5964.05	8.96	10-10-2026	28-09-2035	705.28	1600.08
8	EVEVES	3.49	1131.30	6120.65	9.98	16-04-2028	11-04-2038	738.72	1870.02
9	EVEJS	6.55	655.78	8683.2	9	01-08-2029	01-08-2038	1400.94	2056.72
10	EVVEJS	3	1316.8	7953.6	9.92	09-11-2029	13-10-2039	1190.71	2507.51
11	EVEJS	4.55	2043.2	6626.8	8.9	23-08-2029	16-07-2038	852.44	2895.64
12	EVES	3.5	2349.8	5741.3	9.22	28-02-2029	21-05-2038	659.14	3008.94
13	EEVES	3.85	2462.4	5790	8.39	09-06-2028	31-10-2036	669.08	3131.48
14	EEVES	3.48	2620.67	5861.15	8.05	23-09-2025	13-10-2033	683.75	3304.42
15	EVES	7	3067	5956.35	8.97	21-06-2024	11-06-2033	703.65	3770.65
16	EJS	5	3963.2	3508.2	10	01-01-2030	01-01-2040	290.92	4254.12
17	EVEEVS	3.5	3809.52	6461	10	15-12-2029	16-12-2039	814.24	4623.76
18	EVEEJS	3.96	3984.84	6546.7	10	20-06-2023	19-06-2033	833.87	4818.71
19	EEVEES	2.36	4403.66	5715.63	9.17	01-09-2027	05-11-2036	653.93	5057.59
20	EJS	3.99	4899	3509	10	31-12-2029	01-01-2040	291.03	5190.03
21	EVEEJS	7	4210.47	7760.3	9	16-11-2029	16-11-2038	1137.86	5348.33

Orbit Insertion phase, it is verified that the cost of the PRM is higher than that of T2. Moreover, the post-insertion orbit has an inclination that is higher than that of T2, resulting in a longer moon tour due to the need to perform plane-changes at Titan to reduce the inclination of the orbit to the orbital plane of Titan. Finally, due to the high arrival declination of T1, the spacecraft's Titan arrival V_∞ (around 4 km/s) is also higher than that of trajectory 2 (around 3 km/s), resulting in a costlier moon tour.

Relatively to T1, T2 has the disadvantage of having a higher interplanetary ΔV and time of flight. Additionally, T2 is launched, and arrives, two years earlier than T1 which has an impact in mission operations and planning that needs to be considered but is out of the scope of this work. Since the advantages in the final phases outweigh the disadvantages of a higher cost in phase one, trajectory 2 was considered as option **A** for the mission to Enceladus and will be used to obtain the results for the next sections. A detailed description of T2 is available in Appendix B, table B.1.

7.2 Phase II - Saturn Orbit Insertion

This section describes the results of the analysis of the Saturn orbit insertion geometry. The orbit insertion calculator receives the arrival epoch and arrival V_∞ vector with respect to Saturn as inputs from phase 1. The user-defined parameters, r_p and e, are set to 1.705 Saturn Radii and 0.99, respectively, to allow the spacecraft to safely cross the ring plane in the F-G gap. Figure 7.1 and table 7.3 contain the main results for the orbit insertion phase. A close-up of figure 7.1 can be seen in Appendix B along with a detailed description of the trajectory.

Upon arrival at Saturn, the spacecraft descends through the ring plane and safely crosses it at around 2.5 Saturn Radii, well within the F-G gap, halfway between the F ring and the Janus ring, with about 4000 km of clear space to each side. The capture manoeuvre occurs after ring crossing, at 1.705 Saturn Radii and places the probe into a highly elliptical orbit with a period of 370 days. The PRM takes place near the apoapsis of the post-capture ellipse and brings the spacecraft to Titan after 230 days, bringing the duration of this phase to a lengthy total of 383 days. This duration can be reduced at the expense of extra ΔV, by reducing the period of the post-capture orbit, by decreasing the allowable post-PRM time of flight or by relaxing the maximum inclination or maximum arrival V_∞ constraints (see Section 5.3.2).

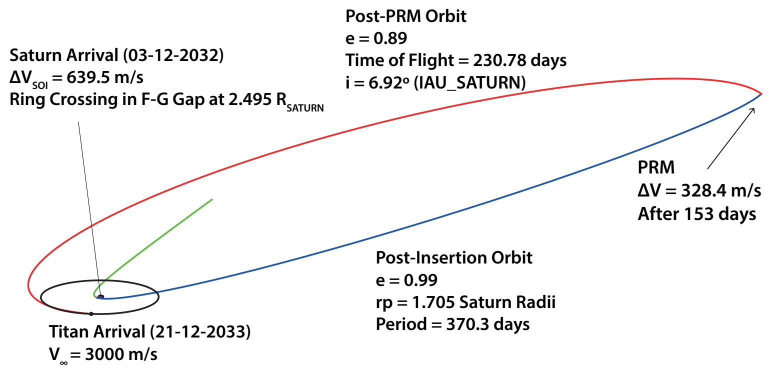


Figure 7.1: Saturn orbit insertion results

Table 7.3: Saturn Orbit Insertion results

Saturn Orbit Insertion	
Total ΔV	967.86 m/s
Time of Flight	383.62 days
Closest Approach	1.705 R_{Saturn}
Ring Crossing	2.495 R_{Saturn}
Titan Arrival V_∞	3 km/s
Titan Arrival Date	21-12-2033
Titan Arrival Inclination	6.92°

7.3 Phase III - Moon Tour

The moon tour starts when the spacecraft reaches Titan, after the Periapsis Raising Manoeuvre. Since the incoming orbit's inclination does not match Titan's orbital plane, it is necessary to perform an inclination "clean-up" flyby before starting the VILT search. Afterwards, the R_a and R_p of the post-flyby orbit are used as starting coordinates for the Titan phase of the design of the moon tour. The properties of this flyby can be consulted in table 7.4. Resulting trajectory images and Tisserand plots of the moon tour can be consulted in Appendix A.

Table 7.4: Inclination change flyby (T0)

Plane-Change Flyby (T0)	Units	
Post-flyby orbital period	days	63.78 ($4T_{Titan}$)
Flyby Height	km	1059.8
Plane-change angle (β)	deg	141.89
Post-Flyby Inclination (IAU)	deg	0.316
Tisserand Coordinates (R_a,R_p)	(m,m)	(5152306304,1001817141)

7.3.1 Titan Phase

The Titan phase of the moon tour starts with the coordinates provided by the inclination change flyby. After the VILT search is performed and trajectories are compared and selected, the best result obtained is shown in table 7.5. Two manoeuvres with a cumulative ΔV cost of 54.57 m/s bring the spacecraft trajectory's V_∞ with respect to Titan from 3 km/s to 2.68 km/s (a 320 m/s decrease). Remember that V_{∞_1} and V_{∞_2} are fixed by the search algorithm (see Section 3.7.2). Afterwards, a transfer orbit departs Titan and reaches Rhea with a $V_{\infty_{Rhea}}$ of 2 km/s, from where the next phase can start. As mentioned in Section 6.2.3, the arrival V_∞ at the next moon in the sequence is a user-defined value.

While some results indicated that it would have been possible to perform the Titan phase with resonant flybys only (no fuel consumption), selecting results for Titan depended on the performance of the algorithm for the Rhea phase. At first, Rhea sequences were not being easily found because the transfer orbits from Titan would reach Rhea in a situation where no resonant flybys or cheap VILTs were possible, resulting in very high ΔV impulsive manoeuvres to continue with the tour. To solve this, the decision was made to spend more fuel in Titan VILTs so that the transfer orbit would bring the spacecraft to Rhea in better conditions. Specifically, the apoapsis of the transfer orbit had to be below 2×10^9 metres to continue the tour. This difficulty could be overcome if the Titan arrival V_∞ could be lowered to values closer to 2 km/s because, as figure A.2 shows, Titan's 2 km/s Tisserand curve intersects Rhea's curves at lower R_a values, closer to available resonant orbits and exterior VILTs.

Table 7.5: Titan Phase. Each row refers to an individual VILT transfer or resonant orbit modify the spacecraft's V_∞ (with respect to the moon) between two V_∞ "levels" (as seen in section 3.7). The quantities on the right are the result of calculating that specific phase-fixed VILT. The last row, which is marked as *TRANSFER*, refers to the inter-moon transfer orbit and its associated cost.

ID	Transfer Type N:M(L)	V_{∞_1} [km/s]	V_{∞_2} [km/s]	ΔV [m/s]	TOF [d]	Flyby Height [km]
T1	ext-OO 3:1(0)	3.0	2.68	54.57	54.47	24291.5
T2	OO 1:1	2.68	2.68	0.0	16.54	846.6
T3	Transfer (OO)	2.68	RHEA 2.0	28.10	9.93	9287.3
	Totals			82.55	75.7	

7.3.2 Rhea Phase

The starting coordinates for Rhea's phase are obtained from the Titan-Rhea transfer orbit: $R_a = 1.58 \times 10^9$ m, $R_p = 0.52327 \times 10^9$ m, corresponding to an initial pump angle $\alpha_{2000} = 19.73$°. The results for the Rhea phase of the moon tour are displayed in table 7.6.

This phase is accomplished with 16 flybys that bring the spacecraft's V_∞ with respect to Rhea from 2 km/s to 1.3 km/s by using 404 m/s of ΔV. This phase is the longest in the tour with a duration of 1 year and 1 month. Solutions for the Rhea phase were challenging to obtain due to the problem mentioned in Section 7.3.1. In addition to lowering Titan arrival V_∞, reducing the safe flyby radius can help by increasing the maximum turn angle of Rhea flybys, decreasing the duration of the phase at the expense of more complex ground and spacecraft systems. This was performed in literature [53] but will not be performed here, with the objective of obtaining a slightly more realistic trajectory.

It is also noteworthy to state that some flybys have less than 10 days of interval (R12,R13,R14), which could lead to a harder navigation environment as there would not be much time to determine the spacecraft's post-flyby position and correct errors. This is in line with literature [53, 57], where short consecutive flybys are also used. In the future, a new rule can be added to the branch and bound search that will eliminate flybys that are too close in time.

This phase ends with a multi-revolution transfer to Dione, where the spacecraft reaches the moon with an arrival $V_{\infty_{Dione}}$ of 1 km/s (with respect to Dione).

Figure A.10 (in appendix A) clearly illustrates the effect that the moon tour has in the spacecraft's

Table 7.6: Rhea Phase. Description as in table 7.5.

ID	Transfer Type N:M (L)	V_{∞_1} [km/s]	V_{∞_2} [km/s]	ΔV [m/s]	TOF [d]	Flyby Height [km]
R1	int-OO 5:2(0)	2	1.9	99.74	22.57	209
R2	Int-OO 9:4(1)	1.9	1.8	99.46	40.62	132.26
R3	Int-OI 5:2(1)	1.8	1.7	98.64	26.53	135.23
R4	IO 11:6	1.7	1.7	0	50.31	252.76
R5	ext-OO 7:4(0)	1.7	1.35	31.38	47.66	261.98
R6	int-OO 8:5(0)	1.35	1.3	49.22	36.11	92.47
R7	OO 3:2	1.3	1.3	0	13.54	292.33
R8	OO 7:5	1.3	1.3	0	31.60	340.25
R9	OI 3:2	1.3	1.3	0	16.96	690.87
R10	IO 6:5	1.3	1.3	0	28.52	136.78
R11	OI 6:5	1.3	1.3	0	29.95	1274.20
R12	IO 1:1	1.3	1.3	0	6.58	279.18
R13	OO 1:1	1.3	1.3	0	4.52	397.1
R14	OI 1:1	1.3	1.3	0	6.13	327.04
R15	IO 5:6	1.3	1.3	0	25.90	85.05
R16	Transfer (OO)	1.3	DIONE 1.0	25.95	21.32	1315.5
	Totals			404.4	408.81	

orbit. For each consecutive flyby, the apoapsis of the orbit is becoming smaller and, consequently, so are the semi-major axis and orbital energy.

7.3.3 Dione Phase

The starting coordinates for Dione's phase are obtained from the Rhea-Dione transfer orbit: $R_a = 5.57 \times 10^8$ m, $R_p = 3.756 \times 10^8$ m, corresponding to an initial pump angle $\alpha_{1000} = 23.27°$. The results for the Dione phase of the moon tour are displayed in table 7.7. This phase has a duration of nearly 6 months.

The Dione phase is interesting as it is composed entirely of resonant flybys. Nevertheless, the search algorithm did find solutions that made it possible to shorten time of flight at the expense of ΔV but it was considered that reducing the potential ΔV cost should be a priority for this phase after an expensive tour of Rhea. Once again, flybys D7, D8 and D9 are separated by less than 10 days, something that may be avoided in a future version of the search algorithm.

7.3.4 Tethys Phase

The starting coordinates for Tethys' phase are obtained from the Dione-Tethys transfer orbit: $R_a = 3.853 \times 10^8$ m, $R_p = 2.947 \times 10^8$ m, corresponding to an initial pump angle $\alpha_{800} = 24.5°$. The results for the Tethys phase of the moon tour are displayed in table 7.8. In this phase, flybys Te6, Te7 and Te8 are very close in time. When working in a full perturbation model this could lead to navigation problems, since orbits are never exactly resonant due to perturbations acting on the spacecraft and moon and 2/3 days could pose a challenge for engineers to be able to accurately determine the position of the spacecraft and execute any correction manoeuvres. A possible solution to this is to force the search algorithm to use higher period resonances. Additionally, some flybys pass low above the surface of Tethys. It was considered that the two initial flybys (at 461 km and 376 km) could be used for mapping

Table 7.7: Dione Phase. Description as in table 7.5.

