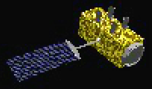
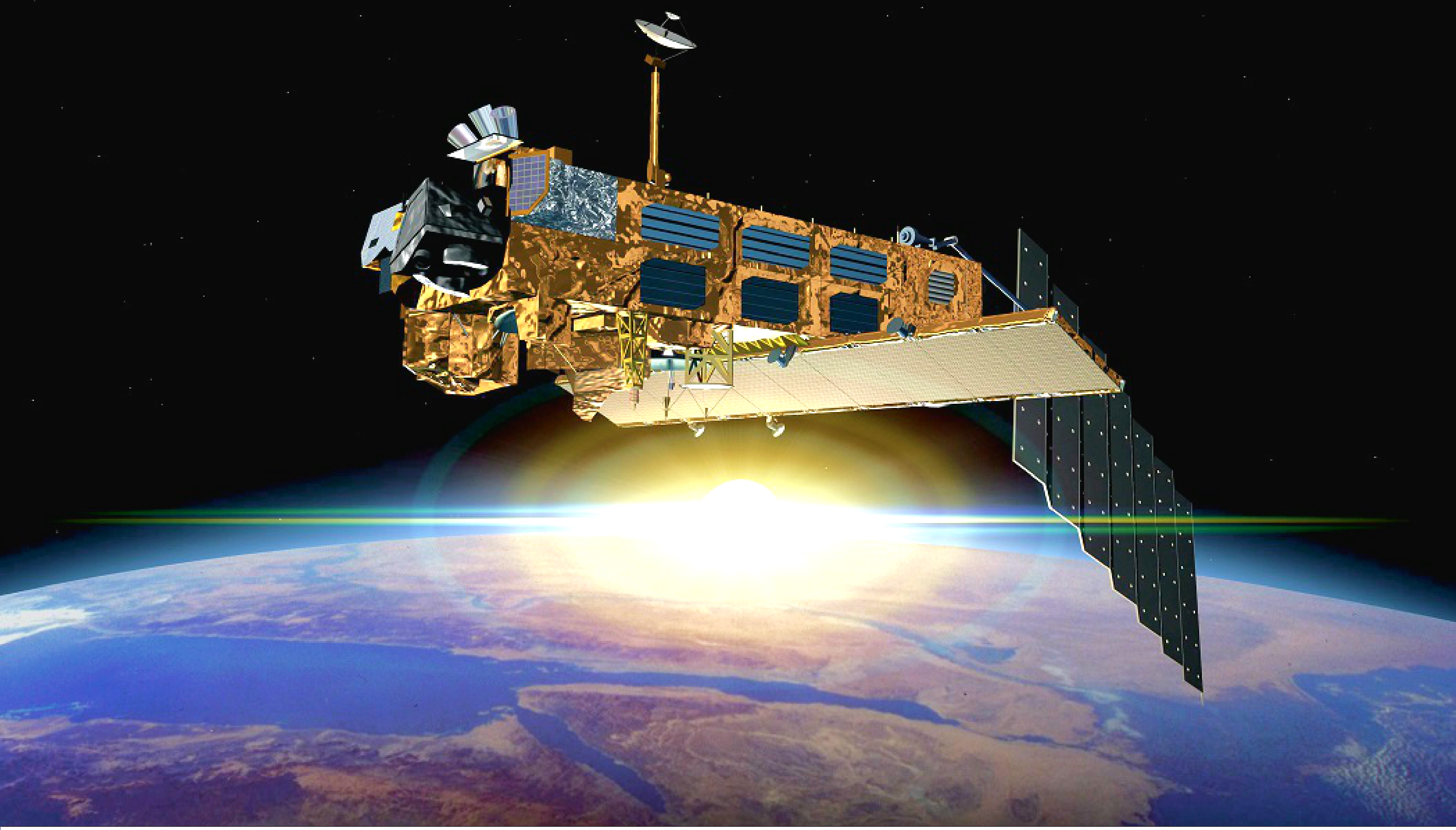
# Convex Guidance for Envisat Rendezvous

MSc thesis



M. R. Bhagat



TU Delft Delft University of Technology

Challenge the future

Cover image credit: Adopted from ESA and Deloo (2014)

# Convex Guidance for Envisat Rendezvous

by

**M. R. Bhagat**

in partial fulfillment of the requirements for the degree of

**Master of Science**
in Aerospace Engineering

at the Delft University of Technology,
to be defended publicly on Wednesday, February 17, 2016 at 11:00 AM.

| | | |
|---|---|---|
| Supervisor: | Dr. ir. E. Mooij, | TU Delft |
| Thesis committee: | Prof. Dr. ir. P. N. A. M. Visser, | TU Delft |
| | ir. K. Cowan, | TU Delft |

TU Delft
Delft
University of
Technology

*"...The necessary number of iterations is one more than the number you have currently done. This is true at any point in time."*

Akin’s law of spacecraft design, No. 3

# Abstract

This thesis is focused on autonomous trajectory planning for Active Debris Removal (ADR), commencing from far-range rendezvous and up to a point just before docking. Unlike typical rendezvous missions, the target in ADR is not designed for rendezvous and capture; it is usually uncooperative and possibly tumbling. If the satellite is unpassivated, then there is the added risk of explosion, while the presence of large appendages can make the satellite fragile. A case in point is Envisat, European Space Agency's largest satellite (about 8000 kg) that went defunct in 2012 and is currently, their prime concern for ADR.

Most past and current technology for guidance is based on the Clohessy-Wiltshire (CW) equations, which are limited in terms of application - they use linearized dynamics, need circular orbits, employ impulsive burns, and require the user to provide some initial guess for the solution. Realistic final approach involves a number of trajectory constraints, like those on docking axis alignment, approach cone angle, keep-out-sphere, station keeping and hold points, which are hard to include with CW-equations. Further, there is a need to make the system autonomous so that it can handle off-nominal situations as well.

Convex optimization problems are a class of problems that are nearly as easy to solve as linear problems, but are more general in their scope of application. They can handle some non-linear constraints, be solved in polynomial time and are guaranteed to give global minimum. Inspired from the recent development of convex-optimization based trajectory-planning technique by Liu (*Autonomous trajectory planning by convex optimization*, PhD thesis, Iowa State University, 2013), a guidance algorithm is developed for the Envisat mission. In comparision with Liu's algorithm, it shows faster convergence (upto 5 times faster), improved accuracy (by an order of 2), and low computational cost (upto 40% in specific cases). Additionally, a method to automate the problem formulation in conic form is provided, so interfacing software can be skipped and the problem can be solved upto 9 times quicker.

The guidance algorithm is able to handle all constraints of the Envisat mission, with the exception of forced motion constraint inside the keep-out-sphere. An alternative formulation is suggested which is moderately successful. The algorithm is also able to generate solutions for various attitude scenarios of Envisat. However, it is found that the method is not reliable, as some times feasible problems are found to be infeasible, or the solution is over/under estimated. Also, despite the improvement in accuracy, it still deviates substantially from the computed path, and needs a path-tracking mechanism. A PID controller is found to be sufficient for this purpose. A novel workaround to include perturbations is successfully demonstrated as well.

To summarize, although the convex-optimization based guidance method has very specific advantages over CW-equations, there is a lot left to be improved with respect to robustness and autonomy.

# PREFACE

This thesis report is the culmination of my Master of Science degree at the faculty of Aerospace Engineering, Delft University of Technology. It focusses on utilizing, assessing and improving a guidance algorithm, based on convex optimization theory, with the aim to contribute towards an autonomous guidance system.

At this point, I would like to extend my gratitude to numerous people who have helped and shaped my work until now. Foremost, to my supervisor, Dr. Erwin Mooij, for giving me an opportunity to work on something as exciting and challenging as this topic, for providing inputs as I worked on this report and for being patient while I took my time to hand it in. Special thanks are in order for Ingo Gerth, for pointing out the topic in the first place and for his valuable feedback. A hearty shout out to fellow students and friends, who have been really kind and helpful all this while (especially with LaTeX). And finally, I would like to thank my family for their continued support and enormous faith.

In many ways, relative orbital motion is counter-intuitive to our expectations. Surely someday, humans will be able to conquer this realm as well and make rendezvous a rather routine process. But, until then, it is an exciting journey filled with touch-and-go instances and tumbling opportunities.

*M. R. Bhagat*
*Delft, February 2016*

# CONTENTS

# LIST OF ACRONYMS

| | |
|---|---|
| C0 | Constraint 0: Initial state |
| C10 | Constraint 10: Station keeping |
| C1 | Constraint 1: Equations of motion |
| C2 | Constraint 2: Maximum thrust magnitude |
| C3 | Constraint 3: Norm of thrust |
| C4 | Constraint 4: Hold point / state |
| C5 | Constraint 5: Approach cone |
| C6 | Constraint 6: Plume impingement |
| C7 | Constraint 7: Rate of thrust change |
| C8 | Constraint 8: Failed - forced motion |
| C9 | Constraint 9: Keep-out-sphere |
| Cf | Constraint f: Final state |
| ADR | Active Debris Removal |
| ANOVA | ANalysis Of VAriance |
| AS | Attitude Scenario |
| ATV | Automated Transfer Vehicle |
| AVGS | Advances Video Guidance Sensor |
| CNES | Centre National d'Études Spatiales |
| CoM | Centre of Mass |
| ConGAL | Convex Guidance Algorithm by Liu |
| CSM | Command Service Module |
| CW | Clohessy-Wiltshire |
| DARPA | Defense Advances Research Projects Agency |
| DART | Demonstration of Autonomous Rendezvous Technology |
| DCM | Direction Cosine Matrix |
| DEOS | Deutsche Orbitale Servicing Mission |
| DLR | Deutsches Zentrum für Luft- und Raumfahrt |
| DOF | Degree of Freedom |
| ECEF | Earth-Centred Earth-Fixed |
| ECI | Earth-Centred Inertial |
| ECOS | Embedded COnic Solver |
| EOL | End of Life |
| EOM | Equations of Motion |
| ESA | European Space Agency |
| ETS | Experiment Test Satellite |
| FP | Final Point |
| GEO | Geosynchronous Earth Orbit |
| GGNCSIM | Generic Guidance Navigation and Control SIMulator |
| GMAT | General Mission Analysis Tool |
| GNC | Guidance, Navigation and Control |
| GPS | Global Positioning System |
| HP1 | Hold Point 1 |
| IADC | Inter-Agency Space Debris Coordination Committee |
| ICRF | International Celestial Reference Frame |
| ISS | International Space Station |
| KOS | Keep-Out Sphere |
| LEO | Low Earth Orbit |
| LM | Lunar Module |
| LP | Linear Program |
| LVLH | Local-Vertical, Local-Horizontal |
| ME | Main Engine |
| MEO | Medium Earth Orbit |
| MOSA | Method Of Successive Approximations |
| MPCV | Multi-Purpose Crew Vehicle |
| MS | Mean Scheme |
| MSFC | Marshall Space Flight Center |
| ODE | Ordinary Differential Equation |
| OFAT | One-Factor-A-Time |
| OP | Orbit Propagator |
| OREKIT | ORbits Exploration KIT |
| PID | Proportional-Integral-Derivative |
| PMD | Post Mission Disposal |
| QCQP | Quadratically Constrained Quadratic Program |
| QP | Quadratic Program |
| RAAN | Right Ascension of the Ascending Node |
| RCA | Re-Computing Algorithm |
| RCS | Reaction Control System |
| RK | Runge-Kutta |
| RPO | Rendezvous and Proximity Operations |
| SA | Sensitivity Analysis |
| SDP | Semi-Definite Program |
| SGP | Simplified General Perturbations |

| | |
|---|---|
| SK | Station Keeping |
| SOCP | Second-Order Conic Program |
| SSN | Space Surveillance Network |
| TLE | Two Line Element |
| TU Delft | Technische Universiteit Delft |
| TUDAT | TU Delft Astrodynamics Toolbox |
| USA | United States of America |
| USSR | Union of Soviet Socialist Republics |
| WP | Work Package |
| XSS | eXperimental Satellite System |

# List of Symbols

| Term | Description | Units |
|---|---|---|
| | **Latin letters** | |
| $\mathbf{1}_n$ | Normalized docking axis | - |
| $\mathbf{A}$ | State matrix for equations of motion | - |
| $\breve{\mathbf{A}}$ | Linear inequality matrix for solvers | - |
| $\mathbf{a}$ | Net acceleration of a body | $\mathrm{m/s^2}$ |
| $a$ | Semi-major axis of orbit | m |
| $\mathbf{C}$ | Direction cosine matrix | - |
| $\mathbf{E}$ | Extraction matrix to refer elements from $\mathbf{W}$ | - |
| $e$ | Orbital eccentricity | - |
| $\mathbf{h}$ | Specific angular momentum | $\mathrm{m^2 s^{-1}}$ |
| $h$ | Integrator step-size | - |
| $i$ | Orbital inclination | rad |
| $I_{sp}$ | Specific impulse | s |
| $J$ | Performance index of optimization problem | varies |
| $N$ | Number of elements | - |
| $n$ | Mean motion of satellite | $\mathrm{s^{-1}}$ |
| $\mathbf{q}$ | Quaternion of rotation | - |
| $\mathbf{R}$ | Unit axis rotation matrix | - |
| $\mathbf{r}$ | Position vector | m |
| $\mathbf{T}$ | Thrust force | N |
| $T$ | Orbital time period | s |
| $U$ | Gravitational potential | $\mathrm{m^2/s^2}$ |
| $\mathbf{V}$ | Velocity of body, mostly used with ConGAL | m/s |
| $\mathbf{v}$ | Velocity of a body | m/s |
| $\mathbf{W}$ | Original decision variable matrix | - |
| $\mathbf{x}$ | General state vector | - |
| $\tilde{\mathbf{x}}$ | New state vector for MOSEK | - |
| $\underline{\mathbf{x}}$ | Fictional state vector for MOSEK | - |
| | **Greek letters** | |
| $\alpha$ | Approch cone half-angle for docking | rad |
| $\chi$ | Velocity heading angle in spherical coordinates | rad |
| $\eta$ | Slack control variable for thrust norm | – |
| $\gamma$ | Flight path angle in spherical coordinates | rad |
| $\Omega$ | Right ascension of ascending node | rad |
| $\omega$ | Argument of periapsis | rad |
| $\phi$ | Azimuth angle in spherical coordinates | rad |
| $\theta$ | True anomaly | rad |
| $\theta$ | Latitude angle in spherical coordinates | rad |
| $\theta$ | Minimum plume impingement angle | rad |
| | **Math characters** | |

| Term | Description | Units |
|---|---|---|
| $\mathcal{C}$ | Cone | - |
| $\mathcal{D}$ | Domain of an optimization problem | - |
| **dom** | Domain of a fucntion | - |
| **epi** | Epigraph of a function | - |
| $\mathbb{R}$ | Set of real numbers | - |
| | **Frame indices** | |
| $\mathcal{B}$ | Body-centred frame | - |
| $\mathcal{G}$ | Body-fixed geometric frame | - |
| $\mathcal{I}$ | Earth-centred-inertial frame | - |
| $\mathcal{T}$ | Frame associated with the target | - |
| $\mathcal{V}$ | Body-centred local vertical local horizontal frame | - |
| | **Indices** | |
| 0 | Initial state/value | - |
| $a$ | Pertaining to apocentre of orbit | - |
| $E$ | Pertaining to Earth | - |
| $f$ | Final state/value | - |
| $g$ | Pertaining to gravity | - |
| $p$ | Pertaining to pericentre of orbit | - |
| $p$ | Pertaining to precessing motion of docking axis | - |
| $r$ | Pertaining to rotational motion of docking axis | - |
| $s$ | Pertaining to spinning motion of docking axis | - |
| $\intercal$ | Transpose of a matrix | - |

# 1

# INTRODUCTION

Autonomous rendezvous, proximity operations and docking is considered as a key enabling technology for a diverse range of future space missions [NRC, 2012]. Almost all mission architectures for planetary exploration, in-orbit servicing, distributed space systems and active space debris removal involve rendezvous and capture. To meet the goals of affordability, safety and sustainability, it is required that such critical capabilities becomes routine and autonomous [Zimpfer et al., 2005].

The entire rendezvous mission consists of several phases, each of which has different trajectory requirements, navigation sensors, operational considerations and nature of manoeuvres. This study is focussed on autonomous[1] trajectory planning for rendezvous missions, commencing from far-range rendezvous phase and leading upto a point just before docking/physical contact. Guidance for rendezvous missions poses a great technical challenge, because of the non-linear trajectory dynamics, stringent constraints of various types and high-precision requirement. Most of the past and current technology for guidance is based on the Clohessy-Wiltshire (CW) equations (more about this is covered in Chapter 3). They are a set of linear, time-invariant differential equations that provide a convenient means to analyse and visualize the relative motion between spacecraft [Clohessy and Wiltshire, 1960]. However, an underlying assumption is that the target spacecraft is in a circular orbit and the two spacecraft are close to each other (so that the linearization is accurate). Thus, it is restrictive in terms of the applications it can be used for. Even small eccentricities in orbit could make it hard to use the CW equations. Although, linearized, time-varying forms of these equations are available for relative motion in non-circular orbits (as discussed by Carter and Humi [1987]), they make the design of rendezvous manoeuvres and control of final approach trajectory much more challenging and less intuitive. One other common problem of CW-based methods is that the user has to specify the number of (impulsive) burns needed during the mission. Often this information is unavailable and/or subject to further analysis, thus making it an iterative process for the user. Realistic final approach and docking also involves a number of trajectory constraints, like those on docking-port alignment, approach-cone angle, acceptable velocity at specific hold points, plume impingement, keep-out-sphere, etc.

All of the above complexities render the problems of guidance a laborious trial-and-error endeavour. Each mission scenario requires substantial (re)planning, which costs effort, money and time. Despite this, there is no guarantee that the planned trajectory will be optimal and robust. To the best of the author's knowledge, currently no single guidance algorithm for autonomous rendezvous handles all these requirements without making substantial simplifying assumptions.

[1] By autonomous, it is implied that the planning of the trajectory and generation of guidance commands are performed on-board without crew intervention or ground support (but with crew/ground monitoring whenever possible)

Recently, Liu [2013] developed a Second-Order Conic Program (SOCP)$^2$-based method that allows numerous constraints of a rendezvous-mission problem to be incorporated in its formulation. It was inspired by the recent remarkable success of SOCP-based methodology in powered planetary landing guidance seen in work by Acikmese and Ploen [2007], Blackmore et al. [2010] and Acikmese et al. [2013]. Using this method, the rendezvous-mission problem is posed as a non-linear optimal-control problem, subject to various state and control inequality constraints on interior points and terminal conditions. This is a marked difference from other traditional methods, which are mostly based on linearized dynamics of relative motion. Some numerical examples of the method's capabilities are presented in the paper by Lu and Liu [2013]. The authors claim that this method is applicable to rendezvous mission problems in any orbit, requires no externally supplied initial guesses, enjoys high reliability of finding an optimal solution and has potential for providing closed loop guidance. For further use throughout this literature study, and owing to the lack of a clear name provided by its original author, we shall term this method as Convex Guidance Algorithm by Liu (ConGAL).

## 1.1 RESEARCH PROBLEM

In the near future, missions for active space debris removal could greatly benefit by using a guidance system based on ConGAL method. Most of the current targets for such missions were not designed for rendezvous and capture – this asks for greater caution in planning and increased reliability in executing the missions. Such missions impose strict safety zones, approach-trajectory corridors and hold points along the way. During inspection, it is possible that the approach strategy has to be modified to accommodate any departure from original assumptions. In addition, the time-frame of these sequence of operations may be really small, making man-in-loop processes unsuitable.

A prime candidate for such debris-removal missions is Envisat, an Earth-observation satellite, that was launched in 2002 by European Space Agency (ESA), and has been defunct (and uncontrollable) since 2012. Due to its bulky nature and current orbit, it poses an extremely high risk of collision. In this regard, ESA has undertaken the e.Deorbit study to assess the debris problem and investigate measures to solve it. Such a mission has never been carried out before and therefore represents the current state of the art in the field of debris removal.

With the above background, a research problem has been identified. The guidance problem of the e.Deorbit mission is beyond the capabilities of any of the previously used guidance algorithms. The ConGAL method may prove to be a helpful option for autonomous trajectory planning. This thesis tries to answer the following research question:

*Can the challenges of the highly constrained Envisat rendezvous mission be handled by a convex optimization based guidance algorithm in a robust, optimal and autonomous manner?*

As mentioned, the focus is on the guidance system – trajectory planning, generation of guidance commands and trajectory tracking for the Envisat mission. In the process, ConGAL method's capabilities are assessed and suitable improvements have been made to cope with the Envisat guidance problem.

## 1.2 OUTLINE OF THE REPORT

This thesis is organized as follows: Chapter 2 consists of systematically identifying the need and challenges of guidance for autonomous rendezvous in the context of space debris removal. It is followed by a survey of relevant past and future missions, to help assess the state of the art. It concludes by defining the debris-removal mission and its scope. Chapter 3 discusses relevant topics in

[2] A standard SOCP problem is a convex optimization problem with a linear cost function, equality constraints and second-order conic inequality constraints [Boyd and Vandenberghe, 2004].

orbital mechanics that are used in this study. This includes reference frames, state vector representations, frame transformations, relative orbital motion and force models. Following this, a primer course on convex theory and optimization is presented in Chapter 4. It defines convex sets and functions, illustrates some benefits of using convex optimization and provides ways to formulate an optimization problem.

Chapter 5 gives a detailed description of the ConGAL method, to help understand the underlying principle. Using this, a ConGAL based algorithm is developed and tested, details of which are covered in Chapter 6. Following which a thorough numerical analysis of ConGAL is presented in Chapter 7. It identifies capabilities and limitations of ConGAL, and suggests improvements. Finally, using our modified version of ConGAL, the Envisat mission is modeled and numerous mission cases are solved in Chapter 8. Chapter 9 presents the conclusions and recommendations from the work carried out in this thesis.

In this report, vectors are written in bold font and matrices are written in capital bold font. For example: vector **r** and matrix **A**.

# 2

# RESEARCH CONTEXT AND MISSION HERITAGE

This chapter provides the research context for defining the thesis project. It highlights the problem of space debris, identifies the challenges of active debris removal, and justifies the need for autonomous guidance systems for such rendezvous missions. This is accompanied by a survey of relevant past and future missions. In the end, based on the e.Deorbit mission, a design case for active debris removal is defined.

## 2.1 ACTIVE DEBRIS REMOVAL AND CHALLENGES

In the last ten years, on an average, 100 satellites were launched every year as noted by Flores-Abad et al. [2014]. While a majority of these satellites from the past decade are still functional, some satellites (which have experienced varying degree of failures) and a large number of spent upper stages continue to orbit without any real use or function. Within another decade, almost all of the currently functional satellites will be decommissioned (due to failures, costs of operations, limited on-board consumables, etc.). If not disposed suitably, these shall continue to remain in orbit and add to the clutter of space junk. This accelerating growth in the number of orbital objects, calls for solutions to keep free operational space in Geosynchronous Earth Orbit (GEO) and Low Earth Orbit (LEO), as well as to avoid endangering of space systems in LEO.

Based on real data until 2007, Liou et al. [2010] estimated that that even if all future launches were withheld, the amount of space debris shall continue to increase for the next 200 years, due to the accidental collision among the existing objects in space. In reality, the orbital debris situation will be worse because satellite launches will continue and thus will provide more opportunities for such accidental collisions to continue. Such a side-effect of the past 50 years of satellite launches, was seen in 2009, when an accidental collision occurred between *Iridium 33*, an operational satellite launched in 1997, and *Kosmos 2251*, a satellite deactivated since 1995 and considered as space debris,. Both the satellites got destroyed and the resulting fragmentation created over 1300 pieces of debris of size > 10 cm [Tan et al., 2013].

If nominal launches continue, and the space debris problem is un-mitigated, then the number of debris objects is expected to rise at an exponential rate. The expected number of objects > 10 cm, in the coming 200 years is shown in Figure 2.1. This estimate was made based on 2010 data and the current debris population has already exceeded the projected values for 2014.

Clearly, mitigating the space debris problem is a matter of prime concern. The Inter-Agency Space Debris Coordination Committee (IADC) suggests Post Mission Disposal (PMD) as a necessary means to curtail orbital longevity of spacecrafts [IADC, 2007]. In LEO, this means de-orbiting and in GEO, this refers to raising the object to graveyard orbit. The most widely accepted guideline is often

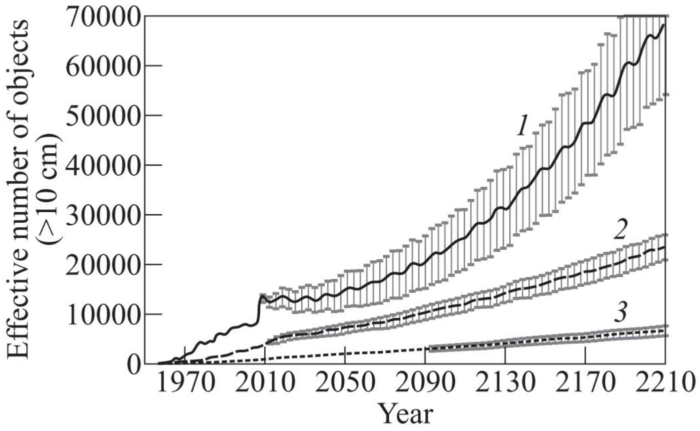


Figure 2.1: Projected growth of > 10 cm debris population for the next 200 years. The simulations assumed nominal launches and no mitigation measures. (1) - LEO: 200-2000 km, (2) - MEO: 2000-35,586 km, (3) - GEO: 35,586-35,986 km [Liou, 2013].

referred to as 25 year PMD rule, wherein a spacecraft in the LEO region should be disposed within 25 years of mission completion. While such attempts for passive debris removal will slow down the growth of future debris populations, it will still be insufficient to constraint the space debris population. Majority of existing satellites (active or not) do not have the capability for 25 year PMD and thus, will remain in orbit for much longer.

On the other hand, post mission disposal through external means, also known as Active Debris Removal (ADR) shows a lot more promise than the other measures. It involves use of a dedicated spacecraft to help dispose the debris object. Work done by Liou et al. [2010] shows that even if only five objects are removed each year through ADR, it is possible to stabilize the space debris population. The two key elements for successful and effective ADR is to focus on critical regions (specifically LEO around 600 km, 800 km and 1000 km) and to target most hazardous debris (*i.e.,* large pieces in critical region). A prime candidate for ADR will thus be Envisat, a satellite that was launched by ESA in 2002 and has been inactive since 2012. It did not have the capability for complying to the 25 year PMD rule. Instead, it is expected to remain in orbit for the next 150 years and due to its bulkiness (almost 8 tons heavy and 26 m long) is very likely to contribute a large amount of debris. It is also in a relatively dense orbit (polar Sun-synchronous at approximately 760 km altitude).

A general ADR concept involves a *servicing spacecraft* (or a chaser) which comes in proximity of the *target* and initiates a set of operations for the target's disposal. Numerous concepts have been developed, which vary in terms of capture mechanisms, proximity distance, means of disposal, type of re-entry, type of target orbits, etc. A study of these was carried out during preliminary analysis and it was concluded that one of the hardest tasks of end-to-end ADR missions is that of rendezvous (and docking) [Bhagat, 2015].

While there is some heritage with rendezvous (along with proximity operations and docking), it is limited to specific missions involving cooperative targets and substantial ground/crew support. Unlike typical rendezvous missions, the expected target is not designed for rendezvous and capture. The target will neither have active sensors for absolute navigation, nor suitable aiding systems for relative navigation, nor any interface for capture and physical connection. It is also possible that the satellite is tumbling. Presence of large appendages like solar arrays, antennas make the system fragile. Further precautions have to be considered, if the satellite is unpassivated, meaning that it still has energy sources (like unused propellant, charged battery) stored inside it. These challenges impose safety zones, approach-trajectory corridors and hold points along the way. Since the object has been in space for a few years, its attitude rate, mass and inertia properties, etc., may be poorly estimated until the chaser is relatively close to it. The target may have to be inspected (for getting a better attitude-rate estimate, for identifying damages, etc.) leading to special manoeuvres around the target. There is a chance that the approach strategy has to be modified after inspection of the target. Further, the time-frame of these sequence of operations may be really small, thus calling upon the requirement of automation. All these factors set challenging requirements on mobility, precision and reliability during proximity operations, because an error could cause collision.

Further, such missions are often constrained heavily by communication and illumination constraints, which can vary significantly should there be any change in mission plan, launch status, etc. ADR missions will typically cater to a variety of targets in different environments. Thus, the rendezvous capability should not be designed specifically for one target and orbit, but must be flexible and adaptable. Otherwise, each mission will require elaborate planning. This increases costs and efforts substantially, as shall be seen in Section 2.2.

Realizing the fore-mentioned challenges, one can now identify that a key-enabling technology for ADR missions it to be able to perform Rendezvous (and docking) *autonomously*. The need for this key-enabler is discussed and supported in greater detail by Polites [1998], NRC [2012], Woffinden and Geller [2007], Zimpfer et al. [2005], Wertz and Bell [2003], Kerambrun et al. [2008], Fehse [2003] and references therein. It is similar to an 'automated' system because it carries out the task without manual control. In addition to that, it allows for on-board decision making such that its mission objective(s) is met, even during contingency operations.

## 2.2 RENDEZVOUS MISSION HERITAGE

Most of the world's data and experience in rendezvous and docking, comes from the American and the Soviet programs. By the end of the 20th century, they had significant capabilities in this domain, but the paths undertaken differed distinctly, owing to the difference in design philosophies. The early American programs relied heavily on crew and ground support to carry our their rendezvous and docking operations, while the Russians took up to automating the process from the very beginning. However, it was not until the materialization of the Japanese and European programs, that the process showed (semi-)autonomy. The following paragraphs describe in brief detail about the growth of rendezvous and docking capabilities through different programs.

### 2.2.1 EARLY AMERICAN PROGRAMS

#### GEMINI PROGRAM

Some of the earliest rendezvous studies were performed by the Americans, and by 1961 they had concluded that rendezvous was technically feasible. In this regard, several experiments were conducted during the Mercury project, when astronauts tried to spot targets, judge distances and estimate attitude using naked eye. Results from these where used for the planning of the Gemini program, which eventually demonstrated the first rendezvous and docking capability. In 1965, Gemini 6 and 7, were manoeuvred within 30 cm of each other, while in 1966 Gemini 8 docked with an unmanned Agena target vehicle. These served as technology demonstrators and provided an experience base for the Apollo program. The Gemini programs performed many 'firsts' for the USA and the world. The technical challenge associated with it included closed loop execution of manual piloting techniques, planning for nominal and off nominal scenarios, selecting launch windows that reduces out-of-plane insertion error and development of contingency procedures. The following paragraphs are based on information from Lunney [1967] and Goodman [2011].

The Gemini program used ground based orbit determination of both chaser and target for computation of chaser orbital adjustments. However, owing to the limitations in ground tracking accuracy, closed-loop rendezvous techniques were developed for use in the terminal phase. A rendezvous radar and associated transponder on the target spacecraft provided range, range rate and line-of-sight angles. In the terminal phase, the on-board computer computed manoeuvre solutions (using the Clohessy-Wiltshire equations) once radar data was available. The docking mechanism consisted of cone and latch hardware on the target to capture 3 fittings on the nose of Gemini spacecraft. It provided high reliability and allowed for a short development schedule.

For Gemini 6, three mission plans had been identified and reviewed. These are discussed below:

1. Tangential orbit:
   This involved launching the Gemini spacecraft into an elliptical orbit, tangential to the target vehicle orbit. Figure 2.2 (top) illustrates this scenario. Rendezvous would occur near apogee of fourth orbit. However, the method did not guarantee proper lighting conditions or consistent relative dynamics in the terminal phase.

2. Coelliptic orbit:
   Rather than intercepting the target orbit from the beginning, this method places the Gemini spacecraft in coelliptic orbit with respect to the target orbit. The intercept manoeuvre is executed once a trajectory criterion (like sufficient lighting during terminal phase, adequate coverage by ground tracking, etc) is met. The central image in Figure 2.2 illustrates the method.

3. Direct rendezvous:
   The launch vehicle (in this case, the Titan II booster) would place the Gemini spacecraft on an intercept trajectory with the target. This restricts the launch window and makes the final state highly sensitive to ascent trajectory deviations. The interception and rendezvous occurs within the first orbit around Earth, hence greatly reducing the amount of time available for on-orbit system check (commissioning phase) and to perform the rendezvous procedure.

Mission plan 2 was chosen for Gemini 6 because it allowed considerable time and control over the rendezvous procedure. The above 3 profiles serve to be the basic plan for most rendezvous missions ever since.

**Apollo program:** Information presented below is based on references [Alexander and Young, 1970] and [Goodman, 2011]. Apollo was the first program to require rendezvous and docking capabilities to meet its operative mission. Following the lunar ascent profile, the Lunar Module (LM) ascent stage performed rendezvous and docking with the Command Service Module (CSM). Mission planning and piloting techniques were essentially those developed and proven during Gemini program. However, Apollo required new rendezvous profile concepts before terminal phase initiation. Also, since safety was a priority, the Apollo program also saw rigorous development of contingency plans. For example, if for some reason the LM could not perform the rendezvous manoeuvres, then the CSM could become the active vehicle and perform rendezvous. As a result, both the vehicles possessed relative navigation sensors, software to process measurements and software for targeting burns. Regardless of being active or passive, both vehicles performed relative navigation and burn targeting functions were exercised. The data and status could be shared (via mission control) for performance monitoring and redundancy. The ranging limit for both the vehicles was about 600 km. Kalman filters were used for relative measurement processing and state vector updates. Both vehicles could compute the manoeuvres foe either vehicles. They made use of a Lambert algorithm for the rendezvous profile. After the Apollo 11 flight, it was found that the CSM pilot has substantial work to perform during rendezvous, and thus, some parts of the process were automated (as observed from Apollo 15 onwards). Numerous rendezvous where considered, however, only two were used - coelleptical (similar to what was used in Gemini 6 and 7) and a short rendezvous profile, that saved two hours. The short profile was a precisely timed orbit insertion that provided a terminal phase initiation (TPI) burn relative position with appropriate terminal phase lighting. Figure 2.3 and Figure 2.4 illustrate the mission profiles.

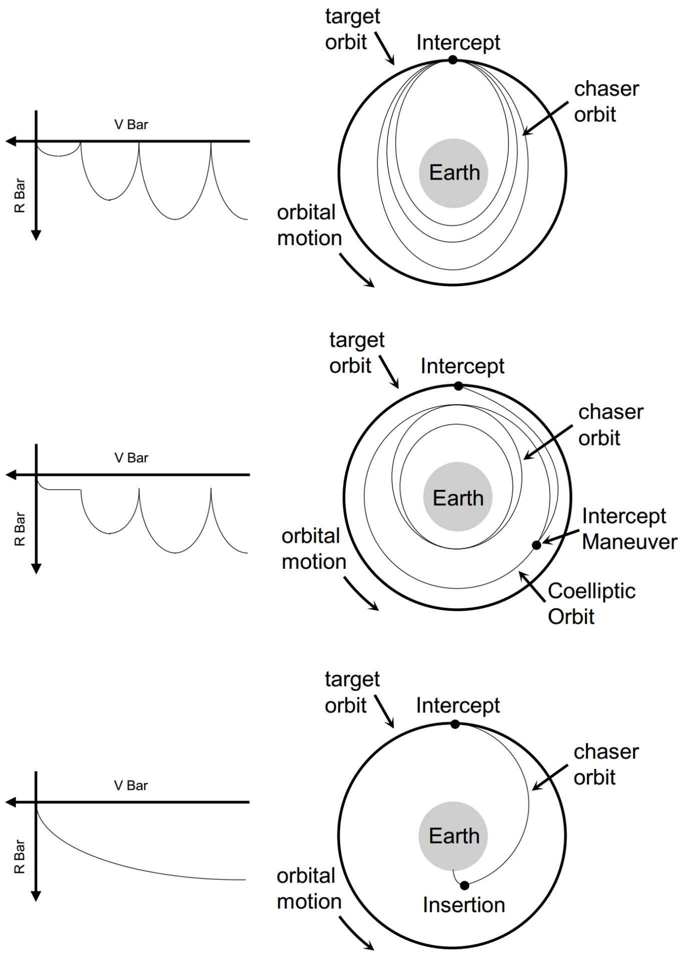


Figure 2.2: Gemini 6 mission profiles. Top: Mission plan 1, Tangential orbit; Centre: Mission plan 2, Coelleptical orbit; Bottom: Mission plan 3, Direct Rendezvous [Goodman, 2011]

### SKYLAB, APOLLO/SOYUZ DOCKING PROGRAM

Between the Apollo program and the Space Shuttle, 2 other programs requires rendezvous, but they did not improve the capability substantially - Skylab (1973-74) and Apollo/Soyuz (1975) docking. Hence, further description is skipped. Hardware and software, for both these programs were almost entirely based on previous programs.