ID	Transfer Type N:M(L)	V_{∞_1} [km/s]	V_{∞_2} [km/s]	ΔV [m/s]	TOF [d]	Flyby Height [km]
D1	OO 4:3	1	1	0	10.90	130.44
D2	OO 9:7	1	1	0	24.53	272.15
D3	OO 5:4	1	1	0	13.63	693.42
D4	OO 6:5	1	1	0	16.35	361.92
D5	OO 7:6	1	1	0	19.08	890.32
D6	OI 9:8	1	1	0	26.35	224.39
D7	IO 1:1	1	1	0	3.922	148.16
D8	OO 1:1	1	1	0	2.73	364.85
D9	OI 1:1	1	1	0	3.75	328.15
D10	II 10:11	1	1	0	27.26	154.17
D11	IO 6:7	1	1	0	18.48	405.39
D12	Transfer (OO)	1	TETHYS 0.8	8.67	10.85	416.62
	Totals			8.67	177.82	

purposes, reducing the risk of performing such low flybys at Tethys.

In this fourth phase of the moon tour, the spacecraft's V_∞ with respect to Tethys is reduced from 0.8 km/s to 0.7 km/s, with a ΔV cost of 22.96 m/s, in 5 months.

Table 7.8: Tethys Phase. Description as in table 7.5.

ID	Transfer Type N:M (L)	V_{∞_1} [km/s]	V_{∞_2} [km/s]	ΔV [m/s]	TOF [d]	Flyby Height [km]
Te1	OO 11:9	0.8	0.8	0	20.78	461.23
Te2	OO 6:5	0.8	0.8	0	11.34	376.58
Te3	ext-OO 7:6(0)	0.8	0.7	17.86	13.18	60.31
Te4	OI 8:7	0.7	0.7	0	16.63	50.98
Te5	II 14:13	0.7	0.7	0	26.45	71.54
Te6	IO 1:1	0.7	0.7	0	2.70	54.79
Te7	OO 1:1	0.7	0.7	0	1.89	519.23
Te8	OI 1:1	0.7	0.7	0	2.62	493.33
Te9	II 14:15	0.7	0.7	0	26.45	77.55
Te10	II 9:10	0.7	0.7	0	17.00	72.28
Te11	II 7:8	0.7	0.7	0	13.22	103.48
Te12	Transfer (IO)	0.7	Enceladus 0.7	5.1	5.58	206.66
	Totals			22.96	157.83	

7.3.5 Enceladus Phase

The initial conditions for the final phase of the moon tour are obtained from the Tethys-Enceladus transfer orbit, which has $R_a = 2.954 \times 10^8$ m and $R_p = 2.386 \times 10^8$ m, resulting in an initial pump angle $\alpha_{700} = 22.54^o$ at Enceladus.

Solutions for the Enceladus phase of the moon tour were found by fixing $L = M - 1$ (the most efficient value [57]) to reduce the large search space created by the high number of allowed resonances. The endgame at Enceladus was performed with the objective of lowering the spacecraft's V_∞ with respect to Enceladus as much as possible (by gradually lowering the apoapsis of the Saturn-centred orbit to that of Enceladus). This phase has a duration of 5.5 months and lowers the spacecraft's V_∞ from 0.7 km/s to 0.35 km/s with a cost of 98.87 m/s (see table 7.9).

If a direct orbit insertion at the beginning of the endgame were to be performed, the capture manoeuvre's ΔV cost would approximately be 578 m/s. By lowering V_∞ to 0.35 km/s, that cost is reduced by approximately 45%, to around 257 m/s, by spending 99.27 m/s. This results in a net ΔV saving of about 220 m/s. Unlike the literature [53, 57], the minimum allowed flyby radius at Enceladus is always above 50 km (as opposed to 25 km [53]), reducing the complexity of the navigation system required to perform the mission.

The complete moon tour ends with the orbit insertion at Enceladus, after 53 flybys across 1054 days and with a total ΔV_{tour} cost of 617 m/s.

Table 7.9: Enceladus Phase. Each row refers to an individual VILT transfer or resonant orbit modify the spacecraft's V_∞ (with respect to the moon) between two V_∞ "levels" (as seen in section 3.7). The quantities on the right are the result of calculating that specific phase-fixed VILT.

ID	Transfer Type N:M(L)	V_{∞_1} [km/s]	V_{∞_2} [km/s]	ΔV [m/s]	TOF [d]	Flyby Height [km]
E1	int-OO 20:17(16)	0.70	0.65	49.80	27.56	262.84
E2	OO 15:13	0.65	0.65	0.00	20.67	170.38
E3	ext-OO 8:7(6)	0.65	0.60	7.83	11.00	61.64
E4	ext-OO 17:15(14)	0.60	0.55	7.69	23.40	55.60
E5	OI 8:7	0.55	0.55	0.00	12.26	69.00
E6	IO 10:9	0.55	0.50	8.17	13.94	69.80
E7	ext-OO 21:19(18)	0.50	0.50	0.00	28.94	275.98
E8	OO 11:10	0.50	0.50	0.00	15.16	688.45
E9	ext-OO 13:12(11)	0.50	0.35	25.38	17.81	51.22
	Totals			98.87	170.745	

7.4 Enceladus Orbital Insertion

When the Enceladus endgame is over, an Enceladus Orbit Insertion manoeuvre is performed to insert the probe into the orbit described in section 6.3. This procedure mirrors the one performed for the Saturn Orbit Insertion, but with a different central body and without the need to perform a PRM. The capture geometry can be seen in figure 7.2 and the main results of this phase can be seen in table 7.10.

A detailed description of the Enceladus Orbit Insertion can be seen in Appendix 2.

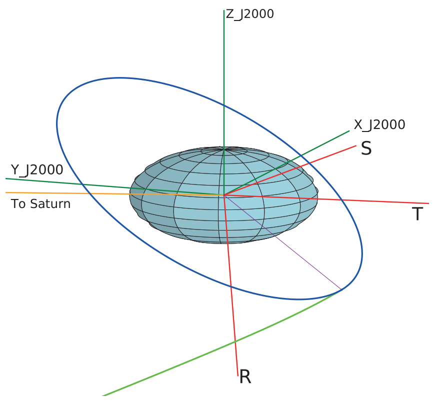


Figure 7.2: Enceladus Orbit Insertion

Table 7.10: Enceladus Orbit Insertion

Enceladus Orbit Insertion	
ΔV_{EOI}	243.5 m/s
Arrival V_∞	350 m/s
Arrival Date	25-Oct-2036
B-Plane Information	
θ_B	58.72 deg
B.T	313.048 km
B.R	515.381 km

There is a considerable reduction in ΔV_{EOI} of around 3.3 km/s, when compared to the cost of capture after a direct Hohmann transfer from Titan (3.55 km/s for a 295 km circular orbit).

7.5 ΔV and Mass Budgets

Table 7.11 details the estimated budgets of all phases of the mission. The following margin philosophies were used:

1. Tour Statistical ΔV: additional 5 m/s per flyby in the moon tour [53].
2. 5% per deterministic manoeuvre (DSMs, insertion manoeuvres) [68]. This margin was not applied to the moon tour's ΔV's.

The mass budget is estimated by taking into account launcher performance. The maximum mass an Atlas 551 rocket can place on an interplanetary trajectory with $C_3 = 16\ km^2/s^2$ is around 5000 kg [68] (see Section 3.4). By assuming a spacecraft dry mass of two tonnes and using Tsiolkovsky's Rocket Equation (2.2), applied to the ΔV budget (after margins), the final wet mass of the spacecraft would be around 4550 kg, leaving a launch mass margin of approximately 10% to account for changes in the mission that may require carrying more weight. The I_{sp} assumed was 326 seconds, based on TSSM's 890N HiPAT bi-propellant high performance liquid apogee thruster [22, 97]. Naturally, if a smaller spacecraft is desired, cheaper and smaller launchers can be used.

Table 7.11: ΔV and mass budgets.

Mission to Enceladus	**ΔV Budget [m/s]**	**ΔV Budget (with margins) [m/s]**	**Propellant Mass [kg]**	**Time of Flight [years]**
Interplanetary Trajectory	455.14	477.90	632.40	9.97
Saturn Orbit Insertion	639.47	671.44	742.92	1.05
Periapsis Raising Manoeuvre	328.39	344.81	324.97	
Titan Phase	82.55	97.55	85.75	0.38
Rhea Phase	404.4	484.40	389.10	1.12
Dione Phase	8.67	68.67	50.54	0.48
Tethys Phase	22.96	82.96	59.62	0.43
Enceladus Phase	98.87	143.87	99.80	0.46
Enceladus Orbit Insertion	243.5	255.68	166.64	—
Total	2283.95	2627.28	2551.73	13.9

Orbit maintenance, TCMs and disposal manoeuvres are out of the scope of this work and were not added to the ΔV budget.

7.6 Discussion

Table 7.12 displays a comparison of the results obtained in this work with those of other missions and mission concepts [21, 22, 53, 57**?**]. The interplanetary trajectory obtained for the Mission to Enceladus is in line with the other state-of-art trajectories, an important factor that supports the quality of the solution obtained. The island model implemented in PyGMO contributed to achieve this result thanks to the ease in creating topologies that allowed the optimisation algorithms to cooperate. While inferior in terms of interplanetary ΔV, this mission's cruise trajectory reaches Saturn in better conditions (lower V_∞) than

Table 7.12: Comparison of Solutions. ΔV results are shown without margins applied.

	Mission to Enceladus	TSSM [22]	Decadal [21]	Cassini [14, 15**?**]	[53]	[57]
Launch Date	12 Dec. 2022	Sep. 2020	Jan. 2023	15 Oct. 1997	—	—
Flight Sequence	EVEES	EEVEES	EVEVES	EVVEJS	—	—
ΔV_{IP} [m/s]	455.13	2750	350	466	—	—
ΔV_{SOI} [m/s]	639.47	746	960	613	1310 (TSSM)	1310 (TSSM)
ΔV_{PRM} [m/s]	328.4	564	534	333		
ΔV_{Tour} [m/s]	617.45	—	527.3	—	492	316
ΔV_{EOI} [m/s]	243.5	—		—	242	129
Total ΔV	2284	—	2371	—	—	—
Dry Mass [kg]	2000 (Atlas V)	3127	1154	2445	—	—
Launch Mass [kg]	4551 (Atlas V)	6203 (Atlas V)	3560	5712	—	—
Moon Tour Time [y]	2.89	—	2.7	—	2	2.7
Cruise Time [y]	9.97	9	8.5	7	—	—

the alternatives presented in table 7.12, resulting in a considerable reduction in ΔV_{SOI}. Despite the advantage in cost, the trajectory obtained is inferior in terms of time of flight (9.97 years), but still within the bounds set for the mission. This could be improved by taking advantage of a higher number of flybys, possibly at the expense of higher ΔV. Since the focus of the work is to minimise ΔV, it was decided that a lower interplanetary ΔV would be preferable to a lower time of flight, as long as the time of flight was within the bounds set for the mission (6 years to 10 years of interplanetary cruise time).

Additionally, the combined higher dry mass of TSSM's Titan orbiter and its in-situ packages of 2446 kg [22] (with margins) reveals that there may be some interest in exploring the use of Solar electric propulsion (SEP) for the interplanetary cruise with the objective of increasing the mass of instruments that can be delivered to Saturn and its moons. This has to be carefully analysed as the inclusion of a separate SEP module may increase the mass of the spacecraft to a point where the fuel required to support the addition of the new propulsion module nullifies its benefits.

The Saturn orbit insertion manoeuvre occurs successfully as the spacecraft descends through the ring plane safely, crossing in the F-G gap, and is then inserted into the post-capture elliptic orbit. The cost of the PRM in this mission is lower than in the missions present in table 7.12 due to the high period of the post-capture orbit, result of a large major axis. As a consequence of this high period capture orbit, the time of flight from capture until the first Titan flyby is high (383 days). This time can be lowered by reducing the period of the post-capture orbit, resulting in a higher ΔV cost associated with capture and the PRM.

The moon tour has been demonstrated to be an essential technique to perform when the objective is to place an orbiter around a low mass moon, such as Enceladus. The V_∞-leveraging technique implemented in this work achieved a net saving of about 2.69 km/s, when compared to a direct Titan-Enceladus Hohmann transfer.

When compared to other tours from literature [21, 53, 57], the moon tour that was designed for the Mission to Enceladus resulted in a higher cost and time of flight. This happens because:

1. The Titan arrival V_∞ is higher than those in literature (3 km/s when compared to 1.46 km/s) [21, 53, 57]. This requires more, or more expensive, VILTs to achieve the necessary reduction in the orbit's major axis for the Titan and Rhea phases of the tour, increasing their cost and time of flight.

2. The addition of the inter-moon transfer orbits, taking the required phasing into account, did not have a negligible effect in the ΔV budget as the literature suggests [53, 57], with these transfer costs summing up to 68 m/s — around 10% of the total cost of the tour. However, the time of flight added by these transfers did not have a large impact in the total time of the moon tour. More importantly, selecting an appropriate transfer orbit was revealed to be an important factor in the quality of the solution. Different transfer orbits allow the spacecraft to access different resonances at arrival to a new moon and, if less expensive VILTs are not available after transfer, this can result in an expensive phase start, such as what happened in this work regarding the Titan-Rhea transfer.
3. A higher number of flybys is required to compensate for the fact that the minimum flyby radii used in this work are more conservative than in literature.
4. It was not possible to exhaustively obtain all the solutions for some phases of the moon tour due to high computational times, a downside of a Python implementation as this potentially eliminated optimal solutions. This limitation could be countered by implementing the branch and bound routine in a compiled language, such as C++, and exposing it to Python via a wrapper. The performance of the algorithms was not evaluated but the Rhea phase was left to run, unconstrained, for four days without having finished (intel core i7). While this may not be a significant computational time in the field of trajectory design, it was significant for the time that was left to perform this work.

It should be noted that the final Enceladus V_∞ could be lowered below 350 m/s through more VILTs, resulting in a cheaper Enceladus Orbit Insertion. While this was done in Campagnola et al. [57] and for the Decadal Survey, it was not explored in this work due to the already lengthy duration of the moon tour (2.9 years). This was a design decision and not a limitation imposed by the models or implementation.

The mission ends with a successful orbital capture at Enceladus, directly into the science orbit presented in Section 6.3. Associating the trajectory designed here with the use of an Atlas V 551 launcher for departure, it is possible to place a spacecraft with around two tonnes of mass into Enceladus orbit. This is close to the dry mass of the Cassini spacecraft (2485 kg) and an increase over the dry mass of the Decadal Survey's orbiter (1154 kg) but the latter would be launched in a smaller launcher, the Atlas 521. The use of solar electric propulsion enables the TSSM spacecraft to have a higher dry mass, as the required fuel mass is lower. TSSM's orbiter and two landers would also be launched atop an Atlas V 551 rocket and have a combined dry mass of 2446 kg (after margins).

In an overall analysis, the results of the mission to Enceladus are comparable in quality to other state-of-art trajectories in literature [21, 22**?**], in terms of ΔV cost and mass placed in orbit. The design process resulted in a lengthy mission, nearly 14 years. In comparison, Cassini's mission will end in 2017, 19 years after being launched from Earth [14, 15]. The mission's duration is close to the recommended time of use of RTGs (14 years) but new technologies, such as ASRGs, that have a higher recommended time (17 years) [21, 94], could power the mission to Enceladus. Finally, the time spent in orbit of Enceladus can be complemented with valuable scientific data that may be gathered from close flybys of other moons in the Saturn system during the 3 years of the moon tour, resulting in a science-rich exploration mission to this dynamic and fascinating moon system.

Chapter 8

Conclusions

The feasibility of an orbiter mission to Enceladus was successfully demonstrated through a preliminary spacecraft trajectory study that includes all phases of the mission: interplanetary, capture and moon tour to reach the final destination. The interplanetary trajectory was obtained through an optimisation procedure that relied in a cooperative topology of stochastic algorithms and a method based on Tisserand graphs to find potential flight sequences. To obtain a moon tour trajectory, a complex V_∞-Leveraging technique, based on Tisserand graphs, was studied and implemented. This resulted in a moon tour solution that accomplishes a significant reduction in the cost of Enceladus Orbit Insertion of 2.69 km/s, when compared to a traditional Titan-Enceladus Hohmann transfer. This shows that VILTs are a powerful technique for designing trajectories in low-gravity moon systems. Furthermore, in a step beyond current literature, the effect of taking into account the need for phasing in inter-moon transfer orbits was studied and found to be potentially non-negligible as it accounts for around 10% of the total ΔV budget of the designed moon tour. Finally, the primary objective of placing an orbiter around Enceladus was successfully achieved since the example trajectory in this work is capable of placing a payload up to two tonnes (with current launchers) in orbit of this icy moon. This mass is similar to that of other large interplanetary spacecraft or spacecraft concepts, such as Cassini-Huygens and TSSM.

8.1 Future Work

The branch and bound and other time critical algorithms should be implemented in a compiled language, such as C++, to minimise some of the computational limitations encountered during this work and to permit the designer to use the branch and bound technique to its full potential to find better moon tour trajectories. Furthermore, after results comparison, it was clear that there may be some advantage in using a hybrid approach to the mission, with solar electric propulsion being used for the interplanetary phase. This could lead to a higher maximum mass at Enceladus and bring other benefits associated with the use of low-thrust propulsion. Additionally, it would be interesting to remove the 2D approach to the VILT technique and investigate how orbital cranking (changing orbital plane) using flybys could improve the quality of the solution [98]. Issues such as communications, navigation and scientific observations also need to be analysed to obtain a realistic trajectory. Finally, a fully integrated trajectory in a multiple body environment, using the solution obtained as an initial guess, should be determined.

References

[1] W. E. Wiesel. *Spaceflight dynamics*. Boston, Mass. : Irwin/McGraw-Hill, 2nd ed edition, 1997. ISBN 0071156313 (International ed. : pbk.).

[2] B. Conway. The Problem of Spacecraft Trajectory Optimization. In B. Conway, editor, *Spacecraft Trajectory Optimization*, pages 1–36. Cambridge University Press, 2014.

[3] J. S. Kargel, J. Z. Kaye, J. W. H. III, G. M. Marion, R. Sassen, J. K. Crowley, O. P. Ballesteros, S. A. Grant, and D. L. Hogenboom. Europa's crust and ocean: Origin, composition, and the prospects for life. *Icarus*, 148(1):226 – 265, 2000. ISSN 0019-1035. doi: http://dx.doi.org/10.1006/icar.2000.6471. URL `http://www.sciencedirect.com/science/article/pii/S0019103500964716`.

[4] J. R. Spencer and F. Nimmo. Enceladus: An Active Ice World in the Saturn System. *Annual Review of Earth and Planetary Sciences*, 41(1):693–717, 2013. ISSN 0084-6597. doi: 10.1146/annurev-earth-050212-124025. URL `http://adsabs.harvard.edu/abs/2013AREPS..41..693S`.

[5] T. B. McCord and C. Sotin. Ceres: Evolution and current state. *Journal of Geophysical Research*, 110:1–14, 2005.

[6] M. D. Rayman, T. C. Fraschetti, C. A. Raymond, and C. T. Russell. Dawn: A mission in development for exploration of main belt asteroids Vesta and Ceres. *Acta Astronautica*, 58(11):605–616, 2006. ISSN 00945765. doi: 10.1016/j.actaastro.2006.01.014.

[7] C. Zimmer, K. K. Khurana, and M. G. Kivelson. Subsurface Oceans on Europa and Callisto: Constraints from Galileo Magnetometer Observations. *Icarus*, 147(2):329–347, 2000. ISSN 00191035. doi: 10.1006/icar.2000.6456.

[8] J. Saur, S. Duling, L. Roth, X. Jia, D. F. Strobel, P. D. Feldman, U. R. Christensen, K. D. Retherford, M. A. McGrath, F. Musacchio, et al. The search for a subsurface ocean in Ganymede with Hubble Space Telescope observations of its auroral ovals. *Journal of Geophysical Research: Space Physics*, 120(3):1715–1737, 2015.

[9] NASA's Jet Propulsion Laboratory. Ocean worlds: An infographic, . URL `http://www.jpl.nasa.gov/infographics/infographic.view.php?id=11262`. Retrieved on 18 April 2016.

[10] NASA. Hubble Sees Evidence of Water Vapor at Jupiter Moon., . URL `http://www.jpl.nasa.gov/news/news.php?release=2013-363`. Retrieved on 18th April 2016.

[11] Jet Propulsion Laboratory. Europa Study 2012 Report. *National Aeronautics and Space Administration*, (D-71990), 2012.

[12] A. Boutonnet and J. Schoenmaekers. JUICE : Consolidated Report on Mission Analysis (CReMA). (WP-578), 2014.

[13] NASA's Jet Propulsion Laboratory. About Saturn and Its moons: Titan. `http://saturn.jpl.nasa.gov/science/index.cfm?SciencePageID=73`, . Retrieved on April 2016.

[14] NASA's Jet Propulsion Laboratory. Cassini Solstice Mission. `http://saturn.jpl.nasa.gov/`, . Retrieved on April 2016.

[15] European Space Agency. Cassini-Huygens Overview. `http://sci.esa.int/cassini-huygens/`. Retrieved on April 2016.

[16] A. I. Razzaghi, D. A. Di Pietro, D. A. Quinn, A. A. Simon-Miller, and S. D. Tompkins. Mission concepts for studying Enceladus. *AIP Conference Proceedings*, 969(301):388–395, 2008. ISSN 0094243X. doi: 10.1063/1.2844992.

[17] S. Matousek, C. Sotin, D. Goebel, and J. Lang. JET: Journey to Enceladus and Titan. California Institute of Technology, Low Cost Planetary Missions Conference, June 2013.

[18] K. Lunine, J.I.; Waite, J.H.; Postberg, F.; Spilker, L.; Clark. Enceladus Life Finder: The Search For Life In A Habitable Moon. *46th Lunar and Planetary Science Conference*, pages 2–3, 2015.

[19] P. Tsou, D. E. Brownlee, C. P. McKay, A. D. Anbar, H. Yano, K. Altwegg, L. W. Beegle, R. Dissly, N. J. Strange, and I. Kanik. LIFE: Life Investigation For Enceladus A Sample Return Mission Concept in Search for Evidence of Life. *Astrobiology*, 12(8):730–742, 2012. ISSN 1531-1074. doi: 10.1089/ast.2011.0813.

[20] K. Konstantinidis, C. L. F. Martinez, B. Dachwald, A. Ohndorf, P. Dykta, P. Bowitz, M. Rudolph, I. Digel, J. Kowalski, K. Voigt, and R. Förstner. A lander mission to probe subglacial water on Saturn's moon Enceladus for life. *Acta Astronautica*, 106:63–89, 2015. ISSN 00945765. doi: 10.1016/j.actaastro.2014.09.012. URL `http://dx.doi.org/10.1016/j.actaastro.2014.09.012`.

[21] R. Kinsey. Mission Concept Study: Planetary Science Decadal Survey (Enceladus Orbiter). Technical report, NASA, 2010.

[22] K. Reh, T. Magner, D. Matson, A. Coustenis, J. Lunine, J. Lebreton, C. Jones, and J. Sommerer. Titan Saturn System Mission Study Final Report. (NMO710851), 2009. NASA/ESA.

[23] A. Coustenis. TandEM: Titan and Enceladus mission. *Experimental Astronomy*, 23(3):893–946, 2008. ISSN 1572-9508. doi: 10.1007/s10686-008-9103-z. URL `http://dx.doi.org/10.1007/s10686-008-9103-z`.

[24] Johns Hopkins Applied Physics Laboratory. New Horizons. `http://pluto.jhuapl.edu/`. Retrieved on April 2016.

[25] NASA. Voyager: Triton Overview. http://voyager.jpl.nasa.gov/science/neptune_triton.html, . Retrieved on April 2016.

[26] European Space Research and Technology Centre. GTOC: Global Trajectory Optimisation Competition. http://sophia.estec.esa.int/gtoc_portal/. Retrieved on April 2016.

[27] J. T. Betts. Survey of Numerical Methods for Trajectory Optimization. *Journal of Guidance, Control, and Dynamics*, 21(2):193–207, 1998. ISSN 0731-5090. doi: 10.2514/2.4231.

[28] D. Myatt, V. Becerra, S. Nasuto, J. Bishop, and D. Izzo. Advanced Global Optimisation for Mission Analysis and Design. Technical report, European Space Agency and The University of Reading, 2004.

[29] J. Sims and S. Flanagan. Preliminary Design of Low-Thrust Interplanetary Missions. *AAS/AIAA Astrodynamics Specialist Conference*, AAS 99-338, 1999.

[30] C. H. Yam, D. Izzo, and F. Biscani. Towards a High Fidelity Direct Transcription Method for Optimisation of Low-Thrust Trajectories. *4th International Conference on Astrodynamics Tools and Techniques*, pages 1–7, 2010. URL http://arxiv.org/abs/1004.4539.

[31] L. Casalino, G. Colasurdo, and M. Rosa Sentinella. Indirect Optimization Method for Low-Thrust Interplanetary Trajectories. In *30th International Electric Propulsion Conference (September 17-20, 2007, Florence, Italy)*, 2007. URL http://erps.spacegrant.org/uploads/images/images/iepc_articledownload_1988-2007/2007index/IEPC-2007-356.pdf. IEPC-2007-356.

[32] F. Jiang, Y. Chen, Y. Liu, H. Baoyin, and J. Li. GTOC5: Results from the Tsinghua University. *Acta Futura*, 8:37–44, 2014. ISSN 2309-1940. doi: 10.2420/AF08.2014.37.

[33] A. E. Petropoulos and J. M. Longuski. Shape-based algorithm for the automated design of low-thrust, gravity assist trajectories. *Journal of Spacecraft and Rockets*, 41(5):787–796, 2004.

[34] J. Seabra. Spacecraft Trajectory Optimization. Master's thesis, Instituto Superior Técnico, 2015.

[35] X. Yao. Global optimisation by evolutionary algorithms. In *Parallel Algorithms/Architecture Synthesis, 1997. Proceedings., Second Aizu International Symposium*, pages 282–291. IEEE, 1997.

[36] M. Rosa Sentinella. *Development of new procedures and hybrid algorithms for space trajectories optimisation*. PhD thesis, Politecnico di Torino, Turin, Italy, 2008.

[37] D. Izzo. Global Optimization and Space Pruning for Spacecraft Trajectory Design. In B. Conway, editor, *Spacecraft Trajectory Optimization*, pages 178–201. Cambridge University Press, 2014.

[38] S. Koziel and Z. Michalewicz. Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. *Evolutionary computation*, 7(1):19–44, 1999.

[39] D. Izzo, V. Becerra, D. Myatt, S. Nasuto, and J. Bishop. Search space pruning and global optimisation of multiple gravity assist spacecraft trajectories. *Journal of Global Optimisation*, (38):283–296, 2007.

[40] D. Izzo, L. F. Simões, M. Märtens, G. C. H. E. de Croon, A. Heritier, and C. H. Yam. Search for a Grand Tour of the Jupiter Galilean Moons. *Genetic and Evolutionary Computation Conference (GECCO 2013)*, pages 1301–1308, 2013. doi: 10.1145/2463372.2463524. URL `http://sophia.estec.esa.int/gtoc{_}portal/wp-content/uploads/2012/11/gtoc6{_}316.pdf$\delimiter"026E30F$nhttp://dl.acm.org/citation.cfm?doid=2463372.2463524$\delimiter"026E30F$nhttp://www.genetic-programming.org/hc2013/Izzo-Paper.pdf`.