### SPACE SHUTTLE

From June 1983, to July 2011, at least 78 missions of Space Shuttle had atleast one rendezvous and proximity operations objective. In many ways, Space Shuttle rendezvous and docking operations represented a significant departure from the Gemini and Apollo programs. Rendezvous was considered a secondary service while payload deployment was primary service. Most rendezvous targets did not possess active navigation aids (transponders or lights), and not many of them were designed to support rendezvous, retrieval and on-orbit servicing. Relative chaser and target spacecraft sizes were significantly different. Barring Mir and International Space Station, the other targets where always smaller than the Space Shuttle. Whereas, in the Apollo and Gemini programs, the target and chaser were relatively the same size. Further, the the relative velocity during docking for Gemini and Apollo, was of the order 0.3 m/s. For Space Shuttle, it was one order smaller while docking with Mir and International Space Station (ISS). There were tighter tolerances on time of docking and contact velocity. Gemini and Apollo crew had the advantage of docking alone line-of-sight, however, for

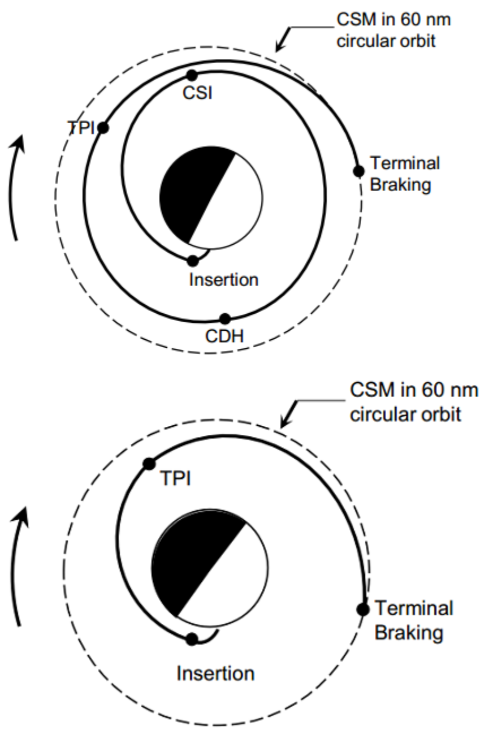


Figure 2.3: Apollo Program mission profiles. Top: Coelleptical orbit rendezvous; Bottom: Short Rendezvous. CSI: Coelliptical Sequence Inititation; TPI: Terminal Phase Initiation [Goodman, 2011].

Space Shuttle, the crew made use of cameras. The main references for the Space Shuttle program are Goodman [2006] and Goodman [2011].

Clearly, the complexity of the Space Shuttle missions was higher than Gemini and Apollo programs. These challenges were only gradually understood as the 1970s proceeded after much of the Space Shuttle hardware design was complete. Thus, despite being a more capable and flexible spacecraft than either of Gemini and Apollo, these challengers required extensive mission specific procedure and trajectory development. This prohibited the process from becoming routine as the solutions to these challenges involved crew and ground support procedural workarounds. New piloting techniques and procedures were preferred over hardware and software development, to meet cost and schedule constraints. The trajectory profile varied significantly for different missions, but some common practices existed. For example, when on a mission to the ISS, the space shuttle took upto 3 days from orbit insertion to docking, while requiring as many as 4 astronauts to carry out the task. The process consisted of 3 parts: far-field, mid-field and proximity operations. During the far-field stage, the space shuttle used internal IMUs and star trackers for computing its position. This was supported by extensive ground based radar updates. Solutions for burn manoeuvres (generally, twice a day) were computed by the ground crew and uplinked to the shuttle. The on-board crew set up the attitude and configured the propulsion system for ignition. The burn itself is executed automatically by the space shuttle. For the mid-field stage, the on-board crew initiated targeting routines. Using a set of sensors, on-board navigation was possible. All of this was manually coordinated with a pre-printed timeline, anchored to major events. In the final stage, the translatory motion of the space shuttle is controlled manually. Only the attitude control was automated. Based on video camera feed and laser range measurements, relative position and velocities were available to the on-board crew. 4 astronauts, with months of training together, were required to dock the space shuttle to the ISS. The space shuttle used Lambert targeting for all rendezvous missions. A

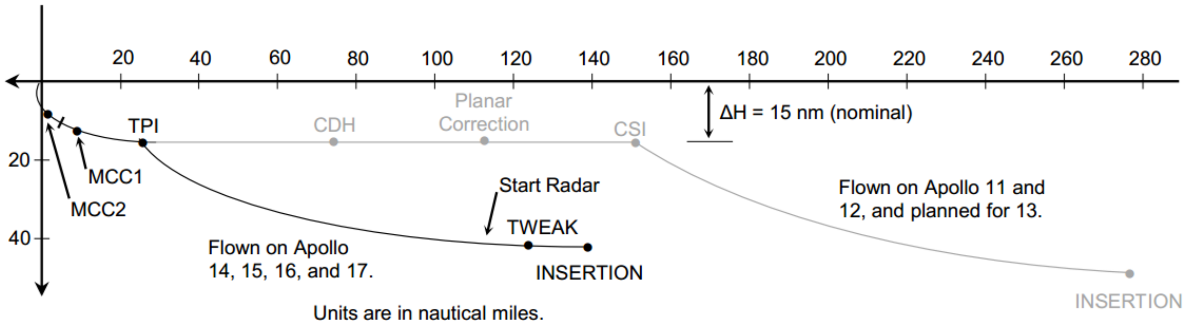


Figure 2.4: Apollo program: Comparision of coelliptic (grey line) and short rendezvous (dark line) profiles in local-vertical and local-horizontal frame. CSI is Coellipticsl Sequence Inititation, CDH is Constant Delta Height, MCC is Mid Course Correction and TPI is Terminal Phase Initiation [Goodman, 2011].

typical +R bar approach to Mir / ISS is shown in Figure 2.5.

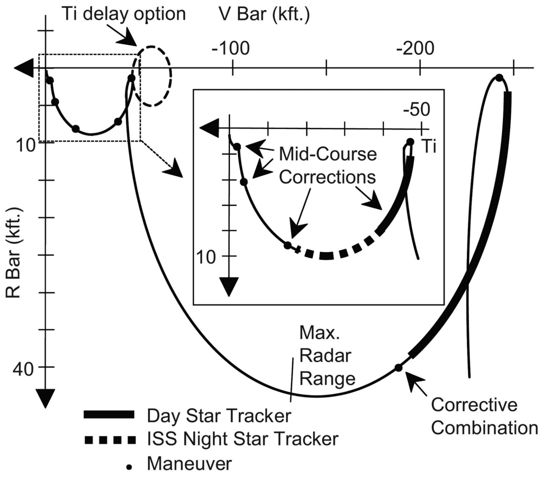


Figure 2.5: Space Shuttle: Typical profile for rendezvous with Mir / ISS using +R bar approach [Goodman, 2006].

### 2.2.2 SOVIET / RUSSIAN PROGRAMS

In the long run, the US approach has been inherently more labour intensive, expensive and required extensive training of crew and ground support. On the other hand, the Soviet program went on to standardize and automate its rendezvous and docking operations. The systems were primarily automated and the crew only had the function to monitor and if necessary, over-ride the controls. In 1967, *Cosmos 186* and *Cosmos 188* performed an unmanned automated rendezvous and docking operation. Two years later, the *Soyuz 4* and *Soyuz 5* spacecraft performed an automated manned docking. Most rendezvous and docking operations of Progress and Soyuz to the Salyut, Mir and ISS space stations, were automated as well.

Earlier, the IGLA (Russian for needle) radio telemetry system was used for docking of active spacecraft (mostly Soyuz) with Soyuz and / or Salyut space station. In 1986, it was replaced by the more powerful and accurate Kurs system, which is used on-board Soyuz, Progress, Mir and ISS. It was designed to provide all required navigation measurements during the entire approach - from hundreds of kilometres, all the way upto final contact. It is an example of the combination of various RF-sensor principles into one navigation system. It determines range, range-rates, line-of-sight angles, and relative attitudes. The docking system consisted of a probe/drogue mechanism, slightly different from the US programs. The mission profile generally involved, near circular orbits which are raised at appropriate times. This made it easier to plan and schedule the manoeuvres. A standard phasing manoeuvres consisted of three sets of one to three boosts, taking into account the location and communication window constraints. This is shown in Figure 2.6. The main references for this are Woffinden [2008] and Fehse [2003]

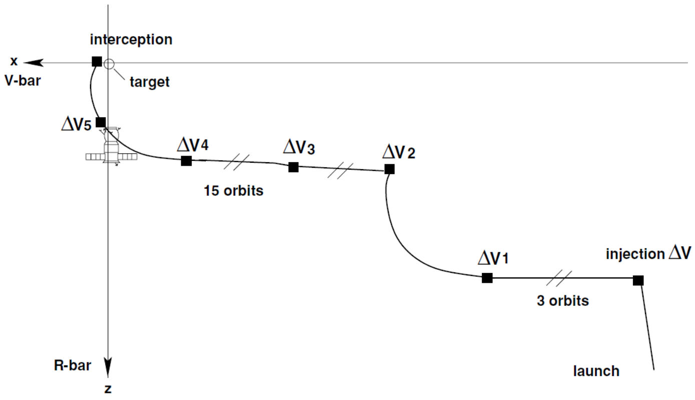


Figure 2.6: Soyuz / Progress program: Typical mission plan, V-bar approach [Fehse, 2003].

### 2.2.3 EXPERIMENTAL TEST SATELLITE

Although, the Soviet space program went on to automate the process, it did not make the rendezvous procedure autonomous. The very first capability to perform autonomous rendezvous and capture was demonstrated by the Japanese Experiment Test Satellite (ETS) VII in 1997. The satellite had two parts - *Orihime* (the chaser) and *Hikoboshi* (the target), which were separated once in orbit, to carry out rendezvous and capture experiments. The chaser was the main satellite body and it was equipped with a 6-Degree of Freedom (DOF) robotic arm, to capture the target. The ETS system used relative Global Positioning System (GPS) for navigation. The guidance laws were based on Clohessy-Wiltshire equations. The first experiment involved a station-keeping manoeuvre at a distance of 2 m for about 15 minutes, followed by docking. During the second experiment, an error was encountered which triggered the abort command, making the chaser fly to a retreat point. Two further attempts were thwarted as well, and it was established by the ground crew that there was a thruster valve malfunctioning. Upon a system reconfiguration, the second rendezvous and docking experiment (from a distance of 500 m), was also successful. The main reference for this mission is Oda [2000]

### 2.2.4 OTHER ATTEMPTS IN THE US

#### XSS 10 AND 11

Between 2003 and 2007, USA had 4 missions to test and demonstrate autonomous rendezvous and docking / capture capability. The eXperimental Satellite System (XSS) series of microsatellites, developed by the US Air Force, was their first such attempt. In 2003, XSS 10 was launched, a 31 kg satellite launched as a secondary payload on a Delta II rocket. After getting ejected from the second stage of the rocket, the XSS-10 performed a semi-autonomous inspection manoeuvre of the Delta II second stage. The satellite had a visible camera, GPS navigation and a star sensor. It made use of relative state navigation and the state was propagated using Clohessy-Wiltshire Equations [Davis and Melanson, 2004].

Two years after the XSS 10 mission, its successor XSS 11 was launched. Its objective was to demonstrate increased autonomy during rendezvous and proximity operations. Its mass was thrice that of XSS 10, and it was expected to build upon what was learnt and tested during the XSS 10 mission. It finished 75 circumnavigation of its expended parent rocket stage. Later its orbit was raised to perform rendezvous and proximity operations with other US-owned decommissioned satellites.

Further details are unavailable in public domain.

#### DART

In 2005, NASA had the Demonstration of Autonomous Rendezvous Technology (DART) mission. Its objective was to demonstrate hardware and software necessary for autonomous rendezvous. The mission failed because of a navigation system error in the initial phase of the rendezvous, which resulted in incorrect thruster commands and excessive use of propellant. Also, the collision avoidance manoeuvre failed to initiate, because the spacecraft collided with its target. The DART spacecraft made use of GPS for navigation and an Advances Video Guidance Sensor (AVGS) for close range navigation [NASA, 2005].

#### ORBITAL EXPRESS

Adding to these missions, was the Defense Advances Research Projects Agency (DARPA) project Orbital Express, which was launched in 2007. Its goal was to validate technical feasibility of robotic, autonomous on-orbit refuelling and reconfiguration of satellites. The mission had two spacecraft, Astro (the servicer/chaser) and NextSat (the client/target). Astro performed station keeping at a distance of 120 m, followed by a few fly-arounds for inspecting the client. A second station keeping point (at 10 m) followed, immediately after which the servicer approached the client for capture. Multiple transfers or propellant and batteries were carried out successfully. The servicer had a 6-DOF manipulator arm on it, which helped carry out the operations. On-board the servicer was a star sensor, GPS receiver, lidar for long range navigation, and AVGS (also used in DART) from NASA Marshall Space Flight Center (MSFC). Capture was possible directly or with use of the robotic arm. Numerous world or US firsts in space were achieved during this mission, despite facing some major anomalies. It was the first mission to carry out rendezvous, proximity operations and capture with full autonomy [DARPA, 2007].

### 2.2.5 AUTOMATED TRANSFER VEHICLE

The European Automated Transfer Vehicle (ATV) program was active from 2008 to 2014, delivering five unmanned cargo vehicles to the ISS. It possessed some highly sophisticated automation techniques, and for far range rendezvous phase, it was also autonomous. It is also the largest vehicle ever to perform automatic rendezvous and docking. At far range, it used relative GPS for navigation, and for close range and docking, it used optical sensors and telegoniometry data. As a back up, the Soviet *Kurs* navigation system was also present onboard (however, it never got to be used). Docking mechanism is similar to that of the Progress vehicle. Special collision avoidance manoeuvres, on fault tolerant chains with dedicated computers and sensors, can be performed during the rendezvous and departure phases. During its years of operation, no major anomalies occurred. A typical mission plan is shown in Figure 2.7.

### 2.2.6 FUTURE MISSIONS

Numerous missions involving rendezvous and docking are currently under development. In-orbit satellite servicing, active debris removal, space exploration and space assembly, are possible applications which require these capabilities. Two major programs deserve our attention:

#### DEOS

DLR's DEOS mission, is a demonstration of an autonomous servicer for debris handling. It is being developed for handling passive, uncooperative and tumbling targets. All operations, inclusive of far rendezvous, close approach, proximity operations (like fly-around, inspections, formation flying), capture, stabilization and controlled de-orbit will be in (supervised) autonomous mode. The test flight consists of an active spacecraft, the servicer, and a target / client. Once in orbit, the stack shall separate to a distance of 2 km and rendezvous and berthing operations will be demonstrated.

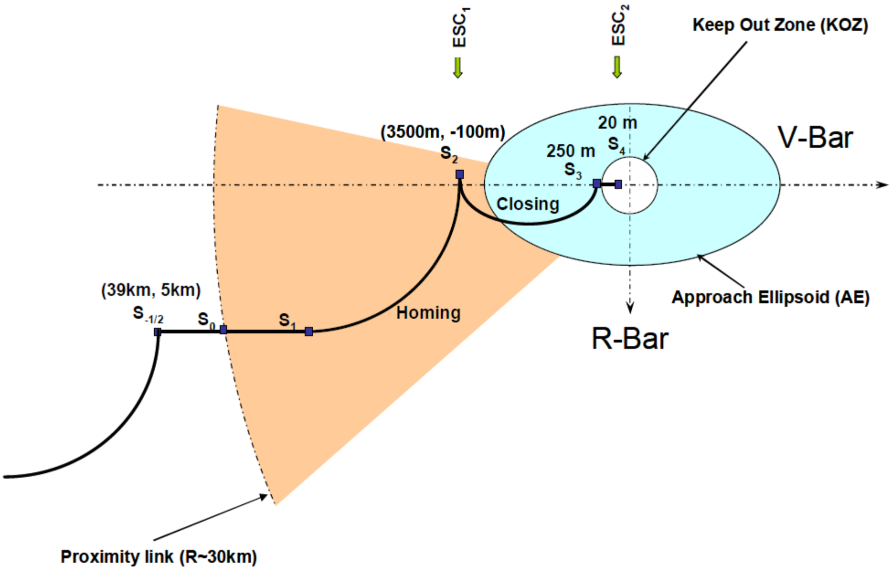


Figure 2.7: ATV Jules Verne: Mission manoeuvre plan for rendezvous and docking with the ISS [Labourdette et al., 2009].

The servicer makes use of GPS, star tracker, cameras and lidar for navigation. The satellites are scheduled to be launched in 2018 [Benninghoff et al., 2012].

### ORION MPCV PROGRAM

Orion Multi-Purpose Crew Vehicle (MPCV) will be a crewed spacecraft, designed to carry upto four astronauts [Ruiz and Hart, 2010]. Its applications range from missions to ISS, to crewed missions to Mars by docking to an additional *Deep Space Habitat* module. Safe and automated rendezvous and docking are critical requirements for the missions, whereas autonomy is desirable for long range missions. Orion shall benefit from improved technology, thus allowing greater onboard computation power, as compared to previous crewed spacecraft. During the far range rendezvous, it shall make use of three IMUs, two star trackers and four GPS antennas. Using GPS, Orion will be able to access a highly accurate navigation state. This improved accuracy and the added computer power, allows Orion to utilize manoeuvre targeting algorithms onboard. Orion will make use of auto-trim capability, which will have the Reaction Control System (RCS) to make up for any residual Delta-V after a programmed burn, to correct any burn execution errors. During mid range rendezvous, Orion shall make use of RF ranging system, star trackers and docking camera. Navigation data from these three will be compared for consistency, and will be forwarded to the manoeuvre targeting algorithms. For proximity operations and docking, Orion will be semi-autonomous as the system will be dynamically piloted. Orion will use a suite of sensors, called the Vision Navigation Sensor. The docking mechanism will be similar to the current systems on Soyuz and ISS, amongst other types.

Table 2.1 summarizes the information presented in the previous subsections. The missions / programs are enlisted chronologically. It is clear that majority of the rendezvous and docking missions have involved cooperative targets - a major difference from what is expected in ADR concepts. All the missions involved a nearly circular target orbit – this provided the option of basing guidance algorithms on Clohessy-Wiltshire equations. There is a clear trend that shows the extent of autonomy in the rendezvous process has been rising.

Table 2.1: Summary of rendezvous related parameters of some missions/programs

| Mission / Program | Target orbit | Target characteristics | Autonomy / support | Guidance algorithm / targeting principle | Comments |
|---|---|---|---|---|---|
| Gemini [a] | Circular | Cooperative and mostly active target | Man-in-loop, extensive support from ground | Based on Clohessy-Wiltshire equations for near-range; far range with Hohmann-like transfers | |
| Apollo [a] | Circular | Cooperative and active target | Man-in-loop, extensive support from ground | Based on Lambert targetting | |
| Soyuz / Progress missions [b] | Circular | Cooperative and mostly active target | Automated, ground and crew support when required | Based on Clohessy-Wiltshire equations for near-range; far range with Hohmann-like transfers [c] | |
| Space Shuttle program [a] | Circular | Mostly cooperative target; occasional passive targets | Far-range rendezvous automated, near-range required man-in-loop, (extensive support from ground) | Based on Lambert targeting | |
| ETS-VII [d] | Nearly Circular | Cooperative and active target | Autonomous | Clohessy-Wiltshire equations for near-range; far range with Hohmann-like transfers | |
| ATV [e] | Nearly Circular | Cooperative and active target | Highly automated with occasional ground support | Based on Clohessy-Wiltshire equations for near-range; far range with Hohmann-like transfers [c] | |
| XSS [f] | Nearly Circular | Cooperative but passive target | Semi-autonomous | Partly based on Clohessy Wiltshire equations; further information unavailable | |
| DART [g] | Nearly Circular | Cooperative but passive target | Autonomous | Information unavailable | Mission failed |
| Orbital Express [h] | Circular | Cooperative but passive target | Autonomous | Information unavailable | |
| DEOS [i] | Circular | Uncooperative target | Autonomous | Based on Clohessy-Wiltshire equations for near-range | Launch in 2018 |
| Orion program [j] | Nearly Circular | Cooperative and active target | Semi-autonomous | Information unavailable | Program starts after 2021 |

[a] Goodman [2011]; [b] Fehse [2003]; [c] Luo et al. [2014]; [d] Ohkami and Kawano [2003]; [e] Labourdette et al. [2009]; [f] Davis and Melanson [2004]; [g] NASA [2005]; [h] DARPA [2007]; [i] Benninghoff et al. [2012]; [j] Ruiz and Hart [2010]

## 2.3 ADR mission definition

As a design case for the thesis, it is decided to make use of ESA's e.Deorbit mission study, part of its CleanSpace initiative. The objective of this mission is to perform the active removal of a big ESA-owned object. Based on data from the e.Deorbit mission Phase A and Phase B studies by ESA [2012], ESA [2014], ESA [2015], and related work contributed by Deloo [2014], a mission scenario is defined.

### 2.3.1 e.Deorbit mission

The mission looks to remove a single, large, heavy (> 4000 kg), ESA-owned space debris from the LEO protected zone. The mission includes a chaser, that shall be launched by a small or medium launcher, rendezvous with the ESA-owned debris (the target), capture the target and remove it from the LEO protected zone. The mission aims to contribute to mitigation of risk posed by the space debris problem, and to demonstrate technologies for future ADR missions.

The mission study assessed that the target to be removed will be Envisat. The target is approximately 8 tonnes heavy, and has a large cross-section area of about 63 m$^2$. The reasons that encouraged the decision are summarized as follows [ESA, 2012]:

1. Envisat is a well-known ESA owned object, and documentation of the satellite is readily available in-house.

2. Envisat is located in a 'hot-spot' region. The SSO is a high-risk region, due to the large number of satellites present there. with available in-house

3. Envisat's platform (SPOT), is used for several other European satellites.

The model of Envisat, the chaser, and top-level requirements pertinent to this study, are discussed in the following subsections.

### 2.3.2 Envisat

Envisat, short for ENVIronmental SATellite, is an inoperative Earth-observing satellite, that was launched in 2002 by the European Space Agency. In April 2012, after ten years of service – twice the planned life span of Envisat – the mission ended, following the unexpected loss of all contact with the satellite.

#### Envisat model

Visual and radar observations of Envisat showed that the satellite is intact and has not suffered any mechanical or structural degradation. Figure 2.8 shows a computer graphic model of Envisat. It is clear that Envisat has three major appendages - the solar array, SAR antenna (visible in the top image) and Ka-band dish antenna (visible in the bottom image).

Some physical parameters of Envisat are listed in Table 2.2. We see that the End of Life (EOL) mass is a little short of 8000 kg, making it one of the heaviest civilian satellites in orbit. A body-fixed geometric reference frame (denoted by $\mathcal{TG}$, short for Target Geometric), originating at the centre of the launch adaptor, is used to define the CoM of the satellite (More about reference frames is discussed in Section 3.1). The $X$-axis points towards the solar Array, the $Y$-axis is parallel to to the SAR antenna but points away from the Ka-band dish antenna. The $Z$-axis completes the right-handed system.

#### Envisat orbit

In 2002, Envisat was launched into a Sun-synchronous orbit with an altitude of about 790km. Later in 2010, it was lowered to a slightly less crowded orbit of 768km. The orbit changes slightly over a period of time, hence for the study, it is decided to make use of an initial reference orbit. ESA [2014]

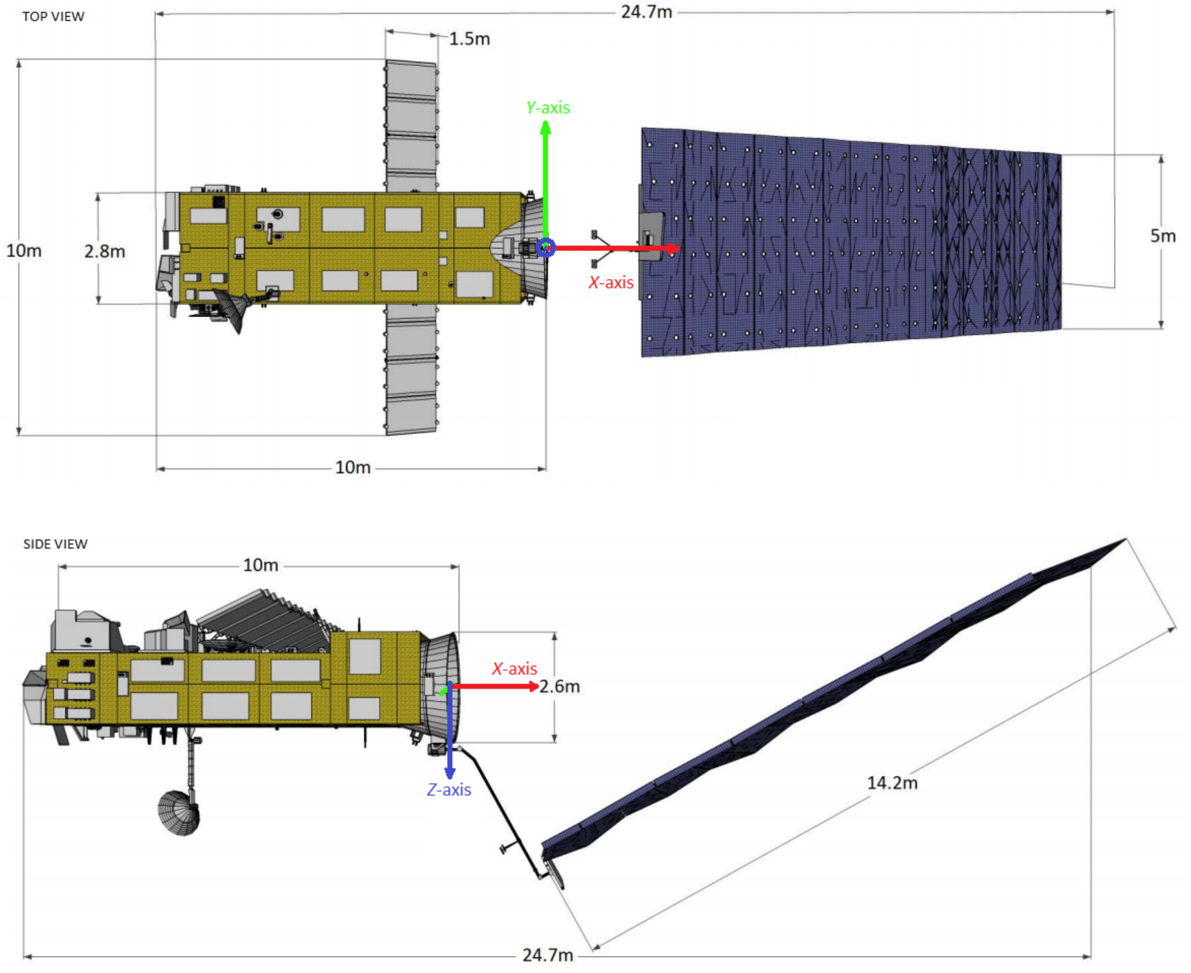


Figure 2.8: Envisat: Computer model showing top and side view of the satellite. A body-fixed geometric reference frame is shown [ESA, 2012].

defines the reference initial state of Envisat based on TLE data from 13 August 2014. The provided epoch, initial position and velocity of the propagation are given in Table 2.3.

Using this, a generic orbit is defined with a mean value of 760 km for altitude, 98.15° for inclination, 0.0015 for eccentricity. The suggested model for orbit propagation includes a Earth gravity model of order 30 (or better), third body perturbation from the Sun and Moon, atmospheric model of MSIS-77 (or better). Such a detailed orbit propagator might not improve overall accuracy, because the initial state uncertainty is comparatively large (Two Line Element (TLE) data uncertainty is 0.7 km ($1\sigma$) in along-track direction, 0.2 km ($\sigma$) in radial and cross-track directions). However, the influence of this uncertainty (on force models) is acceptable for the scope of this study.

### Envisat Attitude

Observations indicate that Envisat's attitude is changing and currently unstable. Virgili et al. [2014] shows there is little or no correlation between the measurements and results from simulations, thus indicating a micro-meteoroid impact and/or energy release from the non-passivated satellite has affected the rotational motion of the satellite. Radar measurements, performed in the end of 2013 revealed that the satellite is oriented with its body-fixed geometric frame $Z$-axis normal to the orbital plane. The main motion corresponds to about 3.5°/s rotation about the $Z$-axis. Additionally, Envisat is tumbling around other axes as well. There is uncertainty on the future evolution of En-

Table 2.2: Envisat physical parameters [ESA, 2014]

| Parameter | Value | Units |
|---|---|---|
| Launch mass | 8211 | kg |
| EOL mass | 7828 | kg |
| $x_{com}$ | -3.905 | m |
| $y_{xom}$ | -0.009 | m |
| $z_{com}$ | 0.003 | m |
| Cross-section area | 63.05 | $m^2$ |
| Drag coefficient | 2.2 | [-] |
| Reflexivity coefficient | 1.1 | [-] |

Table 2.3: Envisat initial state of propagation, provided in EME2000 reference frame [ESA, 2014]

| Parameter | Value | Units |
|---|---|---|
| Initial epoch | 2014-08-13 03:37:37.00 | [-] |
| | 5338.1511226 | MJD2000* |
| Initial position | [637.862,-4385.952,5596.390] | km |
| Initial velocity | [-2.54065, 5.39303, 4.50519] | km/s |

*MJD2000 is Modified Julian Day with epoch year 2000

visat's attitude and currently ESA is analysing it to predict future motion. For the study, 3 rotation scenarios are considered, as defined by Wieser [2014]. These are summarized in Table 2.4, using $\mathcal{TB}$ frame, which is the $\mathcal{TG}$ frame but its origin is shifted to the CoM of Envisat.

Table 2.4: Scenarios for Envisat attitude motion [Wieser, 2014]

| No. | Spin Axis | Reference Axis | Spin rate (°/s) | Enclosed Angle (°) | Precession rate (°/s) |
|---|---|---|---|---|---|
| 1 | $+Z$-axis of $\mathcal{TB}$ frame | Angular momentum | 5.0 | 0 | |
| 2 | | | 5.0 | 45 | 0.15 |
| 3 | | | 5.0 | 90 | 0.15 |

### ENVISAT AS A RENDEZVOUS TARGET

As a rendezvous target, Envisat is considered to be uncooperative. Its trajectory and attitude cannot be controlled, it does not provide any active information to the chaser (say, for relative navigation) and does not have any dedicated passive sensor aids on-board (like reflectors). Further, the satellite is unpassivated, and hence possesses a threat of explosion if collided with. It is concluded the batteries are mostly empty, and thus a break-up due to the power system is unlikely; however, a potential risk of break-up is posed by the propulsions system, since about 40kg of pressurized propellant still remains on-board.

### CLAMPING LOCATION ON ENVISAT

An ideal clamping location for the chaser should be rigid, easily accessible, allow for safe clamping (without damage to main structure, appendages; or create more debris). It should also align the

Centre of Mass (CoM) of the chaser with that of the target, to avoid any unnecessary torque produced during thrusting. Envisat's launch adaptor would make for an ideal clamping location, were it not for the solar array's blocking access to it. The alternative, is to clamp along one of the longer sides of Envisat. The face with an outward normal along $+Z$-axis of $\mathcal{TG}$ frame, is considered to be a suitable clamping location. There are no instruments, or appendages on that face along the $+Z$ direction.

### 2.3.3 CHASER

The chaser design is based on the rigid-link baseline chaser defined by ESA [2012] and also used by Deloo [2014]. The design foresees a boxed configuration with a robotic arm and tentacles, as shown in Figure 2.9. The design trade-off are described by ESA [2012] and references therein. The main propulsion system will consist of a bi-propellant system with four 425 N engines (2 active, 2 redundant), having an $I_{sp}$ of 320 s. In addition, 24 thrusters (12 active, 12 redundant) each delivering 22 N, will be used as RCS thrusters for attitude control and rendezvous and docking. These thrusters are fired in pairs, thus capable of delivering 44 N along any axes of the Chase reference frame.

An overview of the various subsystems is shown in Table 2.5. In Figure 2.10, one can see the chaser clamped to Envisat.

Table 2.5: Chaser spacecraft overview [ESA, 2012]

| System/Parameter | Characteristics | |
|---|---|---|
| Mass | Dry mass | 762 kg |
| | Propellant mass | 826 kg |
| AOCS/GNC | Attitude control | 3-axis stabilized |
| | Actuators | 4 × Reaction wheels |
| | RCS | 24 × Thrusters (22 N); $I_{sp} = 290$ s |
| Propulsion | Type | Chemical, bi-propellant system |
| | Engine | 4 × Thruster (425 N); $I_{sp} = 320$ s |
| Capture | Mechanism | Berthing: robotic arm<br>Clamping: tentacles |

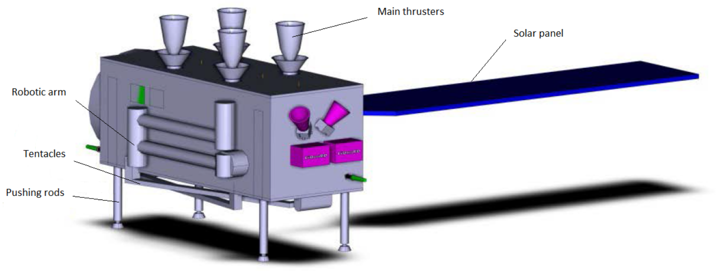


Figure 2.9: Chaser baseline design [ESA, 2012].

### 2.3.4 SCOPE OF MISSION SCENARIO

Figure 2.11, shows the complete ADR mission. The mission scenario considered for this study commences once the chaser satellite is released into a 300 km × 300 km × 98.15° orbit (the injection orbit) and its commissioning phase is completed. The scenario will terminate upon making contact with the target spacecraft. The launch and de-orbiting phase are not considered as part of the scenario. The chaser shall employ an autonomous guidance system. To simplify the analysis, it is assumed

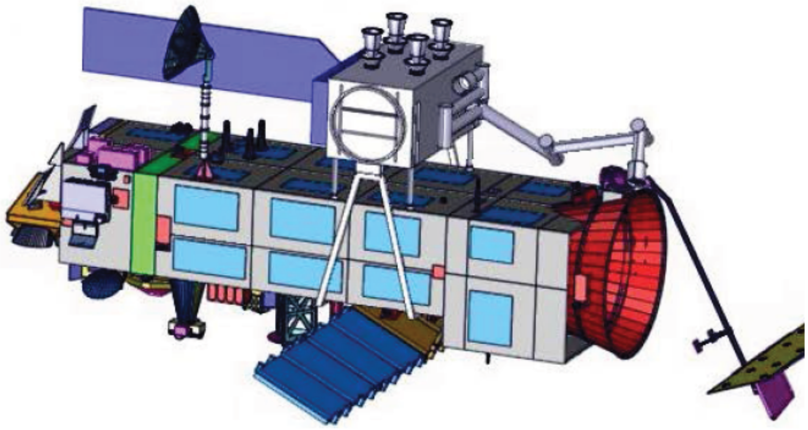


Figure 2.10: Chaser clamped to Envisat [ESA, 2012].

that any illumination and communication constraints are relaxed. Also, only the translation motion of the chaser spacecraft is considered.

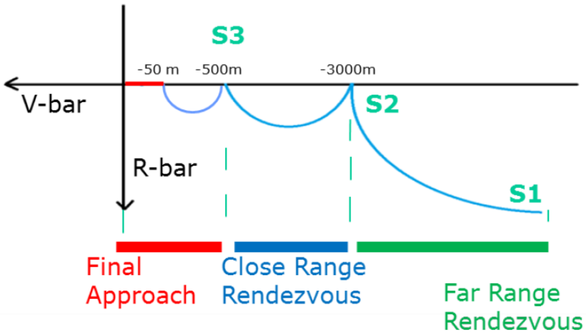


Figure 2.11: Rendezvous phases for e.Deorbit mission as defined in [ESA, 2012].

The study by [ESA, 2012] suggests a V-bar approach for the mission, and identifies three three major phases: far-range rendezvous, close-range rendezvous and final approach. Multiple hold points are deemed necessary for the rendezvous sequence, as it provided an opportunity to characterize the motion and state of Envisat, as well as allows a safe approach to the satellite. The phases are defined as follows:

1. Far-range rendezvous: This phase begins from point $S_1$ on the injection orbit, to point $S_2$ (-3000,0,0) m in the LVLH or $\mathcal{F}_{\mathcal{V}}$ frame (see Section 3.1.2 for more on reference frames). The exact location of $S_1$ is a design choice that needs to be made based on the results. Besides the initial and final state, and the underlying dynamics, the only constraint active in this phase is that on maximum thrust.

2. Close-range rendezvous: From point $S_2$ to $S_3$ (-500,0,0) m in $\mathcal{F}_{\mathcal{V}}$ frame. The chaser shall switch to its RCS system for the transfer. Here too, besides initial and final hold points and underlying dynamics, the only constraint active is that on maximum thrust.

3. Final approach: From $S_3$ to final docking. This phase is highly constrained with constraints on maximum thrust, approach cone, plume impingement, keep-out-sphere, forced motion, etc. The target point is defined to be at 3 m along the spin axis, where the relative chaser velocity should be zero.

# 3

# ORBITAL MECHANICS

Orbital mechanics, an offshoot of celestial mechanics, is the study of motion of orbital bodies, more generally artificial bodies. It consists of *kinematics*, which describes the motion without reference to forces, and *dynamics*, which studies the rules governing the interactions of bodies and helps determine the relationship between the motion and its causes. This chapter presents and discusses the orbital mechanics required to study the motion involved in rendezvous applications. Guidance algorithms are based on the dynamics of the system and are designed to optimally harness the systems' capabilities and constraints. Thus, a thorough understanding of the involved mechanics, is key to developing an efficient and effective guidance system.

Motion cannot be described without a reference frame, and thus this chapter begins by defining a few reference frames relevant to this study. It is followed by discussing some state representations, which describe the position and velocity of a body. Since rendezvous problem may make use of different types of reference frames, it is helpful to be able to convert a state from one frame to another. Hence, reference frame transformations are discussed next. Lastly, equations of motions and force models are described as well.

## 3.1 REFERENCE FRAMES

The motion of an object can be described unambiguously, only if there exists a *reference frame*, with respect to which the object's state varies with time. Reference frames are a physical concept that define the space related to the motion of a body. Often, they are also referred to as *Observational frame of references*. They can be (pseudo-)inertial or non-inertial, based on their own motion. A pseudo-intertial frame of reference is a non-accelerating frame of reference - in other words, it is either at rest or is under rectilinear motion, with respect to other inertial frames. It is hard to define an inertial frame of reference unambiguously in the scope of this report, hence for further information, it is recommended that the reader refer to Morin [2008] and references therein. A non-inertial frame is one, which is accelerating with reference to an inertial frame of reference. This includes frames that are rotating as well.