[41] G. Colasurdo, A. Zavoli, A. Longo, L. Casalino, and F. Simeoni. Tour of Jupiter Galilean moons: Winning solution of GTOC6. *Acta Astronautica*, 102:190–199, 2014. ISSN 00945765. doi: 10.1016/j.actaastro.2014.06.003. URL `http://dx.doi.org/10.1016/j.actaastro.2014.06.003`.

[42] M. R. Sentinella and L. Casalino. Cooperative evolutionary algorithm for space trajectory optimization. *Celestial Mechanics and Dynamical Astronomy*, 105(1):211–227, 2009. ISSN 09232958. doi: 10.1007/s10569-009-9223-4.

[43] M. Vasile and P. D. Pascale. Preliminary design of multiple gravity-assist trajectories. *Journal of Spacecraft and Rockets*, 43(4):794–805, 2006.

[44] B. A. Conway, C. M. Chilan, and B. J. Wall. Evolutionary principles applied to mission planning problems. *Celestial Mechanics and Dynamical Astronomy*, 97(2):73–86, 2006. ISSN 09232958. doi: 10.1007/s10569-006-9052-7.

[45] B. J. Wall and B. A. Conway. Genetic algorithms applied to the solution of hybrid optimal control problems in astrodynamics. *Journal of Global Optimization*, 44(4):493–508, 2008. ISSN 09255001. doi: 10.1007/s10898-008-9352-4.

[46] M. Vasile, E. Minisci, and M. Locatelli. On testing global optimization algorithms for space trajectory design. In *AIAA/AAS Astrodynamics Specialist Conference and Exhibit, Honolulu, Hawaii*, 2008.

[47] M. Vasile, E. Minisci, and M. Locatelli. An inflationary differential evolution algorithm for space trajectory optimization. *IEEE Transactions on Evolutionary Computation*, 15(2):267–281, April 2011. ISSN 1089-778X. doi: 10.1109/TEVC.2010.2087026.

[48] B. Addis, A. Cassioli, M. Locatelli, and F. Schoen. A global optimization method for the design of space trajectories. *Computational Optimization and Applications*, 48(3):635–652, 2009. ISSN 1573-2894. doi: 10.1007/s10589-009-9261-6. URL `http://dx.doi.org/10.1007/s10589-009-9261-6`.

[49] A. Gad. Space Trajectories Optimization Using Variable-Chromosome-Length Genetic Algorithms. Master's thesis, Michigan Technological University, 2011.

[50] G. A. Rauwolf and V. L. Coverstone-Carroll. Near-optimal low-thrust orbit transfers generated by a genetic algorithm. *Journal of Spacecraft and Rockets*, 33(6):859–862, 1996.

[51] B. Dachwald. Optimization of interplanetary solar sailcraft trajectories using evolutionary neurocontrol. *Journal of Guidance, Control, and Dynamics*, 27(1):66–72, 2004.

[52] B. J. Wall and B. A. Conway. Shape-Based Approach to Low-Thrust Rendezvous Trajectory Design. *Journal of Guidance, Control, and Dynamics*, 32(1):95–101, 2009. ISSN 0731-5090. doi: 10.2514/1.36848.

[53] N. J. Strange, S. Campagnola, and R. P. Russell. Leveraging flybys of low mass moons to enable an Enceladus orbiter. *Advances in the Astronautical Sciences*, 135:2207–2225, 2010. ISSN 00653438.

[54] NASA Space Science Data Coordinated Archive. Planetary Fact Sheets. `http://nssdc.gsfc.nasa.gov/planetary/planetfact.html`. Retrieved on July 2016.

[55] SETI Institute. Vital Statistics for Saturn's Rings and Inner Satellites. `http://pds-rings.seti.org/saturn/saturn_tables.html`. Retrieved on July 2016.

[56] International Astronomical Union Working Group for Planetary System Nomenclature. Ring and Ring Gap Nomenclature. `http://planetarynames.wr.usgs.gov/Page/Rings`. Retrieved on August 2016.

[57] S. Campagnola, N. J. Strange, and R. P. Russell. A fast tour design method using non-tangent V-Infinity Leveraging Transfers. *Advances in the Astronautical Sciences*, 136(2):983–1003, 2010. ISSN 00653438. doi: 10.1007/s10569-010-9295-1.

[58] W. J. Larson and J. R. Wertz. *Space Mission Analysis and Design*. 1999. ISBN 1881883108. URL `http://www.osti.gov/energycitations/product.biblio.jsp?osti{_}id=7369177`.

[59] D. Brown, M. Martinez, and G. Napier. Juno Launch. 2011.

[60] G. H. Fountain, D. Y. Kusnierkiewicz, C. B. Hersman, T. S. Herder, T. B. Coughlin, W. C. Gibson, D. A. Clancy, C. C. Deboy, T. A. Hill, J. D. Kinnison, D. S. Mehoke, G. K. Ottman, G. D. Rogers, S. A. Stern, J. M. Stratton, S. R. Vernon, and S. P. Williams. The new horizons spacecraft. *New Horizons: Reconnaissance of the Pluto-Charon System and the Kuiper Belt*, (July 2015):23–47, 2009. ISSN 00386308. doi: 10.1007/978-0-387-89518-5_3.

[61] NASA. Galileo spacecraft facts, . URL `http://solarsystem.nasa.gov/galileo/facts.cfm`. Retrieved on July 2016.

[62] D. Doody. *Deep Space Craft: An Overview of Interplanetary Flight*. Springer Praxis Books. Springer Berlin Heidelberg, 2010. ISBN 9783540895107. URL `https://books.google.pt/books?id=Gp-2B7AUPVUC`.

[63] S. Kemble. *Interplanetary Mission Analysis and Design*. Springer-Praxis, 2006. doi: 10.1007/3-540-37645-3.

[64] H. D. Curtis. *Orbital Mechanics for Engineering Students*. 2014. ISBN 9780080977478. doi: 10.1016/B978-0-08-097747-8.00007-4. URL `http://www.sciencedirect.com/science/article/pii/B9780080977478000074`.

[65] D. Izzo. Revisiting Lambert's problem. *Celestial Mechanics and Dynamical Astronomy*, 121(1), 2014. ISSN 15729478. doi: 10.1007/s10569-014-9587-y.

[66] R. H. Gooding. A procedure for the solution of Lambert's orbital boundary-value problem. *Celestial Mechanics and Dynamical Astronomy*, 48(2):145–165, 1990. ISSN 09232958. doi: 10.1007/BF00049511.

[67] N. Arora and R. P. Russell. A fast and robust multiple revolution Lambert algorithm using a cosine transformation. In *Advances in the Astronautical Sciences*, volume 150, pages 411–430, 2014. ISBN 9780877036050.

[68] R. Biesbroek. *Lunar and Interplanetary Trajectories*. Springer, 2016. ISBN 978-3-319-26981-8. doi: 10.1007/978-3-319-26983-2. URL `http://link.springer.com/10.1007/978-3-319-26983-2`.

[69] G. R. Hintz. *Orbital Mechanics and Astrodynamics: Techniques and Tools for Space Missions*. Springer, 2015. ISBN 978-3-319-09444-1. doi: 10.1007/978-3-319-09444-1.

[70] B. A. Archinal, M. F. A'Hearn, E. Bowell, A. Conrad, G. J. Consolmagno, R. Courtin, T. Fukushima, D. Hestroffer, J. L. Hilton, G. A. Krasinsky, G. Neumann, J. Oberst, P. K. Seidelmann, P. Stooke, D. J. Tholen, P. C. Thomas, and I. P. Williams. Report of the IAU Working Group on Cartographic Coordinates and Rotational Elements: 2009. *Celestial Mechanics and Dynamical Astronomy*, 109 (2):101–135, 2011. ISSN 09232958. doi: 10.1007/s10569-010-9320-4.

[71] W. M. Folkner, J. G. Williams, D. H. Boggs, R. S. Park, and P. Kuchynka. *The Planetary and Lunar Ephemerides DE430 and DE431*. 2014. ISBN 2013072201.

[72] Jet Propulsion Laboratory. HORIZONS Web Interface. `http://ssd.jpl.nasa.gov/horizons.cgi`. Retrieved on October 2016.

[73] NAIF (Navigation and Ancillary Information Facility). SPICE Toolbox. `http://naif.jpl.nasa.gov/naif/`. Retrieved on August 2016.

[74] A. Annex, J. McAuliffe, and The Gitter Badger. SpiceyPy. March 2016. doi: 10.5281/zenodo.48437.

[75] NASA's Jet Propulsion Laboratory. Approximate Position of the Major Planets, . URL `http://ssd.jpl.nasa.gov/txt/aprx_pos_planets.pdf` and `http://ssd.jpl.nasa.gov/?planet_pos`. Retrieved on August 2016.

[76] N. Arora and R. P. Russell. A fast, accurate, and smooth planetary ephemeris retrieval system. *Celestial Mechanics and Dynamical Astronomy*, 108(2):107–124, 2010. ISSN 09232958. doi: 10.1007/s10569-010-9296-0.

[77] SciPy. SciPy Toolbox. `https://www.scipy.org/`. Retrieved on August 2016.

[78] N. J. Strange and J. M. Longuski. Graphical Method for Gravity-Assist Trajectory Design. *Journal of Spacecraft and Rockets*, 39(1):9–16, 2002. ISSN 0022-4650. doi: 10.2514/2.3800.

[79] J. Sims, J. M. Longuski, and A. J. Staugler. V8 Leveraging for Interplanetary Missions: Multiple-Revolution Orbit Techniques. *Journal of Guidance, Control, and Dynamics*, 20(3):409–415, 1997. ISSN 0731-5090. doi: 10.2514/2.4064.

[80] S. Campagnola and R. P. Russell. The Endgame problem part A: V-Infinity Leveraging technique and the Leveraging graph. *Advances in the Astronautical Sciences*, 134(2):1867–1886, 2009. ISSN 00653438. doi: 10.2514/1.44258.

[81] ESA. PyKEP. `https://esa.github.io/pykep/`, . Retrieved on February 2016.

[82] ESA. PyGMO. `https://esa.github.io/pygmo/`, . Retrieved on February 2016.

[83] D. Izzo. PyGMO and PyKEP: open source tools for massively parallel optimization in astrodynamics (the case of interplanetary trajectory optimization). *Proceed. Fifth International Conf. Astrodynam. Tools and Techniques, ICATT*, 2012. URL `https://scholar.google.com.br/citations?view{_}op=edit{_}citation{&}hl=en{&}citation{_}for{_}view=ykp4zwQAAAAJ:0U6Ihb5iCvQC{&}continue=/scholar{%}3Fhl{%}3Den{%}26as{_}sdt{%}3D0,5{%}26scilib{%}3D1{%}26scioq{%}3Dpygmo{&}citilm=1`.

[84] T. Vinkó, D. Izzo, and C. Bombardelli. Benchmarking different global optimisation techniques for preliminary space trajectory design. In *58th International Astronautical Congress, International Astronautical Federation (IAF)*, 2007.

[85] Advanced Concepts Team - European Space Agency. Global Trajectory Optimisation Problems database. `http://www.esa.int/gsp/ACT/inf/projects/gtop/gtop.html`. Retrieved on August 2016.

[86] R. Storn and K. Price. Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *Journal of Global Optimization*, 11(4):341–359, 1997. ISSN 1573-2916. doi: 10.1023/A:1008202821328. URL `http://dx.doi.org/10.1023/A:1008202821328`.

[87] J. Brest, V. Zumer, and M. S. Maucec. Self-adaptive differential evolution algorithm in constrained real-parameter optimization. In *2006 IEEE International Conference on Evolutionary Computation*, pages 215–222. IEEE, 2006.

[88] A. Corana, Marchesi M., C. Martini, and S. Ridella. Minimizing Multimodal Functions of Continuous Variables with the 'Simulated Annealing' Algorithm. *ACM Transactions on Mathematical Software*, 13(3):262–280, 1987. ISSN 00983500. doi: 10.1145/66888.356281.

[89] D. Kraft. *A Software Package for Sequential Quadratic Programming*. Deutsche Forschungs- und Versuchsanstalt fur Luft- und Raumfahrt Koln: Forschungsbericht. Wiss. Berichtswesen d. DFVLR, 1988. URL `https://books.google.pt/books?id=4rKaGwAACAAJ`.

[90] Northwestern McCormick School of Engineering. Sequential Quadratic Programming. `https://optimization.mccormick.northwestern.edu/index.php/Sequential_quadratic_programming`. Retrieved on September 2016.

[91] Burkardt, John. The Compass Search Optimization Algorithm. https://people.sc.fsu.edu/~jburkardt/c_src/compass_search/compass_search.html. Retrieved on August 2016.

[92] B. Correia. Multi-revolution transfers in orbital mechanics. Master's thesis, Instituto Superior Técnico, May 2016.

[93] T. R. Spilker, R. C. Moeller, C. S. Borden, W. D. Smythe, R. E. Lock, J. O. Elliott, J. A. Wertz, and N. J. Strange. Analysis of architectures for the scientific exploration of Enceladus. *IEEE Aerospace Conference Proceedings*, 2009. ISSN 1095323X. doi: 10.1109/AERO.2009.4839317.

[94] NASA. Advanced Stirling Radioisotope Generator — NASA Fact Sheet. https://solarsystem.nasa.gov/rps/docs/APP%20ASRG%20Fact%20Sheet%20v3%209-3-13.pdf, . Retrieved on October 2016.

[95] R. Russell and M. Lara. On the design of an enceladus science orbit. *Acta Astronautica*, (65): 27–39, March 2009. doi: 10.1016/j.actaastro.2009.01.021.

[96] F. Peralta and S. Flanagan. Cassini Interplanetary Trajectory Design. *Control Engineering Practice*, 3(11):1603–1610, 1995.