A reference frame is a set of axes that has a defined origin and relative orientation. It serves as a reference to coordinate system(s), which are mathematical concepts used to describe the motion of the object in that reference frame. An observer in an observational frame of reference can choose to employ any convenient coordinate system (for example, Cartesian, polar, spherical, curvilinear, etc.) to describe the observations made. Due to the unlikely usage of a reference frame without a coordinate system, reference frames are also (mis-)referred to as coordinate system or frame, in common usage. In this report, reference frames, by default will make use of a right-handed Cartesian coordinate system (further explained in section 3.2). An example of a Cartesian frame of reference is shown in Figure 3.1. The Cartesian system, a widely used coordinate system, consists of 3 intersecting, mutually orthogonal unit vectors. The point of intersection is the *origin* of the system. Along the three unit vectors, lie the reference axes of the system. They are generally paired as, $X$-axis with $\hat{\mathbf{i}}$, $Y$-axis with $\hat{\mathbf{j}}$ and $Z$-axis with $\hat{\mathbf{k}}$. An object's position (or its coordinates), as measured from the origin, are the the signed distances of the object's projection on the reference axes. As a convention, a general frame of reference will be denoted by $\mathcal{F}$.

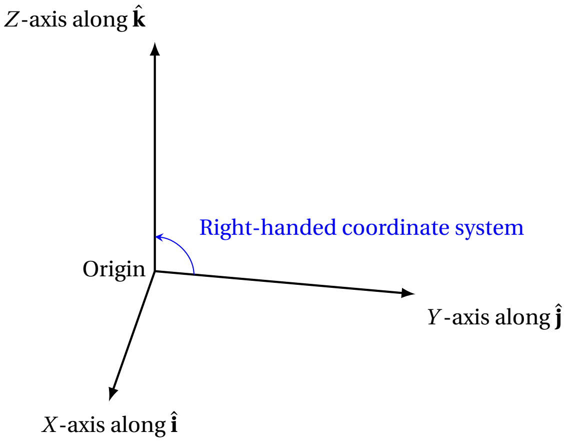

$Z$-axis along $\hat{\mathbf{k}}$
Right-handed coordinate system
Origin
$Y$-axis along $\hat{\mathbf{j}}$
$X$-axis along $\hat{\mathbf{i}}$

Figure 3.1: Example of a Cartesian frame of reference, denoted by $\mathcal{F}$. The three axes are orthogonal to each other.

The choice of a reference system depends on the application involved. An appropriately defined reference frame, makes the description of the involved motion simple and convenient. For instance, applications which involve helio-centric orbits, benefit from using the International Celestial Reference Frame (ICRF)[3], originating from the barycentre of our solar-system. On the other hand, an ill-defined reference frame, for example, a Venus-centred reference frame for a spacecraft in helio-centric orbit, might provide information about the state of the spacecraft, but it would be hard to comprehend and un-intuitive for the user. Further, the equations of motion would be unnecessarily complex.

The following subsections define the relevant reference frames for use in the rendezvous problem. We shall begin the list with Earth-centred reference frames. It is followed by spacecraft body-centred reference frames, to describe spacecraft attitude, motion between the target and spacecraft, etc. The descriptions are based on the author's understanding and standard definitions found in Fehse [2003] and Mooij [2013], unless stated otherwise.

### 3.1.1 EARTH-CENTRED FRAMES

These reference frames have the origin at the geometric centre of Earth. These make excellent choices for describing motion of satellites in orbit around Earth. Based on the intended application, two reference frames can be considered:

[3] The ICRF is the best estimate of an inertial frame of reference [Vallado, 2001]. Its axis are precisely defined using extra-galactic radio sources like quasars. Thus, other inertial frame of reference in our solar system, can defined with respect to ICRF. For example, the axes orientation of the EME2000, a commonly used Earth-centred inertial frame, matches almost perfectly with that of the ICRF. The difference in the orientation of the two frames is less than 0.1", which for most practical purposes, can be neglected.

#### Earth-centred inertial frame, index $\mathcal{I}$

A very commonly used Earth-Centred Inertial (ECI) frame[4] is the Earth Mean Equator and Equinox frame (commonly referred to as J2000/EME2000) defined on epoch 1 January 2000. This frame has its $X$-axis directed towards the J2000 vernal equinox, while the $Z$-axis is collinear with the rotational axis of the Earth. Using the rule of right-handed systems, the direction of $Y$-axis is fixed. The $X$-$Y$ plane lies in the equatorial plane of Earth. The term 'mean' in its definition indicates that the oscillation of the equator and the ecliptic has been averaged. The frame allows estimation of state vector of object under consideration, with respect to the Earth's CoM and against the (relatively) fixed stars in the background. The frame is visualized in Figure 3.2.

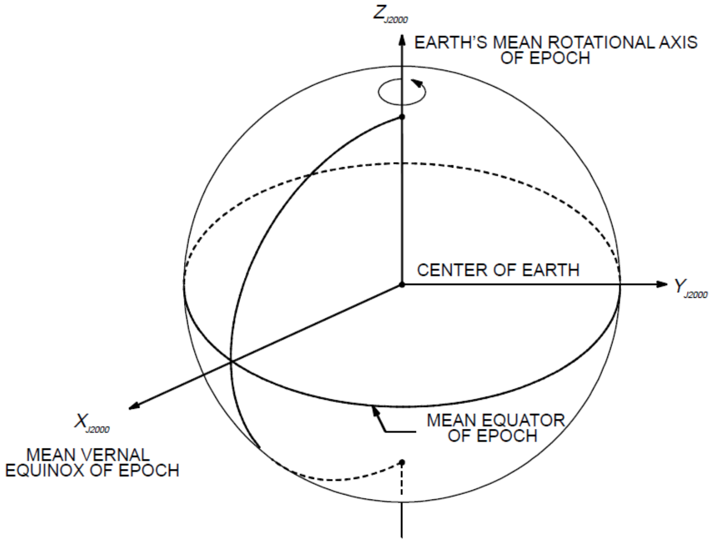


Figure 3.2: Earth-centred inertial frame ($\mathcal{I}$), EME2000 cartesian coordinate system, with origin at centre of Earth [Sauceda, 2001].

#### Earth-centred rotating frame, index $\mathcal{R}$

This reference frame co-rotates with the Earth, so the axes of the frame are fixed with respect to points on Earth. Thus, it is also referred to as Earth-Centred Earth-Fixed (ECEF) frame. Similar to the ECI frame, its $Z$-axis, is along the mean axis of rotation of Earth. The $X$-axis, however, is along the Greenwich meridian, which passes the equator at 0° longitude. The $Y$-axis is then suitably defined, as in a right-handed system. This frame of reference is of particular use when estimating line-of-sight contact between satellite and ground stations on Earth. Figure 3.3 shows the ECEF frame of reference. At the beginning of every rotation, the ECEF frame coincides with the ECI frame.

### 3.1.2 Body centric frames

The origin of these reference frames are fixed on the spacecraft, and thus these frames move along with the spacecraft. Frames defined on the target spacecraft will have $\mathcal{T}$ as a prefix to its index.

#### Body-centred rotating frame, index $\mathcal{V}$

It is also referred to as Local-Vertical, Local-Horizontal (LVLH) frame. It originates at the centre of mass of the spacecraft. The $Z$-axis is defined along the radial direction from the CoM of spacecraft to the centre of Earth, and is also known as R-bar. The $Y$-axis is in the direction opposite to the

[4] Due the orbital motion of Earth around the Sun, the Earth is always accelerating, and thus, any frame on Earth would not be considered strictly inertial with respect to the ICRF. However, for most applications this acceleration can be neglected and, an inertial frame can be defined.

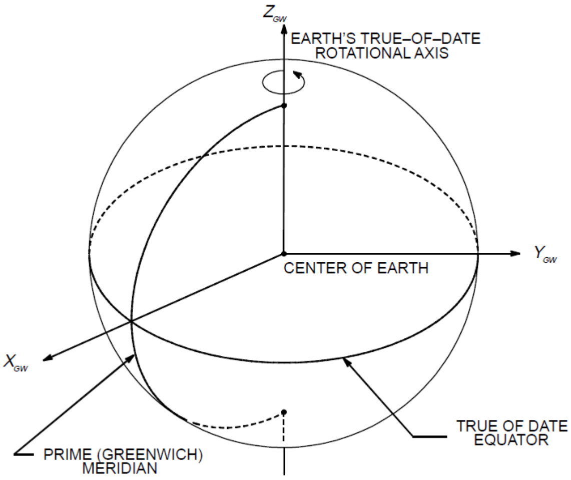

$Z_{GW}$
EARTH'S TRUE–OF–DATE ROTATIONAL AXIS
$Y_{GW}$
CENTER OF EARTH
$X_{GW}$
TRUE OF DATE EQUATOR
PRIME (GREENWICH) MERIDIAN

Figure 3.3: Earth-centred Earth-fixed frame, $X$-axis along Greenwich meridian, origin at centre of Earth [Sauceda, 2001].

angular momentum vector, and referred to as H-bar. Lastly, the $X$-axes, suitably fixed by the right-handed system, is called the V-bar. Having the target spacecraft position fixed (at the origin of the frame) makes it convenient for relative navigation. Figure 3.4 shows the reference frame. The orbital plane lies in the plane of R-bar and V-bar.

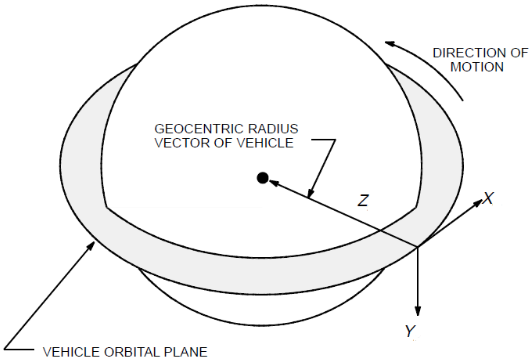


Figure 3.4: Local Vertical / Local Horizontal, or spacecraft body centred intertial frame. Origin considered at CoM of spacecraft [Sauceda, 2001]

### BODY-FIXED FRAME, INDEX $\mathcal{B}$

The frame is used to describe the orientation of the body. Its origin is fixed to the body (say, the CoM of the body) whereas many conventions exist to define the axes. If the frame is defined using the geometrical features of the body, then it can also be referred to as a body-fixed geometric frame. In Section 2.3, we had already defined such a frame (denoted by $\mathcal{TG}$) with its origin at the centre of the launch-adaptor of the satellite.

For our case, we define the Envisat fixed frame, denoted by index $\mathcal{TB}$. Its axis orientation is similar to that of $\mathcal{TG}$ frame, but the origin is placed at the CoM of Envisat. This is shown in Figure 3.5, along with the Envisat LVLH frame (denoted by $\mathcal{TV}$). The origin of both these frames is the CoM of the satellite, but for ease of visualization, the frames are placed outside of the body.

## 3.2 STATE REPRESENTATIONS

The kinematic information of a point mass or a rigid body is termed as its *state*. For a point mass, this consists of translational kinematics (position and motion in three degree-of-freedom), while

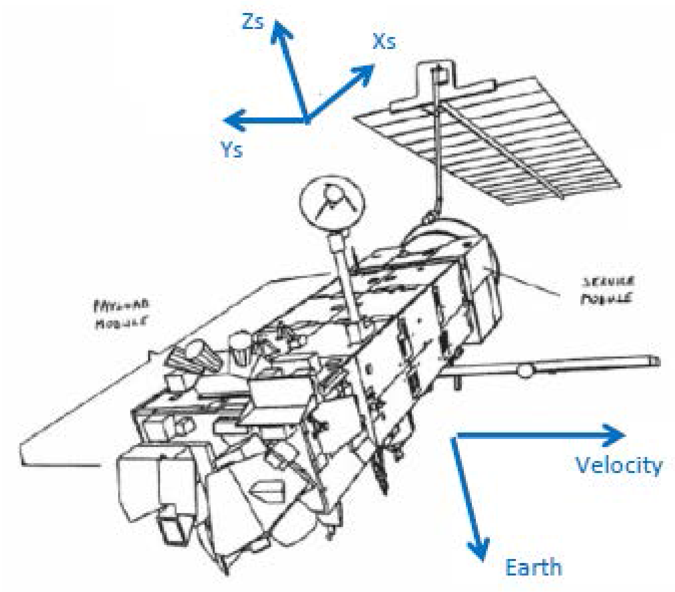


Figure 3.5: Envisat: Axis orientation of LVLH frame, index $\mathcal{TV}$ show in the bottom right of the image. Axis orientation of body fixed frame, index $\mathcal{TB}$ shown on top left. The origin of the both these frames lies at the centre of mass of the satellite, however, for ease of visualization, the frames are shown outside the body [ESA, 2012]

for a rigid body it includes rotational kinematics (upto six degree-of-freedom). For most guidance applications, one considers only the translational motion and neglects the rotational degrees-of-freedom. This restricts the analysis to translational kinematics only. The same practise is followed in this study as well. The state of a body is mathematically represented by its *state vector*. For example, the state vector can include the position and velocity of the body, as described by a coordinate system in a reference frame. They allow a convenient means to analyse and display problems. This section explains three state vector representations that are used in this report, while the conversions between them is given in Appendix A

### 3.2.1 CARTESIAN REPRESENTATION OF STATE VECTOR

The Cartesian system, as mentioned earlier, is the most commonly used coordinate system. It is easy to understand and intuitive in approach for a majority of applications. It consists of 3 mutually orthogonal unit vectors, which intersect at the origin of the system. Along these vectors, lie the reference axes of the system. An object's position vector is the scaled linear combination of the object's projections on the reference axes. Mathematically, this is defined as:

$$\mathbf{r} = \mathbf{r}^{\mathcal{F}} = x\hat{\mathbf{i}}_{\mathcal{F}} + y\hat{\mathbf{j}}_{\mathcal{F}} + z\hat{\mathbf{k}}_{\mathcal{F}} \tag{3.1}$$

where $\mathcal{F}$ is the reference frame, and $x, y, z$ are the object's projections on the three axes. Figure 3.6 illustrates a Cartesian representation of state vector.

The state vector in Cartesian coordinates includes position and velocity of the object. The position vector $\mathbf{r}$, is defined as above. The simplest way to express the velocity is to take the time derivative of position and decompose it along the three axes. This gives us:

$$\mathbf{r} = \begin{bmatrix} x & y & z \end{bmatrix}^{\mathsf{T}}; \quad \dot{\mathbf{r}} = \begin{bmatrix} \dot{x} & \dot{y} & \dot{z} \end{bmatrix}^{\mathsf{T}} \tag{3.2}$$

One can define the accelerations by the time derivative of the velocity vector.

And the state vector is then defined as follows:

$$\mathbf{x} = \begin{bmatrix} \mathbf{r}^{\mathsf{T}} & \dot{\mathbf{r}}^{\mathsf{T}} \end{bmatrix}^{\mathsf{T}} \tag{3.3}$$

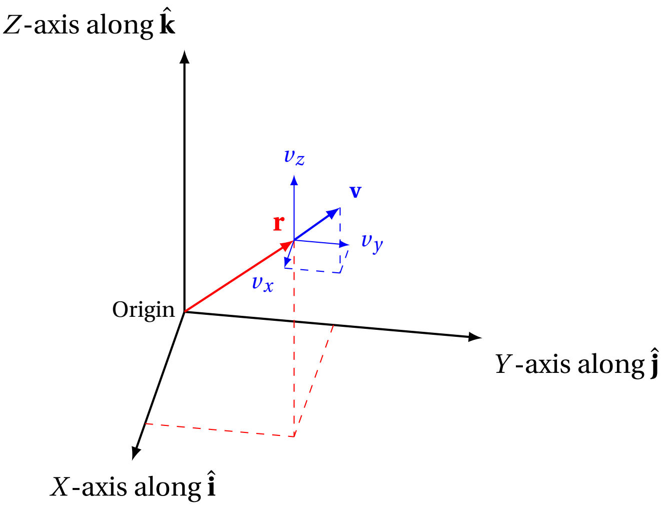

$Z$-axis along $\hat{\mathbf{k}}$
$v_z$
$\mathbf{v}$
$\mathbf{r}$
$v_y$
$v_x$
Origin
$Y$-axis along $\hat{\mathbf{j}}$
$X$-axis along $\hat{\mathbf{i}}$

Figure 3.6: The Cartesian coordinate system for representing translational state vector

### 3.2.2 SPHERICAL COORDINATES REPRESENTATION OF STATE VECTOR

Spherical coordinates are suitable for orbital mechanics, because often references axes originate from the centre of spherical bodies. The position of a point is defined by the radial distance **r** of the point from the origin, the latitude angle $\theta$ made by the point as measured from the $XY$-plane, and the azimuth angle $\phi$ of its orthogonal projection in $XY$-plane. This can be seen in Figure 3.7. The velocity, in spherical components, is defined by the magnitude of the velocity of the object **v**, its heading angle $\chi$, measured from North on the local horizontal plane and the flight-path angle $\gamma$.

$$\mathbf{r} = \begin{bmatrix} r & \phi & \theta \end{bmatrix}^\mathsf{T}; \quad \dot{\mathbf{r}} = \begin{bmatrix} \dot{r} & \chi & \gamma \end{bmatrix}^\mathsf{T} \tag{3.4}$$

And the state vector is then defined as follows:

$$\mathbf{x} = \begin{bmatrix} \mathbf{r}^\mathsf{T} & \dot{\mathbf{r}}^\mathsf{T} \end{bmatrix}^\mathsf{T} \tag{3.5}$$

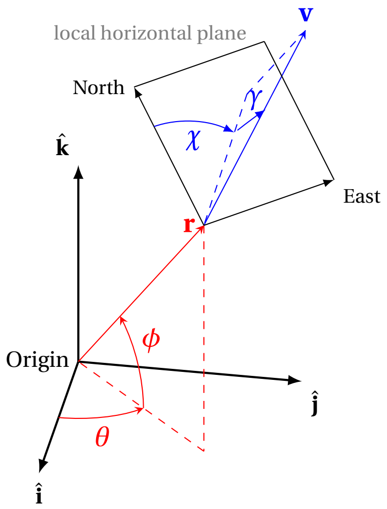
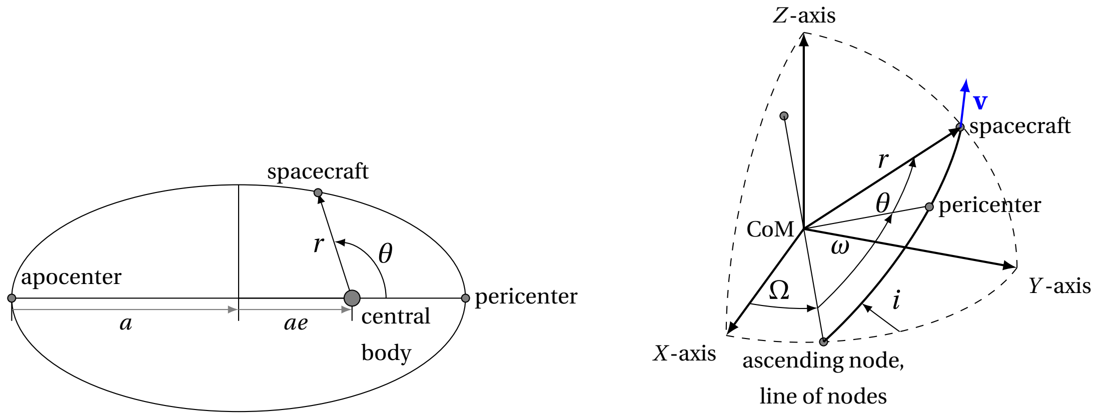


Figure 3.7: An example of state representation in spherical coordinate system. Adapted from Gerth [2014].



(a) Planar ellipse

(b) 3D representation

Figure 3.8: State representation using Keplerian elements. Adapted from Gerth [2014].

### 3.2.3 ORBITAL ELEMENTS AS STATE VECTOR

While Cartesian coordinates provide a simple method for state representation, often it is not very intuitive to understand orbital motion. On the other hand, orbital elements can be used for state representation of orbital motion. They require some extra effort to be calculated initially but provide greater understanding. Orbital motion, in its simplest form, can be modelled as a Keplerian orbit. The spacecraft's motion takes the shape of a conic section as it revolves around a central body, which is placed at one of the foci of the said conic. For rendezvous operations in Earth orbits, it is sufficient to restrict our discussions to elliptical orbits.

A commonly used set of orbital elements are the Keplerian elements, which consists of six parameters. Figure 3.8 shows an elliptical orbit and marks these parameters. For every point in orbit around the central body, astrodynamics dictates the velocity that an object must have to sustain a given orbit. Thus, if the gravitational parameter $\mu$ of the central body is known, then determining the location of the object in the defined orbit, can be sufficient to define its state vector.

Formal definitions of the six parameters can be found in books on astrodynamics like, Vallado [2001]. The six parameters are listed in 3.1. The semi-major axis indicates how big the orbit is, while the eccentricity is a measure how much the elliptical orbit deviates from a circle. If $e = 0$, then the orbit is a circle and a parabola at $e = 1$. For our study, we shall consider only elliptical orbits, hence $e < 1$. Inclination, as the name suggests, indicates the inclination of the orbit with respect to the equatorial plane. Right Ascension of the Ascending Node (RAAN) is a measure of the deviation in angle of the ascending node of the orbit, from the vernal equinox. Argument of pericentre, defines the position of the pericentre with respect to the ascending node, whereas the true anomaly $\theta$, is used to define the position of the object in the orbit, with respect to the pericentre. Sometimes, instead of the true anomaly, one can also use the mean anomaly M, to define the state of the spacecraft.

## 3.3 FRAME TRANSFORMATIONS

Each of the frames described above is used for a specific task, and it would be helpful to be able to transform between them. The frames might be moved and/or rotated with respect to each other, and thus it is important to know their relative position and orientation. The relative distance between two frames can be represented by a Cartesian state vector, for example. However, to discuss the relative orientation between frames, we need to make use of *attitude parameters*, which are discussed below.

Table 3.1: Chaser spacecraft overview

| Parameter | Symbol | Units | Range |
|---|---|---|---|
| Semi-major axis | $a$ | km, AU, etc. | $a > R$* |
| Eccentricity | $e$ | – | $0 \leq e \leq 1$ |
| Inclination | $i$ | deg or radians | $0° \leq i \leq 180°$ |
| RAAN | $\Omega$ | deg or radians | $0° \leq \Omega \leq 180°$ |
| Argument of pericentre | $\omega$ | deg or radians | $0° \leq \omega \leq 180°$ |
| True anomaly | $\theta$ | deg or radians | $0° \leq \theta \leq 360°$ |

*$R$ is radius of central body

### 3.3.1 ATTITUDE PARAMETERS

Attitude parameters are a set of coordinates which are used to define the orientation of a rigid body in a frame. In the thesis, attitude parameters are required for defining the orientation of frames used and to define the instantaneous orientation of the docking axis. Shuster [1993] states that the attitude parameterization of a body requires atleast 3 dimensions. However, this may lead to singularities wherein the coordinates are undefined or not unique. To avoid these singularities, some methods make use of a more than 3 coordinates. We make use of a combination of the following three methods in the report.

#### DIRECTION COSINES

In geometry, direction cosines of a vector, are the cosines of the angles between the vector and the three coordinate axes of the frame it is defined in. A vector **r**, in some frame $\mathcal{F}$ can be written as:

$$\mathbf{r} = (\cos\alpha_1\hat{\mathbf{i}} + \cos\alpha_2\hat{\mathbf{j}} + \cos\alpha_3\hat{\mathbf{k}})\ \|\mathbf{r}\| \tag{3.6}$$

where $\alpha_1$ is the the angle between the vector and $\hat{\mathbf{i}}$, and so on for $\alpha_2$ and $\alpha_3$. Knowing the direction cosines of a vector, its unit vector can be uniquely defined.

Using the same concept, one can define the orientation of the axes of (some) frame $\mathcal{B}$ with respect to some other frame $\mathcal{I}$. The direction cosines can be found by the dot product of the axes of the two frames. Since there are 3 axes each, in all 9 direction cosines will be formed. The transformation between the two frames is given by

$$\begin{bmatrix} \hat{\mathbf{i}}_{\mathcal{B}} \\ \hat{\mathbf{j}}_{\mathcal{B}} \\ \hat{\mathbf{k}}_{\mathcal{B}} \end{bmatrix} = \begin{bmatrix} \cos\alpha_{11} & \cos\alpha_{12} & \cos\alpha_{13} \\ \cos\alpha_{21} & \cos\alpha_{22} & \cos\alpha_{23} \\ \cos\alpha_{31} & \cos\alpha_{32} & \cos\alpha_{33} \end{bmatrix} \cdot \begin{bmatrix} \hat{\mathbf{i}}_{\mathcal{I}} \\ \hat{\mathbf{j}}_{\mathcal{I}} \\ \hat{\mathbf{k}}_{\mathcal{I}} \end{bmatrix}^{\mathsf{T}} = \mathbf{C}_{\mathcal{I}}^{\mathcal{B}} \begin{bmatrix} \hat{\mathbf{i}}_{\mathcal{I}} \\ \hat{\mathbf{j}}_{\mathcal{I}} \\ \hat{\mathbf{k}}_{\mathcal{I}} \end{bmatrix}^{\mathsf{T}} \tag{3.7}$$

where $\alpha_{ij}$ is the angle between between the $i$-th coordinate axis of frame $\mathcal{B}$ and $j$-th coordinate axis of frame $\mathcal{I}$. The matrix $\mathbf{C}_{\mathcal{I}}^{\mathcal{B}}$ is thus called the Direction Cosine Matrix (DCM). It can be used to transform a vector defined in frame $\mathcal{I}$ to frame $\mathcal{B}$ in the following way:

$$\mathbf{r}_{\mathcal{B}} = \mathbf{C}_{\mathcal{I}}^{\mathcal{B}}\mathbf{r}_{\mathcal{I}} \tag{3.8}$$

Since the magnitude of angles between two vectors remain same whether measured from one frame or the other, the backward transformation from frame $\mathcal{B}$ to frame $\mathcal{I}$ is straightforward as well:

$$\mathbf{r}_{\mathcal{I}} = (\mathbf{C}_{\mathcal{I}}^{\mathcal{B}})^{\mathsf{T}}\mathbf{r}_{\mathcal{B}} = \mathbf{C}_{\mathcal{B}}^{\mathcal{I}}\mathbf{r}_{\mathcal{B}} \tag{3.9}$$

Combining the above two equations, we also notice that $\mathbf{C}\mathbf{C}^{\mathsf{T}} = \mathbf{I}_{3\times3}$ (which makes $\mathbf{C}$ to be orthonormal). The DCM contains 9 elements (6 more than a minimal parameterization) and is thus is computationally expensive for use.

### EULER ANGLES

Euler angles are a popular set of attitude parameters used in aerospace engineering. A frame $\mathcal{I}$ can be transformed to a frame $\mathcal{B}$ by three successive rotations. A commonly used rotation sequence is $Z$–$Y$–$X$, *i.e.*, rotate by the required angle about the $Z$-axis of the $I$ frame, followed by rotations about $Y$ and $X$-axes of the intermediate frames. In all, three angles are specified thus, making Euler angles a minimal set in terms of representing attitudes. However, they can suffer from singularities in some cases, thus they are not completely reliable. More about the different types of Euler angle sequences, derivations of equations and commonly used conventions can be found in Shuster [1993], Singla et al. [2005] and Wie [2008].

The combination of Euler angles with DCMs results in a convenient and an intuitive means for frame transformations. The DCM of an intermediate coordinate axis formed by rotating the reference frame by an angle $\psi$ around its $Z$-axis is given by:

$$\mathbf{R}_z = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.10}$$

where $\mathbf{R}$ is called the unit-axis rotation matrix. Similarly, for rotation around $Y$ and $X$-axes, we have the following:

$$\mathbf{R}_y = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{3.11}$$

$$\mathbf{R}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & \sin\varphi \\ 0 & -\sin\varphi & \cos\varphi \end{bmatrix} \tag{3.12}$$

Using successive unit-axis rotations, the reference frame can be rotated to any relative orientation:

$$\mathbf{r}_\mathcal{B} = \mathbf{R}_x(\varphi)\mathbf{R}_y(\theta)\mathbf{R}_z(\psi)\mathbf{r}_\mathcal{I} = \mathbf{R}_\mathcal{I}^\mathcal{B}\mathbf{r}_\mathcal{I} \tag{3.13}$$

where $\mathbf{R}_\mathcal{I}^\mathcal{B}$ is called the rotation matrix from frame $\mathcal{I}$ to $\mathcal{B}$. It is noted that the rotation matrix (which is just a different form of DCM), is orthonormal as well. Hence, the inverse transformation is:

$$\mathbf{r}_\mathcal{I} = (\mathbf{R}_\mathcal{I}^\mathcal{B})^{-1}\mathbf{r}_\mathcal{B} = \mathbf{R}_\mathcal{B}^\mathcal{I}\mathbf{r}_\mathcal{B} \tag{3.14}$$

### QUATERNIONS

Quaternion of rotation consist of four dependent, normalised parameters representing a single rotation around an axis called as Euler's eigenaxis. The Euler's eigenaxis rotation theorem states that "by rotating a rigid body about an axis that is fixed to the body and stationary in an inertial reference frame, the rigid body attitude can be changed from any given orientation to any other orientation. Such an axis of rotation, whose orientation relative to both an inertial reference frame and the body remains unchanged throughout the motion, is called the Euler axis or eigenaxis."

The first three elements of the quaternion define the orientation of the Euler eigenaxis in the reference frame whereas the fourth scalar component represents the angle of rotation $\theta$. This is in contrast with the Euler angles, where three successive rotations describe the orientation, the Euler eigenaxis is a single axis around which the reference frame is rotated once. There is no singularity involved in its usage, but due to having 4 parameters in its formulation, it is slightly more computationally expensive, compared to Euler angles.

The orientation of the Euler's eigenaxis remains unchanged in the motion, relative to both the frames. Thus, a unit vector $e$ along this axis can be defined as:

$$\begin{aligned} \hat{\mathbf{e}} &= e_1\hat{\mathbf{i}}_\mathcal{I} + e_2\hat{\mathbf{j}}_\mathcal{I} + e_3\hat{\mathbf{k}}_\mathcal{I} \\ &= e_1\hat{\mathbf{i}}_\mathcal{B} + e_2\hat{\mathbf{j}}_\mathcal{B} + e_3\hat{\mathbf{k}}_\mathcal{B} \end{aligned} \tag{3.15}$$

for which it holds that $e_1^2 + e_2^2 + e_3^2 = 1$. The quaternion of rotations is then defined by:

$$\mathbf{q} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \begin{bmatrix} e_1 \sin(\theta/2) \\ e_2 \sin(\theta/2) \\ e_3 \sin(\theta/2) \\ \cos(\theta/2) \end{bmatrix} \tag{3.16}$$

where $\theta$ is the rotation angle around the Euler's eigenaxis. The four parameters of a quaternion are not independent of each other, as a quick glance shows us that:

$$q_1^2 + q_2^2 + q_3^2 + q_4^2 = 1 \tag{3.17}$$

The DCM can now be written in the quaternion parameterized form as follows [Wie, 2008]:

$$\mathbf{C} = \begin{bmatrix} q_4^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 + q_3q_4) & 2(q_1q_3 - q_2q_4) \\ 2(q_1q_2 - q_3q_4) & q_4^2 + q_2^2 - q_1^2 - q_3^2 & 2(q_2q_3 + q_1q_4) \\ 2(q_1q_3 + q_2q_4) & 2(q_2q_3 - q_1q_4) & q_4^2 + q_3^2 - q_1^2 - q_2^2 \end{bmatrix} \tag{3.18}$$

Composite rotations are made possible by multiplications of quaternions (similar to DCMs and Euler angle rotations) to form a new quaternion. This new quaternion is equivalent to the successive rotations of the parent quaternions. The multiplication, however, is unlike normal matrix multiplication. It makes of multiplication for complex numbers, as shown below:

$$\begin{aligned} \mathbf{q} \otimes \mathbf{r} &= \begin{bmatrix} q_4 \begin{bmatrix} r1 \\ r2 \\ r3 \end{bmatrix} + r_4 \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} - \begin{bmatrix} r1 \\ r2 \\ r3 \end{bmatrix} \times \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} \\ q_4 r_4 - \begin{bmatrix} r1 \\ r2 \\ r3 \end{bmatrix} \cdot \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} \end{bmatrix} \\ &= \begin{bmatrix} q_4 & q_3 & -q_2 & q_1 \\ -q_3 & q_4 & q_1 & q_2 \\ q_2 & -q_1 & q_4 & q_3 \\ -q_1 & -q_2 & -q_3 & q_4 \end{bmatrix} \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \end{bmatrix} \end{aligned} \tag{3.19}$$

#### USAGE

In this thesis, the three attitude parameters are used in combination with each other. It is easy to understand Euler angles by the user, and thus, they are used for the interface to and from the simulator. Quaternions allow easy propagation of state and singularity free transformation. Hence, they are used for kinematic propagation of target state, as it spins. Lastly, DCMs are used for general purpose transformation because of the ease with which they can be used and represented.

### 3.3.2 TRANSFORMATION EXAMPLES

Based on the discussion in the previous section, some frame transformations are provided below. For ease of visualization, the transformations are shown in unit-axis rotation matrix.

#### BETWEEN EARTH CENTRED INERTIAL AND ROTATING FRAMES

Earth centred inertial frame, index $\mathcal{I}$ and Earth centred rotating frame, index $\mathcal{R}$ share the same origin, but are rotated about the $Z$-axis. Hence, a simple rotational transformation is sufficient to toggle between the two frames. Earth rotating frame $\mathcal{R}$ rotates with an angular velocity $\omega_E$, and at any given point of time, the angle between the frames $\mathcal{R}$ and $\mathcal{I}$, $\Omega$, is given by:

$$\Omega = \Omega_0 + \omega_E t \tag{3.20}$$

where, $\Omega_0$ is the value of $\Omega$ at a given epoch, and $t$ is the time since last epoch. Thus, the unit-axis rotation matrix is:

$$\mathbf{R}_I^R = \begin{bmatrix} \cos\Omega & \sin\Omega & 0 \\ -\sin\Omega & \cos\Omega & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.21}$$

And the transformation between the two frames is:

$$\mathbf{r}^{\mathcal{R}} = \mathbf{R}_I^R \mathbf{r}^{\mathcal{I}} \qquad \text{From frame } \mathcal{I} \text{ to } \mathcal{R}. \tag{3.22}$$

$$\mathbf{r}^{\mathcal{I}} = \mathbf{R}_R^I \mathbf{r}^{\mathcal{R}} \qquad \text{From frame } \mathcal{R} \text{ to } \mathcal{I}. \tag{3.23}$$

### Between Earth centred rotating and body centred inertial (LVLH) frames

The Earth centred rotating frame $\mathcal{R}$ and the LVLH frame $\mathcal{V}$, are oriented in different directions and moving with respect to each other. As seen from frame $\mathcal{R}$, the position of the origin of frame $\mathcal{V}$ is given by the translation vector $\mathrm{T}_{\mathcal{R}}^{\mathcal{V}}$. The origin of frame $\mathcal{V}$, has a latitude $\delta$ and longitude $\lambda$, as seen from frame $\mathcal{R}$. The rotation matrix between the two frames is defined by:

$$\mathbf{R}_{\mathcal{V}}^{\mathcal{R}} = \mathbf{R}_z(-\pi/2)\mathbf{R}_x \pi/2)\mathbf{R}_y(\lambda)\mathbf{R}_x(\delta) \tag{3.24}$$

The overall transformation equation is:

$$\mathbf{r}_{\mathcal{R}} = \mathrm{T}_{\mathcal{R}}^{\mathcal{V}} + \mathbf{R}_{\mathcal{V}}^{\mathcal{R}} \mathbf{r}_{\mathcal{V}} \tag{3.25}$$

And the inverse is given by:

$$\mathbf{r}_{\mathcal{V}} = (\mathbf{R}_{\mathcal{V}}^{\mathcal{R}})^{\mathsf{T}} (\mathbf{r}_{\mathcal{R}} - \mathrm{T}_{\mathcal{R}}^{\mathcal{V}}) \tag{3.26}$$

### Between body centred inertial (LVLH) and body fixed frames

Consider the orientation of the body fixed frame $\mathcal{B}$ with the LVLH frame $\mathcal{V}$ is described by the roll ($\phi$), pitch ($\theta$) and yaw ($\psi$) angles. Then using the unit-axis rotation matrix, we have:

$$\mathbf{r}_{\mathcal{B}} = \mathbf{R}_x(\phi)\mathbf{R}_y(\theta)\mathbf{R}_z(\psi)\mathbf{r}_{\mathcal{V}} = \mathbf{R}_V^B \mathbf{r}_{\mathcal{V}} \tag{3.27}$$

Since a unit-axis rotation matrix is orthonormal, the inverse transformation is straightforward

$$\mathbf{r}_{\mathcal{V}} = (\mathbf{R}_V^B)^{\mathsf{T}} \mathbf{r}_{\mathcal{B}} \tag{3.28}$$

## 3.4 Equations of motion

This section focusses on the equations used to describe motion of objects in orbits. These can be expressed in various forms, based on the choice of state representation and frame of motion. We shall revisit the previously discussed Cartesian representations and Keplerian orbital elements and discuss relevant equations of motions. Equations of Motion (EOM) in spherical co-ordinate frames, are not discussed as they are not considered to be directly applicable to the rendezvous problem modelling. A detailed section on Clohessy-Wiltshire equations is presented to give an idea about relative orbital motion. In the end, the kinematic equations for describing orientation of rotating frames is discussed.

### 3.4.1 CARTESIAN COMPONENTS

As seen in equations (3.2) and (3.3), the Cartesian state representation of a body is $\mathbf{x} = \begin{bmatrix}\mathbf{r}^\intercal & \dot{\mathbf{r}}^\intercal\end{bmatrix}^\intercal$, where

$$\mathbf{r} = \begin{bmatrix}x & y & z\end{bmatrix}^\intercal; \quad \dot{\mathbf{r}} = \begin{bmatrix}\dot{x} & \dot{y} & \dot{z}\end{bmatrix}^\intercal$$

Differentiating it with respect to time, we get the velocity and net acceleration of the body.

$$\mathbf{v} = \frac{d\mathbf{r}}{dt} = \begin{bmatrix}\dot{x} & \dot{y} & \dot{z}\end{bmatrix}^\intercal \tag{3.29}$$

$$\mathbf{a} = \frac{d^2\mathbf{r}}{dt^2} = \begin{bmatrix}\ddot{x} & \ddot{y} & \ddot{z}\end{bmatrix}^\intercal = \frac{\mathbf{F}_{\text{net}}}{m} \tag{3.30}$$

where, $\mathbf{F}_{\text{net}}$ is the net external force on the body, and $m$ is its (instantaneous) mass. These equations hold true for motion in inertial frames. For a given initial state, a state at a final time can be calculated by integrating the above equation of motion. One has to note that the above equations are defined in an inertial frame. It is not suggested to integrate the Equations of Motion (EOM) in a rotating frame, as one would have to consider the effect of apparent forces as well. This makes the problem unnecessarily complicated. The alternative is to transform the initial state to an inertial frame, integrate its EOM, and transform the final state back to the rotating frame.

### 3.4.2 KEPLERIAN ORBITAL ELEMENTS

Keplerian orbits prove to be a good baseline approximation for describing orbital motion. The main assumptions related to a Keplerian orbit are:

1. There are no internal or external forces except gravity.
2. The gravitating bodies are spherical and have constant density.
3. There are no tidal forces.
4. The mass of the orbiting body is negligible compared to the primary body's mass.
5. The gravitational force is Newtonian

We can derive the equation of orbital motion using Newton's law of gravitation and his second law of motion. We begin with Newton's law of gravitation,

$$\mathbf{F}_g = -G\frac{Mm}{\mathbf{r}^2} \tag{3.31}$$

where $\mathbf{F}_g$ is the gravitational force between two bodies of mass $M$ and $m$ separated by a distance $r$. $G = 6.674 \times 10^-11\text{Nm}^2/\text{kg}^2$ is the universal gravitational constant. Next, we move on to Newton's second law of motion, that relates forces and acceleration,

$$\mathbf{F} = m\ddot{\mathbf{r}} \tag{3.32}$$

where $\mathbf{F}$ is the net force acting on a body of mass $m$, and $\ddot{\mathbf{r}}$ is its acceleration. Combining the above two equations, we get the equation of orbital motion (considering only gravity) of a satellite:

$$\ddot{\mathbf{r}} = -\frac{\mu}{\mathbf{r}^2} \tag{3.33}$$

where, $\mu = G \cdot M$, is the gravitational parameter. For Earth, this value is $\mu_E = 398600.441\text{km}^3/\text{s}^2$ A solution to (3.33) is a conic section, described by:

$$r = \frac{a(1-e^2)}{1+e\cos\theta} \tag{3.34}$$

where $a$ is the semi-major axis, $e$ is eccentricity, $\theta$ is true anomaly. The derivation can be found in standard books on astrodynamics like, Vallado [2001]. Based on the value of eccentricity $e$, one can define the type of conic section. For our discussion, we shall restrict to cases where the eccentricity is in the range, $0 \leq e < 1$, corresponding to circular and elliptical orbits only.

Using (3.34), the radius of apocentre and preicentre, are given by considering $\theta = \pi$ and $\theta = 0$, respectively.

$$\begin{aligned} r_A &= a(1+e); \quad \text{Radius of apocentre} \\ r_P &= a(1-e); \quad \text{Radius of pericente} \end{aligned} \tag{3.35}$$

Time period, the time taken for one orbital revolution, is given by:

$$T = 2\pi\sqrt{\frac{a^3}{\mu}} \tag{3.36}$$

The total energy of the system, as described by the vis-viva equation is given by:

$$v^2 = \mu\left(\frac{2}{r} - \frac{1}{a}\right) \tag{3.37}$$

### 3.4.3 CLOHESSY-WILTSHIRE EQUATIONS

Hill [1878] presented a set of linear differential equations, which describe relative orbital motion between satellites. These have analytical solutions which allow propagation of the chaser in the target reference frame (LVLH frame). Later, Clohessy and Wiltshire [1960] found a linearized solution to Hill's equations, specifically for use with terminal guidance for rendezvous. The solution, commonly referred to as the Clohessy-Wiltshire (CW) equations, is easy to implement, substantially accurate (in its scope of application) and has low computational cost. It is of little surprise that they are the primary choice for guidance (as was observed in Table 2.1) in rendezvous missions. The equations of motion of the chaser spacecraft in LVLH frame are as follows (based on Fehse [2003])

$$\begin{aligned} x(t) &= \left(\frac{4\dot{x}_0}{n} - 6z_0\right)\sin(nt) - \frac{2\dot{z}_0}{n}\cos(nt) + (6nz_0 - 3\dot{x}_0)t + \left(x_0 + \frac{2\dot{z}_0}{n}\right) \\ y(t) &= y_0\cos(nt) + \frac{\dot{y}_0}{n}\sin(nt) \\ z(t) &= \left(\frac{2\dot{x}_0}{n} - 3z_0\right)\cos(nt) + \frac{\dot{z}_0}{n}\sin(nt) + \left(4z_0 - \frac{2\dot{x}_0}{n}\right) \end{aligned} \tag{3.38}$$

Here the subscript 0 refers to the initial value, $t$ refers to time since last epoch and $n = 2\pi/T$ is the mean motion of the target spacecraft.

The CW equations are applicable for circular orbits, and small relative distances between the two satellites (as compared to the point-mass central body). Further, only free-drift trajectories and impulsive manoeuvres can be derived from them, as one is restricted to use impulsive, and constant thrust accelerations. Solutions with continuous (or finite-time) thrust manoeuvres are possible, but by solving the Hill's equations (although, often the names are wrongly presumed to be the same).

To give an idea of the relative motion between satellites, two cases of free-drift motion and two main types of impulsive manoeuvres for transfer along V-bar are discussed[5].

#### FREE-DRIFT MOTION

A free-drift (FD) motion is one where the chaser drifts away from its initial state (as seen in LVLH frame), due to small difference in its orbit with respect to the target. Four cases are presented, to get a feel of the natural tendency of relative motion. It is assumed that there are no perturbations acting,

[5] The remaining transfers, while important, are skipped from discussion for lack of direct relevance to this study. The interested reader is suggested to read the work by Fehse [2003] for a detailed analysis.

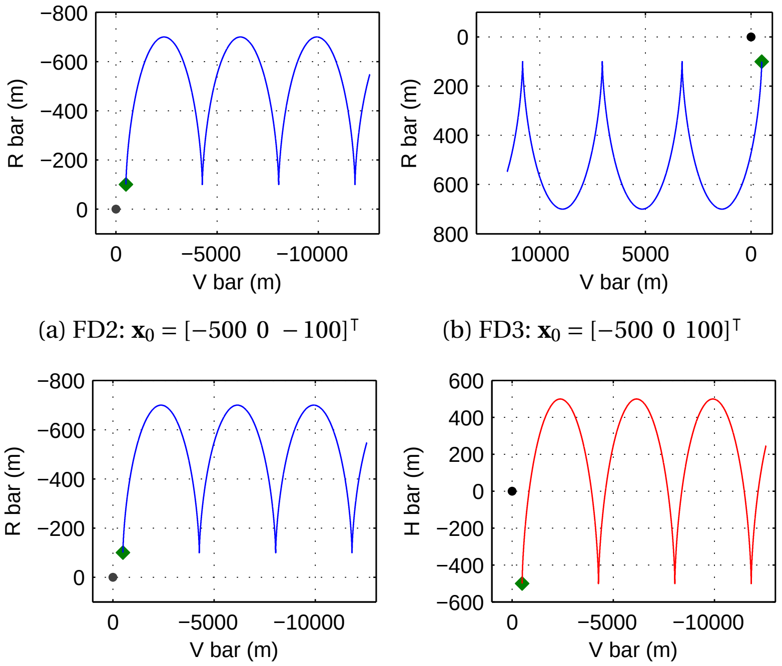
and unless explicitly specified, all initial state values are zero. The results are plot in Figure 3.9. The target (Envisat) is a black circle, and the initial chaser state is a green mark.

1. CW-FD case 1: The chaser is initially on the V-bar axis, at some distance $x_0$. The state remains unchanged, and thus, can be used for passive station-keeping.

2. CW-FD case 2: The chaser's initial position is behind and above the target (*i.e.*, negative values for $x_0$ and $z_0$). On a short time scale, the chaser appears to move away from the target. In the long scale, the chaser has a periodic motion in negative $R$-bar (with period $T$) and a secular motion in negative V-bar direction. The same can be said, if the initial value of $x_0$ is positive. See Figure 3.9a for flight when $t_f > 3T$.

3. CW-FD case 3: The chaser's initial position is behind and below the target (*i.e.*, negative $x_0$ and positive $z_0$). The free-drift motion is similar to the one seen in CW-FD case 2, but with inverted signs. The chaser sees periodic motion in positive $R$-bar (with period $T$) and secular motion in positive V-bar. The same holds when the initial value of $x_0$ is positive. See Figure 3.9b for the trajectory.

4. CW-FD case 4: The chaser is on a point along the H-bar axis. (*i.e.,* $y_0$ is non-zero). The chaser has a tendency to move back towards the target, and further on to the other side. The motion is similar to that of an oscillating pendulum. The period of motion is $T$, and the maximum amplitude remains $\pm y_0$. If $x_0$ and $z_0$ are non-zero too, then this motion gets superimposed with their free-drift motion. See Figure 3.9c and Figure 3.9d for the trajectory.



(a) FD2: $\mathbf{x}_0 = [-500\;\;0\;\;-100]^\mathsf{T}$

(b) FD3: $\mathbf{x}_0 = [-500\;\;0\;\;100]^\mathsf{T}$

(c) FD4: $\mathbf{x}_0 = [-500\;\;-500\;\;100]^\mathsf{T}$, View 1

(d) FD4: $\mathbf{x}_0 = [-500\;\;-500\;\;100]^\mathsf{T}$, View 2

Figure 3.9: Free drift trajectories for various initial states of chaser spacecraft, as calculated by CW equations. Target is black dot, initial position of chaser is green dot, orbital motion is along the the left side of the reader.

### IMPULSIVE MANOEUVRE

A typical impulse transfer along V-bar involves two-impulses that are fired just after $t_0$ and before $t_f$, with free-drift motion in between. We assume the following initial state for the chaser:

$$y_0, z_0 = 0; \qquad \dot{x}_0, \dot{y}_0, \dot{z}_0 = 0 \tag{3.39}$$

and a final state as:

$$y(t_f), z(t_f) = 0; \qquad \dot{x}(t_f), \dot{y}(t_f), \dot{z}(t_f) = 0 \tag{3.40}$$

**Tangential impulse transfer:**

The tangential impulse transfer, as the name suggests, involves burns in the tangential direction or the V-bar axis. Just after the first impulse, let $\dot{x}_1 = \Delta V_1$. Using the defined values, we can simplify the CW-equations (Equation (3.38) to propagate state of the chaser:

$$\begin{aligned} x(t) &= \frac{\dot{x}_0}{n}\big(4\sin(nt) - 3nt\big) + x_0 \\ y(t) &= 0 \\ z(t) &= \frac{2\dot{x}_0}{n}\big(\cos(nt) - 1\big) \end{aligned} \tag{3.41}$$

Substituting $z(t_f) = 0$, and the other final states in (Equation (3.38), we get:

$$\cos(nt_f) = 1 \Rightarrow nt_f = 0, 2\pi, 4\pi, \dots, 2N\pi \tag{3.42}$$

This means that over every orbital period, the chaser returns to a point on V-bar axis (a motion similar to the one seen in Figure 3.9a). Using this value, we get:

$$\Delta V_1 = -\frac{n\Delta x}{6N\pi} \tag{3.43}$$

where $\Delta x = x(t_f) - x_0$. Setting $x(t_f) = 0$, $N = 1$, we get:

$$\dot{x}_1 = \Delta V_1 = \frac{nx_0}{6\pi} \tag{3.44}$$

Here, $\dot{x}_1$ refers to the velocity of the chaser *just* after the impulse burn. It is the $\Delta V$ required to initiate the motion from $x_0$ to origin of LVLH (for docking). A second impulse would be needed to ensure that the relative velocity of the chaser is zero when it reaches the target. To find the magnitude of $\Delta V$ for the second burn, we first differentiate Equation (3.38) to get:

$$\begin{aligned} \dot{x}(t) &= -6n(1 - \cos(nt))z_0 - 2\sin(nt)\dot{z}_0 + (4\cos(nt) - 3)\dot{x}_0 \\ \dot{y}(t) &= -n\sin(nt)y_0 + \cos(nt)\dot{y}_0 \\ \dot{z}(t) &= 3n\sin(nt)z_0 + \cos(nt)\dot{z}_0 + 2\sin(nt)\dot{x}_0 \end{aligned} \tag{3.45}$$

By substituting Equation 3.44 and simplifying, we get:

$$\dot{x}_2 = \Delta V_2 = -\frac{nx_0}{6\pi} \tag{3.46}$$

where $\dot{x}_2$ is the velocity *just* before $t_f = T$. It is in the opposite direction of the first burn. The total $\Delta V$ for the transfer is:

$$\begin{aligned} \Delta V_T &= |\Delta V_1| + |\Delta V_2| \\ &= \left|\frac{nx_0}{3\pi}\right| \\ &\approx |0.1061 nx_0| \end{aligned} \tag{3.47}$$

Thus, the $\Delta V$ for tangential impulse transfer to the target depends solely on the initial position of chaser on V-bar axis, and the mean motion of the target. It takes 1 orbital period to complete the transfer. Figure 3.10a shows the transfer for $x_0 = -500$ m with Envisat as target.

**Radial impulse transfer:**
The radial impulse transfer consists of a pair of impulses in the radial direction, or R-bar axis. The equations are derived in a similar method as that for tangential impulse transfer, with the difference that we set $\dot{z}_1 = \Delta V_1$. This leads us to:

$$\begin{aligned} x(t) &= \frac{2\dot{z}_0}{n}(1-\cos(nt)) + x_0 \\ y(t) &= 0 \\ z(t) &= \frac{\dot{z}_0}{n}\sin(nt) \end{aligned} \tag{3.48}$$

Setting $z(t_f) = 0$, and substituting other values, we get:

$$\sin(nt_f) = 0 \Rightarrow nt_f = 0, \pi, 2\pi, \ldots, N\pi \tag{3.49}$$

This shows that every half orbital period, the chaser returns to V-bar axis. Again, setting N = 1 and $x(t_f) = 0$, we get:

$$\Delta V_1 = \frac{nx_0}{4} \tag{3.50}$$

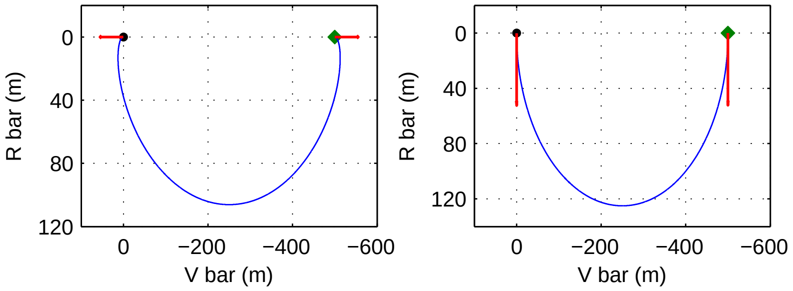
The second impulse is also calculated to:

$$\Delta V_2 = \frac{nx_0}{4} \tag{3.51}$$

This has the same direction as the first burn. The total $\Delta V$ is:

$$\begin{aligned} \Delta V_{T/2} &= |\Delta V_1| + |\Delta V_2| \\ &= \left|\frac{nx_0}{2}\right| \\ &\approx |0.5nx_0| \end{aligned} \tag{3.52}$$

The transfer takes only $T/2$ seconds, but it is nearly 4.7 times more expensive than tangential impulse transfer. Figure 3.10b shows the transfer for $x_0 = -500$ m with Envisat as target.



(a) Tangential impulse transfer from $x_0 = -500$ m

(b) Radial impulse transfer from $x_0 = -500$ m

Figure 3.10: Impulsive transfers from point on V-bar.

### 3.4.4 KINEMATIC EQUATIONS FOR ROTATIONAL MOTION

Envisat is tumbling, and thus the orientation of the body-fixed frame $\mathcal{TB}$ is changing with time. In Section 3.3.1, we discussed the various forms of attitude parameters. These are used to describe the orientation of a reference frame with another. However, time-dependent relations between two frames were not discussed there. Since we chose quaternions as the primary attitude parameter, its kinematic equation is presented below. These will be used to determine the orientation of the docking axis at a given time.

Consider a frame $\mathcal{B}$ rotating with respect to another frame $\mathcal{V}$, with an angular velocity $\boldsymbol{\omega}$. The quaternion is defined as $\mathbf{q} = \begin{bmatrix} q_1 & q_2 & q_3 & q_4 \end{bmatrix}^{\mathrm{T}}$ and its derivative $\dot{\mathbf{q}} = \begin{bmatrix} \dot{q}_1 & \dot{q}_2 & \dot{q}_3 & \dot{q}_4 \end{bmatrix}^{\mathrm{T}}$. The kinematic relation between the two is:

$$\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\Omega}\mathbf{q}, \qquad \text{where } \boldsymbol{\Omega} = \begin{bmatrix} 0 & \omega_3 & -\omega_2 & \omega_1 \\ -\omega_3 & 0 & \omega_1 & \omega_2 \\ \omega_2 & -\omega_1 & 0 & \omega_3 \\ -\omega_1 & -\omega_2 & -\omega_3 & 0 \end{bmatrix} \tag{3.53}$$

This can also be approximated as as an analytical expression:

$$\mathbf{q}(t+\Delta t) = \left\{ \cos(\omega\Delta t/2)\mathbf{I}_4 + \left[ \frac{\sin(\omega\Delta t/2)}{\omega} \right] \boldsymbol{\Omega} \right\} \mathbf{q}(t) \tag{3.54}$$

where $\omega$ is the magnitude of the angular veclocity.

## 3.5 FORCE MODELS

This section focusses on force models that allow to estimate the forces acting on the spacecraft. Gravitational forces, thrust forces, atmospheric drag, solar radiation and third-body perturbations are discussed here.

### 3.5.1 GRAVITY-FIELD

The driving force for orbital motion is gravity - it is one of the most dominant forces on the spacecraft, along with thrust forces.(As a result, other forces, can well be considered as perturbations). Based on the required accuracy and acceptable complexity, numerous gravity-field models exist. We shall focus on some of the more relevant models, in this section. These models do not include relativistic effects.

#### NEWTONIAN GRAVITY

This is the simplest of the gravity-models and is based on Newton's inverse-square law of gravity. It accurately models the gravitational force of a uniform sphere, and thus is only an approximation of the gravity of an actual celestial body. The resulting gravitational acceleration $\mathbf{a}_g$ is thus, given by:

$$\mathbf{a}_g = \ddot{\mathbf{r}} = -\frac{\mu}{r^2}\hat{\mathbf{r}} \tag{3.55}$$

where $\mu$ is the central body's gravitational parameter. In terms of gravitational potential $U$, it is:

$$\ddot{\mathbf{r}} = \nabla U; \qquad U = \mu/r \tag{3.56}$$

Keplerian orbits consider Newtonian gravity. As such, the applications of Newtonian gravity is used to define the *reference orbit* of the spacecraft, while the effect(s) of non-uniform gravity is considered as a perturbation, in which case the equation (3.55) is modified to:

$$\mathbf{a}_g = -\frac{\mu}{r^2}\hat{\mathbf{r}} + \mathbf{f} \tag{3.57}$$

where $\mathbf{f}$ is the perturbing force.

### SPHERICAL HARMONICS

Since Earth, is not a homogeneous sphere, but rather a non-uniform, oblate spheroid, its gravity field cannot be modelled accurately by the Newtonian gravity model. To account for these variations, the spherical harmonics model is suggested. The gravitational potential for Earth in terms of spherical harmonics are Wakker [2010a]:

$$U = -\frac{\mu}{r}\left[1 + \sum_{n=2}^{\infty}\left(\frac{R}{r}\right)^n J_n P_n(\sin\theta) + \sum_{n=2}^{\infty}\sum_{m=1}^{n}\left(\frac{R}{r}\right)^n J_{n,m} P_{n,m}(\sin\theta)\cos m(\phi - \phi_{n,m})\right] \tag{3.58}$$

where, $(r,\phi,\theta)$ are the spherical coordinates of the point that is considered, $R$ is the mean equatorial radius of Earth, $J_n, J_{n,m}$ and $\phi_{n,m}$ are model parameters. Functions $P_n(x)$ and $P_{n,m}(x)$, are *Legendre polynomials* of degree $n$ and *associated Legendre functions of the first kind* of degree $n$ and order $m$, respectively. For a given $x$, they are defined as:

$$P_n(x) = \frac{1}{(-2)^n n!}\frac{d^n}{dx^n}(1-x^2)^n \tag{3.59}$$

$$P_{n,m}(x) = (1-x^2)^{m/2}\frac{d^m P_n(x)}{dx^m} \tag{3.60}$$

Equation (3.58) only holds, when $r > R$. The summation index of $n$ begins at 2. For $n = 1$, we have $J_1$ and $J_{1,1}$ as zero, by definition. This is because, the origin of the spherical coordinate system coincides with the Earth's CoM. Thus, the first perturbing term is $J_2$, with a value of approximately $1.083 \times 10^{-3}$. Of all the $J$-coefficients, it is the largest, and yet the resulting perturbing force is a good 3 orders of magnitude smaller than Newtonian gravity.

The first term in equation (3.58), is equivalent to equation (3.56). The second term represents the influence of deviations of the mass density distribution in north-south direction (also known as *zonal harmonics*, while the third terms represents the influence of deviations of the mass density distribution in north-south and east-west directions (also knowns as *tesseral harmonics*. However, when $m = n$, it is referred to as *sectorial harmonics*). These can be seen in Figure 3.11.

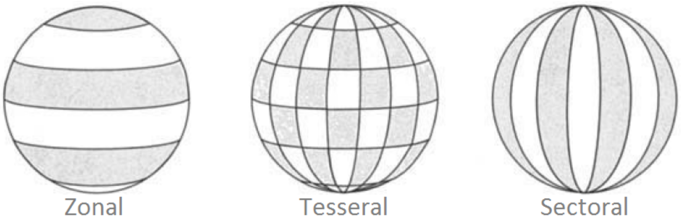


Figure 3.11: Zonal, tesseral and sectorial spherical harmonics [Pisacane, 2005]

### 3.5.2 THRUST FORCES

Thrust, produced by an on-board propulsion system, is the other dominating force that effects the dynamics of the spacecraft. Since the propulsion system is extensively tested prior to flight, the thrust force is relatively easier to model with high fidelity. One of the simplest equations for thrust force is given by:

$$\mathbf{T} = -\dot{m}\mathbf{v}_e + (p_e - p_{amb})A_e\hat{r}_e \tag{3.61}$$

where, thrust $\mathbf{T}$ is the sum of momentum thrust and pressure thrust. The momentum thrust is the product of $\dot{m}$, the mass flow rate of propellant and $\mathbf{v}_e$, the exhaust velocity. Pressure thrust, which opposes momentum thrust, is the force experienced at the nozzle exhaust (of cross-section area $A_e$) due to difference in pressure of exhaust gases, $p_e$ and ambient pressure, $p_{amb}$. In vacuum, $p_{amb} \approx 0$, thus pressure thrust shall be dropped. A more convenient method to express the thrust is to make use of specific impulse ($I_{sp}$) in its formulation. $I_{sp}$ is defined as the impulse produced per unit weight of propellant. The higher its value, more efficient is the engine. Thus, the thrust is now defined as:

$$\mathbf{T} = \dot{m} I_{sp} g_0 \hat{\mathbf{T}}; \qquad \text{where} I_{sp} = \frac{\int T \mathrm{dt}}{\int \dot{m} g_0 \mathrm{dt}} = \frac{\int (-\dot{m} \mathbf{v}_e \mathrm{dt}}{\int \dot{m} g_0 \mathrm{dt}} \tag{3.62}$$

$I_{sp}$ is merely a scaling factor, that accounts for both momentum thrust and pressure thrust. It is defined with respect to $g_0$ and hence, the thrust equation must include $g_0$, regardless of the value of gravitational acceleration at the position of actual flight. $I_{sp}$ values vary with respect to the pressure difference at the nozzle exhaust. Hence care has to be exercised while using values of $I_s p$, for example at sea-level and at vacuum.

The above equations are applicable to an ideal thruster. A real thruster, however, has losses that reduce the overall performance of the engine. These losses can be determined accurately by testing, and can be incorporated into the force model by an updated value of *actual* $I_{sp}$. Also, real thrusters take a finite time for firing and this effects the overall impulse delivered. These rise and fall times for the thruster can be upto a few hundred milliseconds.

### 3.5.3 ATMOSPHERIC DRAG

There is no clear demarcation that defines the end of Earth's atmosphere. While most (> 99.99% by mass) of the atmosphere is found below the Kármán line, there is still a faint trace of gas molecules that extends upto a few hundreds of kilometres above the Earth's surface. This is responsible for producing aerodynamic forces, most notably drag, on the surface of a satellite. Like in aeronautics, we can write the resulting drag force to be:

$$\mathbf{a}_{\text{drag}} = -\frac{1}{2}\left(\frac{C_D A}{m}\right) \rho v_{\text{rel}} \mathbf{v}_{\text{rel}} \tag{3.63}$$

where, $A$ and $m$ are the frontal area and mass of the spacecraft, $\rho$ is the atmospheric density, $C_D$ is the dimensionless drag coefficient and $\mathbf{v}_{\text{rel}}$ is the relative velocity of the satellite with rotating earth. $C_D$ has a value in the range of 2-3; the value of $\rho$ is dependent on solar flux, diurnal variation, seasonal variation, geomagnetic activity, etc. At an altitude of 750 km, it is in the range of $10^{-12}$kg/m$^3$ during maximum solar activity. Below, the altitude of 200 km, it is the most dominant perturbation while above 1000 km, it can almost always be neglected.

### 3.5.4 RADIATION PRESSURE

Solar radiation pressure is the pressure exerted upon any surface exposed to solar radiation. On Earth, these forces are too small to be detected under normal circumstances, however, the motion of an object in orbit can be perturbed by it (especially for bodies with a large surface area and low mass). The generation of radiation pressure results from the exchange in momentum, when a photon strikes a surface. Using equations from Wakker [2010a], we have the acceleration due to solar radiation pressure, $\mathbf{a}_{\text{rad}}$, as:

$$\mathbf{a}_{\text{rad}} = -C_R \frac{WA}{mc} \hat{\mathbf{r}}_{\text{sat,rad}} \tag{3.64}$$

where, $C_R$ is the satellite's reflectivity, $W$ is the incoming radiation flux or power density, $A$ is area of satellite exposed to the radiation, $c$ is the speed of light and $\mathbf{r}_{\text{sat,rad}}$ is a unit vector from satellite to the source of radiation. The power density $W$ is predominantly from the Sun (appr. 1350 W/m$^2$ for satellites in Earth orbit), and partly from Earth's albedo and infrared radiations (about $300 - 400$ W/m$^2$). Variation in Solar activity, effective time-varying cross-section exposed to radiation, time-varying coefficients of reflectivity are some reasons why modelling the radiation pressure can be a hard exercise.

### 3.5.5 THIRD-BODY ACCELERATION

Until now, we only considered the gravitational forces in a two-body problem. However, in reality, the orbits of bodies are perturbed by numerous other bodies. These perturbations, while small, can manifest to be of substantial magnitude in a few limiting cases or while considering orbits over longer periods. In a n-body problem, the perturbing potential experienced by a satellite (body $i$ of negligible mass, orbiting body $k$) due to other bodies $j$, is given by:

$$U_{\text{pert}} = -\sum_{j \neq k,i} \mu_j \left( \frac{1}{r_{ij}} - \frac{\mathbf{r}_i \cdot \mathbf{r}_j}{r_j^3} \right) \tag{3.65}$$

where $\mathbf{r}_i$ and $\mathbf{r}_i$ are position vectors of bodies $i$ and $j$ from body $k$; $r_{ij}$ is the distance between body $i$ and $j$, and $\mu_j$ is the gravitational parameter of body $j$. The perturbing acceleration is found by:

$$a_{\text{pert}} = -\frac{1}{m_i} \nabla U_{\text{pert}} \tag{3.66}$$

### 3.5.6 CONCLUSION

Numerous forces come into play, while determining accurate osculating orbit of a satellite. The choice to include or exclude some perturbations, is a trade-off between associated increase in accuracy and efforts required. Figure 3.12 gives a good idea of the relative magnitudes of various perturbing forces. It becomes evident that all perturbation forces are a few orders magnitude below nominal gravitational acceleration. Atmospheric drag, $J$-coefficients, Earth albedo radiation pressure decreases with altitude, whereas third body perturbations increase. Solar radiation pressure increases only marginally. At the altitude of Envisat, the strongest contributors are perturbations due to $J_2, J_{2,2}$ coefficients and atmospheric drag, with accelerations in the order of $10^{-2}$, $10^{-4}$, $10^{-5}$ m/s$^2$ respectively. In the timeline of a typical rendezvous mission, which is expected to be a few (<5) hours, the lower magnitude perturbations will have negligible effects on the orbit. Thus, they shall discarded from the analysis.

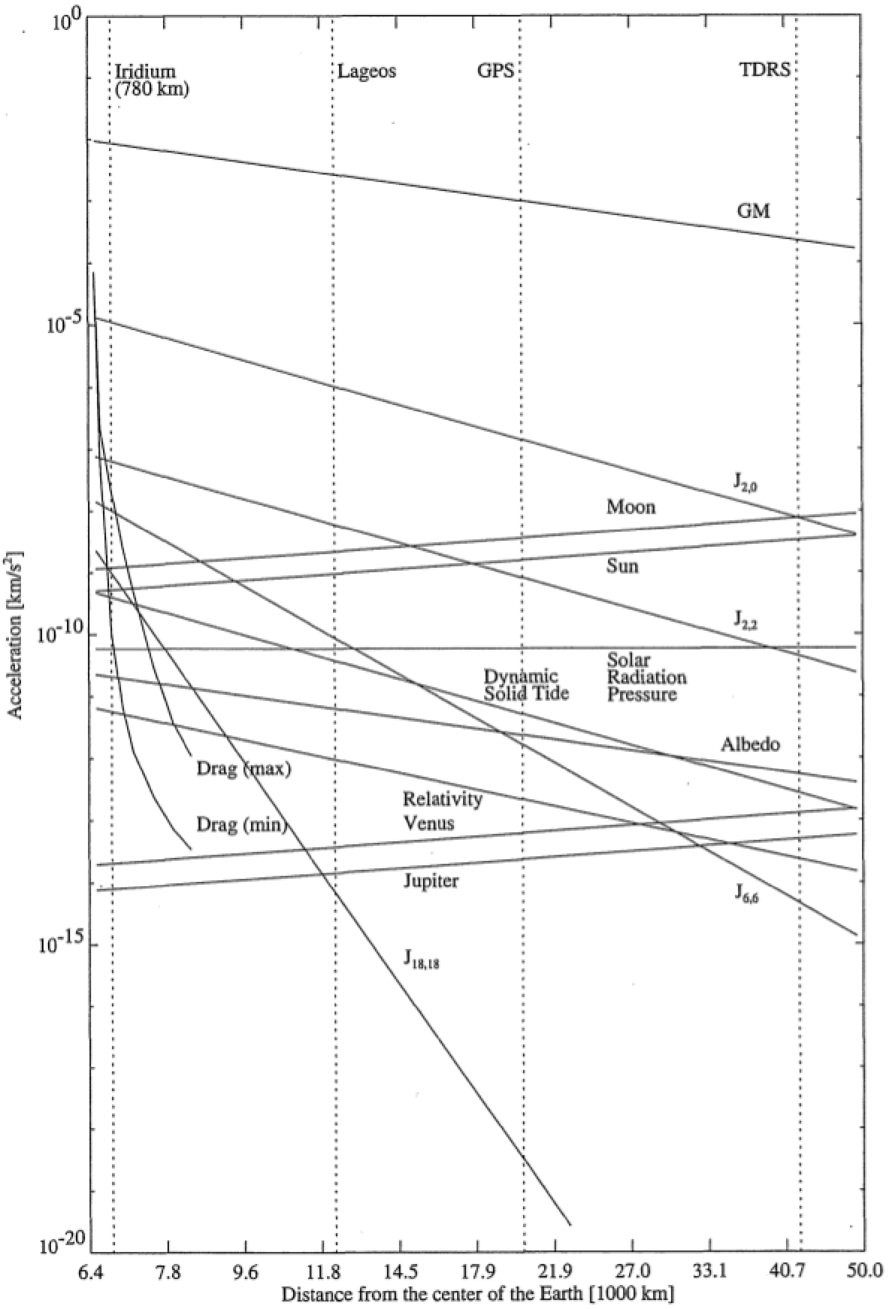


Figure 3.12: Magnitude of acceleration for Earth orbiting satellites [Montenbruck and Gill, 2000].

# 4

# Convex Theory and Optimization

Convex programming unifies linear and quadratic programming into a general form, which is nearly as easy to solve as linear problems but includes a large number of non-convex problems. The ConGAL method employs a Second-Order Conic Program (SOCP), which is a particular form of a convex optimization problem, for optimizing the control variables. To be able to fully understand and appreciate the ConGAL method, a brief overview of convex theory and optimization is presented here. Furthermore, methods to identify a convex function, mathematical operations that retain convexity and transformation of optimization problems from one convex form to another are presented here. These shall be used while trying to modify/improve the ConGAL method. For ease of use and understanding, the choice of symbols in this chapter conform with the conventional choice used in mathematics. However, this causes some symbols to overlap with their applications elsewhere in this thesis. In case of conflict, the scope of symbols defined in this chapter, is restricted to this chapter only.

## 4.1 Introduction

A general optimization problem is of the form

$$\begin{aligned} &\text{minimize} && f_0(\mathbf{x}) \\ &\text{subject to} && f_i(\mathbf{x}) \leq b_i, \qquad i = 1, \ldots, m \end{aligned} \tag{4.1}$$

Here, the vector $\mathbf{x} = (x_1, \ldots, x_n)$ is the optimization variable, the function $f_0 : \mathbb{R}^n \to \mathbb{R}$ is the objective function (or the cost function), the functions $f_i : \mathbb{R}^n \to \mathbb{R}, i = 1, \ldots, m$ are the (inequality) constraints and the constants $b_i, i = 1, \ldots, m$ are the limits or bounds of the constraints. A vector $\mathbf{x}^*$ is called the optimal (or solution) of the problem in Equation (4.1), if it has the smallest value of the objective function among all vectors that satisfy the constraints, i.e., for any $\mathbf{z}$ with $f_1(\mathbf{z}) \leq b_1, \ldots, f_m(\mathbf{z}) \leq b_m$, we have $f_0(\mathbf{z}) \geq f_0(\mathbf{x}^*)$. A maximization problem can be considered by simply taking the negative of the objective function, $-f_0$.

In general, optimization problems can be very hard to solve, especially when the optimization variable, $\mathbf{x}$, has a large number of elements. There are several reasons for this difficulty, as stated by Hindi [2004]:

1. The problem may have local optima, thus the obtained solution may not be the one required.
2. It might be hard to find a feasible point, or the feasible set is not fully connected or in the worst case, the feasible set is empty.
3. Stopping criteria in general optimization algorithms could be arbitrary

4. Optimization problems might have poor convergence rates

5. Numerical problems could cause the minimization algorithm to stop or wander.

However, exceptions exist and not all optimization problems prove to be hard to solve. Some special classes of problems, like the least-squares and linear programming problems are relatively 'easier' to solve. These problems posses special characteristics, which can be exploited to solve them very efficiently and reliably. For example, in linear programming, a local minima is in fact, the global minima; or for a bounded objective function, the optimum value is always attained on the boundary of optimum-level set. Owing to well established theory and numerous approaches that have been developed since the 1950s, solving such problems can be considered as 'mature technology', as noted by Boyd and Vandenberghe [2004].

Another exception to the rule, is the convex optimization problem, a type of non-linear problem. They can be solved efficiently and reliably in polynomial time. They cover a wide range of applications such as automatic control systems, estimation and signal processing, communications and networks, electronic circuit design, statistics, finance, etc.

Least-squares and linear programming are actually, subsets (or special cases) of the more generic convex optimization problem. By definition, convex problems are not linear, but they are conceptually very similar to linear problems. For example, Nesterov and Nemirovskii [1994] showed that interior-point methods, which are very efficient at solving linear problems, can be extended to solve convex problems as well. The 'theory of convex problems is far more straightforward (and complete) than the theory of general non-linear optimization.' as observed by Boyd and Vandenberghe [2004]. In this context, it is best to quote Rockafeller [1993] :

> *..the great watershed in optimization isn't between linearity and nonlinearity, but convexity and nonconvexity*

Convex problems are more general and have greater range of applications than linear problems while being more tractable than non-linear problems. If a problem is formulated in the convex form, it is almost certain that it can be solved [Boyd and Vandenberghe, 2004]. The following sections are based on material found in standard textbooks of convex optimization, like those of Boyd and Vandenberghe [2004] and Nesterov and Nemirovskii [1994].