[97] A. RocketDyne. Bi-propellant data sheets. http://www.rocket.com/files/aerojet/documents/Capabilities/PDFs/Bipropellant%20Data%20Sheets.pdf. Retrieved on September 2016.

[98] N. Strange, R. Russell, and B. Buffington. Mapping the v-infinity globe. *AAS Paper*, pages 07–277, 2007.

Appendix A

Images

A.1 Tisserand Plots

This section contains Tisserand plots for the Solar System and the moons of Saturn:

- Figure A.1 displays an example Tisserand plot of the Solar System (between Venus and Saturn).
- Figure A.2 is the Tisserand Graph for Saturn's moon system.
- Figures A.3 through A.7 are Tisserand graphs for the moons of interest in Saturn's system. Not all resonances are displayed for the sake of clarity.

  Descriptions of each particular plot are in the figure's caption.

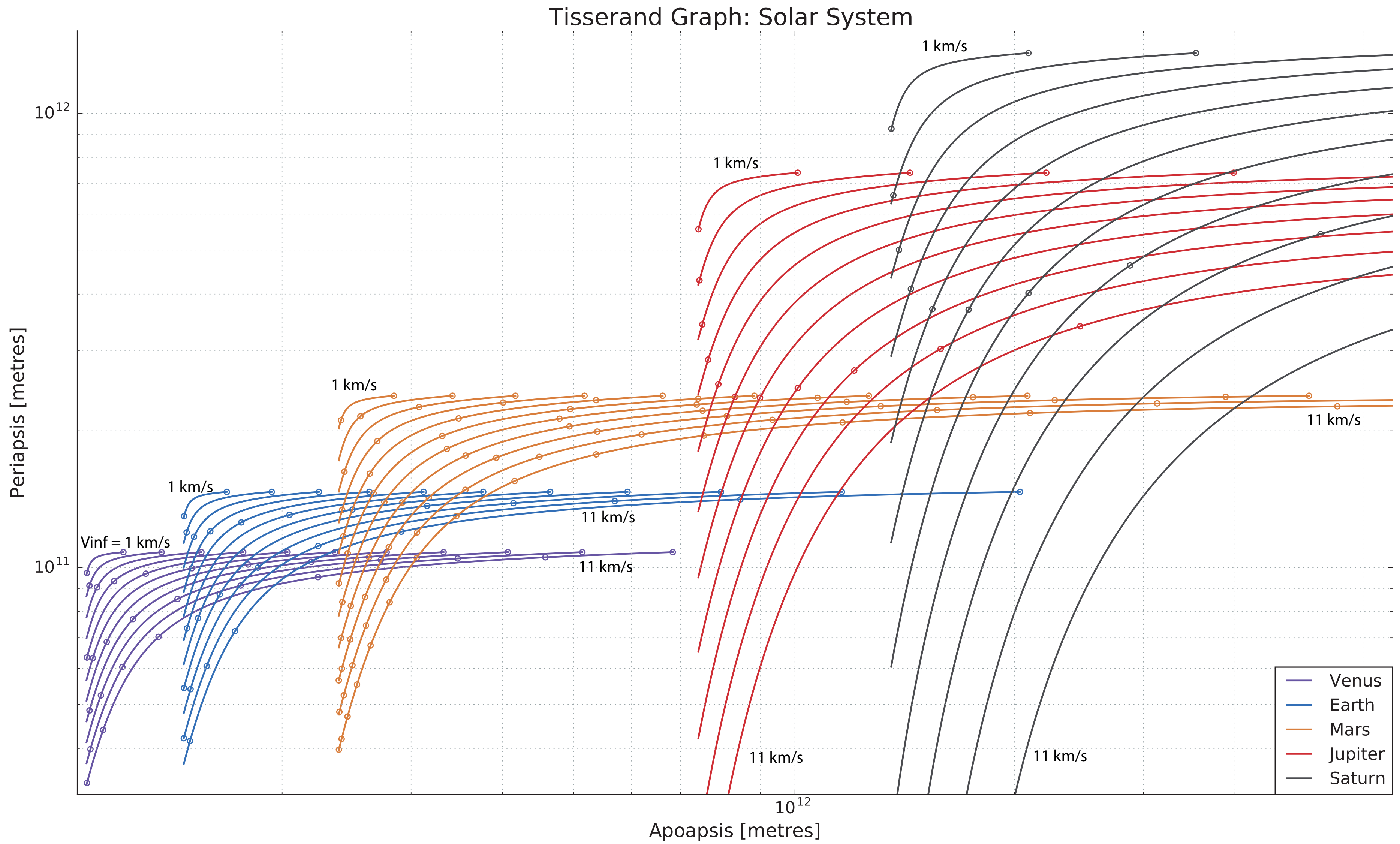


Figure A.1: Logarithmic Tisserand Graph of the Solar System. Curves between $V_\infty = 1$ km/s and $V_\infty = 11$ km/s at a step of 1 km/s. The circles are visual aids that help understand the maximum movement that can be performed along a curve due to the minimum flyby radius constraint.

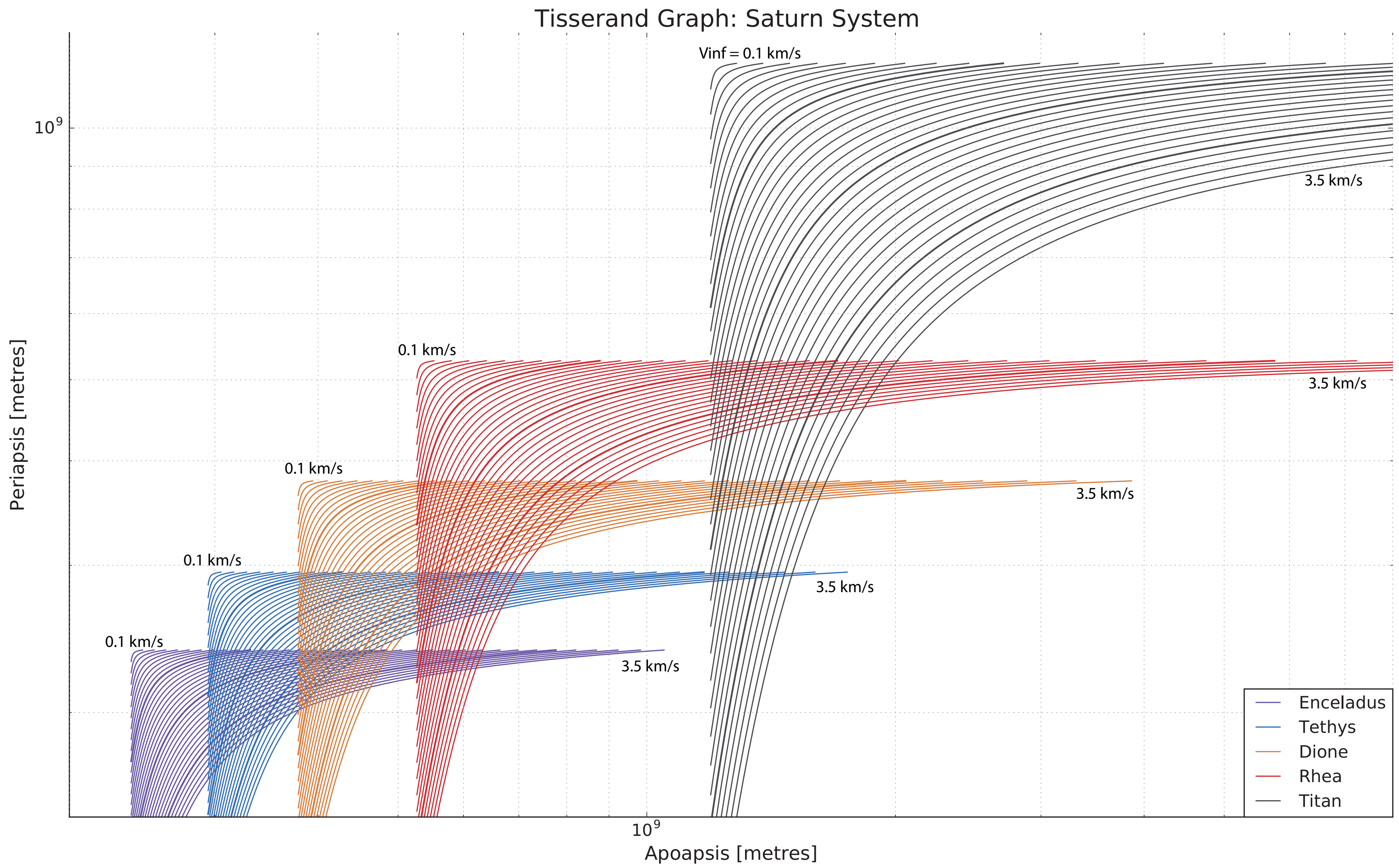


Figure A.2: Logarithmic Tisserand Graph of Saturn's moon system, from Enceladus to Titan. Curves between $V_\infty = 100$ m/s and $V_\infty = 3.5$ km/s at a step of 100 m/s. Darker curves mark $V_\infty = 1$ km/s, 2 km/s and 3 km/s.

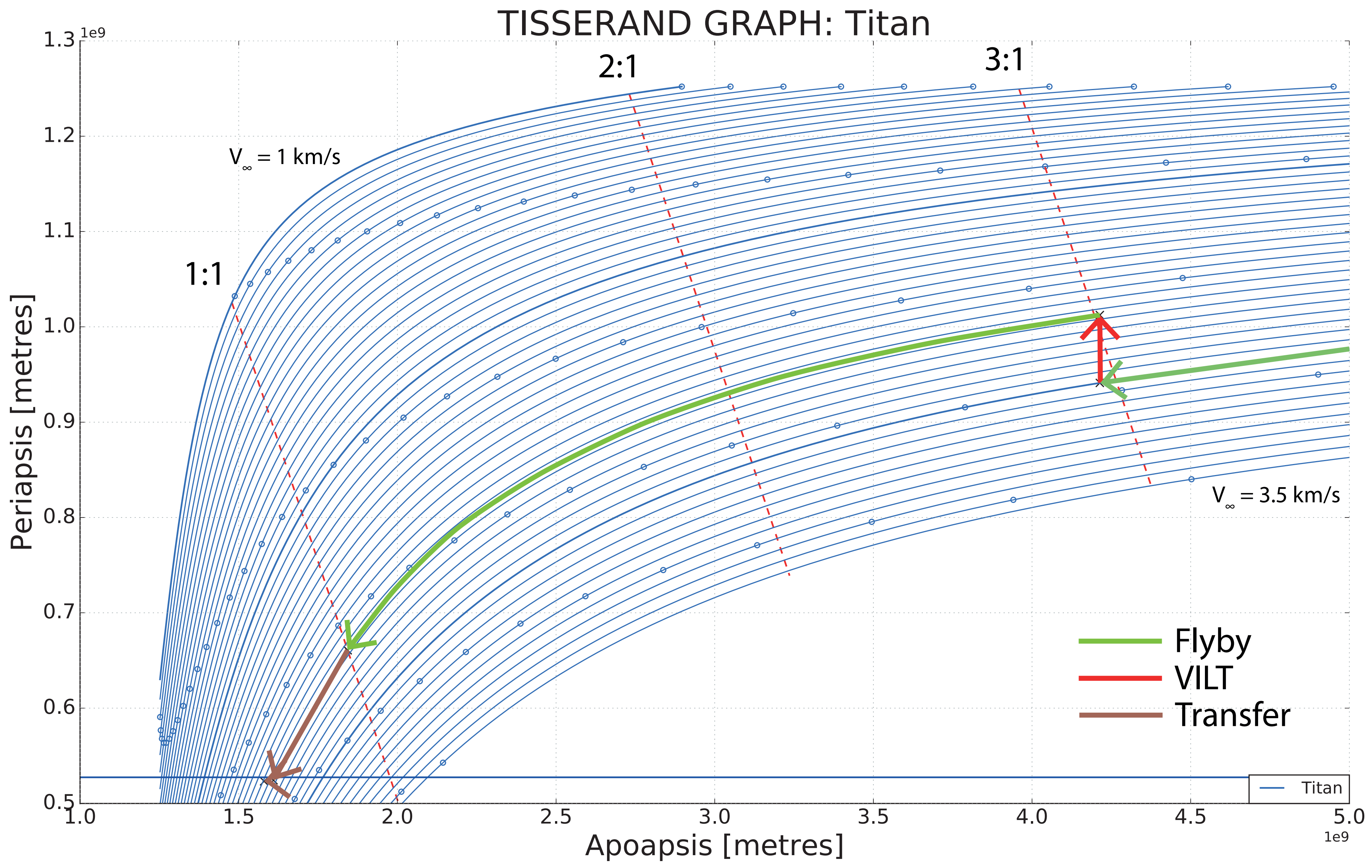


Figure A.3: Tisserand Graph for Titan. Curves start at the top with $V_\infty = 1000$ m/s and end at $V_\infty = 3.5$ km/s with a step of 50 m/s. Dotted lines specify resonant orbits. The blue line marks the orbital radius of the next body in the sequence (Rhea). The solution to the moon tour is overlaid on the Tisserand plot.

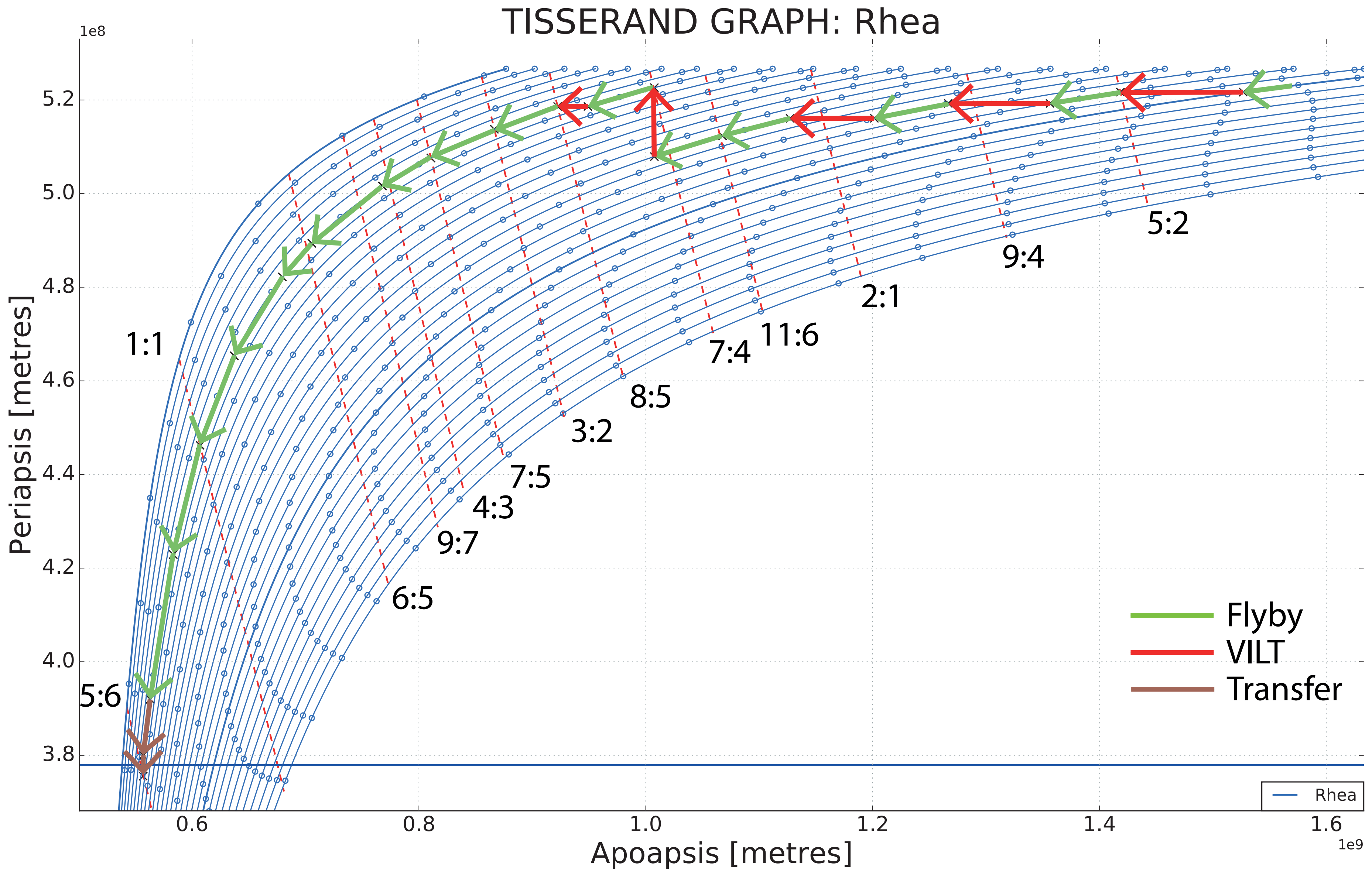


Figure A.4: Tisserand Graph for Rhea. Curves start at the top with $V_\infty = 1000$ m/s and end at $V_\infty = 2.5$ km/s with a step of 50 m/s. Dotted lines specify resonant orbits. The blue line marks the orbital radius of the next body in the sequence (Dione). The solution to the moon tour is overlaid on the Tisserand plot.

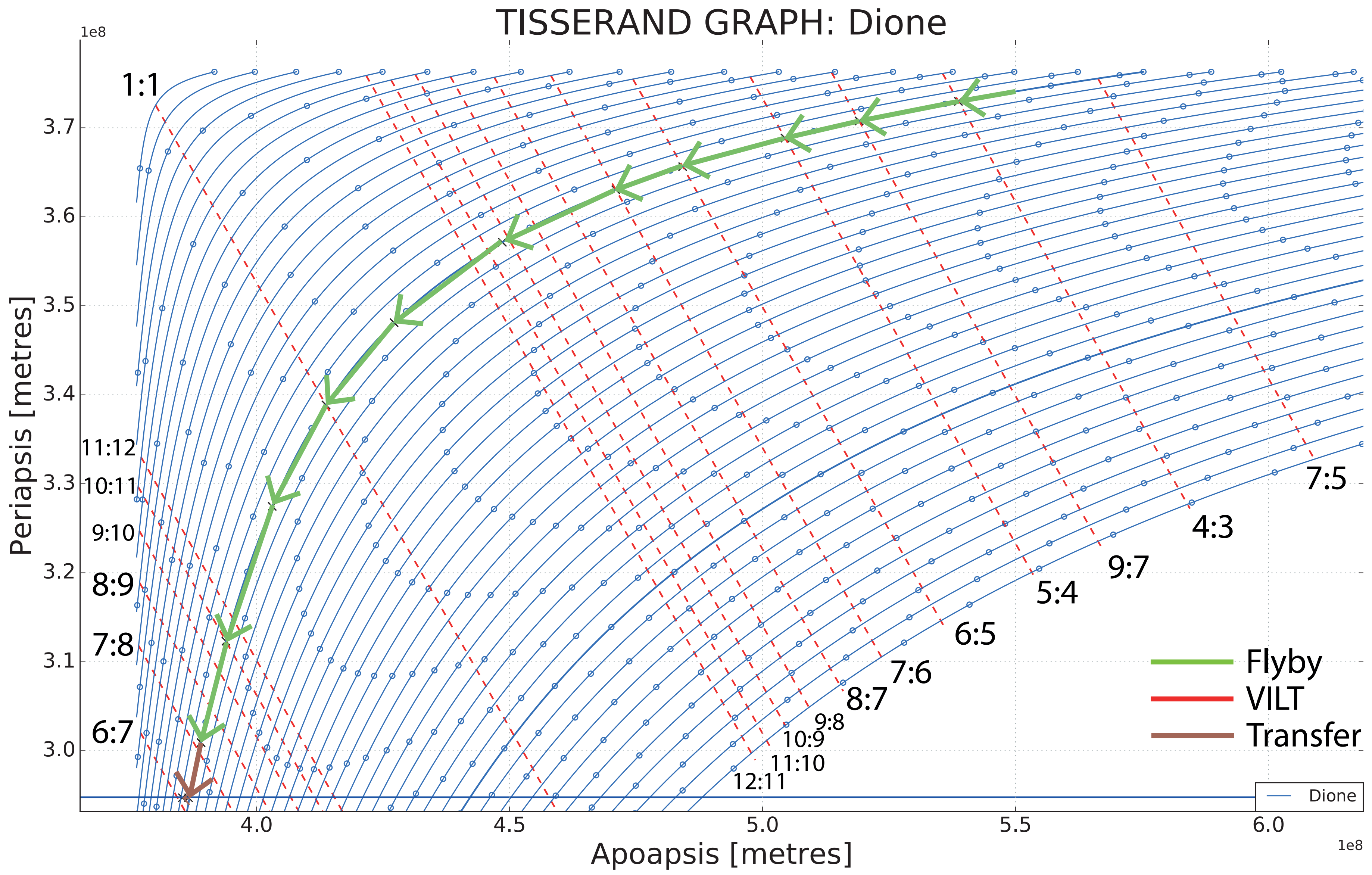


Figure A.5: Tisserand Graph for Dione. Curves start at the top with $V_\infty = 100$ m/s and end at $V_\infty = 2.5$ km/s with a step of 50 m/s. Dotted lines specify resonant orbits. The blue line marks the orbital radius of the next body in the sequence (Tethys). The solution to the moon tour is overlaid on the Tisserand plot.

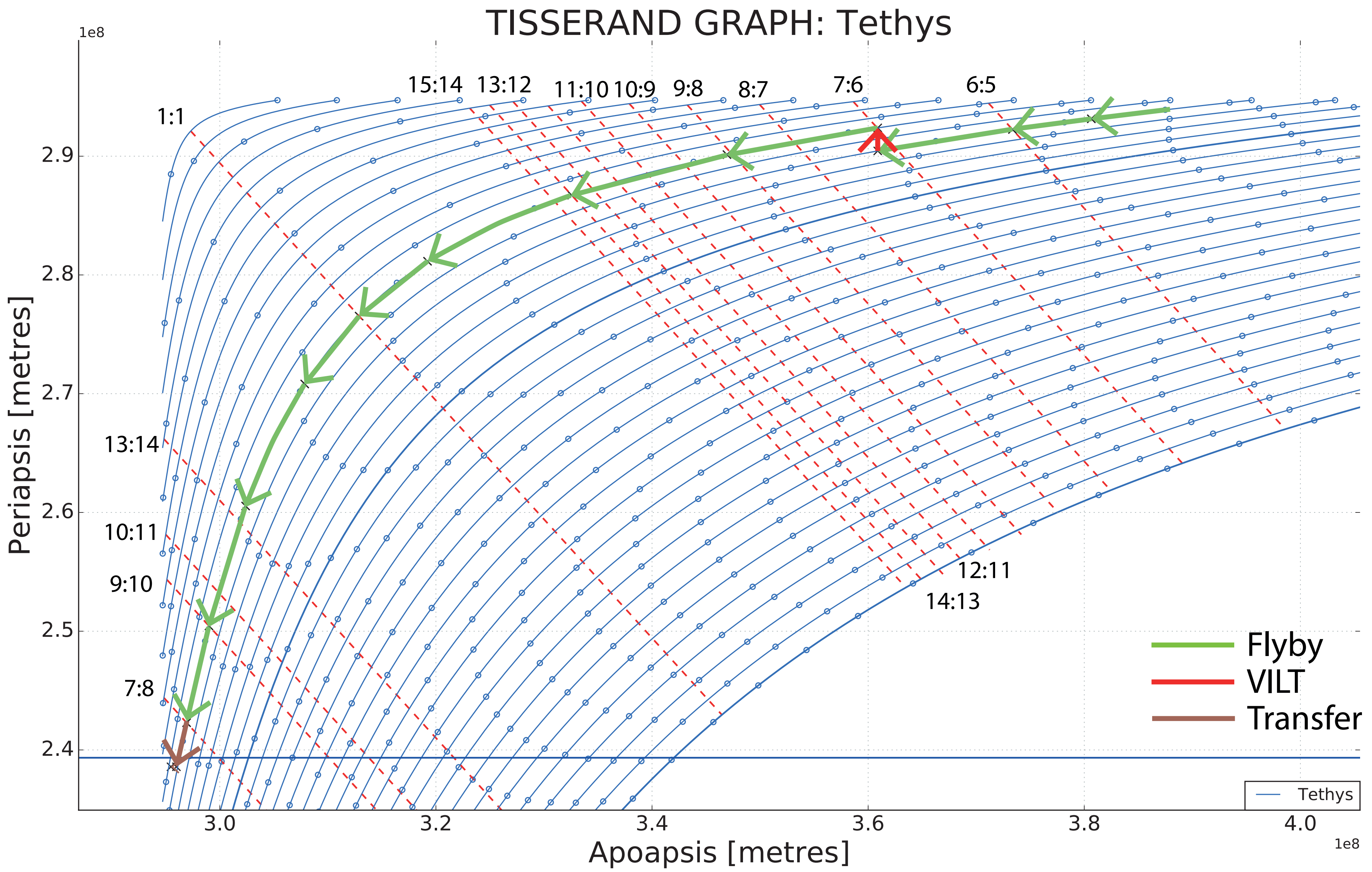


Figure A.6: Tisserand Graph for Tethys. Curves start at the top with $V_\infty = 100$ m/s and end at $V_\infty = 2$ km/s with a step of 50 m/s. Dotted lines specify resonant orbits. The blue line marks the orbital radius of the next body in the sequence (Enceladus). The solution to the moon tour is overlaid on the Tisserand plot.

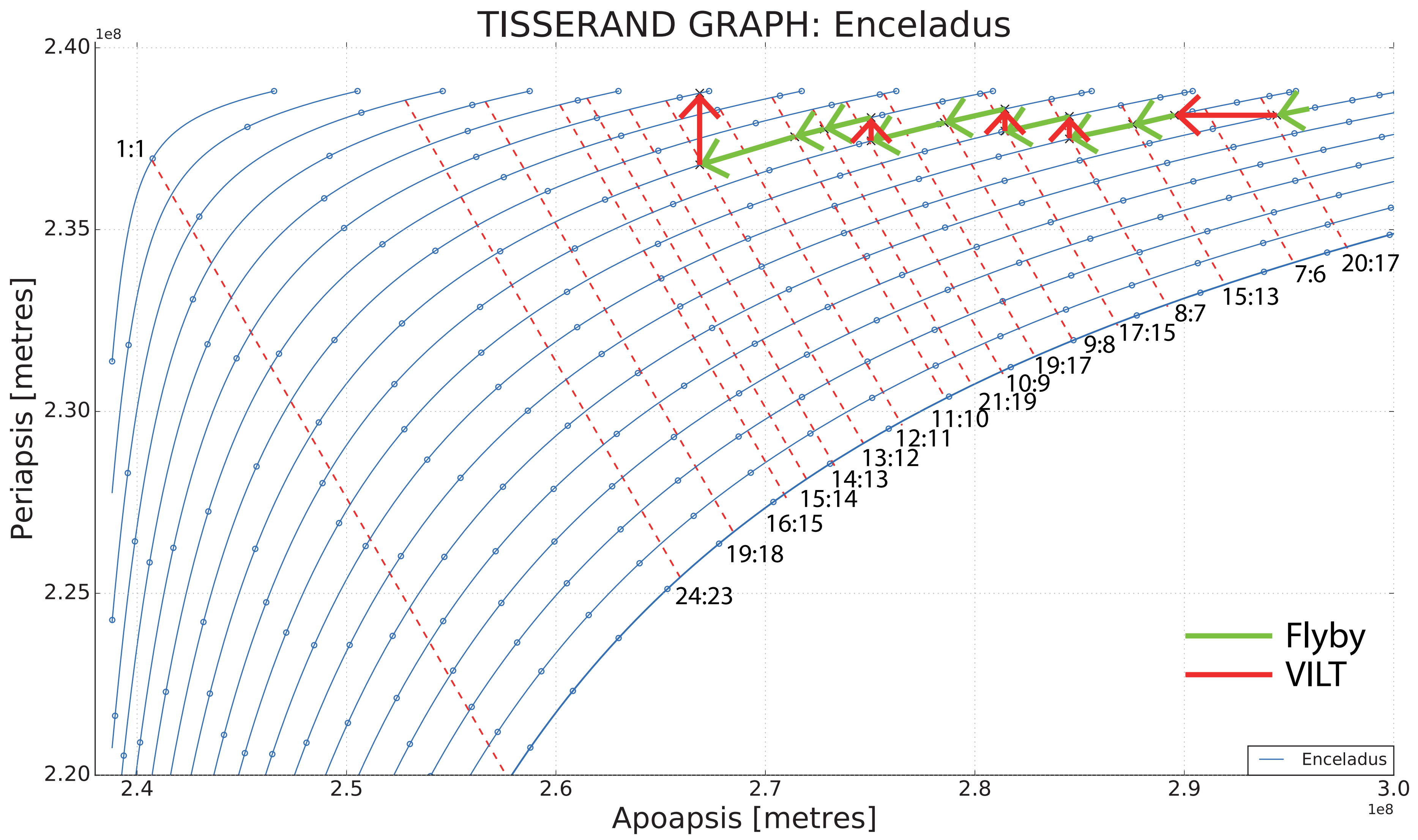


Figure A.7: Tisserand Graph for Enceladus. Curves start at the top with $V_\infty = 100$ m/s and end at $V_\infty = 1$ km/s with a step of 50 m/s. Dotted lines specify resonant orbits. The solution to the moon tour is overlaid on the Tisserand plot.