## 4.2 CONVEX THEORY

This section provides a brief background of the concepts and notation necessary for understanding convex optimization problem. While an immense amount of analysis (with rigorous proofs) of this field is easily available in literature, the scope of this thesis restricts foraying into too much detail.

### 4.2.1 CONVEX SETS

The general equation of a line is $y = \theta x_1 + (1-\theta)x_2$, with $x_1, x_2 \in \mathbb{R}^n$, $\theta \in \mathbb{R}$ and $x_1 \neq x_2$. If we consider only $0 \leq \theta \leq 1$, then the line becomes a line segment. This helps us set up the definition of affine and convex sets.

A set $C \subseteq \mathbb{R}^n$ is *affine*, if for any $\mathbf{x}_1, \mathbf{x}_2 \in C$ and $\theta \in \mathbb{R}$, we have $\theta\mathbf{x}_1 + (1-\theta)\mathbf{x}_2 \in C$, i.e., a line through (or a linear combination of) any two points in $C$ lies in $C$, provided the coefficients in the linear combination add up to 1. This can be generalized to more than two points, such as $\theta_1\mathbf{x}_1 + \theta_2\mathbf{x}_2 + \ldots + \theta_k\mathbf{x}_k$ for $\theta_1 + \theta_2 + \ldots + \theta_k = 1$. If $\mathbf{x}_i$ is in C, then all its affine combinations must also belong in C.

*Convex* sets are similar to affine sets, but instead of a line, only the line segment between any two $\mathbf{x}_1, \mathbf{x}_2 \in C$ lies in $C$. A set $C \subseteq \mathbb{R}^n$ is convex for any $\mathbf{x}_1, \mathbf{x}_2 \in C$ and any $\theta \mid 0 \leq \theta \leq 1$, it holds

$$\theta\mathbf{x}_1 + (1-\theta)\mathbf{x}_2 \in C \tag{4.2}$$

Like in affine sets, a convex combination of points such as $\theta_1\mathbf{x}_1 + \theta_2\mathbf{x}_2 + \ldots + \theta_k\mathbf{x}_k$ also lies in $C$ for $\theta_1 + \theta_2 + \ldots + \theta_k = 1$ and $\theta_i \geq 0, i = 1, \ldots, k$. Geometrically, convex sets are considered to be 'bulging outward, with no dents or kinks in them'. Boyd and Vandenberghe [2004] define a set as convex 'if every point in the set can be seen by every other point, along an unobstructed straight path between them'. Convex sets must possess a non-empty interior, as otherwise line segments between some points will not be contained in the set. Thus, for example, a solid sphere is a convex set whereas a hollow sphere is not. Figure 4.1 gives examples of some convex and non-convex sets.

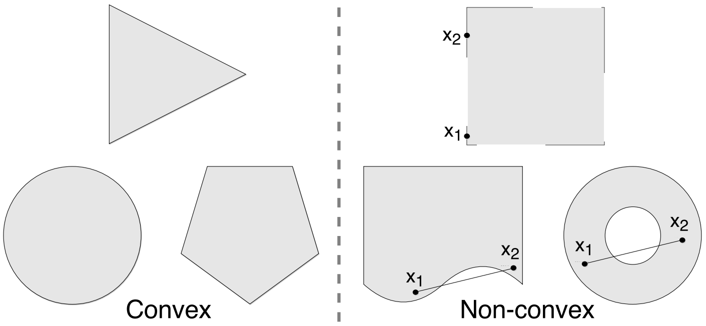


Figure 4.1: Examples of convex and non-convex sets. The set on the right is non-convex because a line joining two interior points lies outside the set. Adapted from Hindi [2004]

A set $C$ is called a *cone* if for every $\mathbf{x} \in C$ and $\theta \geq 0$, we have $\theta\mathbf{x} \in C$. In other words, a ray emanating from its origin and passing through any point in the cone, lies in the cone. The cone is said to be pointed if it contains the origin $\mathbf{0}$. Like in affine and convex sets, one can also define a conic combination. A cone $C$ is a *convex cone*, if for any $\mathbf{x}_1, \mathbf{x}_2, \in C$, and $\theta_1, \theta_2 \geq 0$, we have $\theta_1\mathbf{x}_1 + \theta_2\mathbf{x}_2 \in C$. Thus, besides being a cone, a convex cone should also contain the line segment between any two points in the cone. Essentially, for any two points $\mathbf{x}_1, \mathbf{x}_2 \in C$, the pie-slice between them and the origin, lies in $C$ as well.

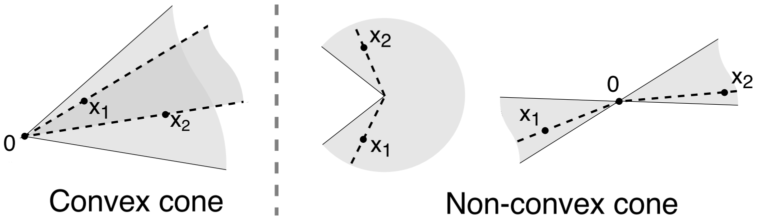


Figure 4.2: Examples of convex and non-convex cones. The dotted lines show that for every $\mathbf{x} \in C$ and $\theta \geq 0$, we have $\theta\mathbf{x} \in C$. In the convex cone, we also see that the pie-slice between any two points and the origin, also lies inside the cone. Adapted from Hindi [2004]

The *norm cone* is defined as

$$C = \{(\mathbf{x}, t) \mid \|\mathbf{x}\| \leq t\} \subseteq \mathbb{R}^{n+1}$$

where $\|\cdot\|$ is any norm on $\mathbb{R}^n$. If we consider the the Euclidean norm, then we have what is called as a *second-order cone*. It is defined as

$$\begin{aligned} C &= \{(\mathbf{x}, t) \mid \|\mathbf{x}\|_2 \leq t\} \\ &= \left\{ \begin{bmatrix} \mathbf{x} \\ t \end{bmatrix} \middle| \begin{bmatrix} \mathbf{x} \\ t \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} I & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ t \end{bmatrix} \leq 0, t \geq 0 \right\} \end{aligned} \tag{4.3}$$

The second-order cone in $\mathbb{R}^3$ is also called as *ice-cream cone* or *Lorentz cone*. Figure 4.3 illustrates the cone.

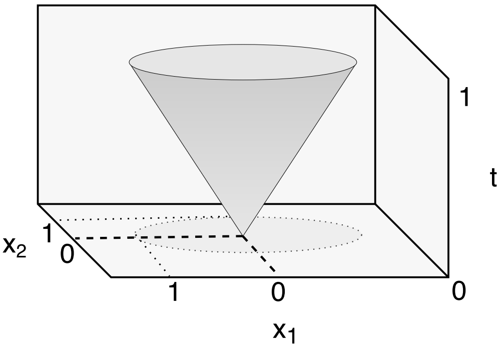


Figure 4.3: Lorentz cone, a second-order cone in $\mathbb{R}^3$. Adapted from Hindi [2004]

To conclude, consider a set $C$, $x_1, x_2 \in C$ and $\theta_1, \theta_2 \in \mathbb{R}$. Assume $y = \theta_1 x_1 + \theta_2 x_2$ lies in $C$. The set $C$ is said to be:

- Affine, if $\{\theta_1 + \theta_2 = 1\}$
- Convex, if $\{\theta_1 + \theta_2 = 1; \theta_1, \theta_2 \geq 0\}$
- Cone, if $\{\theta_1, \theta_2 \geq 0\}$

### 4.2.2 GENERALIZED INEQUALITY

At this point it is important to introduce generalized inequality. The normal (or scalar) inequality, or simply just the inequality $\leq$ and $\geq$ is used to relate between magnitude of two scalar elements. For vectors or matrices, however, such inequalities make little sense (or often, causes confusion). Thus, a generalized inequality is often employed to relate elements, regardless of their type.

A generalized inequality is denoted by $\succeq$ and $\preceq$. For two vectors, $\mathbf{x}, \mathbf{y}$, the generalized inequality represents component wise inequality. Thus,

$$\mathbf{x} \preceq \mathbf{y} \Longleftrightarrow x_i \leq y_i, i = 1, 2, \ldots, n$$

In cones, we have the generalized inequality as $\succeq_K$ and $\preceq_K$ , where the $K$ identifies the cone with which it is defined. It is defined as:

$$\mathbf{y} \in K \Longleftrightarrow 0 \preceq_K \mathbf{y} \tag{4.4}$$

This formulation implies that

$$\mathbf{x} \preceq_K \mathbf{y} \Longleftrightarrow \mathbf{y} - \mathbf{x} \in K$$

If for a vector $\mathbf{y}$, we know that

$$0 \preceq_K \mathbf{y} \tag{4.5}$$

then we say that $\mathbf{y}$ belongs to the cone $K$. As an inequality, it also implies the following:

$$\mathbf{x} \succeq_K \mathbf{y} \quad \Longrightarrow \quad \alpha\mathbf{x} \succeq_K \alpha\mathbf{y} \quad \forall \alpha > 0$$

$$\mathbf{x} \succeq_K \mathbf{y} \quad \Longrightarrow \quad \alpha\mathbf{x} \preceq_K \alpha\mathbf{y} \quad \forall \alpha < 0$$

$$\mathbf{x} \succeq_K \mathbf{y},\ \mathbf{x} \preceq_K \mathbf{y} \quad \Longrightarrow \quad \mathbf{x} = \mathbf{y}$$

Sometimes, when the meaning is clear, $\preceq_K$ is simply written as $\preceq$. It is interesting to note that when $K = \mathbb{R}_+$ (the non-negative orthant), the generalized inequality $\preceq_K$ is simply the element-wise inequality

$$\mathbf{x} \preceq_{\mathbb{R}_+} \mathbf{y} \Longleftrightarrow x_i \leq y_i, i = 1, 2, \ldots, n$$

### 4.2.3 CONVEX FUNCTIONS

Now that the convex set and related concepts are defined, its easier to define a convex function and understand some of its special properties, which make it easier to solve convex optimization problems.

A function $f : \mathbb{R}^n \to \mathbb{R}$ is a *convex fucntion* if **dom** $f$ (the domain of $f$) is a convex set, and if for all $\mathbf{x}, \mathbf{y} \in$ **dom** $f$, and $\theta \in [0,1]$, we have

$$f(\theta\mathbf{x} + (1-\theta)\mathbf{y}) \leq \theta f(\mathbf{x}) + (1-\theta) f(\mathbf{y}) \tag{4.6}$$

The function $f$ is strictly convex if strict inequality holds in equation (4.6), $\mathbf{x} \neq \mathbf{y}$ and $0 < \theta < 1$. Also, if a function $f$ is (strictly) convex, then $-f$ is (strictly) concave [6].

Consider a convex function $f : \mathbb{R} \to \mathbb{R}$. Then geometrically, it implies that line segment between $(x, f(x)$ and $y, f(y)$, lies above the graph of $f$. This can be seen in Figure 4.4.

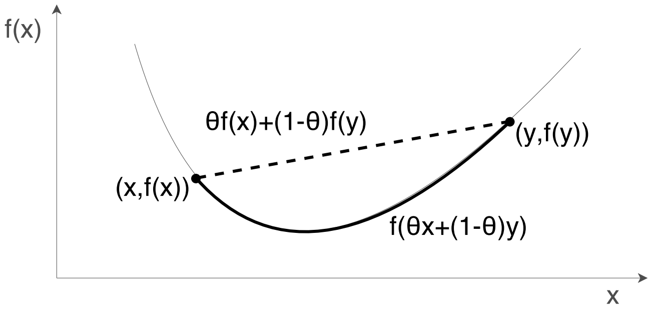


Figure 4.4: The graph of a convex function $f : \mathbb{R} \to \mathbb{R}$. Line joining any two points lies above the graph.

It is not always easy to determine if a function is convex. For a function with higher order or large number of variables, one cannot really plot the graph and check for convexity (it might even be impossible). Also, it becomes harder to determine whether the **dom** $f$ is a convex set or not. Fortunately, the above definition is not the only way in which the convexity of a function can be characterized. Convex functions can also be defined in the following ways:

- **First-order condition**:
  For a function $f : \mathbb{R}^n \to \mathbb{R}$ that is differentiable, it is convex if and only if **dom** $f$ is convex and

  $$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^{\mathsf{T}}(\mathbf{y} - \mathbf{x}) \tag{4.7}$$

  holds for all $x, y \in$ **dom** $f$. This is illustrated in Figure 4.5.

- **Second-order condition**:
  For a function $f : \mathbb{R}^n \to \mathbb{R}$ that is twice-differentiable, it is convex if and only if **dom** $f$ is convex and its Hessian is positive semidefinite.

  $$\forall \mathbf{x} \in \mathbf{dom}\, f, \quad \nabla^2 f(\mathbf{x}) \geq 0.$$

- **Epigraph is convex**:
  For a function $f : \mathbb{R}^n \to \mathbb{R}$, the epigraph is defined as

  $$\mathbf{epi}\, f = \{(x,t) \mid x \in \mathbf{dom}\, f, f(x) \leq t\}$$

  which is a subset of $\mathbb{R}^{n+1}$. 'Epi' means above, so epigraph means above the graph. See Figure 4.6 for illustration. The function $f$ is convex if and only if its epigraph is a convex set.

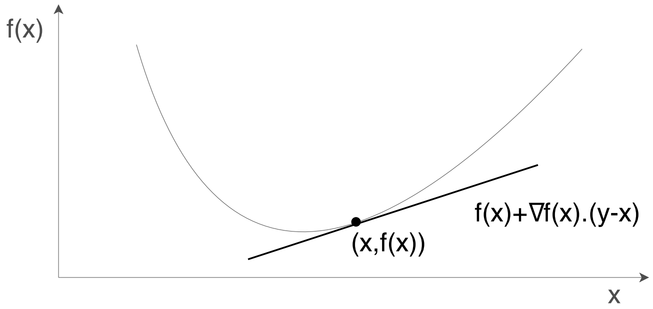


Figure 4.5: The first order condition for a convex function $f : \mathbb{R} \to \mathbb{R}$. The slope at any point lies below the curve.

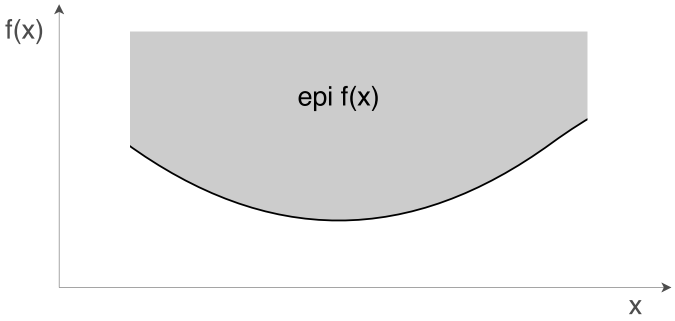


Figure 4.6: The epigraph of a function $f : \mathbb{R} \to \mathbb{R}$ is shaded. If the epigraph is a convex set then the function is convex.

The first-order condition is an affine function of **y**, which can be identified as the first-order Taylor approximation of $f$ near **x**. One can thus conclude that the first-order Taylor approximation is a global under-estimator of the convex function. As Boyd and Vandenberghe [2004] puts it, '$ldots$from *local information* about a convex function (*i.e.*, its value and derivative at a point) we can derive *global information*'. This is an important property of convex functions, because it goes onto say that if $\nabla f(\mathbf{x}) = 0$, then for all $\mathbf{y} \in \mathbf{dom}\, f$, $f(\mathbf{y}) \geq f(x)$, *i.e.*, $x$ is a global minimizer of the function $f$. Thus, if for a local minimizer we have, $\nabla f(x) = 0$, we can conclude with certainty that it is also the global minimizer

The second-order condition can be better understood with an example. For a function $f : \mathbb{R} \to \mathbb{R}$, this condition reduces to the simple condition $f''(x) \geq 0$ and the **dom** $f$ is an interval. Thus, the derivative is non-decreasing and the graph has a positive or upward curvature at $x$.

Note that one cannot drop the requirement **dom** $f$ be a convex set, in the first- or second-order characterization of a convex function. As an example, consider $f(x) = 1/x^2$. It is not a convex function despite having $f''(x) > 0 \forall x \in$ **dom** $f$. This is because **dom** $f = \{x \in \mathbb{R} \mid x \neq 0\}$ is a non-convex set.

## 4.3 EXAMPLES OF CONVEX FUNCTIONS

As discussed in the previous section, it is hard to determine if a function is convex or not. In this section, some common functions are classified as convex and concave. Reformulating a seemingly hard function into any of these convex functions could help identify convexity. This section does not prove if these functions are convex or not, but rather states the results from Boyd and Vandenberghe [2004].

[6] This is an important observation because the maximization of a concave function, is similar to the minimization of the respective convex formulation.

It is already mentioned that linear functions are convex functions. For linear functions, the equality in Equation (4.6) would hold true and thus, it would be a convex function.

The following functions on domain $\mathbb{R}$, with variable $x$ are convex:

1. *Exponential*: $e^{ax}$ is convex on $\mathbb{R}$, for any $a \in \mathbb{R}$.
2. *Powers*: $x^a$ is convex on $\mathbb{R}_{++}$ (positive real numbers) when $a \geq 1$ or $a \leq 0$ and concave for $0 \leq a \leq 1$.
3. *Powers of absolute value*: $|x|^p$, for $p \geq 1$, is convex on $\mathbb{R}$
4. *Logarithm*: $\log(x)$ is concave on $\mathbb{R}_{++}$
5. *Negative entropy*: $xlog(x)$ on $\mathbb{R}_+$ with the function defined as 0 for $x = 0$, is convex.

The following functions on $\mathbb{R}^n$ are convex:

1. *Norms*: Every norm on $\mathbb{R}^n$ is convex.
2. *Max functions*: $f(\mathbf{x}) = \max\{x_1, \ldots, x_n\}$ is convex.
3. *Quadratic-over-linear function*: The function $f(x, y) = x^2/y$ is a convex function when $\mathbf{dom}\, f = \mathbb{R} \times \mathbb{R}_{++}$.
4. *Log-determinant*: The function $f(\mathbf{X}) = \log(\det \mathbf{X})$ is concave on $\mathbf{dom}\, f = S^n_{++}$.

It is also important to note the following operations that preserve convexity:

1. *Non-negative scaling*: If a function $f$ is convex, and $\alpha \geq 0$, then the function $\alpha f$ is convex too.
2. *Addition*: If $f_1$ and $f_2$ are convex, then their sum $f_1 + f_2$ is convex too.
3. *Non-negative weighted sums*: Obviously, from the above two operations, one can see that

$$f = w_1 f_1 + \ldots + w_m f_m$$

   is convex too if $w_i \geq 0$ and $f_i$ is convex, for $i = 1, \ldots, m$.
4. *Composition with affine mapping*: Consider $f : \mathbb{R}^n \to \mathbb{R}$, $\mathbf{A} \in \mathbb{R}^{n \times m}$, and $\mathbf{b} \in \mathbb{R}^n$. We define $g : \mathbb{R}^m \to \mathbb{R}$ by

$$g(\mathbf{x}) = f(\mathbf{Ax} + \mathbf{b})$$

   with $\mathbf{dom}\, g = \{\mathbf{x} \mid \mathbf{Ax} + \mathbf{b} \in \mathbf{dom}\, f\}$. If $f$ is convex (or concave), then $g$ is also convex (or concave).
5. *Point-wise maximum and supremum*: If $f_1$ and $f_2$ are convex function then their point-wise maximum $f$, defined by

$$f(\mathbf{x}) = \max\{f_1(\mathbf{x}), f_2(\mathbf{x})\},$$

   with $\mathbf{dom}\, f = \mathbf{dom}\, f_1 \cap \mathbf{dom}\, f_2$, is also convex.
6. *Minimization*: If $f$ is convex in $(\mathbf{x}, \mathbf{y})$, and $C$ is a convex non-empty set, then the function

$$g(\mathbf{x}) = \inf_{\mathbf{y} \in C} f(\mathbf{x}, \mathbf{y})$$

   is convex in $\mathbf{x}$, provided $g(\mathbf{x}) > -\infty$ for all $x$. Here, inf is short for infimum, and refers to the greatest lower bound.

## 4.4 CONVEX OPTIMIZATION PROBLEM

### 4.4.1 GENERAL PROBLEM FORMULATION

A convex optimization problem is one of the form

$$\begin{aligned} \text{minimize} \quad & f_0(\mathbf{x}) & \\ \text{subject to} \quad & f_i(\mathbf{x}) \leq 0, & i = 1,\dots,m \\ & \mathbf{a}_i^\mathsf{T}\mathbf{x} = \mathbf{b}_i, & i = 1,\dots,p, \end{aligned} \tag{4.8}$$

where $f_0,\dots,f_m$ are convex functions. It is different from the standard optimization problem in Equation (4.1) because the objective and the inequality constraint functions are convex and the equality constraint functions are affine. It is quite evident that the feasible set of the convex optimization problem is convex, as it is an intersection of the domain of the problem. The domain of the problem is given by

$$\mathcal{D} = \bigcap_{i=0}^{m} \mathbf{dom}\, f_i,$$

which is a convex set.

A fundamental property of convex optimization problems is that any locally optimal point is also globally optimal. Thus, it is suffice to find a point that is 'optimal' and not worry about local or global optimal. The first order condition used to define the convex function, offers an optimality criteria, which can be used to identify an optimal point. Let us assume that the objective $f_0$ in Equation (4.8) is differentiable. From the first-order condition, for all $x, y \in \mathbf{dom}\, f_0$, we have

$$f_0(y) \geq f_0(x) + \nabla f_0(x)^\mathsf{T}(y-x).$$

Then $x$ is optimal if and only if $x \in \mathcal{D}$ and

$$\nabla f_0(x)^\mathsf{T}(y-x) \geq 0 \;\; \forall y \in \mathcal{D}$$

### 4.4.2 SPECIAL FORMS

Based on the type of objective function and constraints involved in a convex problem, one can distinguish some commonly occurring families of problems. These families share similar structure of problems and have dedicated solvers designed for them. Being able to transform a general convex problem into one of these families could help understand the problem better and possibly even solve it quicker.

1. Linear program (LP):
   As mentioned earlier, one can consider a linear optimization problem to be a special case of convex problems. A linear problem is one in which the onbjective and constraint functions are all affine. A general linear problem has the form:

   $$\begin{aligned} \text{minimize} \quad & \mathbf{c}^\mathsf{T}\mathbf{x} + \mathbf{d} \\ \text{subject to} \quad & \mathbf{G}\mathbf{x} \preceq \mathbf{h} \\ & \mathbf{A}\mathbf{x} = \mathbf{b} \end{aligned} \tag{4.9}$$

   where $\mathbf{G} \in \mathbb{R}^{m\times n}$ and $\mathbf{A} \in \mathbb{R}^{p\times n}$. It is common to omit the $\mathbf{d}$ as it does not affect the optimal set. Other formulations of LP exist as well, like *standard* form (only inequalities are componentwise non-negativity constraints), or *inequality* form (no equality constraints).

2. Quadratic program (QP):
   A convex problem is called QP if the objective function is convex quadratic and the constraint

functions are affine. Its general form is:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\mathbf{x}^{\mathsf{T}}\mathbf{P}\mathbf{x} + \mathbf{q}^{\mathsf{T}}\mathbf{x} + \mathbf{r} \\ \text{subject to} \quad & \mathbf{G}\mathbf{x} \preceq \mathbf{h} \\ & \mathbf{A}\mathbf{x} = \mathbf{b} \end{aligned} \tag{4.10}$$

where $\mathbf{P} \in \mathbf{S}_{+}^{n}$, $\mathbf{G} \in \mathbb{R}^{m\times n}$ and $\mathbf{A} \in \mathbb{R}^{p\times n}$. If the QP has quadratic inequality constraints (like $(1/2)\mathbf{x}^{\mathsf{T}}\mathbf{P}_i\mathbf{x}+\mathbf{q}_i^T\mathbf{x}+\mathbf{r}_i \leq 0$), then it is called quadratically constrained quadratic program (QCQP). Linear program is a special case of QCQP (and QP) with $\mathbf{P}_i = 0$, whereas least-square problems are unconstrained QPs. Thus, QCQPs are more general than LPs

3. Second-order cone programming (SOCP):
This problem has a second-order cone constraint, and hence the name. Its general form is:

$$\begin{aligned} \text{minimize} \quad & \mathbf{f}^{\mathsf{T}}\mathbf{x} \\ \text{subject to} \quad & \|\mathbf{A}_i\mathbf{x} + \mathbf{b}_i\|_2 \leq \mathbf{c}_i^T\mathbf{x} + d_i \qquad i = 1,\ldots,m \\ & \mathbf{F}\mathbf{x} = \mathbf{g} \end{aligned} \tag{4.11}$$

where $\mathbf{A}_i \in \mathbb{R}^{n_i\times n}$ and $\mathbf{F} \in \mathbb{R}^{p\times n}$. When $\mathbf{c}_i = 0; i = 1,\ldots,m$, the SOCP (4.11) is equivalent to QCQP (as obtained by squaring each constraint). Similarly, if we have $\mathbf{A}_i = 0; i = 1,\ldots,m$, the SOCP is equivalent to a general LP. SOCP are more general than QCQP, and thus, LP.

4. Conic programming:
They have a linear objective and one generalized inequality constraint over the cone $K \subseteq \mathbb{R}^m$.

$$\begin{aligned} \text{minimize} \quad & \mathbf{f}^{\mathsf{T}}\mathbf{x} \\ \text{subject to} \quad & \mathbf{M}\mathbf{x} + \mathbf{p} \preceq_K 0 \\ & \mathbf{F}\mathbf{x} = \mathbf{g}. \end{aligned} \tag{4.12}$$

When $K$ is a non-negative orthant (an orthant is the analogue in $n$-dimensional space of a quadrant in a plane), then the problem reduces to LP.

At this point, one can easily conclude that conic problem is the most general formulation of the above described families of convex problems. A solver meant for conic problems, should have no difficulty handling these other problems as well. As observed, it is possible to inter-convert between these problems by suitably considering some elements to be zero or restricting their range. However, an exception to that is the conversion between SOCP and conic problem. A conic problem uses generalized inequalities over cones (discussed in sub-section 4.2.2) whereas the SOCP uses the commonly used element-wise inequality. This leap in usage of inequalities needs to be handled specially.

Consider a SOCP like the one described above in Equation (4.11) that needs to be written in a form similar to Equation (4.12). Comparing the two forms, we can already see, that the optimization function ($\mathbf{f}^{\mathsf{T}}\mathbf{x}$) and the equality constraint ($\mathbf{F}\mathbf{x} = \mathbf{g}$) are the same, and thus, these remain unchanged. The inequality constraint in Equation (4.11) is to be rewritten as a generalized inequality in Equation (4.12) and that relation is found by having a look at Equation (4.3). We re-write Equation (4.3) for some vector $\mathbf{X} = \begin{bmatrix} X_0 & X_1 & \ldots & X_N \end{bmatrix}^{\mathsf{T}} = \begin{bmatrix} X_0 & \mathbf{Y} \end{bmatrix}^{\mathsf{T}}$ as:

$$\begin{aligned} & C = \{\mathbf{X} \mid \|\mathbf{Y}\|_2 \leq X_0\} \\ \Rightarrow\ & \mathbf{X} \in C \\ \Rightarrow\ & \mathbf{X} \succeq_C 0 \end{aligned} \tag{4.13}$$

Next we define the inequality constraint of Equation (4.11) to be a part of cone $\mathcal{Q}^{n_i}$, and re-write in the form shown in Equation 4.13:

$$\begin{aligned} \mathcal{Q}^{n_i} &= \left\{ \begin{bmatrix} \mathbf{c}_i^T\mathbf{x} + d_i & \mathbf{A}_i\mathbf{x} + \mathbf{b}_i \end{bmatrix}^{\mathsf{T}} \mid \|\mathbf{A}_i\mathbf{x} + \mathbf{b}_i\|_2 \leq \mathbf{c}_i^T\mathbf{x} + d_i \right\} \\ \Rightarrow \begin{bmatrix} \mathbf{c}_i^T\mathbf{x} + d_i \\ \mathbf{A}_i\mathbf{x} + \mathbf{b}_i \end{bmatrix} &= \begin{bmatrix} \mathbf{c}_i^T \\ \mathbf{A}_i \end{bmatrix} \mathbf{x} + \begin{bmatrix} d_i \\ \mathbf{b}_i \end{bmatrix} \succeq_{\mathcal{Q}^{n_i}} 0 \end{aligned} \tag{4.14}$$

We see the cone $\mathcal{Q}^{n_i} \subseteq \mathbb{R}^{n_i} \times \mathbb{R}$ and has a dimension $n_i + 1$. For each $i$ in Equation (4.11), we can define such a cone. The resulting cone, which is a union of these cones, is given as:

$$K = \mathcal{Q}^{n_1} \times \mathcal{Q}^{n_2} \times \cdots \times \mathcal{Q}^{n_m} \tag{4.15}$$

Cone $K$ denotes the feasible set for the optimization problem. We further define **M** and **p** as block matrices with vector and matrix elements from **A**, **b**, **c**, **d** as[7]:

$$M = -\begin{bmatrix} \mathbf{A}_1 \\ \mathbf{c}_1 \\ \mathbf{A}_2 \\ \mathbf{c}_2 \\ \vdots \\ \mathbf{A}_m \\ \mathbf{c}_m \end{bmatrix} \qquad p = -\begin{bmatrix} \mathbf{b}_1 \\ d_1 \\ \mathbf{b}_2 \\ d_2 \\ \vdots \\ \mathbf{b}_m \\ d_m \end{bmatrix} \tag{4.16}$$

The negative sign helps reverse the $\succeq$ to $\preceq$, so that the equation is of the form shown in Equation (4.12).

[7] Grant, M., personal communication, May 2015

# 5 Convex Guidance

The highly constrained nature of a realistic rendezvous mission problem make it a very challenging optimal control problem to solve numerically. Liu [2013] developed a guidance algorithm based on convex optimization theory that has the potential to solve such problems. This chapter describes and derives the Convex Guidance Algorithm by Liu (ConGAL) in a step-by-step process; the intent is to understand the underlying principle so that a similar algorithm can be implemented for solving the Envisat rendezvous problem. The essential components of the methodology include lossless relaxation of the original rendezvous problem and a novel successive solution procedure, termed as the Method Of Successive Approximations (MOSA). The resulting set of problems can be solved with a primal-dual interior point algorithm, which is a reliable method to handle convex optimization problems. Some salient features of ConGAL, as supported by Lu and Liu [2013], include application to any non-circular orbit without the need of user supplied initial guesses, propagation of state in inertially, and the possibility of including perturbations. To avoid repeated references, it is stated that this chapter is entirely based, but duly re-phrased, from the description found with Liu [2013], Lu and Liu [2013] and Liu and Lu [2013].

## 5.1 Problem formulation

The three-dimensional equation of motion of a point-mass spacecraft in inverse-square gravity field (*i.e.*, Newtonian gravity) in Earth-centred inertial reference frame ($\mathcal{I}$) is

$$\ddot{\mathbf{r}} = -\frac{\mu_E}{r^3}\mathbf{r} + \frac{\mathbf{T}}{m} \tag{5.1}$$

where $\mu_E$ is the gravitational parameter of Earth, $\mathbf{r} \in \mathbb{R}^{3\times 1}$ is a vector from centre of Earth to the spacecraft and $r = \|\mathbf{r}\|$, $\mathbf{T}$ is the thrust vector of the spacecraft, and $m$ is the instantaneous mass. The above equation can be re-written in dimensionless form as[8]:

$$\dot{\mathbf{r}} = \mathbf{V}, \qquad \mathbf{r}(0) = \mathbf{r}_0 \tag{5.2}$$

$$\dot{\mathbf{V}} = -\frac{1}{r^3}\mathbf{r} + \frac{\mathbf{T}}{m}, \qquad \mathbf{V}(0) = \mathbf{V}_0 \tag{5.3}$$

$$\dot{m} = -\frac{1}{v_{\text{ex}}}\|\mathbf{T}\|, \qquad m(0) = m_0 \tag{5.4}$$

where $\mathbf{r}$ is normalized by $R_0$, the radius of Earth; the inertial velocity vector $\mathbf{V}$ is normalized by $\sqrt{g_0 R_0}$ with $g_0$ being the gravitational acceleration at $R_0$; $m$ is normalized by $m_0$, the initial mass of

[8] A proof for the normalized set of equations is not provided by Liu [2013] or his collaborators. They are derived (albeit with a slight modification) in Chapter 7, while discussing an alternative set of scaling factors for normalization.

the spacecraft; $\mathbf{T}$ by $m_0 g_0$; the fuel consumption is dependent on $\|\mathbf{T}\|$ and $v_{\text{ex}} > 0$, the dimensionless constant effective exhaust velocity of the engine. The differentiation in these equations is with respect to a dimensionless time $t$ that is real time normalized by $\sqrt{R_0/g_0}$. The values $\mathbf{r}_0$ and $\mathbf{V}_0$ are initial position and velocity of the spacecraft, respectively.

Let $\mathbf{x} = (\mathbf{r}^\mathsf{T}\mathbf{V}^\mathsf{T})^\mathsf{T}$ be the state vector of the chaser, $\mathbf{x}_t(t) = (\mathbf{r}_t^\mathsf{T}\mathbf{V}_t^\mathsf{T})^\mathsf{T}$ be the state vector of the target (Envisat) and $t_f$ be the specified final time at which docking occurs / mission ends. The docking axis is defined by a unit vector $\mathbf{1}_n \in \mathbb{R}^3$, attached to the target. It is a function of time in the inertial frame.

In an optimization problem, a performance index $J$ is minimized, subject to a set of constraints. For our rendezvous problem, we would like to find a fuel optimal trajectory, but it could well be a different performance index. We start by defining our optimization problem as follows [Lu and Liu, 2013]:

**Problem 1:**

$$J = \int_0^{t_f} \|\mathbf{T}\|\, \mathrm{d}t \tag{5.5}$$

$$\|\mathbf{T}\| \leq T_{\max} \tag{5.6}$$

$$\left\|\mathbf{r}(t) - \mathbf{r}_t(t)\right\| \cos\alpha \leq \mathbf{1}_n^\mathsf{T}(t) \cdot (\mathbf{r}(t) - \mathbf{r}_t(t)) \tag{5.7}$$

$$\mathbf{1}_n^\mathsf{T}(t)\mathbf{T} \leq \|\mathbf{T}\| \cos\beta \tag{5.8}$$

$$\mathbf{C}_f(t_f)\mathbf{x}(t_f) + \mathbf{d}_f(t_f) = 0 \tag{5.9}$$

$$\mathbf{C}_i(t_i)\mathbf{x}(t_i) + \mathbf{d}_i(t_i) = 0, \qquad i = 1, \dots, l, \quad 0 < t_1 < \cdots < t_l < t_f \tag{5.10}$$

The above problem is not complete because it does not involve the equations of motions yet, however, for the time being, let us proceed as it is. The performance index to be minimized is the total impulse delivered, which is a measure of the propellant consumed, as seen in Equation (5.5). It is subject to constraints, Equations (5.6) - (5.10). The thrust produced by the spacecraft has an upper bound $T_{\max}$, which is the maximum thrust divided by $m_0 g_0$. When $T = 0$, the spacecraft is said to be coasting.

Equation (5.7) specifies the rendezvous trajectory approach corridor defined by a cone with half angle $\alpha$ at the docking point of the target. This constraint limits the final approach trajectory inside the conic corridor, in the final phase of the mission (when the distance between the two vehicles is very small). Equation (5.8) restricts the thrust direction, such that there is no thrust plume impingement on the target. The thrust plume is in the opposite direction of the thrust vector $\mathbf{T}$ and it is to be pointed away from the target by a minimum angle $\beta > 0$. When $\beta = 0$, the thrust vector can be in any direction, however when $\beta = \pi$, the thrust vector can only points towards the target along the docking axis (such that the plume points away from the target). For intermediate $\beta$, the angle between thrust vector $\mathbf{T}$ and docking axis $1_n$ should be greater than $\beta$. The above two constraints are applicable only when the chaser is very close to the target.

The constraint(s) in Equation (5.9) specify the terminal condition(s) of the mission. For example,

$$\mathbf{x}(t_f) - \mathbf{x}(t_f) = 0 \tag{5.11}$$

specifies that the state vector of the target and chaser should be equal at the end of the RPO mission. In other words, the chaser has docked to the target. In addition to having terminal conditions, the mission may also have some intermediate conditions like hold points, which need to be defined. This can be accommodated using Equation (5.10). For example,

$$[I_{3\times3}\ 0_{3\times3}]\ \mathbf{x}(t_i) - (\mathbf{r}_t(t_i) + \boldsymbol{\delta}_i) = 0 \tag{5.12}$$

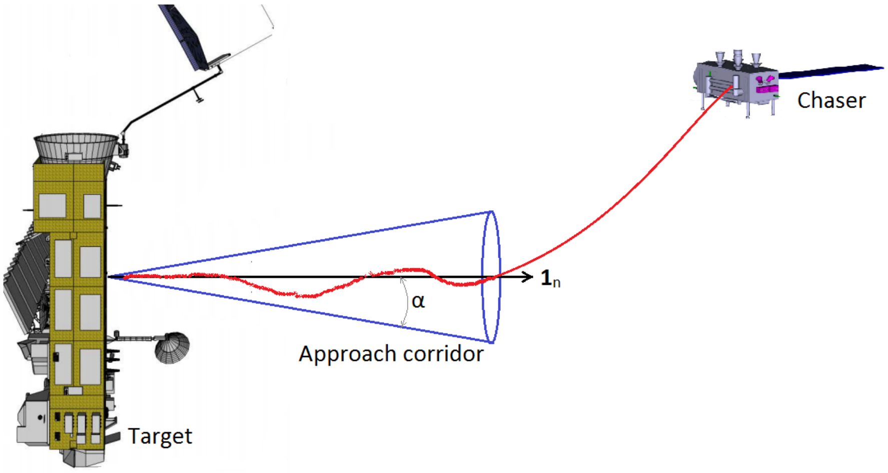


Figure 5.1: Approach corridor for rendezvous defined by a cone with half angle $\alpha$ along the docking axis $\mathbf{1}_n$. Envisat and chaser satellite image from ESA [2012], and based upon Lu and Liu [2013].

specifies a hold point at a distance $\boldsymbol{\delta}_i$ from the target. Note that, $\mathbf{C}_i$ and $\mathbf{d}_i$ at different interior points $t_i$, may have different dimensions. This allows the flexibility of constraining either or both of position and velocity at different points.