A.2 Moon Tour

This section contains the moon tour trajectories. All trajectories are projected in the orbital plane of the moon they refer to.

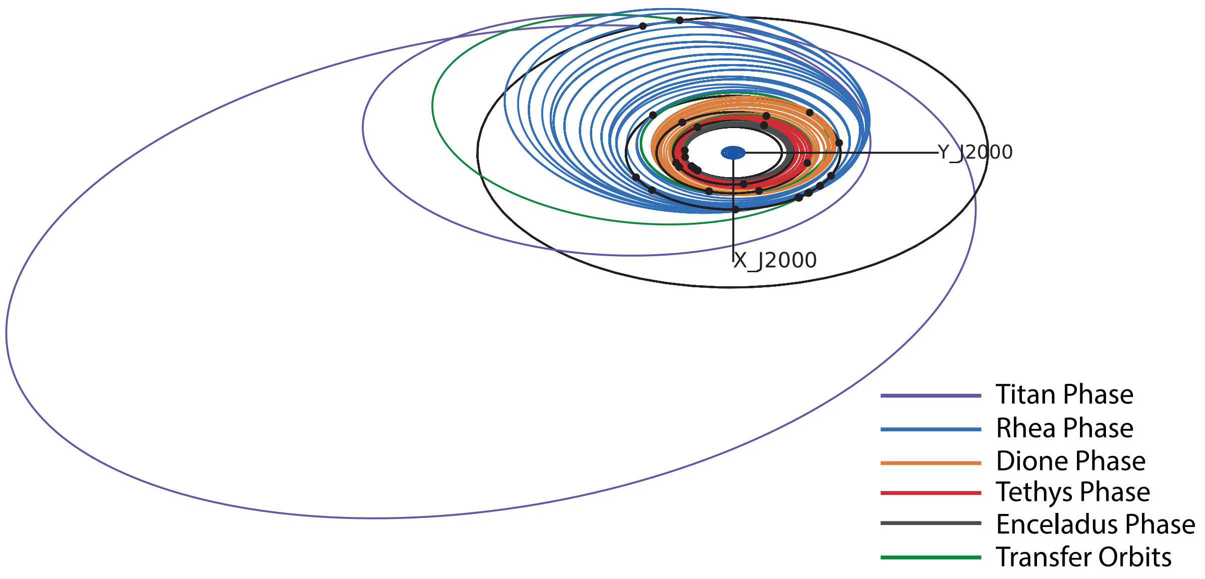


Figure A.8: Complete moon tour.

The Titan phase of the moon tour is shown in figure A.9. The inclination change flyby is not pictured.

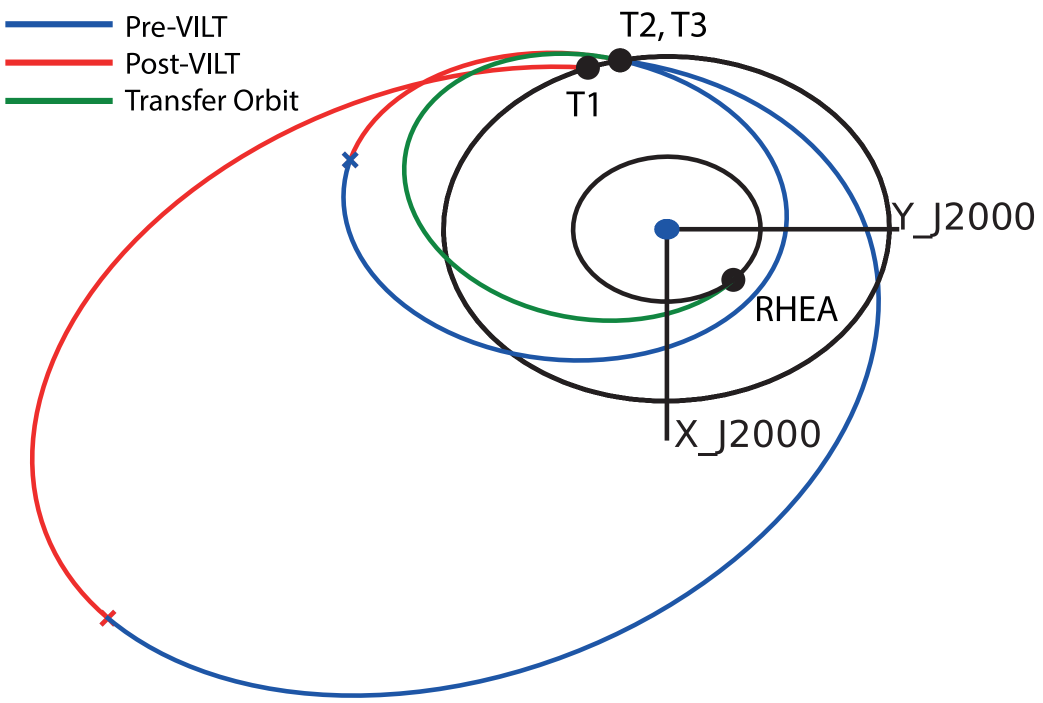


Figure A.9: Moon Tour: Titan Phase. The X's mark the locations of the DSM of each VILT.

Figure A.10 displays the Rhea phase of the moon tour and the transfer orbit to Dione.

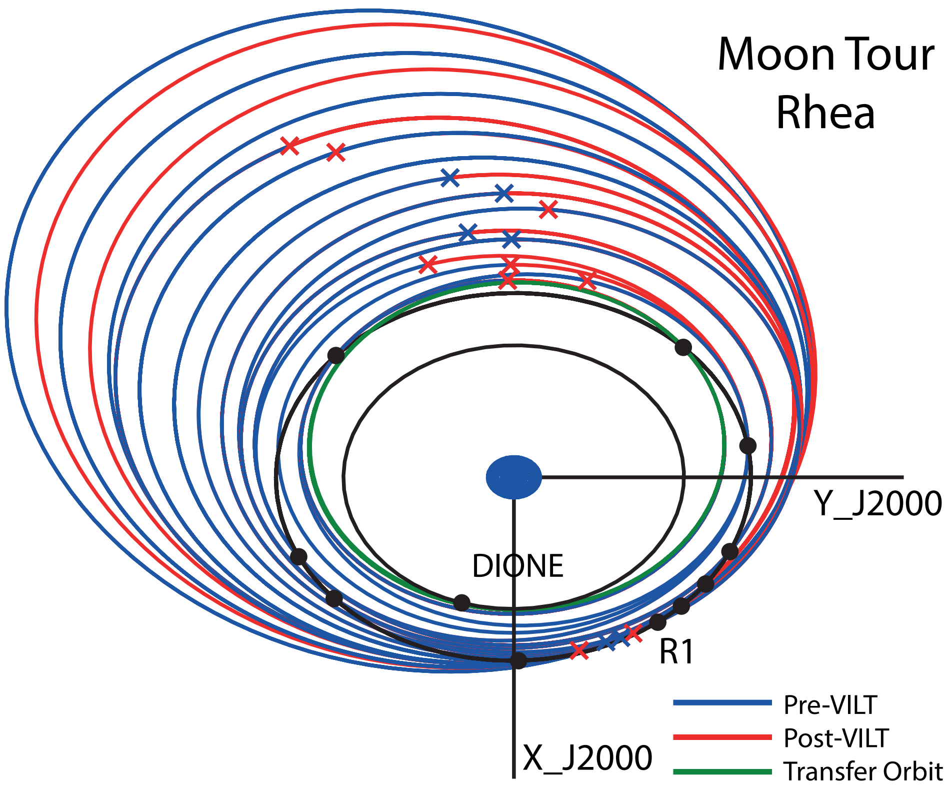


Figure A.10: Moon Tour: Rhea Phase

Figure A.11 displays the Dione phase of the moon tour and the transfer orbit to Tethys.

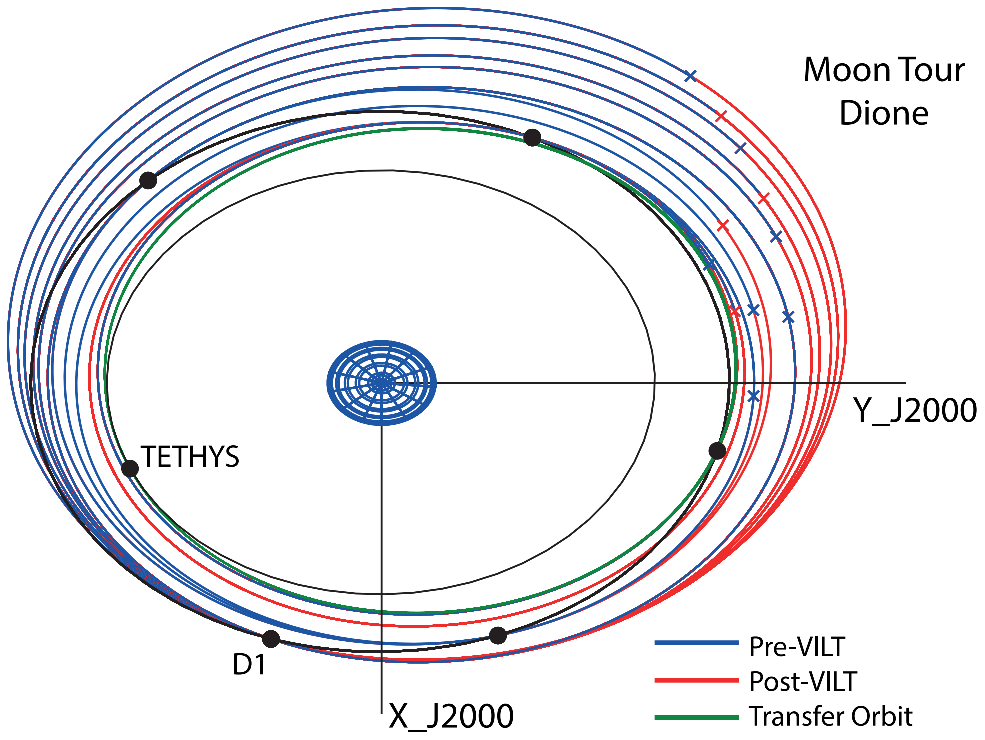


Figure A.11: Moon Tour: Dione Phase

Figure A.12 displays the Tethys phase of the moon tour and the transfer orbit to Enceladus.

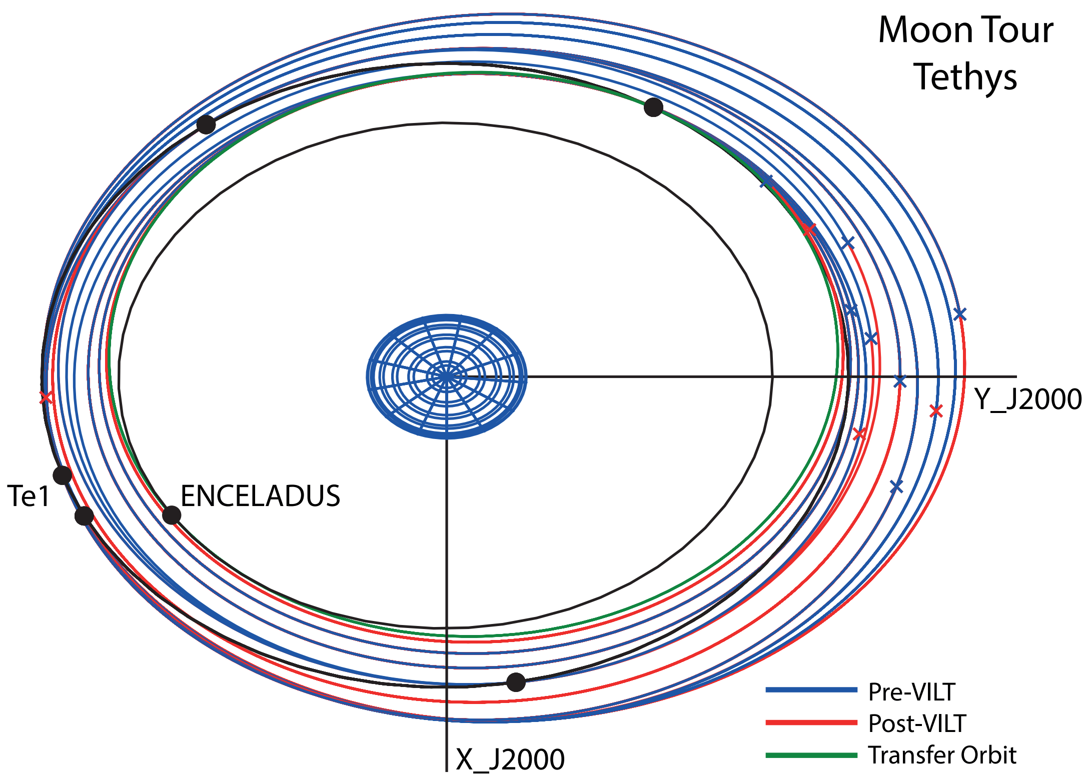


Figure A.12: Moon Tour: Tethys Phase

Finally, figure A.13 shows the last phase of the moon tour.