The rendezvous problem consists of finding the thrust vector history $\mathbf{T}^*(t)$ such that, from given initial condition $\mathbf{x}(0)$ and $m_0$, the chaser trajectory $\mathbf{x}(t)$ from the equations of motions (Equations (5.2) - (5.4)) and $\mathbf{T}^*(t)$ satisfies the constraints (Equations (5.6) - (5.10)) while minimizing the performance index Equation (5.5). If this problem is repeatedly solved onboard with the current chaser state as the initial condition, the solution can provide closed-loop rendezvous guidance.

## 5.2 RELAXATION

To transform the current problem into an SOCP (as shown in Equation (4.11)), it requires that the problem have a linear objective and linear, quadratic or second-order cone constraints (all of which are convex). We see that Equation (5.5) contains the norm of a vector and is thus non-linear. Also, Equation (5.4) is non-linear while Equation (5.8) is non-convex for $\beta \in [0, \pi/2]$. To convexify the problem, we begin by introducing a slack control variable $\eta$ in place of $\|\mathbf{T}\|$. Thus the original problem now changes to the following [Lu and Liu, 2013] :

**Problem 2:**

$$J = \int_0^{t_f} \eta \mathrm{d}t \tag{5.13}$$

$$\|\mathbf{T}\| \le \eta \tag{5.14}$$

$$0 \le \eta \le T_{\max} \tag{5.15}$$

$$\left\|\mathbf{r}(t) - \mathbf{r}_t(t)\right\| \cos\alpha \le \mathbf{1}_n^{\mathsf{T}}(t) \cdot (\mathbf{r}(t) - \mathbf{r}_t(t)) \tag{5.16}$$

$$\mathbf{1}_n^{\mathsf{T}}(t)\mathbf{T} \le \eta \cos\beta \tag{5.17}$$

$$\mathbf{C}_f(t_f)\mathbf{x}(t_f) + \mathbf{d}_f(t_f) = 0 \tag{5.18}$$

$$\mathbf{C}_i(t_i)\mathbf{x}(t_i) + \mathbf{d}_i(t_i) = 0, \qquad i = 1, \ldots, l, \quad 0 < t_1 < \cdots < t_l < t_f \tag{5.19}$$

with the dynamics given by:

$$\dot{\mathbf{r}} = \mathbf{V} \tag{5.20}$$

$$\dot{\mathbf{V}} = -\frac{1}{r^3}\mathbf{r} + \frac{\mathbf{T}}{m} \tag{5.21}$$

$$\dot{m} = -\frac{1}{v_{\mathrm{ex}}}\eta \tag{5.22}$$

We see that the performance index (Equation (5.13)) is now linear. Two new *relaxed* constraints, Equations (5.14) and (5.15) have replaced Equation (5.6). Such a method of relaxation was first introduced by Acikmese and Ploen [2007] and further seen in work by Blackmore et al. [2010] and Acikmese et al. [2013]. The dynamics are linear in an augmented control vector $\mathbf{u} = (\mathbf{T}^\intercal \eta)^\intercal$. The constraint in Equation (5.17) has become convex as well. The EOMs (Equation (5.20) - (5.22)) no longer contain $\|\mathbf{T}\|$. Though, they are still non-linear because of the term $\frac{1}{r^3}\mathbf{r}$ and also because of the of coupling between $\mathbf{T}$ and $m$ as $\mathbf{T}/m$. These will be handled using the Method Of Successive Approximations, presented in the next section.

While the original problem has now been relaxed, one wonders if the two are equivalent problems. In other words, whether the solution of the relaxed problem is in fact, the solution of the original problem. The proof of their equivalence is shown by Lu and Liu [2013]. The proof is detailed and beyond the scope of this study. Hence, it is deemed unnecessary to present it here.

## 5.3 METHOD OF SUCCESSIVE APPROXIMATION

To numerically solve Problem 2, we still have to make the EOMs (Equations (5.20) - (5.22)) linear. The following change of variables is suggested:

$$\boldsymbol{\tau} = \frac{\mathbf{T}}{m}, \quad z = \ln m \quad \sigma = \eta / m \tag{5.23}$$

This allows us to re-write the equations of motions as:

$$\dot{\mathbf{r}} = \mathbf{V} \tag{5.24}$$

$$\dot{\mathbf{V}} = -\frac{1}{r^3}\mathbf{r} + \boldsymbol{\tau} \tag{5.25}$$

$$\dot{z} = -\frac{1}{v_{\mathrm{ex}}}\sigma \tag{5.26}$$

Let $\mathbf{y} = (\mathbf{r}^\intercal \mathbf{V}^\intercal z)^\intercal$ be the new state vector and $\mathbf{u} = (\boldsymbol{\tau}^\intercal \sigma)^\intercal$ be the control vector. The previous set of equations can be cast as:

$$\begin{aligned} \dot{\mathbf{y}} &= \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} \\ -\frac{1}{r^3}\mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & 0 \end{bmatrix} \mathbf{y} + \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & -\frac{1}{v_{\mathrm{ex}}} \end{bmatrix} \mathbf{u} \\ &= \mathbf{A}(r)\mathbf{y} + \mathbf{B}\mathbf{u} \end{aligned} \tag{5.27}$$

Using the new variables, we can also re-write the performance index as follows:

$$J = \int_0^{t_f} \sigma(t)\mathrm{d}t$$

Minimizing $\sigma$ corresponds to maximising $m$, and thus lowest propellant consumption. Also, Equations (5.14) and (5.15) are modified to accommodate the new variables:

$$\|\boldsymbol{\tau}\| \le \sigma$$

$$0 \le \sigma \le T_{\max} e^{-z}$$

However, an unintended consequence is that the right hand side of the above equation is not a conic constraint equation. To approximate it by a conic equation, the term $e^{-z}$ has to be linearized. This can be done using a first-order Taylor expansion of about a reference $z_0(t)$, as suggested by Acikmese and Ploen [2007]. Thus, the new constraint is now

$$0 \leq \sigma \leq T_{\max}\, e^{-z_0(t)}[1-(z(t)-z_0(t))] \tag{5.28}$$

The new relaxed problem is now re-written as [Lu and Liu, 2013]:

**Problem 3:**

$$J = \int_0^{t_f} \sigma(t)\mathrm{d}t \tag{5.29}$$

$$\|\boldsymbol{\tau}\| \leq \sigma \tag{5.30}$$

$$0 \leq \sigma \leq T_{\max} e^{-z_0(t)}[1-(z(t)-z_0(t))] \tag{5.31}$$

$$\left\|\mathbf{r}(t)-\mathbf{r}_t(t)\right\| \cos\alpha \leq \mathbf{1}_n^{\mathsf{T}}(t)\cdot(\mathbf{r}(t)-\mathbf{r}_t(t)) \tag{5.32}$$

$$\mathbf{1}_n^{\mathsf{T}}(t)\boldsymbol{\tau} \leq \sigma\cos\beta \tag{5.33}$$

$$\mathbf{C}_f(t_f)\mathbf{x}(t_f)+\mathbf{d}_f(t_f) = 0 \tag{5.34}$$

$$\mathbf{C}_i(t_i)\mathbf{x}(t_i)+\mathbf{d}_i(t_i) = 0, \qquad i = 1,\ldots,l, \quad 0 < t_1 < \cdots < t_l < t_f \tag{5.35}$$

But, there still remains one non-linearity in the problem. That is of the term $1/r^3$ in $\mathbf{A}(r)$ of Equation (5.27). In many other applications, the percentage of variation for $r$ is very small and the approximation of $r \approx r_0$ may suffice, where $r_0$ is the magnitude of the initial radius vector. This is exactly the bases for the so-called 'linear gravity' model proposed by Jezewski [1971]. However, it can be an inaccurate approximation to make when high precision is required. Thus, a method of successive approximations was developed by Lu and Liu [2013] which is presented below. Here, parameters with superscript $(k)$, refer to value from the $k$-th iteration.

1. Set iteration step $k = 1$; $r^{(0)}(t) = r_0$ and $z^{(0)}(t) = \ln(1-\dot{m}_c t)$, where $\dot{m}_c > 0$ is (an arbitrary, constant, dimensionless) propellant consumption rate.

2. Solve the following optimal control problem to find the pair $\mathbf{u}^{(k)}$, $\mathbf{y}^{(k)}$.

**Problem 4:**
Given the state equations and initial condition:

$$\dot{\mathbf{y}}^{(k)} = \mathbf{A}(r^{(k-1)}(t))\,\mathbf{y}^{(k)} + \mathbf{B}\mathbf{u}^{(k)}, \qquad \mathbf{y}^{(k)}(0) = y(0) \tag{5.36}$$

minimize for the given $t_f$

$$J = \int_0^{t_f} \sigma^{(k)}(t)\mathrm{d}t \tag{5.37}$$

subject to

$$0 \leq \sigma^{(k)}(t) \leq T_{\max} e^{-z^{(k-1)}(t)}\{1-[z^{(k)}(t)-z^{(k-1)}(t)]\} \tag{5.38}$$

$$\left\|\boldsymbol{\tau}^{(k)}(t)\right\| \leq \sigma^{(k)}(t) \tag{5.39}$$

$$\left\|\mathbf{r}^{(k)}(t)-\mathbf{r}_t(t)\right\| \cos\alpha \leq \mathbf{1}_n^{\mathsf{T}}(t)\cdot(\mathbf{r}^{(k)}(t)-\mathbf{r}_t(t)) \tag{5.40}$$

$$\mathbf{1}_n^{\mathsf{T}}(t)\mathbf{T}^{(k)} \leq \sigma^{(k)}(t)\cos\beta \tag{5.41}$$

$$\mathbf{C}_f(t_f)\mathbf{x}^{(k)}(t_f)+\mathbf{d}_f(t_f) = 0 \tag{5.42}$$

$$\mathbf{C}_i(t_i)\mathbf{x}^{(k)}(t_i)+\mathbf{d}_i(t_i) = 0, \qquad i = 1,\ldots,l, \quad 0 < t_1 < \cdots < t_l < t_f \tag{5.43}$$

where $\mathbf{x}^{(k)} = ((\mathbf{r}^{(k)})^\top(\mathbf{V}^{(k)})^\top)^\top$, and $r^{(k-1)}(t) = \left\|\mathbf{r}^{(k-1)}(t)\right\|$ and $z^{(k-1)}(t)$ are already found from previous solution.

3. The solution is then checked to see if the convergence condition has been met for a prescribed tolerrance $\epsilon > 0$

$$\sup_{0\le t\le t_f} \left\|\mathbf{y}^{(k)}(t) - \mathbf{y}^{(k-1)}(t)\right\| \le \epsilon, \qquad k > 1 \tag{5.44}$$

where sup is the *supremum* function, which provides the upper bound of the set. If the previous condition is satisfied, go to step 4, otherwise set $k = k+1$, and go to step 2.

4. The solutuon to the problem is found out to be as $\mathbf{y} = \mathbf{y}^{(k)}$ and $\mathbf{u} = \mathbf{u}^{(k)}$.

Lu and Liu [2013] go on to make the following remarks about the method:

a) For each $k$, the Problem 4, is one with linear time-varying dynamics, linear equality constraints, and second-order-cone inequality constraints. Yet, it is still a non-linear optimal control problem (because of the conic constraints) but is ready to be discretized into an SOCP problem that can be solved numerically.

b) The converged solution found is the one that satisfies exactly (to the accuracy of numerical solution) the original non-linear gravity model.

c) The user need not define the burn-coast structure before hand. The final solution found will automatically determine the optimal number of burn and coast arc, as well as their durations.

Liu [2013] does make an attempt to prove that the MOSA solution converges and that the converged solution in fact solves the original rendezvous problem. To a great extent he is successful (and the interested reader is recommended to read the dissertation, because the proof is too detailed and elaborate to present here), yet the proof cannot be said to be valid for *all* cases. In their defence, the lack of a complete proof is abated by their positive numerical experience. Lu and Liu [2013] state that, 'Our numerical experiences have always shown good convergence of the successful solutions'.

## 5.4 DISCRETIZATION

The set of problems presented until now were considered in continuous time. However, to solve them numerically, one has to discretize the problem. To find the numerical solution to the Problem 4 for each state $k$, the state and control vectors are discretized by $\mathbf{y}_j = \mathbf{y}(t_j)$ and $\mathbf{u}_j = \mathbf{u}(t_j)$, where $j = 1, 2, 3, \ldots, n$, $t_j = jh$ and $h = t_f/n$, for a positive integer $n$. The state equation (Equation (5.36)) is integrated by the trapezoidal rule:

$$\mathbf{y}_j = \mathbf{y}_{j-1} + \frac{1}{2}h\left[f(\mathbf{y}_j, \mathbf{u}_j, t_j) + f(\mathbf{y}_{j-1}, \mathbf{u}_{j-1}, t_{j-1})\right], \qquad j = 1, 2, \ldots, n \tag{5.45}$$

where $f$ is the right-hand side of the Equation (5.36). Let $\mathbf{A}_j = \mathbf{A}[r^{k-1}(t_j)]$, then the above equation can be re-written as:

$$\left(I - \frac{1}{2}h\mathbf{A}_j\right)\mathbf{y}_j = \left(I + \frac{1}{2}h\mathbf{A}_{j-1}\right)\mathbf{y}_{j-1} + \frac{1}{2}h\mathbf{B}\mathbf{u}_j + \frac{1}{2}h\mathbf{B}\mathbf{u}_{j-1} \tag{5.46}$$

where $I$ is a $7 \times 7$ unit matrix. The local errors are proportional to $h^3$. At this point, two options may be considered in choosing the variables for the subsequent finite-dimensional optimization problem.

**Scheme 1:** Choose the optimization vector to be

$$\mathbf{w} = (\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_n, \mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_n) \tag{5.47}$$

In this case, the Equations (5.38) - (5.43), at each $t_j$ form linear equality and second-order conic inequality constraints on **w**. Also, the collection of equations from Equation (5.46) at each $t_j$ forms a linear equality constraint on **w**.

**Scheme 2:** When $\mathbf{y}_0$ is given, all $\mathbf{y}_j$, $j = 1, 2, \ldots, n$ can be recursively solved in terms of $\mathbf{u}_i$, $0 \le l \le j$, from (5.46):

$$\mathbf{y}_j = \left(I - \frac{1}{2}h\mathbf{A}_j\right)^{-1}\left[\left(I + \frac{1}{2}h\mathbf{A}_{j-1}\right)\mathbf{y}_{j-1} + \frac{1}{2}h\mathbf{B}(\mathbf{u}_j + \mathbf{u_{j-1}})\right] \tag{5.48}$$

In this way, the $\mathbf{y}_j$ are replaced as linear functions of $\mathbf{u}_j$. The optimization variables are now just $\mathbf{w} = (\mathbf{u}_0, \ldots, \mathbf{u}_n)$. The discretized state equations (Equation (5.46) at each $t_j$) are no longer part of the optimization problem. The constraints in Equations (5.38) - (5.43), will now be reformulated for $(\mathbf{u}_0, \ldots, \mathbf{u}_n)$.

The discretization scheme 2 results in a significantly smaller dimension for the optimization problem when compared to scheme 1. But the problem under scheme 1 is sparse, while it is dense in scheme 2. The choice of discretization scheme depend on the capabilities of the optimization solver used. Most current solvers favor scheme 1, as they are optimized for use with sparse matrices.

After the discretization, Problem 4 is now reduced to an SOCP of the form [Lu and Liu, 2013]

**Problem 5:**

$$\min_{\mathbf{w}} \mathbf{c}^\mathsf{T}\mathbf{w} \tag{5.49}$$

$$\left\|\mathbf{M}_i\mathbf{w} + \mathbf{b}_i\right\| \le \mathbf{e}_i^\mathsf{T}\mathbf{w} + f_i, \qquad i = 1, \ldots, N \tag{5.50}$$

Note the resemblance with Equation 4.11. This is a $n_w$-dimensional SOCP, where $n_w$ is the dimension of the optimization variable **w**. In the above problem, we have $\mathbf{M}_i \in \mathbb{R}^{n_i \times n_w}$, $\mathbf{c} \in \mathbb{R}^{n_w}$, $\mathbf{b}_i \in \mathbb{R}^i$, $\mathbf{e}_i \in \mathbb{R}^{n_w}$ and $f_i$ is a real number. It may appear at first that the above problem accepts only inequality constraints, however, equality constraints can be re-written as a set of two inequalities. For example, $\mathbf{Ax} = \mathbf{b}$ is the same as the set of equations $\mathbf{Ax} \le \mathbf{b}$ and $\mathbf{Ax} \ge \mathbf{b}$.

The above problem can be solved using a primal-dual interior-point algorithm. Liu [2013] made use of YALMIP, an environment for rapid-algorithm development, and MOSEK, a commercial solver for conic problems. The programming environment was MATLAB. The output of the problem is the optimization variable **w**, which gives the thrust vector history and the propagated state of the chaser.

## 5.5 ADDITIONAL FORMULATIONS

As described by Liu and Lu [2013], the ConGAL method also provides the option to add further constraints like rate of change of thrust vector and collision avoidance. The gravity model can also be expanded to include the $J_2$ and atmospheric drag terms. The following sub-sections briefly describe the modifications to accommodate these formulations.

### 5.5.1 RATE OF CHANGE OF THRUST VECTOR

Spacecraft have a limited capability to quickly change their thrust magnitude or direction. As a result, it is important to constrain the rate of change of **T**. It is achieved by adding two dynamic equations and the corresponding constraint equations as follows

$$\begin{aligned} \dot{\boldsymbol{\tau}} &= \boldsymbol{\nu}, & \dot{\sigma} &= \lambda \\ \|\boldsymbol{\nu}\| &\le \xi, & \left|\lambda\right| &\le \zeta \end{aligned} \tag{5.51}$$

where $\xi$ is the maximum rate of change of thrust direction and $\zeta$ is the maximum rate of change of thrust magnitude. The state vector is now redefined as $\mathbf{y} = \begin{bmatrix}\mathbf{r}^\top & \mathbf{V}^\top & z & \boldsymbol{\tau}^\top & \sigma)\end{bmatrix}^\top$ and the control vector as $\mathbf{u} = \begin{bmatrix}\boldsymbol{\nu}^\top & \lambda\end{bmatrix}^\top$. As a result, the matrices $\mathbf{A}$ and $\mathbf{B}$ are

$$\mathbf{A}(r) = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} \\ -\frac{1}{r^3}\mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & 0 & \mathbf{0}_{1\times3} & -\frac{1}{v_{\text{ex}}} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & 0 & \mathbf{0}_{1\times3} & 0 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & 0 \\ \mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix} \tag{5.52}$$

The overall problem can be written as follows [Lu and Liu, 2013]

**Problem 6:**

$$J = \int_0^{t_f} \sigma(t)\mathrm{d}t \tag{5.53}$$

$$\dot{\mathbf{y}} = \mathbf{A}(r)\mathbf{y} + \mathbf{B}\mathbf{u} \tag{5.54}$$

$$\|\boldsymbol{\tau}\| \leq \sigma \tag{5.55}$$

$$\|\boldsymbol{\nu}\| \leq \xi \tag{5.56}$$

$$|\lambda| \leq \zeta \tag{5.57}$$

$$0 \leq \sigma \leq T_{\max} e^{-z_0(t)}[1 - (z(t) - z_0(t))] \tag{5.58}$$

$$\|\mathbf{r}(t) - \mathbf{r}_t(t)\| \cos\alpha \leq \mathbf{1}_n^\top(t) \cdot (\mathbf{r}(t) - \mathbf{r}_t(t)) \tag{5.59}$$

$$\mathbf{1}_n^\top(t)\boldsymbol{\tau} \leq \sigma\cos\beta \tag{5.60}$$

$$\mathbf{C}_f(t_f)\mathbf{x}(t_f) + \mathbf{d}_f(t_f) = 0 \tag{5.61}$$

$$\mathbf{C}_i(t_i)\mathbf{x}(t_i) + \mathbf{d}_i(t_i) = 0, \qquad i = 1, \ldots, l, \quad 0 < t_1 < \cdots < t_l < t_f \tag{5.62}$$

### 5.5.2 PERTURBATION FROM $J_2$ AND ATMOSPHERIC DRAG

The equation of motion defined in Equation (5.3) considers the problem to be two-body problem in Newtonian gravity. To have more precise solution for the RPO, a more accurate model is necessary. In general, it can be re-written as

$$\ddot{\mathbf{r}} = \dot{\mathbf{V}} = -\frac{\mu}{r^3}\mathbf{r} + \frac{\mathbf{T}}{m} + \mathbf{a}_{\text{p}} \tag{5.63}$$

where $\mathbf{a}_{\text{p}}$ is the sum of all perturbing accelerations causes by other forces. The acceleration due to $J_2$ and atmospheric drag is shown below.

#### PERTURBATION DUE TO $J_2$

Earth, due to its non-spherical shape exhibits a gravity field that differs from Newtonian gravity. This can be modelled using gravity harmonics, of which the biggest component is due to the oblateness of Earth. Amongst the zonal harmonics, $J_2$ is the largest and represents most of Earth's departure from a perfect sphere. $J_2$ is about 400 times larger than $J_3$, and hence this is a valid assumption for time being. The resulting equation of motion in the Earth Centred Inertial frame ($\mathcal{I}$) is

$$\dot{\mathbf{V}} = (-\frac{1}{r^3} + a_1)\mathbf{r} + a_2\mathbf{V} + \frac{\mathbf{T}}{m} + \mathbf{a}_0 \tag{5.64}$$

where

$$\mathbf{a}_0 = \frac{3}{2}J_2\frac{1}{r^4}\sin(2i)\sin\theta\frac{\mathbf{r}\times\mathbf{V}}{|\mathbf{r}\times\mathbf{V}|} \tag{5.65}$$

$$a_1 = \frac{3}{2}J_2\frac{\mathbf{V}^\mathsf{T}\mathbf{r}}{r^6\mathbf{V}_h}\sin^2 i\sin(2\theta) - \frac{3}{2}J_2\frac{1}{r^5}(1-3\sin^2 i\sin^2\theta) \tag{5.66}$$

$$a_2 = \frac{3}{2}J_2\frac{1}{r^4\mathbf{V}_h}\sin^2 i\sin(2\theta) \tag{5.67}$$

and $\mathbf{V}_h$ is the horizontal component of the velocity vector

$$\mathbf{V}_h = \mathbf{V} - (\mathbf{V}^\mathsf{T}\mathbf{r}/r^2)\mathbf{r} \tag{5.68}$$

Thus, the state dynamic equation can be written as

$$\dot{\mathbf{x}} = \tilde{\mathbf{A}}(\mathbf{x})\mathbf{x} + \tilde{\mathbf{B}}\mathbf{u} + \mathbf{b}(\mathbf{x}) \tag{5.69}$$

where

$$\tilde{\mathbf{A}}(\mathbf{x}) = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} \\ \left(-\frac{1}{r^3}+a_1\right)\mathbf{I}_{3\times3} & a_2\mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & 0 \end{bmatrix}, \quad \tilde{\mathbf{B}} = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & -\frac{1}{v_\text{ex}} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \mathbf{0}_{3\times1} \\ \mathbf{a}_0 \\ 0 \end{bmatrix} \tag{5.70}$$

Unlike $\mathbf{A}(r)$ used in the previous formulations, which only depends on the norm of the $\mathbf{r}$ vector, the matrix $\tilde{\mathbf{A}}(\mathbf{x})$ is dependent on both $\mathbf{r}$ and $\mathbf{V}$.

#### PERTURBATION DUE TO ATMOSPHERIC DRAG

Spacecraft in LEO are subject to atmospheric drag, which can cause substantial change in orbital energy, especially over long periods. The acceleration is in the direction opposite of the velocity of the spacecraft. It is given (in $g$) as

$$\mathbf{a}_\text{drag} = -\frac{1}{2}\frac{C_D S}{m}\frac{R_0}{m_0}\rho V\mathbf{V} \tag{5.71}$$

where $C_D$ is the drag coefficient, $S$ is the dimensional cross-sectional area of the spacecraft normal to its velocity $\mathbf{V}$ and $\rho$ is the dimensional atmospheric density. Adding this acceleration to the equation of motion in Equation (5.64), we have the same form of state equation (5.69), but with some changes:

$$\tilde{\mathbf{A}}(\mathbf{x}) = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} \\ \left(-\frac{1}{r^3}+a_1\right)\mathbf{I}_{3\times3} & (a_2+a_3)\mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & 0 \end{bmatrix}, \tag{5.72}$$

where

$$a_3 = -\frac{1}{2}\frac{C_D S}{m}\frac{R_0}{m_0}\rho V \tag{5.73}$$

The matrix $\tilde{\mathbf{A}}$ includes perturbations due to both $J_2$ and atmospheric drag. If $J_2$ is to be neglected, then set $\mathbf{a}_0$, $a_1$ and $a_2$ to zero.

### 5.5.3 COLLISION AVOIDANCE

When the chaser approaches the target, it should ensure that it does not accidentally collide with the target. Thus, a keep-out-zone is defined which is imposed as a set of two constraints in the optimization problem. For simplicity, the keep-out-zone is modelled as a sphere of radius $d_a$ centred at the target in LVLH frame. The first constraint to be enforced is

$$\left\|\mathbf{r}(t) - \mathbf{r}_t(t)\right\| \geq d_a \tag{5.74}$$

The above equation is concave, and hence cannot be handled by the SOCP in its current form. Again, the MOSA is used to solve it over a series of iterations. In the $k$-th iteration described in the method of successive approximation (Section 5.3), the above equation can be linearised about the $k-1$-th solution $\mathbf{r}^{[k]}(t)$

$$\left\| \mathbf{r}^{(k-1)}(t) - \mathbf{r}_t(t) \right\| + \frac{[\mathbf{r}^{(k-1)}(t) - \mathbf{r}_t(t)]^\intercal}{\left\| \mathbf{r}^{(k-1)}(t) - \mathbf{r}_t(t) \right\|} [\mathbf{r}(t) - \mathbf{r}^{(k-1)}(t)] \geq d_a \tag{5.75}$$

The above equation is affine in $\mathbf{r}(t)$, and thus convex. In addition a trust-region constraint on $\mathbf{r}$ is imposed

$$\mathbf{r}(t) - \mathbf{r}^{(k-1)}(t) \leq d_b \tag{5.76}$$

where $d_b > 0$ is chosen, and remains constant in the current iteration. The trust region ensures between successive iterations, $\mathbf{r}(t)$ does not change by more than $d_b$. This helps Equation (5.75) linearise the solution more effectively. The above constraint for each fixed $t$ is a quadratic cone, and can be readily handled in SOCP.

# 6

# Software Development and Testing

One of the deliverables of this thesis project is a software that computes guidance commands and simulates the motion of a spacecraft in an highly constrained rendezvous mission. It establishes a mathematical model of the spacecraft and the environment, and helps in studying the interaction of the guidance system with the environment. The ConGAL method serves as the basis for the guidance system. At the core of this simulator, is a *main engine* that formulates the guidance problem based on inputs from the user, and solves it. This chapter describes the software tool's architecture, its development and verification/validation tests. In the process, numerous discoveries were made with respect to ConGAL's characteristic behaviour and its limitations. Thus, the development of the final version of the software was carried out in tandem with the numerical analysis of ConGAL, which is discussed up next in Chapter 7. Should the need arise, the reader is advised to have a look at the summary of the numerical analysis presented in Section 7.6 for a better understanding of this chapter.

## 6.1 Software architecture

Figure 6.1 provides a quick overview of the relations between the GNC system of the chaser spacecraft, the space flight environment and the target spacecraft. This is a good starting point for designing the software architecture – it answers *what* is it that the software intends to do. We are interested in generating the guidance commands for the rendezvous flight, and thus only those blocks directly relevant to this are considered for modelling in the software. As a result, the navigation system and the control system are not focussed upon (and hence, shown in lighter shade in Figure 6.1). The main block of the software is envisioned to be a ConGAL method based guidance algorithm that computes the guidance commands required to solve the rendezvous mission. A top-level architecture of the software is illustrated in Figure 6.2 - it answers *how* the software will go about to meet its requirements. The blocks are discussed in detail below.

### 6.1.1 Input

The guidance algorithm requires a large number of input parameters, which can be categorized into four major types:

**Input 1:** Chaser parameters
These inputs relate to properties of the chaser spacecraft. For example, its initial mass and state, available propellant, maximum thrust produced by the engine, etc.

**Input 2:** Target parameters
The target itself presents a varieties of inputs to the solver. Most importantly, the target

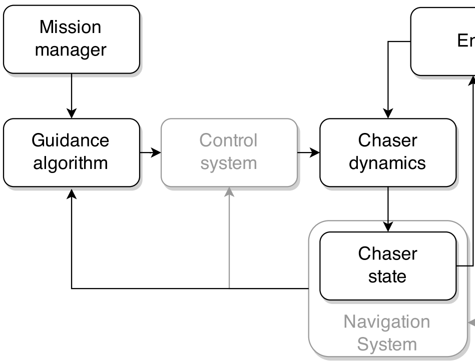


Figure 6.1: Process overview of the software tool. The thesis focusses on the guidance aspects only and hence the navigation and control systems are not fully considered.

state vector over the time of mission is required by the solver. Further, the orientation of docking axis in the inertial frame, constraints on plume-angle impingement, etc.

**Input 3:** Mission case parameters
For each mission case considered, some parameters will vary. Namely, time of flight, hold points, terminal conditions, environmental model for on-board propagation, etc.

**Input 4:** Solver parameters
The solver parameters have to be decided by the user, based on the requirements for that mission cases. This includes, required accuracy of the solver, maximum allowable run-time, integrator used, etc.

This block provides the user an opportunity to customize spacecraft properties and mission constraints.

### 6.1.2 MAIN ENGINE

As the name suggests, this block forms the core of the software as it solves the guidance problem. Figure 6.2 also shows the functional flow diagram inside the `Main Engine` block. The block is programmed in an algorithm development environment and the optimization problem is solved in a *solver*. It is a specialized software that is used to solve conic programs. Some examples include MOSEK, ECOS, SDPT3, (details in MOSEK ApS [2015], Domahidi et al. [2013], Tütüncü et al. [2003]). They employ the primal-dual interior point method group of algorithms to solve the problem. Often, an interfacing software is required to use the solver, and some examples include CVX

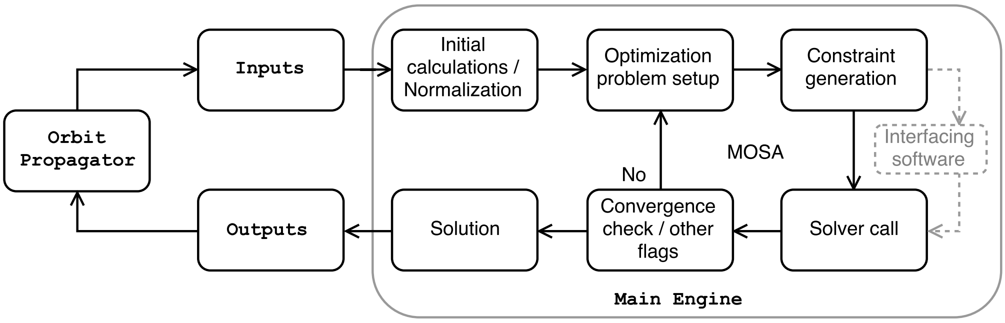


Figure 6.2: Top-level software architecture.

and YALMIP (see Grant et al. [2008], Löfberg [2004]). The inner loop is to reach convergence using the Method of Successive Approximations (MOSA), as discussed in Section 5.3. In each iteration, the problem solves a linear, time-varying dynamics, SOCP problem. However, the converged solution found is expected to satisfy the original problem with non-linear gravity model. The outer loop is to check accuracy of the trajectory and also help correct deviation (which is discussed in Section 8.4.2).

### 6.1.3 OUTPUT AND POST-PROCESSING

The simulator gives the following outputs:

**Output 1:** Guidance commands (Thrust-vector profile)

**Output 2:** Optimized objective function (for example, the amount of propellant used)

**Output 3:** State vector of the chaser

**Output 4:** Performance parameters of the simulator, like accuracy, run-time, etc.

The outputs by itself mean little if unprocessed. For instance, conversion from normalized form to original form, inertial to LVLH frame, preparation for input to orbit propagator. Further processing involves plotting graphs, comparing results, etc.

### 6.1.4 ORBIT PROPAGATOR

There are two main requirements to have a orbit propagator for solving the problem. First, the ConGAL method requires accurate state history of the target throughout the flight. In real flights, this data is available from various sources (tracking stations, on-board sensors, GPS, etc). However, for this thesis, it is simulated by employing an orbit propagator: based on a given initial state of the target, the orbit propagator shall provide the state vector profile of the spacecraft up to end of flight. This is then used in defining constraints for hold points, docking axis orientation, final point of contact, and so on. Secondly, the thrust profile generated by `Main Engine` is sent to it for independent propagation of the chaser state. This is then used to check accuracy of the guidance solution generated by the main engine.

Figure 6.3 shows the architecture of the orbit propagator. Based on the initial state of the spacecraft and the properties of the spacecraft (as described by the spacecraft model), the orbit propagator integrates the EOM to provide the updated state of the spacecraft. The force models along with thrust profile (if any, from the previous computation of guidance problem) provide the accelerations to the EOM block.

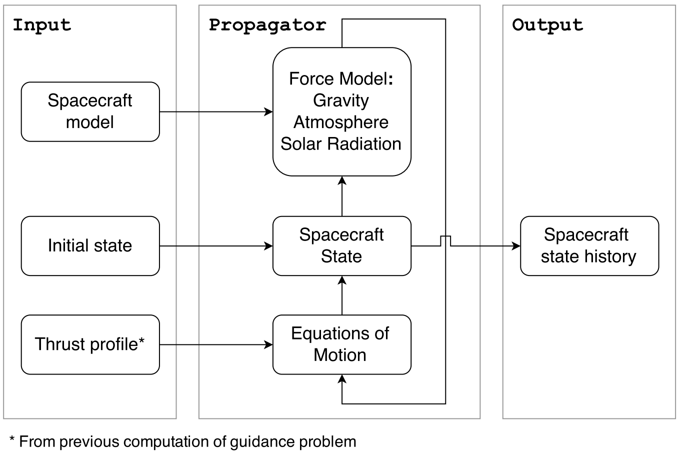


Figure 6.3: Functional flow diagram of the the Orbit Propagator.

## 6.2 AVAILABLE PACKAGES

An extensive analysis and trade-off of the available packages was done during literature study for this thesis. The details are available in Bhagat [2015]. Key take-aways from it are:

- Modelling environment: MATLAB is used as the modelling environment, owing to its versatility. It can easily interface with numerous solvers, which gives it an edge over C++.
- Orbit propagator: GGNCSIM, a set of MATLAB / Simulink libraries, is used to develop the OP, as it allows great flexibility in modelling the simulation environment.
- Conic problem solvers: ECOS is considered as the primary choice, as it is quick and has a low footprint. Either of SeDuMi or MOSEK is considered as backup candidates.
- Interfacing software to solver: CVX is considered as primary choice and YALMIP as secondary.

However, mid-way through this thesis it was decided to switch to MOSEK as the primary solver, and leave out using the interfacing software because of issues with accuracy, convergence and computation time. This is discussed in detail in Chapter 7.

## 6.3 ORBIT PROPAGATOR

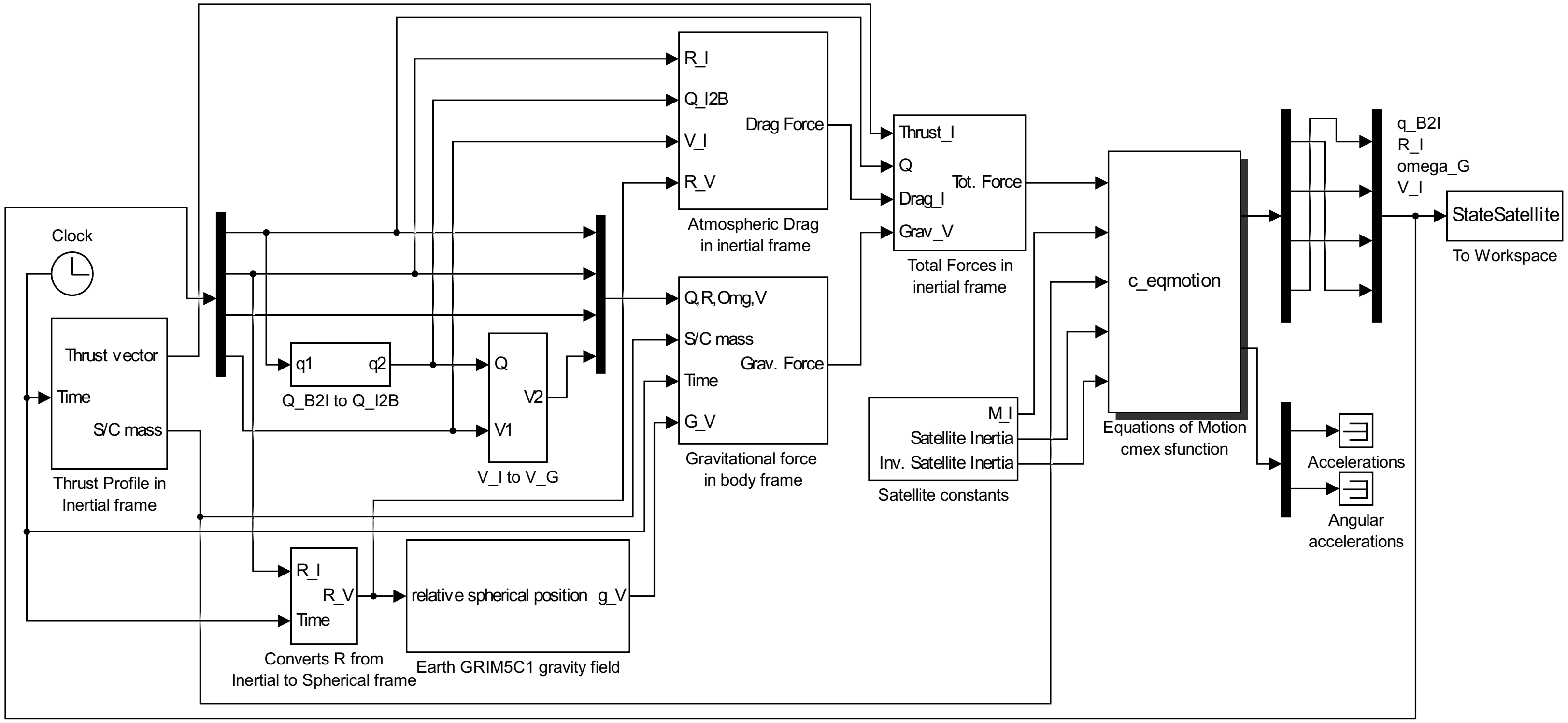
This section discusses in detail how the `Orbit Propagator` (OP) is built and verified. It was decided to make use of Generic Guidance Navigation and Control SIMulator (GGNCSIM), which is a set of MATLAB/Simulink libraries that provides great flexibility in building a simulation environment for spacecraft. It also allows easy interfacing with other MATLAB routines.

The detailed architecture of the Orbit Propagator (OP) is seen in Figure 6.4. The OP is initialized with the the initial state of the spacecraft, the order and degree of the gravity field, thrust profile (if any), time of flight and step size of final results. Other spacecraft properties like drag coefficient, specific impulse of thrusters, cross-section area and initial mass are provided as well. Based on the thrust profile, the spacecraft mass can vary with time and hence, it is calculated at each time step. The integrator uses a variable step, fourth and fifth order Runge Kutta integrator (implemented by `ode45` solver of MATLAB). The flight dynamics are provided by the cmex sfunction of GGNCSIM, which besides translation motion also accommodates rotational motion. However, rotational motion is not considered in this OP, and all parameters pertaining to it are initialized suitably to avoid any errors. For the gravity field, the Earth GRIM5C1 model is used and for the atmosphere, Earth MSIS 1986 nominal density model is used. It is expected that these two library blocks function properly and are skipped from verification tests. The entire OP module, however, cannot be expected to be free of errors right away, and thus need to be verified. The verification tests and their results are covered in the next subsection.

### 6.3.1 VERIFICATION TESTS

Four tests, covering the range of activities expected from OP, are carried out. For these tests, it is assumed that Earth is the central body. The constants or parameters that are used during the tests are shown in Table 6.1. The tests are as follows:

1. Circular and elliptical orbit test: In the presence of a central gravity field, the propagated orbit should match with analytical values within a given tolerance.
2. Nodal regression due to $J_2$ effect: The orbital plane rotates in the presence of the $J_2$ effect of Earth. The propagated orbit's change in RAAN ($\dot{\Omega}$) should match the expected value of analytical solutions.



Indexes in this diagram, as per GGNCSIM convention
G refer to body frame
V refers to spherical frame
I refers to inertial frame

V_I = [vx vy vz] = velocity
R_I = [x y z] = cartesian position
omega_G = [p q r] = angular rotation rate
q_B2I = [q1..q4] = quaternion of I frame

Figure 6.4: Detailed architecture of the `Orbit Propagator`

3. Hohmann-transfer between two circular orbits: The OP accepts a thrust profile corresponding to a Hohmann-transfer between two circular orbits in a central gravity field. The resulting trajectory is checked with expected values.

4. Effect of Atmospheric drag on specific orbital energy: In the presence of atmospheric drag, the orbit decays and specific orbital energy reduces. This change can be measured and compared to analytical values.

Table 6.1: Parameters used for `Orbit Propagator` verification tests

| Parameter | Value | Units |
|---|---|---|
| Earth gravitational parameter | $3.986004415 \times 10^{14}$ | $m^3s^{-2}$ |
| Earth equatorial radius | 6378136 | m |
| Spacecraft initial mass | 500 | kg |
| Spacecraft cross-section area | 5 | $m^2$ |
| Drag coefficient | 2.2 | [-] |
| Specific Impulse | 320 | s |

### OP TEST 1: CIRCULAR AND ELLIPTICAL ORBITS

Consider a uniform central gravity field for Earth. An object moving in a circular or elliptical orbit should continue to do so without change in its orbital elements (except true anomaly, which should vary with time). Table 6.2 shows the input to the OP and its output after 100000 s of flight, sampled at 1 s intervals. We see that there is no change in the relevant orbital elements. This shows that the OP works well for circular and elliptical orbits in a central gravity field.

Table 6.2: OP Test 1 input and output for circular and elliptical orbit after 100000 s of flight.

| Orbital Element | Unit | Circular orbit | | Elliptical orbit | |
|---|---|---|---|---|---|
| | | Input | Output | Input | Output |
| Semi-major axis*, $a$ | m | 300000.00 | 300000.00 | 2000000.00 | 2000000.00 |
| Eccentricity, $e$ | [-] | 0.0000 | 0.0000 | 0.2000 | 0.2000 |
| Inclination, $i$ | ° | 0.00 | 0.00 | 0.00 | 0.00 |
| RAAN, $\Omega$ | ° | 0.00 | 0.00 | 0.00 | 0.00 |
| Argument of perigee, $\omega$ | ° | 0.00 | 0.00 | 0.00 | 0.00 |

* Values show $a - R_E$ for ease of reading

### OP TEST 2: NODAL REGRESSION DUE TO $J_2$ EFFECT

Now, Earth's $J_2$ effect is considered as well in the gravity field[9]. Presence of $J_2$ causes the orbital plane to precess, *i.e.*, RAAN changes during flight. This variation in RAAN ($\Delta\Omega$) shows short-period variation (which cancels after one orbital revolution of the satellite) and a secular variation (which keeps adding). An approximate[10] analytical expression for $\Delta\Omega$ after one orbital revolution of the satellite is [Wakker, 2010b]:

$$\Delta_{2\pi}\Omega = -3\pi J_2 \frac{R_E^2}{r_0^2} \cos i_0 \tag{6.1}$$

[9] This is carried out by changing the order and degree of Earth GRIM5C1 gravity field function to two.

[10] It is an approximation because the effect of short-period variation is neglected. By using values of $r_0$ and $i_0$, we are only considering their average values over 1 period [Vallado, 2001].

where $r_0$ and $i_0$ are initial radius and inclination angle. We see that $\Delta_{2\pi}\Omega < 0$ if $0° < i_0 < 90°$ and $\Delta_{2\pi}\Omega > 0$ if $90° < i_0 < 180°$. For the OP to pass this test, it should give the same change in RAAN as above, within acceptable tolerance values. Table 6.3 shows the inputs and outputs of the test. We see that the difference between the analytical value and actual value from OP is less than 5" over one orbital revolution. As expected, higher the satellite, lower the nodal regression value. Also, as the angle of inclination increases, the nodal regression changes from negative to positive value.

Table 6.3: OP Test 2 input and output for 5 cases checking nodal regression over 1 orbital revolution of the satellite.

(a) Input to OP

| Orbit values | Unit | Case 1 | Case 2 | Case 3 | Case 4 | Case 5 |
|---|---|---|---|---|---|---|
| Semi-major axis*, $a$ | m | 300000.00 | 700000.00 | 700000.00 | 700000.00 | 700000.00 |
| Eccentricity, $e$ | [-] | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| Inclination, $i$ | ° | 30.00 | 30.00 | 60.00 | 90.00 | 120.00 |
| RAAN, $\Omega$ | ° | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| Argument of perigee, $\omega$ | ° | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |

* Values show $a - R_E$ for ease of reading

(b) Output from OP

| Cases | $\Delta_{2\pi}\Omega$ | | |
|---|---|---|---|
| | Expected (°) | Actual (°) | Error (") |
| Case 1 | -0.461829 | -0.460596 | 4.439 |
| Case 2 | -0.411106 | -0.410005 | 3.964 |
| Case 3 | -0.237352 | -0.237195 | 0.565 |
| Case 4 | 0.000000 | 0.000093 | 0.335 |
| Case 5 | 0.237352 | 0.237378 | 0.094 |

### OP TEST 3: HOHMANN-TRANSFER BETWEEN TWO CIRCULAR ORBITS

The `Main Engine` generates thrust commands which will be fed to the OP. It is necessary to check whether the satellite is able to change its trajectory accurately as per the given thrust commands. In this regard, a two-burn Hohmann transfer is carried out between two circular orbits to verify OP's capabilities. Consider a uniform central gravity field for Earth; the radius of the initial orbit to be $r_1$ and the final orbit to be $r_2$ in the same plane. The first burn puts the satellite in an elliptical transfer orbit and the second burn circularizes it to the final orbit. The $\Delta V$ for the burns are as follows:

$$\Delta V_1 = \sqrt{\frac{\mu}{r_1}}\left(\sqrt{\frac{2r_2}{r_1+r_2}} - 1\right) \tag{6.2}$$

$$\Delta V_2 = \sqrt{\frac{\mu}{r_2}}\left(1 - \sqrt{\frac{2r_1}{r_1+r_2}}\right) \tag{6.3}$$

Consider the first burn to be carried out instantaneously at some time $t_1$. The second burn is then carried out at when the satellite is diametrically opposite in orbit. This is given by:

$$t_2 = t_1 + t_H, \text{ where } t_H = \pi\sqrt{\frac{(r_1+r_2)^3}{8\mu}} \tag{6.4}$$

Owing to the finite value of the step-size of integration in the OP and the possible rounding off of $t_H$ to nearest multiple of the step-size, it is expected that there will be a slight variation between the results of the OP and analytical values. Table 6.3 shows the inputs and outputs of the test and Figure 6.5 shows the variation in orbital radius over time. It is observed that the output matches very well with the expected result and thus, the OP passes test 3 as well.

Table 6.4: OP Test 3 input and output for Hohmann transfer between two circular orbits in the same plane.

(a) Input to OP

| Parameter | Unit | Value |
|---|---|---|
| Initial orbit, $r_1$ | m | 700000.00 |
| Final orbit, $r_2$ | m | 1000000.00 |
| Hohmann transfer time, $t_H$ | s | 3057.88 |

(b) Output from OP

| Parameters | Unit | At t = $\frac{t_H}{2}$ | | At t > $t_H$ | |
|---|---|---|---|---|---|
| | | Expected | Actual | Expected | Actual |
| Semi-major axis*, $a$ | m | 850000.00 | 849999.97 | 1000000.00 | 999999.94 |
| Eccentricity, $e$ | [-] | 0.02075 | 0.02075 | 0.0000 | 0.0000 |
| Spacecraft mass, $m$ | kg | 487.808 | 487.808 | 476.035 | 476.035 |

* Values show $a - R_E$ for ease of reading

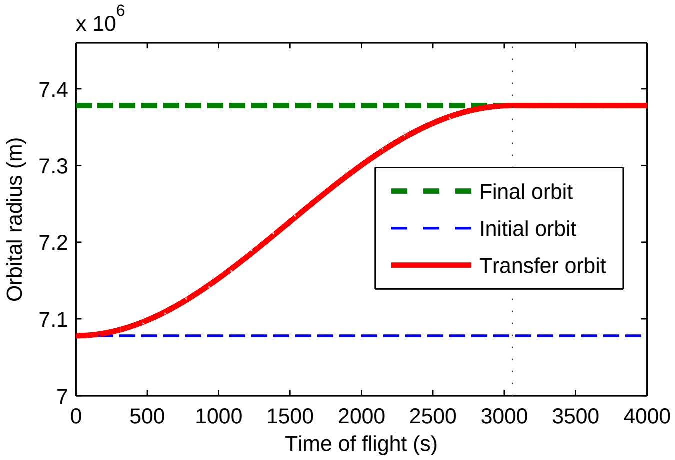


Figure 6.5: OP Test 3: Variation of orbital radius of satellite with respect to time. The vertical dotted-line marks the time for Hohmann-transfer, beyond which the orbital radius (red line) matches exactly with the required final orbit (green dashed line).

### OP TEST 4: VARIATION IN SPECIFIC ORBITAL ENERGY DUE TO ATMOSPHERIC DRAG

The specific orbital energy of an object in orbit is the sum of its gravitational potential energy and kinetic energy per unit mass. It is given by [Wakker, 2010b]:

$$\begin{aligned}\epsilon &= \epsilon_k + \epsilon_p \\ &= \frac{V^2}{2} - \frac{\mu}{r} \\ &= -\frac{\mu}{2a}\end{aligned} \tag{6.5}$$

where $\epsilon$ is the specific orbital energy, indices $_k$ and $_p$ refer to kinetic and potential energies respectively, $V$ is the orbital velocity and $r$ is the orbital radius at each instant. In a uniform central gravity, the specific orbital energy remains constant unless some other force (like perturbations, thrust from engine, etc.) acts on the object. The atmospheric drag, which is a non-conservative force, opposes the natural motion of the object, resulting in reduced velocity and semi-major axis. The work done is observed as a reduction in the specific orbital energy. This can be written as:

$$\frac{d\epsilon}{dt} = \frac{\mathbf{F}_{aero}}{m} \cdot \mathbf{V} \tag{6.6}$$

where, $\mathbf{F}_{aero}$ is the drag force and $m$ is the instantaneous mass. Upon integrating, we get the change in specific orbital energy $\Delta\epsilon$. The verification test involves checking the $\Delta\epsilon$ as estimated by the above equation against the value provided by integrating Equation (6.5). The input and output of the test are shown in Table 6.5. Four cases, with varying inclination and altitude are considered over a flight time of 20000 s. For orbits at same altitude, it is observed that the inclination angle affects the drag experienced by the object. Further, as expected, the orbit at a higher altitude shows a lower decrease in altitude and specific orbital energy, as the density of atmosphere is lower at higher altitudes. The difference between the expected value of $\Delta\epsilon$ and the actual value calculated by the OP, is zero in all cases. Thus, the OP passes this test as well.

Table 6.5: OP Test 4 input and output for variation in specific orbital energy due to atmospheric drag.

(a) Input to OP

| Orbit values | Unit | Case 1 | Case 2 | Case 3 | Case 4 |
|---|---|---|---|---|---|
| Semi-major axis*, $a$ | m | 300000.00 | 300000.00 | 300000.00 | 700000.00 |
| Eccentricity, $e$ | [-] | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| Inclination, $i$ | ° | 0.00 | 60.00 | 180.00 | 0.00 |
| RAAN, $\Omega$ | ° | 0.00 | 0.00 | 0.00 | 0.00 |
| Argument of perigee, $\omega$ | ° | 0.00 | 0.00 | 0.00 | 0.00 |

* Values show $a - R_E$ for ease of reading

(b) Output from OP

| Cases | $\Delta a$ (m) | $\Delta\epsilon$ (J/kg) | | |
|---|---|---|---|---|
| | Actual | Expected | Actual | Error |
| Case 1 | -391.947 | -1751.660 | -1751.660 | 0.00 |
| Case 2 | -374.582 | -1674.050 | -1674.050 | 0.00 |
| Case 3 | -505.741 | -2260.261 | -2260.267 | 0.00 |
| Case 4 | -0.551 | -2.193 | -2.193 | 0.00 |

## 6.4 MOSEK

MOSEK is a primal-dual interior point algorithm for solving conic problems. Liu [2013] used MOSEK along with the interfacing software YALMIP for solving the guidance problem. In the version of the software presented here in this chapter, we make use of MOSEK without any interfacing software. The other versions (with other solver and software) and the reasons to not use them are discussed in Chapter 7.

Each solver accepts the conic problem in a different way, so that it can exploit the underlying algorithm in the most efficient way. The following is an explanation of how to use MOSEK for solving SOCP problems. It is partly based on the MOSEK manual [MOSEK ApS, 2015], and mostly on the author's understanding and experience of using MOSEK. The general form in which MOSEK requires a problem is:

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}^{\mathsf{T}}\mathbf{x} + c^f \\ \text{subject to} \quad & \mathbf{l}^c \le \breve{\mathbf{A}}\mathbf{x} \le \mathbf{u}^c, \\ & \mathbf{l}^x \le \mathbf{x} \le \mathbf{u}^x, \\ & \mathbf{x} \in \mathcal{C} \end{aligned} \tag{6.7}$$

where $\mathbf{x}$ is the decision variable vector and will be optimized by the solver, $\breve{\mathbf{A}}$ is the matrix of linear inequality constraints[11] which is bounded by $\mathbf{u}^c$ and $\mathbf{l}^c$. The cone $\mathcal{C}$ is defined suitably to restrict the domain of the problem as per the given constraints. Let

$$\underline{\mathbf{x}}^t \in \mathbb{R}^{n^t}, \quad t = \text{i}, \text{ii}, \ldots, k \tag{6.8}$$

be vectors comprised of members of the decision variable vector $\mathbf{x}$. MOSEK allows each member of the decision variable vector to be part of exactly *one* such $\underline{\mathbf{x}}^t$ vectors. Based on this, we define the cone $\mathcal{C}$ as follows:

$$\mathcal{C} := \left\{ \mathbf{x} \in \mathbb{R}^n : \underline{\mathbf{x}}^t \in \mathcal{C}_t, \; t = \text{i}, \text{ii}, \ldots, k \right\} \tag{6.9}$$

where sub-cones $\mathcal{C}_t$ must be of either of the following forms:

- $\mathbb{R}$ set:

$$\mathcal{C}_t = \left\{ \mathbf{x} \in \mathbb{R}^{n^t} \right\} \tag{6.10}$$

- Second-order cone:

$$\mathcal{C}_t = \left\{ \mathbf{x} \in \mathbb{R}^{n^t} : x_1 \ge \sqrt{\sum_{j=2}^{n^t} x_j^2} \right\} \tag{6.11}$$

An equivalent formulation, in the form described by Equation (4.3), is as follows:

$$\begin{aligned} & \mathcal{C}_t = \left\{ \mathbf{x} \in \mathbb{R}^{n^t} : x_1 \ge \left\| \begin{bmatrix} x_2 & x_3 & \cdots & x_{n^t} \end{bmatrix}' \right\|_2 \right\} \\ & \mathbf{x} \succeq_{\mathcal{C}_t} 0 \end{aligned} \tag{6.12}$$

In simple words, MOSEK takes in linear inequality constraints by using the first two inequalities of Equation (6.7). To define second-order cone constraints, one has to make use of *fictional* vectors[12] $\underline{\mathbf{x}}^t$ which define the relevant cones $\mathcal{C}_t$. The cone $\mathcal{C}$, defines the final domain of the problem due to

[11] The conventional symbol for linear inequality matrix, $\mathbf{A}$, is being used for defining equations of motion constraint in the ConGAL problem formulation. Hence, we shall make use of $\breve{\mathbf{A}}$.

[12] These vectors store only the indexes to the elements from decision variable vector $\mathbf{x}$, and not the elements themselves. Hence, the author refers them by the name *fictional*.

these conic constraints, and is given by the direct product of the cones $\mathcal{C}_t$. Due to MOSEK's underlying structure, MOSEK can allow the members of the decision variable vector $\mathbf{x}$, to be part of only one fictional vector $\underline{\mathbf{x}}^t$. In practise, there is very little chance that a member of the decision variable vector is part of only one second-order conic constraint and/or that that second-order constraint is of the form described in Equation (6.11) or (6.12). To overcome this problem, one is forced to append additional dummy members to the original decision variable vector. This additional members are either duplicates of the original members or form a linear relation with them. This allows the relevant original members to be part of the first second-order constraint, and the dummy members to be part of the next second-order constraint. Naturally, each of these dummy variables are also restricted to be part of only one fictional vector. Additional second-order constraints require the decision variable vector to have additional dummy variables. Thus, instead of solving the original sized problem, the solver now handles a much larger problem.

The following example illustrates working with MOSEK. Consider an optimization problem:

$$\begin{aligned} \text{minimize} \quad & x_1 + x_2 + x_3 + x_4 \\ \text{subject to} \quad & x_1 + x_2 + x_4 \geq 3, \\ & 3x_2 + x_3 + \geq 2, \\ & x_4 - x_3 >= 0, \\ & x_4 >= 2, \\ & x_1 \geq \sqrt{x_2^2 + x_3^2}, \\ & 2x_1 + 1 \geq \sqrt{4x_2^2 + (x_4 - 1)^2} \end{aligned} \tag{6.13}$$

where $\mathbf{x} = [x_1 \; x_2 \; x_3 \; x_4]'$ is the original decision variable vector; the objective is to minimize the sum of its elements; besides linear constraints, we also see two second-order constraints, one of which is not of the form that is directly acceptable by MOSEK. All four elements are part of these second-order constraints but $x_1$ and $x_2$ are present in both.

The problem in Equation (6.13) needs to be re-written in the form of Equation (6.7) to be accepted by MOSEK. To do so, the problem is first written in a new form:

$$\begin{aligned} \text{minimize} \quad & x_1 + x_2 + x_3 + x_4 && (6.14) \\ \text{subject to} \quad & x_1 + x_2 + x_4 \geq 3, && (6.15) \\ & 3x_2 + x_3 + \geq 2, && (6.16) \\ & x_4 - x_3 >= 0, && (6.17) \\ & 2x_1 + 1 = x_5 && (6.18) \\ & 2x_2 = x_6 && (6.19) \\ & x_4 - 1 = x_7 && (6.20) \\ & x_4 \geq 2, && (6.21) \\ & x_1 \geq \sqrt{x_2^2 + x_3^2}, && (6.22) \\ & x_5 \geq \sqrt{x_6^2 + x_7^2} && (6.23) \end{aligned}$$

Three new variables $x_5, x_6, x_7$ are introduced, so that the constraint $2x_1 + 1 \geq \sqrt{4x_2^2 + (x_4 - 1)^2}$ can be written in an equivalent form, as seen in Equation (6.23), which is more suitable for MOSEK. These new variables bring in three new linear constraints, as defined by Equations (6.18) – (6.20). Re-organizing the problem in matrix form, allows us to write it as:

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}^{\mathsf{T}}\tilde{\mathbf{x}} \\ \text{subject to} \quad & \mathbf{l}^{c} \leq \breve{\mathbf{A}}\tilde{\mathbf{x}} \leq \mathbf{u}^{c}, \\ & \mathbf{l}^{x} \leq \tilde{\mathbf{x}} \leq \mathbf{u}^{x}, \\ & \tilde{\mathbf{x}} \in \mathcal{C} \end{aligned} \tag{6.24}$$

where:

$$\begin{aligned} \mathbf{c} &= \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}^{\mathsf{T}} \\ \breve{\mathbf{A}} &= \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 3 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & -1 \end{bmatrix} \\ \mathbf{l}^{c} &= \begin{bmatrix} 3 & 2 & 0 & -1 & 0 & 1 \end{bmatrix}^{\mathsf{T}} \\ \mathbf{u}^{c} &= \begin{bmatrix} \infty & \infty & \infty & -1 & 0 & 1 \end{bmatrix}^{\mathsf{T}} \\ \mathbf{l}^{x} &= \begin{bmatrix} -\infty & -\infty & -\infty & 2 & -\infty & -\infty & -\infty \end{bmatrix}^{\mathsf{T}} \\ \mathbf{u}^{x} &= \begin{bmatrix} \infty & \infty & \infty & \infty & \infty & \infty & \infty \end{bmatrix}^{\mathsf{T}} \end{aligned} \tag{6.25}$$

and,

$$\begin{aligned} \mathcal{C} &= \left\{ \mathbf{x} \in \mathbb{R}^{7} : \underline{\mathbf{x}}^{t} \in \mathcal{C}_{t}, \;\; t = \mathrm{i}, \mathrm{ii} \right\} \\ \underline{\mathbf{x}}^{\mathrm{i}} &= \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}^{\mathsf{T}} \\ \underline{\mathbf{x}}^{\mathrm{ii}} &= \begin{bmatrix} x_5 & x_6 & x_7 \end{bmatrix}^{\mathsf{T}} \\ \mathcal{C}_{t} &= \left\{ \mathbf{x} \in \mathbb{R}^{3} : x_1 \geq \sqrt{x_2^2 + x_3^2} \right\}; \qquad t = \mathrm{i}, \mathrm{ii} \end{aligned} \tag{6.26}$$

The new decision variable vector is $\tilde{\mathbf{x}} = \begin{bmatrix} x_1\, x_2\, x_3\, x_4\, x_5\, x_6\, x_7 \end{bmatrix}'$, where the last three elements are additional dummy variables. The objective function $\mathbf{c}$ is such that only the sum of first four elements of $\tilde{\mathbf{x}}$ are considered, as required by Equation (6.14). Matrix $\breve{\mathbf{A}}$ takes in coefficients from the left-hand-side of the Equations (6.15) – (6.20). These 6 (in)equality constraints are bounded below and above by $\mathbf{l}^{c}$ and $\mathbf{u}^{c}$. Since the Equations (6.18) – (6.20) are equality constraints, their upper and lower bound values are kept equal. Two fictional vectors $\underline{\mathbf{x}}^{\mathrm{i}}$ and $\underline{\mathbf{x}}^{\mathrm{ii}}$, each having distinct members, are used to define the second-order cones of Equations (6.22) and (6.23). The only member of the decision variable vector $\mathbf{x}$ to have a lower or upper limit is $x_4$, as seen in Equation (6.21). This is represented by $\mathbf{l}^{x}$ and $\mathbf{u}^{x}$.

As Equation (6.24) is of the form Equation (6.7), it is acceptable by MOSEK. Its parameters are communicated to MOSEK using a MATLAB structure data type. The MATLAB code for solving this problem is shown in Code 6.1 and its solution is presented in Table 6.6. In the code, a struct `prob` is defined using values from Equations (6.25) and (6.26).

MATLAB Code 6.1: Example run in MOSEK

```matlab
%%MOSEK example run
%Defining the structure 'prob' which will be sent to MOSEK
prob.c = [1 1 1 1 0 0 0]';                      %Objective function
prob.a = [1 1 0 1 0 0 0; 0 3 1 0 0 0 0; 0 0 -1 1 0 0 0; 2 0 0 0 -1 0 0; ...
          0 2 0 0 0 -1 0; 0 0 0 1 0 0 -1];      %Linear constraints
prob.blc = [3 2 0 -1 0 1]';                     %Lower bound on constraints
prob.buc = [inf inf inf -1 0 1]';                %Upper bound on constraints
prob.blx = [-inf -inf -inf 2 -inf -inf -inf]';   %Lower bound on x
prob.bux = [inf inf inf inf inf inf inf]';       %Upper bound on x
conarr(1) = struct('type','MSK_CT_QUAD','sub',[1 2 3]); %First cone
conarr(2) = struct('type','MSK_CT_QUAD','sub',[5 6 7]); %Second cone
prob.cones = cell(2,1);
prob.cones{1} = conarr(1);
prob.cones{2} = conarr(2);

%MOSEK Call
[r,res]=mosekopt('minimize',prob);  %r = status, res = results&info struct

%Solution
xtilde = res.sol.itr.xx;                  %Solution extracted from res struct
x = xtilde(1:4);                       %Original decision vector variable
Obj = res.sol.itr.pobjval;           %Objective value of solved problem
```

Table 6.6: Results from MOSEK example run (Code 6.1)

| Parameter | Value | |
|---|---|---|
| `Obj` | 3.2899 | |
| `xtilde` | 0.8165 | $\left.\right\}$ x |
| | 0.7633 | |
| | -0.2899 | |
| | 2.0000 | |
| | 2.1009 | |
| | 1.5264 | |
| | 1.0000 | |

In the above example, since there were only two second-order constraints, of which only one needed additional dummy variables, it was easy to define the additional dummy variables (and the additional linear constraints associated with them) and the fictional vectors. However, when hundreds of such constraints need to be considered, it becomes a difficult and tedious process to generate the elements of the members of the `prob` structure. This has to be automated and in such a way, that no member of the decision variable vector repeats itself in the fictional vectors. Also, as the additional dummy variables are added to the original decision variable vector, all vectors and matrices in Equation (6.24) need to be updated and re-sized.

Looking at the constraints defined in Chapter 5, 8, it is obvious that all elements of the original decision variable vector are part of more than one second-order constraint. Thus, each constraint would need additional dummy variables. Further, almost all[13] second-order constraints are not of the form described in Equation (6.11) or (6.12). The general form of second-order conic constraints is:

$$\left\|\mathbf{M}\mathbf{x}+\mathbf{b}\right\|_2 \leq \mathbf{e}^{\intercal}\mathbf{x}+f \tag{6.27}$$

where $\mathbf{M} \in \mathbb{R}^{m\times n}$ and $\mathbf{x} \in \mathbb{R}^{\widetilde{N}}$, is the original decision variable vector. Such constraints necessarily need additional dummy variables to convert to the form that is acceptable to MOSEK. Thus, regardless of whether a member of the decision variable vector is repeated in the fictional vectors, the presence of a second-order constraint in its general form, necessitates additional dummy variables. This further calls for the need of automating the process.

The automation process is described as follows. Consider an optimization problem, where $\mathbf{x} \in \mathbb{R}^n$ is the original decision variable vector; the initial set of linear (in)equalities are represented

[13] The only exception to this is the constraint on maximum thrust (C3), as shall be observed in the upcoming sections.

by $\breve{\mathbf{A}}_0 \in \mathbb{R}^{r\times n}$. Let there be $k$ second-order constraints (of the form shown in Equation (6.27)) which need additional dummy variables (for the time being constraints that do not need additional dummy variables are skipped). The additional dummy variables (and the associated fictional vectors) can be easily defined once the constraint is converted to the general conic form (discussed in Section 4.4.2). Let the cone associated with each such second-order constraint Equation (6.27) be $\mathcal{C}_t$. Using Equation (4.14), the constraint can be written using generalized inequality as:

$$\begin{bmatrix}\mathbf{e}^\mathsf{T}\\ \mathbf{M}\end{bmatrix}_t \mathbf{x} + \begin{bmatrix} f\\ \mathbf{b}\end{bmatrix}_t \succeq_{\mathcal{C}_t} 0 \tag{6.28}$$

Now, we introduce a slack variable vector $\mathbf{s} \in \mathcal{C}_t$ consisting of $(m_t + 1)$ elements, and re-arrange the terms:

$$\begin{aligned}\begin{bmatrix} f\\ \mathbf{b}\end{bmatrix}_t &= -\begin{bmatrix}\mathbf{e}^\mathsf{T}\\ \mathbf{M}\end{bmatrix}_t \mathbf{x} + \mathbf{s}\\ \mathbf{s}_t &\succeq_{\mathcal{C}_t} 0\end{aligned} \tag{6.29}$$

This closely resembles the general conic form, for some cone $K$:

$$\begin{aligned}\mathbf{Gx} + \mathbf{s} &= \mathbf{h}\\ \mathbf{s} &\succeq_K 0\end{aligned} \tag{6.30}$$

We define $\mathbf{G}_t = -\begin{bmatrix}\mathbf{e}^\mathsf{T}\\ \mathbf{M}\end{bmatrix}_t$ and $\mathbf{h}_t = \begin{bmatrix} f\\ \mathbf{b}\end{bmatrix}_t$ and rewrite the second-order constraint in the conic form

$$\begin{aligned}\mathbf{G}_t\mathbf{x} + \mathbf{s}_t &= \mathbf{h}_t\\ \mathbf{s} &\succeq_{\mathcal{C}_t} 0\end{aligned} \tag{6.31}$$

This slack vector $\mathbf{s}_t$ provides the additional variables to the original decision variable vector. It is to be noted that each slack vector is defined using a linear relation with $\mathbf{x}$, the original decision variable vector, only. The fictional vector $\underline{\mathbf{x}}_t$ is defined as the index of the elements of the slack vector. With the additional dummy variables and the fictional vectors defined for the $t$-th second-order conic constraint, it is now possible to update and resize the other vectors and matrices of Equation (6.24). This is done as follows:

1. The decision variable vector $\tilde{\mathbf{x}}$ is updated by appending $\mathbf{s}_t$ to it.

2. Objective function $\mathbf{c}$ is updated to include a zero-vector of $(m_t + 1)$ elements.

3. Linear (in)equality matrix $\breve{\mathbf{A}}$ needs to include the linear relations between the original members of decision variable vector and the additional dummy variables. This is illustrated in Figure 6.6. The initial linear inequality matrix $\breve{\mathbf{A}}_0$ gets appended by $\mathbf{G}_t$ from Equation (6.31). Each of these newly added row, is a linear relation of original decision variable vector $\mathbf{x}$ to define the corresponding elements of the slack variable vector $\mathbf{s}_t$. The slack variables are addressed using an Identity matrix of size $(m_t + 1)$. In the process, the original set of (in)equalities get padded by a zero-matrix, as they have no connection with the additional members. All other empty spaces are filled by zeroes as well. The update and resize of $\breve{\mathbf{A}}$ can be traced as per the serial numbers at the bottom left of each matrix in the said figure.

4. The bounds on the constraints $\mathbf{l}^c$ and $\mathbf{u}^c$ get appended by the $\mathbf{h}_t$, as bounds for the new linear equality relations.

5. The bounds on the members of the decision variable vector, $\mathbf{l}^x$ and $\mathbf{u}^x$ get appended by vector of $(m_t + 1)$ elements consisting of $-\infty$ and $\infty$ respectively.

The process continues for $(t = \text{i}, \text{ii}, \ldots, k)$. Now, if there are any second-order constraints that do not need additional dummy variables and are of the form in Equation (6.11) or (6.12), then the process is simpler. No additions are required to any of the vectors and matrices in Equation (6.24), and fictional vectors $\underline{\mathbf{x}}_t$ can be defined straight away by storing the index of the relevant elements of the original decision variable vector, $\mathbf{x}$.

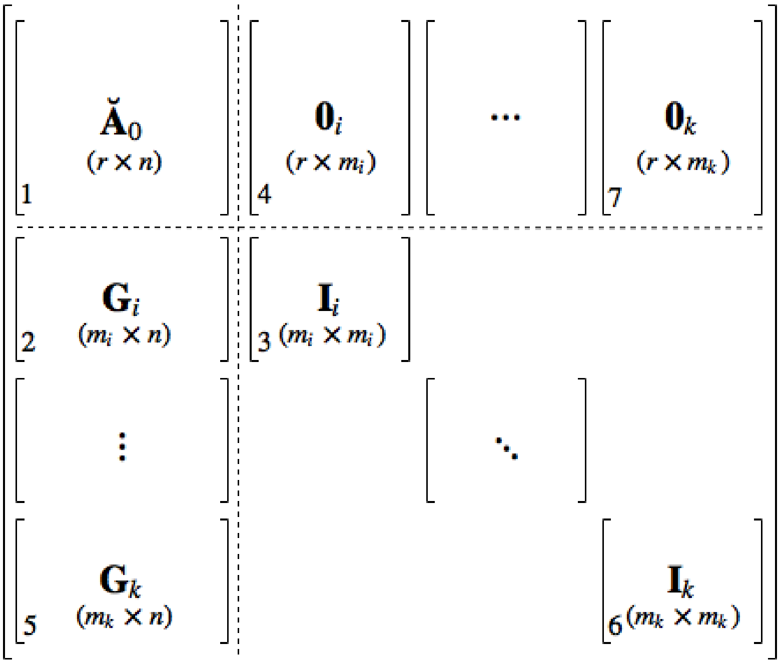


Figure 6.6: Update and resize of MOSEK matrix $\breve{\mathbf{A}}$, consisting of linear (in)equalities constraints of the optimization problem. For each conic constraint $(t = \text{i,ii}, \ldots, k)$ that requires additional dummy variables, the matrix has $\mathbf{G}_t$ appended to it, along with associated padding provided by the zero-matrix and the identity matrix. The serial numbers at bottom left of each sub-matrix guide the flow.

## 6.5 MAIN ENGINE

This section explains the functional sub-blocks of `Main Engine` (ME) block (see Figure 6.2). There is special focus on the constraint generation, as it essentially formulates the problem for the solver. Also covered, are two verification tests that were carried out to ensure that the block functions as desired. We make use of MOSEK as the solver, while the conversion from SOCP formulation to conic formulation is carried out without the use of an interfacing software. The versions with different solver / interfacing software are discussed in Chapter 7.

### 6.5.1 INITIAL CALCULATIONS AND SETUP

The initial calculation / normalization block, is primarily used for carrying out pre-MOSA calculations. This involves converting user-friendly input (like chaser state in LVLH state, initial orientation of target body frame in Euler angles, etc.) to the form required by the ConGAL method. In addition, the docking axis vector propagation using initial attitude and state of target is carried out (details in Section 8.1). Lastly, the relevant parameters are normalized (for minimizing numerical error) for use with the ConGAL based algorithm.

The optimization problem setup is the first step of MOSA, and at the beginning of each iteration, the program checks if the parameters are in order, and resets counters. It is also used for applying the mean-scheme, which is discussed ahead in Section 7.3.