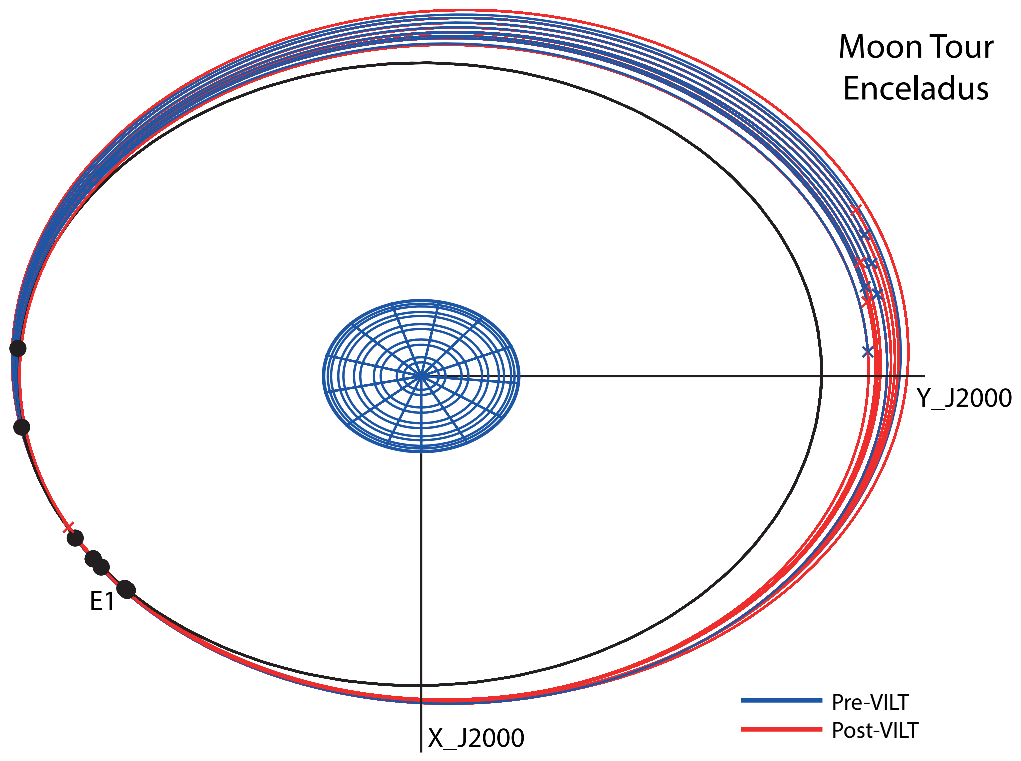


Figure A.13: Moon Tour: Enceladus Phase

A.3 Other images

This section includes images from this work that would break the layout of the document.

Figure A.14 shows a close-up of a typical entry geometry at Saturn. The planet is not to scale and the PRM is not shown.

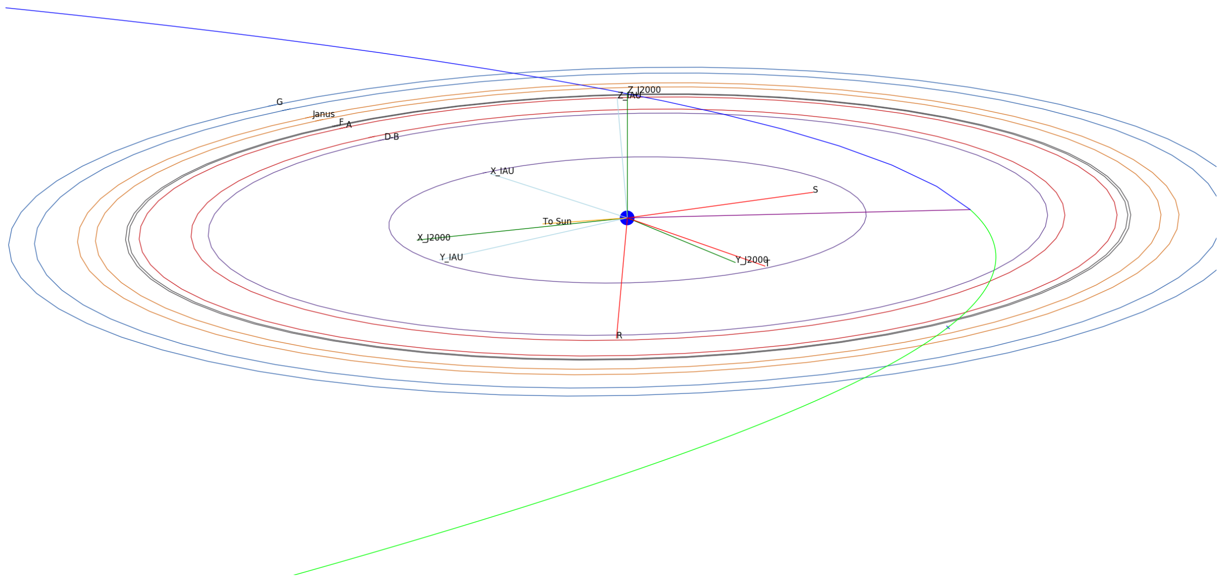


Figure A.14: Typical capture geometry at Saturn. The orbit comes from above and leaves below the ring plane. Green: Arrival Hyperbola. Blue: Post-capture ellipse. Gaps are present where rings change color. The blue x in the arrival hyperbola marks the spot where the spacecraft crosses the ring plane.

Appendix B

Trajectory Information

This appendix contains detailed information about the trajectories obtained.

B.1 Interplanetary Option A (Trajectory 2)

Table B.1: Interplanetary Trajectory T2: Detailed Results

Interplanetary Trajectory 2 (EVEES)			
		Results (Summary)	**Units**
	Mission Time	Launch on 12-12-2022 and Arrival on 03-12-2032	
	ΔV	455.136	m/s
	Time of Flight	9.969	years
	Launch V_∞	4	km/s
	Arrival V_∞	5.761	km/s
		Results (Detailed)	**Units**
Leg 1	Leg	**Earth to Venus (L)**	
	Launch Epoch	8381.617 (12-12-2022)	mjd2000
	Duration	430.4	days
	DSM After	173.5	days
	ΔV_{DSM}	0.160	m/s
Leg 2	Leg	**Venus to Earth (V1)**	
	Flyby Epoch	9208.397 (16-02-2024)	mjd2000
	Flyby Radius	1.585	Venus Radii
	Duration	396.39	days
	DSM After	21.96	days
	ΔV_{DSM}	0.037	m/s
Leg 3	Leg	**Earth to Earth (E1)**	
	Flyby Epoch	9208.397 (18-03-2025)	mjd2000
	Flyby Radius	2.36	Earth Radii
	Duration	850.49	days
	DSM After	—	days
	ΔV_{DSM}	—	m/s
Leg 4	Leg	**Earth to Saturn (E2)**	
	Flyby Epoch	10058.884 (16-07-2027)	mjd2000
	Flyby Radius	1.03	Earth Radii
	Duration	1966.45	days
	DSM After	69.033	days
	ΔV_{DSM}	454.94	m/s
Arrival Details	Epoch	12025.335 (03-12-2032)	mjd2000
	Arrival V_∞	[5723.044293324046, 661.5850449698022, -57.275309120671864]	m/s

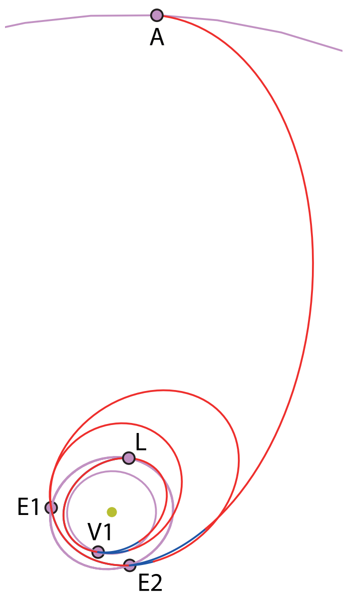


Figure B.1: Projection of trajectory 2 in the ecliptic plane

B.2 Saturn Orbit Insertion

This section presents detailed information regarding the Saturn Orbit Insertion solution achieved in this work.

Table B.2: Saturn Orbit Insertion: Detailed Information

		Saturn Orbit Insertion	
		Results (Summary)	
	Mission Time	Capture on 12-12-2022. Arrival at Titan on 03-12-2022	**Units**
	ΔV	967.864	m/s
	Time of Flight	383.62	days
	Titan Arrival V_∞	3	km/s
	Titan Arrival Inclination	6.92	deg
		Results (Detailed)	
		Arrival Information	**Units**
	Arrival Epoch	12025.334 (03-12-2032)	mjd2000
	Arrival V_∞	[-5723.044293324046, -661.5850449698022, 57.275309120671864]	m/s
Arrival	Arrival Declination	4.79	deg
		B-Plane Information	**Units**
	B-Plane Angle	0.0	rad
	BdotT	486584721.05	m
	ΔV_{SOI}	639.472	m/s
SOI	Periapsis Vector (J2000)	(-95585185.91535664, 27410790.98265934, -1504928.3554632142)	m
	R_p	1.708	Saturn Radii
	Ring Crossing	2.495	Saturn Radii
	Eccentricity	0.99	—
Post-Capture Orbit	Inclination (IAU_SATURN)	4.788	deg
	Orbital Period	370.3	days
	ΔV_{PRM}	328.392	m/s
PRM and Post-PRM	$\Delta T_{Apoapsis}$	-32.31	days
	Time of Flight	230.78	days
	Inclination (IAU_SATURN)	6.92	deg
	Arrival Epoch	12408.95 (21-12-2033)	mjd2000
Titan Arrival	Arrival Velocity Vector (J2000)	(734.2955882450592, -7388.4583396920825, 1357.8212107142867)	m/s
	Arrival V_∞	3000	m/s

Figure B.2 shows a close-up of the Saturn orbit insertion trajectory. The red cross marks the spot where the spacecraft crosses the ring plane. The green trajectory is the arrival hyperbola and the blue trajectory is the post-capture ellipse. Rings are between circles with the same colours.

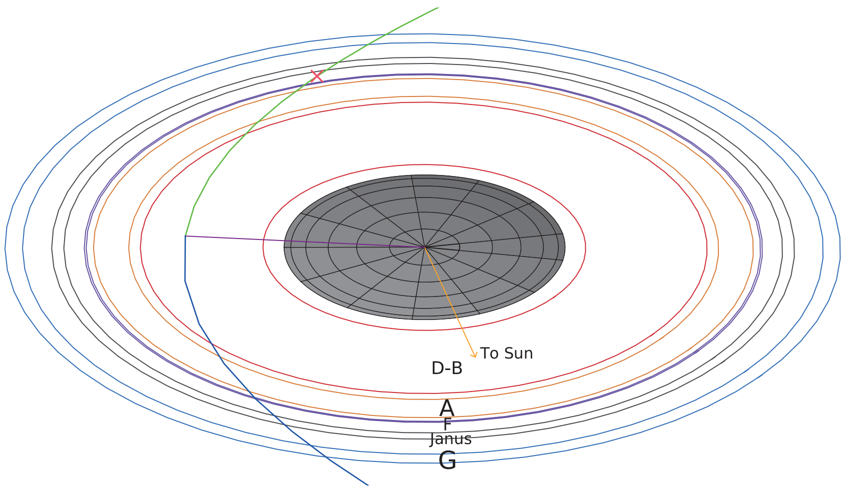


Figure B.2: SOI, looking down on the planet's north pole.

B.3 Enceladus Orbit Insertion

Table B.3 displays detailed information regarding the Enceladus Orbit Insertion manoeuvre.

Table B.3: Enceladus Orbit Insertion

		Enceladus Orbit Insertion	
		Results (Summary)	
	Arrival Date	25-Oct-2036	**Units**
	ΔV_{EOI}	243.5	m/s
	Enceladus Arrival V_∞	350.0	m/s
		Results (Detailed)	
		Arrival Information	**Units**
	Arrival Epoch	13447.2858 (25-Oct-2036)	mjd2000
	Arrival V_∞	[348.60501534750983, -8.92687915852457, -29.302763609082604]	m/s
Arrival	Arrival Declination	-0.01	deg
		B-Plane Information	**Units**
	B-Plane Angle	58.72	rad
	BdotT	313048.95	m
	BdotR	515381.51	m
	Eccentricity	0.076	—
	Inclination (IAU_ENCELADUS)	58.725	deg
Post-Capture Orbit	Orbital Period	0.436	days
	Periapsis Vector (J2000)	[-9841.998383029713, -342113.4170946134, -479358.50936710427]	m
	Initial Altitude	295	km