### 6.5.2 CONSTRAINT GENERATION

This is the most important part of the program as it basically defines the problem; it also takes the longest time to get evaluated as hundreds of matrices involving thousands of elements need to be defined. For MOSEK, the formulation is elaborate, as we saw in the previous section. We start be defining the notations used for the program, which differ slightly from the ones used in the previous sections and Chapter 5.

The mission with time of flight $t_f$, is divided into $N$ parts or $N+1$ nodes (inclusive of initial point) with uniform step size $h$. At each node, $i = 0, 1, 2, \ldots, N$, the optimization problem solves for a $(11 \times 1)$ vector $\mathbf{w}_i = \begin{bmatrix} \mathbf{y}_i^\top & \mathbf{u}_i^\top \end{bmatrix}^\top$, where $\mathbf{y}_i = \begin{bmatrix} \mathbf{r}^\top & \mathbf{V}^\top & z \end{bmatrix}_i^\top$ is the $(7 \times 1)$ state vector consisting of inertial radius vector, velocity vector and logarithm of spacecraft mass. The $(4 \times 1)$ control vector $\mathbf{u}_i = \begin{bmatrix} \boldsymbol{\tau}^\top & \sigma \end{bmatrix}_i^\top$ consists of thrust acceleration vector and norm of thrust acceleration. We introduce variables $\widetilde{N} = N + 1$, $n_w = 11$, $n_y = 7$, $n_u = 4$, and $\widetilde{n_w} = \widetilde{N} \cdot n_w$, $\widetilde{n_y} = \widetilde{N} \cdot n_y$, $\widetilde{n_u} = \widetilde{N} \cdot n_u$. The decision variable vector is defined as:

$$\mathbf{W} = \begin{bmatrix} \mathbf{y}_0^\top & \mathbf{y}_1^\top & \mathbf{y}_2^\top & \ldots & \mathbf{y}_N^\top \mathbf{u}_0^\top & \mathbf{u}_1^\top & \mathbf{u}_2^\top & \ldots & \mathbf{u}_N^\top \end{bmatrix}^\top \tag{6.32}$$

which is a $(\widetilde{n_w} \times 1)$ vector. All constraints are formulated for use with $\mathbf{W}$, making it necessary to use what is termed as *extraction* matrices. These matrices help reach out to the specific element of state / control vector at a given node. For example, to reach out to the position vector at third node, we use:

$$\mathbf{E}_r(2)\mathbf{W} \tag{6.33}$$

where

$$\mathbf{E}_r(i) == \underbrace{\begin{bmatrix} \underbrace{0 \;\cdots\; 0}_{i \cdot n_y} & 1 & 0 & 0 & 0 & \cdots & 0 \\ & & 1 & & & & \\ & & & 1 & & & \end{bmatrix}}_{\widetilde{n_w}} \tag{6.34}$$

Other extraction matrices shall be defined as they come up.

Each of the many types of constraints is identified by the acronym $\mathsf{Cq}$, which is written in sans-serif font and the $\mathsf{q}$ refers to an index (generally a number) assigned to it. All associated parameters with that particular constraint use the same index as a subscript. Any pre-superscript / pre-subscript that is associated with a parameter is merely used for ensuring unique names. Linear (in)equalities make use of $\mathbf{L}$ and $\mathbf{B}$, while conic inequalities use $\mathbf{G}$ and $\mathbf{h}$. In each iteration $k$, the following set of constraints gets formulated:

1. $\mathsf{C0}$, constraint for defining initial state: This is a linear constraint that is used to define the initial state of chaser in inertial frame. It is given by:

$$\mathbf{L}_0\mathbf{W} = \mathbf{B}_0 \tag{6.35}$$

where

$$\mathbf{L}_0 = \begin{bmatrix} \mathbf{E}_r(0) \\ \mathbf{E}_V(0) \\ \mathbf{E}_z(0) \end{bmatrix} \tag{6.36}$$

where

$$\mathbf{E}_V(i) = \underbrace{\begin{bmatrix} \underbrace{0 \;\cdots\; 0}_{i \cdot n_y} & 0 & 0 & 0 & 1 & 0 & 0 & 0 & \cdots & 0 \\ & & & & & 1 & & & & \\ & & & & & & 1 & & & \end{bmatrix}}_{\widetilde{n_w}} \tag{6.37}$$

$$\mathbf{E}_z(i) = \Big[\underbrace{\underbrace{0 \ \cdots \ 0}_{i\cdot n_y} \ \underbrace{0 \ \cdots \ 0}_{6} \ 1 \ 0 \ \cdots \ 0}_{\widetilde{n_w}}\Big] \tag{6.38}$$

are the extraction vectors for velocity and spacecraft mass. The initial inertial state of the chaser is provided as:

$$\mathbf{B}_0 = \begin{bmatrix} \mathbf{r}^\mathsf{T} & \mathbf{V}^\mathsf{T} & z \end{bmatrix}_0^\mathsf{T} \tag{6.39}$$

2. C1, constraint for equations of motion: This is a linear constraint that relates the state of chaser at each node with the dynamics of the system and the control vector. This is based on the discussion in Section 5.4 and makes use of the trapezoidal rule for integration. Its general form is given as:

$$\left(-\mathbf{I} + \frac{1}{2}h\mathbf{A}_i\right)\mathbf{y}_i + \left(\mathbf{I} + \frac{1}{2}h\mathbf{A}_{i-1}\right)\mathbf{y}_{i-1} + \frac{1}{2}h\mathbf{B}\mathbf{u}_i + \frac{1}{2}h\mathbf{B}\mathbf{u}_{i-1} = 0 \tag{6.40}$$

and sent to the solver as:

$$\mathbf{L}_1\mathbf{W} = \mathbf{0}_{\widetilde{n_y}\times 1} \tag{6.41}$$

with

$${}^{(1)}\mathbf{L}_1 = \begin{bmatrix} \mathbf{0}_{7\times7} & \mathbf{0}_{7\times7} & \mathbf{0}_{7\times7} & \cdots & \cdots & \mathbf{0}_{7\times7} \\ \mathbf{I} + \frac{h}{2}\mathbf{A}_1 & -\mathbf{I} + \frac{h}{2}\mathbf{A}_2 & \mathbf{0}_{7\times7} & \cdots & \cdots & \mathbf{0}_{7\times7} \\ \mathbf{0}_{7\times7} & \mathbf{I} + \frac{h}{2}\mathbf{A}_2 & -\mathbf{I} + \frac{h}{2}\mathbf{A}_3 & \mathbf{0}_{7\times7} & \cdots & \mathbf{0}_{7\times7} \\ \vdots & \mathbf{0}_{7\times7} & \mathbf{I} + \frac{h}{2}\mathbf{A}_3 & -\mathbf{I} + \frac{h}{2}\mathbf{A}_4 & \mathbf{0}_{7\times7} & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots \\ \mathbf{0}_{7\times7} & \cdots & \cdots & \mathbf{0}_{7\times7} & \mathbf{I} + \frac{h}{2}\mathbf{A}_{N-1} & -\mathbf{I} + \frac{h}{2}\mathbf{A}_N \end{bmatrix} \tag{6.42}$$

$${}^{(2)}\mathbf{L}_1 = \begin{bmatrix} \mathbf{0}_{7\times4} & \mathbf{0}_{7\times4} & \mathbf{0}_{7\times4} & \cdots & \cdots & \mathbf{0}_{7\times4} \\ \frac{\mathbf{B}h}{2} & \frac{\mathbf{B}h}{2} & \mathbf{0}_{7\times4} & \cdots & \cdots & \mathbf{0}_{7\times4} \\ \mathbf{0}_{7\times4} & \frac{\mathbf{B}h}{2} & \frac{\mathbf{B}h}{2} & \mathbf{0}_{7\times4} & \cdots & \mathbf{0}_{7\times4} \\ \vdots & \mathbf{0}_{7\times4} & \frac{\mathbf{B}h}{2} & \frac{\mathbf{B}h}{2} & \mathbf{0}_{7\times4} & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots \\ \mathbf{0}_{7\times4} & \cdots & \cdots & \mathbf{0}_{7\times4} & \frac{\mathbf{B}h}{2} & -\frac{\mathbf{B}h}{2} \end{bmatrix} \tag{6.43}$$

$$\mathbf{L}_1 = \begin{bmatrix} {}^{(1)}\mathbf{L}_1 & {}^{(2)}\mathbf{L}_1 \end{bmatrix} \tag{6.44}$$

Here, ${}^{\mathbf{y}}\mathbf{L}_1 \in \mathbb{R}^{\widetilde{n_y}\times\widetilde{n_y}}$ and ${}^{\mathbf{u}}\mathbf{L}_1 \in \mathbb{R}^{\widetilde{n_y}\times\widetilde{n_u}}$. The topmost row of $\mathbf{L}_1$ consists of all-zeros, as the initial state is exempt from the equations of motion.

3. C2, constraint on maximum thrust acceleration magnitude: This is an inequality constraint that limits the value of magnitude of thrust acceleration. It accounts for the change in mass the spacecraft experiences due to any previous thrust burns. Its general form is given by:

$$0 \le \sigma(t) \le T_{\max} e^{-z^{(k-1)}(t)}\{1 - [z(t) - z^{(k-1)}(t)]\} \tag{6.45}$$

and is formulated for MOSEK as:

$$\mathbf{L}_2 \equiv \begin{bmatrix} {}^{(1)}\mathbf{L}_2 \\ {}^{(2)}\mathbf{L}_2 \end{bmatrix} \le \begin{bmatrix} {}^{(1)}\mathbf{B}_2 \\ {}^{(2)}\mathbf{B}_2 \end{bmatrix} \equiv \mathbf{B}_2 \tag{6.46}$$

for $i = 1, 2, \ldots, N$, where

$$ {}^{(1)}\mathbf{L}_2 = \begin{bmatrix} \mathbf{E}_\sigma(1) + T_{\max}\exp(-\mathbf{E}_z(1)\mathbf{W}^{(k-1)})\cdot\mathbf{E}_z(1) \\ \mathbf{E}_\sigma(2) + T_{\max}\exp(-\mathbf{E}_z(2)\mathbf{W}^{(k-1)})\cdot\mathbf{E}_z(2) \\ \vdots \\ \mathbf{E}_\sigma(N) + T_{\max}\exp(-\mathbf{E}_z(N)\mathbf{W}^{(k-1)})\cdot\mathbf{E}_z(N) \end{bmatrix} \qquad {}^{(2)}\mathbf{L}_2 = -\begin{bmatrix} \mathbf{E}_\sigma(1) \\ \mathbf{E}_\sigma(2) \\ \vdots \\ \mathbf{E}_\sigma(N) \end{bmatrix} \tag{6.47} $$

$$ {}^{(1)}\mathbf{B}_2 = \begin{bmatrix} T_{\max}\exp(-\mathbf{E}_z(1)\mathbf{W}^{(k-1)})\cdot(1+\mathbf{E}_z(1)) \\ T_{\max}\exp(-\mathbf{E}_z(2)\mathbf{W}^{(k-1)})\cdot(1+\mathbf{E}_z(2)) \\ \vdots \\ T_{\max}\exp(-\mathbf{E}_z(N)\mathbf{W}^{(k-1)})\cdot(1+\mathbf{E}_z(N)) \end{bmatrix} \qquad {}^{(2)}\mathbf{B}_2 = \mathbf{0}_{N\times 1} \tag{6.48} $$

and

$$ \mathbf{E}_\sigma(i) = \begin{bmatrix} \underbrace{\underbrace{0 \ \cdots \ 0}_{\widetilde{n_y}} \quad \underbrace{0 \ \cdots \ 0}_{i\cdot n_u} \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0 \ \cdots \ 0}_{\widetilde{n_w}} \end{bmatrix} \tag{6.49} $$

In the above equations, parameters with superscript $k-1$ refer to values from the previous iteration, The constraint is defined only from $i = 1$ onwards, as the initial node's control vector is set to zero.

4. C3, constraint on thrust acceleration vector: This is a second-order-conic constraint that ensures that the norm of the thrust accleration vector is equal to or less than the thrust acceleration magnitude defined in C2. The general form is:

$$ \left\| \boldsymbol{\tau}(t) \right\| \leq \sigma(t) \tag{6.50} $$

It is the first conic constraint in our formulation and also of the form described in Equation 6.12. Thus, it does not need any dummy variables for its formulation. We can straight away define the fictional vector as:

$$ \underline{\mathbf{x}}_3 = \begin{bmatrix} \mathbf{B}(1)^\intercal & \mathbf{B}(2)^\intercal & \cdots & \mathbf{B}(N)^\intercal \end{bmatrix}^\intercal \tag{6.51} $$

where

$$ \mathbf{B}(i) = \begin{bmatrix} \widetilde{n_y} + i\cdot n_u \\ \widetilde{n_y} + i\cdot n_u - 3 \\ \widetilde{n_y} + i\cdot n_u - 2 \\ \widetilde{n_y} + i\cdot n_u - 1 \end{bmatrix} \tag{6.52} $$

and pass it on to MOSEK during the solver-call block. Note that the maximum thrust constraint is jointly carried out by C2, which sets maximum thrust magnitude, and C3, which ensures norm of thrust vector is less than that value.

5. C4, constraint for defining hold points: It is a linear constraint that momentarily ensures that the chaser state matches a particular value. Its general form is:

$$ \mathbf{x}(t) = \mathbf{d} \tag{6.53} $$

and it it is formulated as:

$$ \mathbf{L}_4\mathbf{W} = \mathbf{B}_4 \equiv \mathbf{d}_4 \tag{6.54} $$

where

$$ \mathbf{L}_4 = \begin{bmatrix} \mathbf{E}_r(i) \\ \mathbf{E}_V(i) \end{bmatrix} \tag{6.55} $$

The extraction matrices help match the value of position and velocity at the node $i$, which corresponds to the hold point time, with vector $\mathbf{d}_4$, which is the normalized, inertial value of state calculated from the user-input LVLH state.

6. C5, constraint for defining approach cone: It is a conic constraint that ensures that the chaser lies within a cone that starts at the target and is pointed along the docking axis vector, $\mathbf{1}_n(t)$. It is specified by the half-cone angle, $\alpha$. Its general form is:

$$\left\| \mathbf{r}(t) - \mathbf{r}_t(t) \right\| \cos\alpha \leq \mathbf{1}_n^\top(t) \cdot (\mathbf{r}(t) - \mathbf{r}_t(t)) \tag{6.56}$$

It is not in the standard form we defined in Equation 6.12, but of the form in Equation 6.27. Hence, it is converted to conic form with the following steps:

$$\left\| \mathbf{r}(t) - \mathbf{r}_t(t) \right\| \leq \frac{\mathbf{1}_n^\top(t)}{\cos\alpha} \cdot (\mathbf{r}(t)) - \frac{\mathbf{1}_n^\top(t)}{\cos\alpha} (\mathbf{r}_t(t)) \tag{6.57}$$

$$-\begin{bmatrix} \frac{\mathbf{1}_n^\top(t)}{\cos\alpha} \cdot (\mathbf{r}(t)) \\ \mathbf{r}(t) \end{bmatrix} + \mathbf{s} = \begin{bmatrix} -\frac{\mathbf{1}_n^\top(t)}{\cos\alpha} (\mathbf{r}_t(t)) \\ -\mathbf{r}_t(t) \end{bmatrix} \tag{6.58}$$

$$\text{with} \quad \mathbf{s}_5 \succeq_{K_5} 0$$

This constraint is defined individually at each node (unlike the previous constraints) as:

$$\mathbf{G}_5(i)\mathbf{W} + \mathbf{s}_5(j) = \mathbf{h}_5(i) \tag{6.59}$$

where

$$\mathbf{G}_5(i) = -\begin{bmatrix} \frac{\mathbf{1}_n^\top(i)}{\cos\alpha} \cdot \mathbf{E}_r^\top(i) \\ \mathbf{E}_r(i) \end{bmatrix} \tag{6.60}$$

$$\mathbf{h}_5(i) = \begin{bmatrix} -\frac{\mathbf{1}_n^\top(i)}{\cos\alpha} (\mathbf{r}_t(i)) \\ -\mathbf{r}_t(i) \end{bmatrix} \tag{6.61}$$

Note that $\mathbf{G}_5 \in \mathbb{R}^{4\times\widetilde{n_w}}$, $\mathbf{h}_5 \in \mathbb{R}^{4\times 1}$ and $i \in \{i_{\min}, i_{\max}\}$, the bounds where the constraint is active. The extraction matrix $\mathbf{E}_r$ is the same as defined in C4; the subscript $t$ refers to target parameters. The slack vector $\mathbf{s}_5(j)$ is defined at the very end of this list (after all constraints are defined).

7. C6, constraint on plume impingement angle: It is an inequality constraint that is used to ensure that the chaser thrust plume does not damage the target. This is enforced by restricting the angle which the thrust vector makes with the docking axis vector to be above a certain value, $\beta$. Its general form is:

$$\mathbf{1}_n^\top \boldsymbol{\tau} \leq \sigma \cos\beta \tag{6.62}$$

and it is sent to MOSEK as:

$$\mathbf{L}_6 \mathbf{W} \leq \mathbf{B}_6 \tag{6.63}$$

where

$$\mathbf{L}_6 = \begin{bmatrix} \mathbf{1}_n^\top(i_{\min})\mathbf{E}_\tau(i_{\min}) \\ \mathbf{1}_n^\top(i_{\min}+1)\mathbf{E}_\tau(i_{\min}+1) \\ \vdots \\ \mathbf{1}_n^\top(i_{\max})\mathbf{E}_\tau(i_{\max}) \end{bmatrix} \qquad \mathbf{B}_6 = \cos\beta \begin{bmatrix} \mathbf{E}_\sigma(i_{\min}) \\ \mathbf{E}_\sigma(i_{\min}+1) \\ \vdots \\ \mathbf{E}_\sigma(i_{\max}) \end{bmatrix} \tag{6.64}$$

The extraction matrix $\mathbf{E}_\tau$ is given as:

$$\mathbf{E}_\tau(i) = \underbrace{\begin{bmatrix} \underbrace{0 \ \cdots \ 0}_{\widetilde{n_y}} & \underbrace{0 \ \cdots \ 0}_{i\cdot n_u} & 1 & 0 & 0 & 0 & \cdots & 0 \\ & & & 1 & & & & \\ & & & & 1 & & & \end{bmatrix}}_{\widetilde{n_w}} \tag{6.65}$$

8. C7, constraint on thrust rate change: This is a modified form of the constraint defined by Liu [2013]. The reasons and details are discussed in Section 7.3. Its general form is:

$$|\dot{\sigma}| \leq \lambda \tag{6.66}$$

and it is formulated as:

$$\mathbf{L}_7 \equiv \begin{bmatrix} {}^{(1)}\mathbf{L}_7 \\ -{}^{(1)}\mathbf{L}_7 \end{bmatrix} \leq \begin{bmatrix} {}^{(1)}\mathbf{B}_7 \\ {}^{(1)}\mathbf{B}_7 \end{bmatrix} \equiv \mathbf{B}_7 \tag{6.67}$$

where

$${}^{(1)}\mathbf{L}_7 = \begin{bmatrix} \mathbf{E}_\sigma(i_{\min}+1) - \mathbf{E}_\sigma(i_{\min}) \\ \mathbf{E}_\sigma(i_{\min}+1) - \mathbf{E}_\sigma(i_{\min}+1) \\ \vdots \\ \mathbf{E}_\sigma(i_{\max}) - \mathbf{E}_\sigma(i_{\max}-1) \end{bmatrix} \qquad {}^{(1)}\mathbf{B}_7 = \lambda \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \tag{6.68}$$

9. Cf, constraint for defining final state: This is a linear constraint that is used to define the final state of chaser in inertial frame. It is given by:

$$\mathbf{L}_f \mathbf{W} = \mathbf{B}_f \tag{6.69}$$

where

$$\mathbf{L}_f = \begin{bmatrix} \mathbf{E}_r(N) \\ \mathbf{E}_V(N) \end{bmatrix} \qquad \mathbf{B}_f = \begin{bmatrix} \mathbf{r}^\mathsf{T} & \mathbf{V}^\mathsf{T} \end{bmatrix}^\mathsf{T}_{(N)} \tag{6.70}$$

These are the constraints that we shall encounter most often during this report. Other constraints are explained / defined as and when required (along with necessary background). After generating the above constraints, we are still left to generate the new decision variable vector $\widetilde{\mathbf{W}}$ consisting of the original decision variable vector $\mathbf{W}$ and the yet-to-be-defined slack vectors (for each conic constraint at a given node). We now define the linear (in)equality matrix $\breve{\mathbf{A}}_0$ discussed in Section 6.4:

$$\breve{\mathbf{A}}_0 = \begin{bmatrix} \mathbf{L}_0^\mathsf{T} & \mathbf{L}_1^\mathsf{T} & \mathbf{L}_2^\mathsf{T} & \mathbf{L}_4^\mathsf{T} & \mathbf{L}_6^\mathsf{T} & \mathbf{L}_6^\mathsf{T} & \mathbf{L}_f^\mathsf{T} \end{bmatrix}^\mathsf{T} \tag{6.71}$$

Its size depends on the number of linear constraints defined for the problem. Let it be defined as a $n_{A0} \times \widetilde{n_w}$ matrix. For every conic constraint (index $q$) at a given node $i$, we append $\mathbf{G}_q(i)$ to this matrix, with the end result being a $(n_{A0} + j_m \cdot 4 \times \widetilde{n_w})$ matrix, where $j_m$ is the total number of such constraints. The slack vector at each such constraint is given by:

$$\mathbf{s}_q(j) = \left[ \underbrace{\begin{matrix} \underbrace{0 \ \cdots \ 0}_{(j-1)\cdot 4} & 1 & 0 & 0 & 0 & \cdots & 0 \\ & & 1 & & & & \\ & & & 1 & & & \\ & & & & 1 & & \end{matrix}}_{\widetilde{n_w}} \right] \tag{6.72}$$

for $j = 1, 2, \ldots, j_m$. The stacked form of these vectors is an identity vector of size $(j_m \cdot 4 \times j_m \cdot 4)$. This helps us write the final form of linear inequality matrix $\breve{\mathbf{A}}$ as:

$$\breve{\mathbf{A}} = \begin{bmatrix} \breve{\mathbf{A}}_0 & \mathbf{0}_{n_{A0} \times j_m \cdot 4} \\ \mathbf{G} & \mathbf{I} \end{bmatrix} \tag{6.73}$$

where $\mathbf{G}$ is the stack of all matrices $\mathbf{G}_q(i)$ from all relevant constraints and nodes. Next, we define the stacked fictional vector as:

$$\underline{\mathbf{x}} = \begin{bmatrix} \mathbf{B}(1)^\mathsf{T} & \mathbf{B}(2)^\mathsf{T} & \cdots & \mathbf{B}(j_{\max})^\mathsf{T} \end{bmatrix}^\mathsf{T} \tag{6.74}$$

where

$$\mathbf{B}(j) = \begin{bmatrix} \widetilde{n_y} + j \cdot 4 \\ \widetilde{n_y} + j \cdot 4 + 1 \\ \widetilde{n_y} + j \cdot 4 + 2 \\ \widetilde{n_y} + j \cdot 4 + 3 \end{bmatrix} \tag{6.75}$$

Note that the order in which we define conic constraints does not matter. Only the linear relation between the $\mathbf{G}_q(i)$ to corresponding $\mathbf{s}_q(j)$ is important. As long as each conic constraint gets its share of additional dummy variables, the placement of the linear set of equations is irrelevant.

Our new decision variable vector is, $\widetilde{\mathbf{W}}$ is of the size $(\widetilde{n_w} + j_m \cdot 4 \times 1)$. All constraints were defined with respect to $\mathbf{W}$ but were updated to work with $\widetilde{\mathbf{W}}$. We are still left to define the objective function, which is to minimize the propellant consumption. Its general form is given as:

$$J = \int_0^{t_f} \sigma \mathrm{dt} \tag{6.76}$$

and is modelled as:

$$\mathbf{L}_{ob}\widetilde{\mathbf{W}} \tag{6.77}$$

where

$$\mathbf{L}_{ob} = h \begin{bmatrix} \underbrace{\underbrace{0 \ \cdots \ 0}_{\widetilde{n_w}+4} \ \ \underbrace{0 \ \ 0 \ \ 0 \ \ 1}_{\times N} \ \ 0 \ \ 0 \ \cdots \ 0}_{\widetilde{n_w}+j \cdot 4} \end{bmatrix} \tag{6.78}$$

This extracts the $\sigma$ element from the $\widetilde{W}$ vector for $i = 1, 2, \ldots, N$ and sums them.

### 6.5.3 SOLVER CALL AND SOLUTION

The solver call is carried out by gathering the constraints and objective function in the 'Prob' variable of structure data type in MATLAB and invoking the 'mosekopt' function, as explained in Section 6.4. MOSEK tries to solve the problem and provides output which can fall into one of these three categories: An optimal solution is found; a stall / near-optimal solution is found; problem is infeasible. Stall / near-optimal solution is one where some of the constraints of the problem are violated and the solution found can have a reduced accuracy. This is not an acceptable solution (despite its deceptive name, 'near-optimal').

From the generated solution for $\widetilde{\mathbf{W}}$, we extract $\mathbf{W}$ and check for convergence. The problem is said to be converged if the maximum difference (in position and velocity) between consecutive iterations is below a particular tolerance value. This is written as:

$$\left\| \mathbf{E}_r(i) \cdot \left( \mathbf{W} - \mathbf{W}^{(k-1)} \right) \right\| \leq \epsilon_r \tag{6.79}$$

$$\left\| \mathbf{E}_V(i) \cdot \left( \mathbf{W} - \mathbf{W}^{(k-1)} \right) \right\| \leq \epsilon_V \tag{6.80}$$

If the problem has converged, then it sends relevant parameters back to the `Outputs` block, where operations like un-normalizing and plotting graphs are carried out. If the operation has not converged, then the problem carries out another iteration of the MOSA loop.

### 6.5.4 VERIFICATION TESTS

Having built the `Main Engine`, it is put to test for verification. In particular, two tests are carried out here while further analysis is carried out in Chapter 7. For these tests, it is assumed that Earth is the central body and there are no perturbations acting on the spacecraft. The constants or parameters that are used during the tests are shown in Table 6.7. The tests are as follows:

1. Liu numerical example 1: Test with numerical example from Liu [2013] for constraints C0, C1, C2, C3, Cf.

2. Liu numerical example 2: Test with numerical example from Liu [2013] for constraints C0, C1, C2, C3, C4, C5, C6, C7, Cf.

Table 6.7: Parameters used for `Main Engine` verification tests

| Parameter | Value | Units |
|---|---|---|
| Earth gravitational parameter ($\mu$) | $3.986004415 \times 10^{14}$ | $m^3s^{-2}$ |
| Earth equatorial radius ($R_0$) | $6.378136 \times 10^6$ | m |
| Gravitational acceleration ($g_0$) | 9.80665 | $m/s^2$ |
| Spacecraft initial mass ($m_0$) | 1385 | kg |
| Spacecraft dry mass ($m_s$) | 1108 | kg |
| Specific Impulse ($I_{sp}$) | 200 | s |

### ME TEST 1: LIU NUMERICAL EXAMPLE 1

ME Test 1 test compares the result with a numerical example from Liu, specifically the example in section 3.4.2, page 43 of Liu [2013]. The problem consists of a far-range rendezvous mission, and the initial parameters for the problem are shown in Table 6.8. The target is in a highly eccentric orbit ($e = 0.5$), with perigee at 370 km altitude. The chaser is initially 100 km below and behind it and has to reach 5 km behind it after 1800 s of flight. Besides the initial and final condition, the only constraints acting on the chaser are those of maximum thrust, which is restricted to 831 N.

Table 6.8: Input to the ME

| Parameter | Value | Units |
|---|---|---|
| | Mission inputs | |
| Time of flight | 1800 | s |
| Step size | 9 | s |
| Convergence | [ 50  1 ] | [m,m/s] |
| | Target inputs | |
| Initial orbit | 370.00 × 13866.27 | km |
| Initial true anomaly | 5.0 | ° |
| | Chaser inputs | |
| Initial LVLH position | [ −100  0  100 ] | km |
| Initial LVLH velocity | [ 1  0  0.5 ] | m/s |
| Final LVLH position | [ −5  0  0 ] | km |
| Final LVLH velocity | [ 0  0  0 ] | m/s |
| Maximum thrust | 831 | N |

The results from the test are shown in Figure 6.7. The `Main engine` is able to solve the problem and converges at 9 iterations. The thrust profile, as seen in Figure 6.7a, (with legend entry, Bhagat), is a two burn manoeuvre. The chaser state, as propagated by the algorithm, is shown in the LVLH frame in Figure 6.7b. It is observed the final state condition (Cf) and the maximum thrust constraint (C2,C3) are easily met. The flight consumed 201.6 kg of propellant. For validation, the generated thrust command is sent to the `Orbit Propagator` (OP), and the propagated state is also shown in

Figure 6.7a (with legend entry, OP Bhagat). We see that the trajectory matches greatly and zooming in, as seen in Figure 6.7c, we see there is a slight drift of about 200 m. Considering the range of the flight, and keeping in mind that the integrator in the `Main Engine` is only second-order, the results are acceptable.

In comparision, the results from Liu [2013] are plotted as well (with legend entries, Liu). We see that the thrust profile between the two simulators matches well, with only tiny variations at points of abrupt changes in thrust magnitude. Correspondingly, the chaser states in LVLH frame, match well too. However, upon feeding Liu's generated thrust profile to the OP, we see that there is notable variation in the LVLH trajectory (legend entry, OP Liu). This difference was eventually attributed to minor differences in scaling factors used by Liu [2013] (this is further explained in Chapter 7)

#### ME TEST 2: LIU NUMERICAL EXAMPLE 2

ME Test 2 compares the result with another numerical example from Liu, specifically the example in section 3.4.3, page 46 of Liu [2013]. The problem consists of a highly constrained close-range rendezvous mission, and its initial parameters are shown in Table 6.9. The target is in a circular orbit of 370 km altitude. The chaser is initially 5000 m ahead and 200 m above it and has to match the target's state after 3000 s of flight. The following constraints are acting on it:

1. Maximum thrust constraint (C2, C3): Thrust is restricted to 21 N during the entire flight.
2. Hold point constraint (C4): The chaser should reach $[200,0,0]$ at 1800 s and have a velocity less than 0.2 m/s.
3. Approach cone (C5): From 1800 s onwards, the chaser should remain inside a 10° half-angle cone along the $+V$-bar axis.
4. Plume impingement constraint (C6): From 1800 s to 2520 s, the plume angle should be greater than 60° and from 2520 onwards until end of flight, it should be 89°.
5. Rate of change of thrust (C7): From 1800 s to 2520 s, the thrust magnitude should not change by a rate greater than 0.1 N/s, and from then on the rate should be restricted to 0.05 N/s.

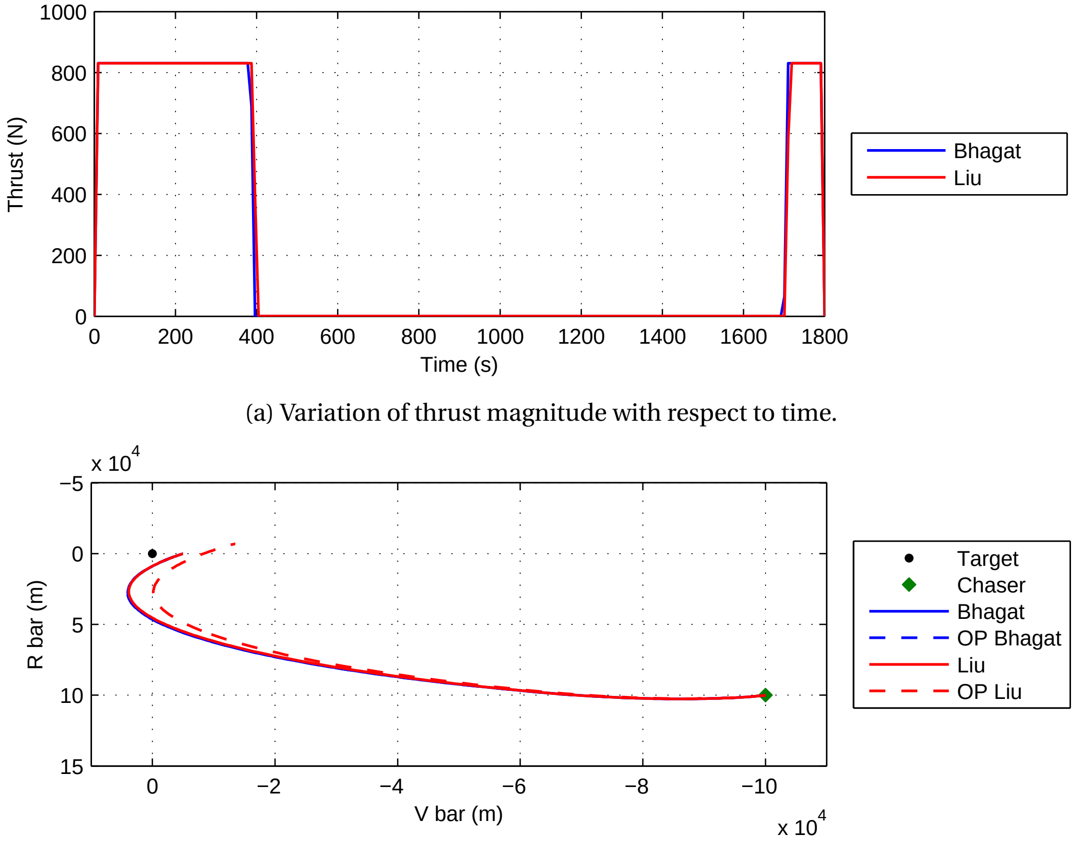
Table 6.9: Input to the ME

| Parameter | Value | Units |
|---|---|---|
| | Mission inputs | |
| Time of flight | 3000 | s |
| Step size | 10 | s |
| Convergence | $[\,50 \quad 1\,]$ | [m,m/s] |
| | Target inputs | |
| Initial orbit | $370.00 \times 370$ | km |
| Initial true anomaly | 0.0 | ° |
| | Chaser inputs | |
| Initial LVLH position | $[\,5000 \quad 0 \quad -200\,]$ | m |
| Initial LVLH velocity | $[\,0.5 \quad 0 \quad 0.06\,]$ | m/s |
| Final LVLH position | $[\,0 \quad 0 \quad 0\,]$ | m |
| Final LVLH velocity | $[\,0 \quad 0 \quad 0\,]$ | m/s |
| Maximum thrust | 21 | N |

The results from the test are shown in Figure 6.8. Initially, the ME falls into a repeating loop of iterations which does not converge on its own. However, a method for faster convergence was developed and applied, called the *mean-scheme* (the details are explained in Section 7.3.1). This made the ME to converge in 14 iterations. In Figure 6.8a, we see the generated thrust profile (with legend entry, Bhagat). Up until 1800 s, when the constraint on thrust rate change kicks in, the thrust profile looks like a two-burn manoeuvre with maximum thrust magnitude. After which, the thrust is constrained to change gradually, as observed by the incline near 1800 s. There are two other burns of smaller magnitude performed at about 2600 seconds and end of flight (although, this burn is not discernible in the current scale of the image). In Figure 6.8b, we see the trajectory of the chaser in LVLH frame (with legend entry, Bhagat) and a zoomed-in version is shown in Figure 6.8c. The chaser manages to remain well within the approach cone. The burn at 2600 seconds is carried out to ensure that the chaser does not violate the approach cone constraint, whereas the burn just before 3000 s kills the relative velocity between the chaser and the target. Overall, the propellant consumption was about 4.44 kg. The thrust profile generated was provided to the OP for validation and its results are shown as well (with legend entry, OP Bhagat). For most part, the trajectory matches and the drift at end of flight is about 50 m. Thus, the ME passes this test as well.

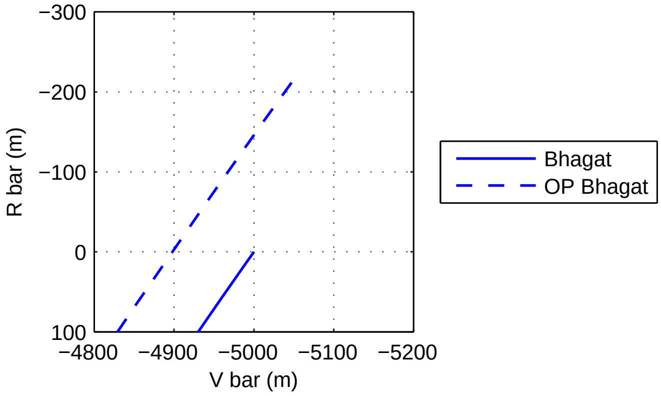
In comparision, the results from Liu [2013] are plotted as well (with legend entries, Liu). There is absolutely nothing that matches between his solution and the ME solution. Liu's solution converged in 4 iterations which is much lesser than what was taken by the ME to converge. However, Liu's solution uses about 4 times as much propellant as compared to the ME solution. The thrust profile looks much different, qualitatively as well. For example, from 1200 s onwards and upto the end of flight, the thrust magnitude is non-zero. Liu's solution does not violate any constraints, although the thrust magnitude at the end of flight remains non-zero (which is expected to create problems as it takes a finite time to stop the thrusters). Also, in Figure 6.8c, we see that the chaser trajectory is, for most part, at the boundary of the approach cone.

Initially, it was presumed that this difference in his and our solutions is due to some mistake in ME. However, that would not explain the similarity between OP trajectory and ME trajectory. Upon feeding Liu's thrust profile to the OP, we see that the resulting trajectory does not match with the trajectory expected by Liu's solution. The OP trajectory quickly goes outside the frame of the plot and at end of flight, the distance between the expected state and real state is almost 44 km! Clearly, there appears to be a mistake from Liu's side and the example provided is faulty. It was eventually discovered that not only are his scaling factors different from ours, but they are also inconsistent (see Section 7.4).



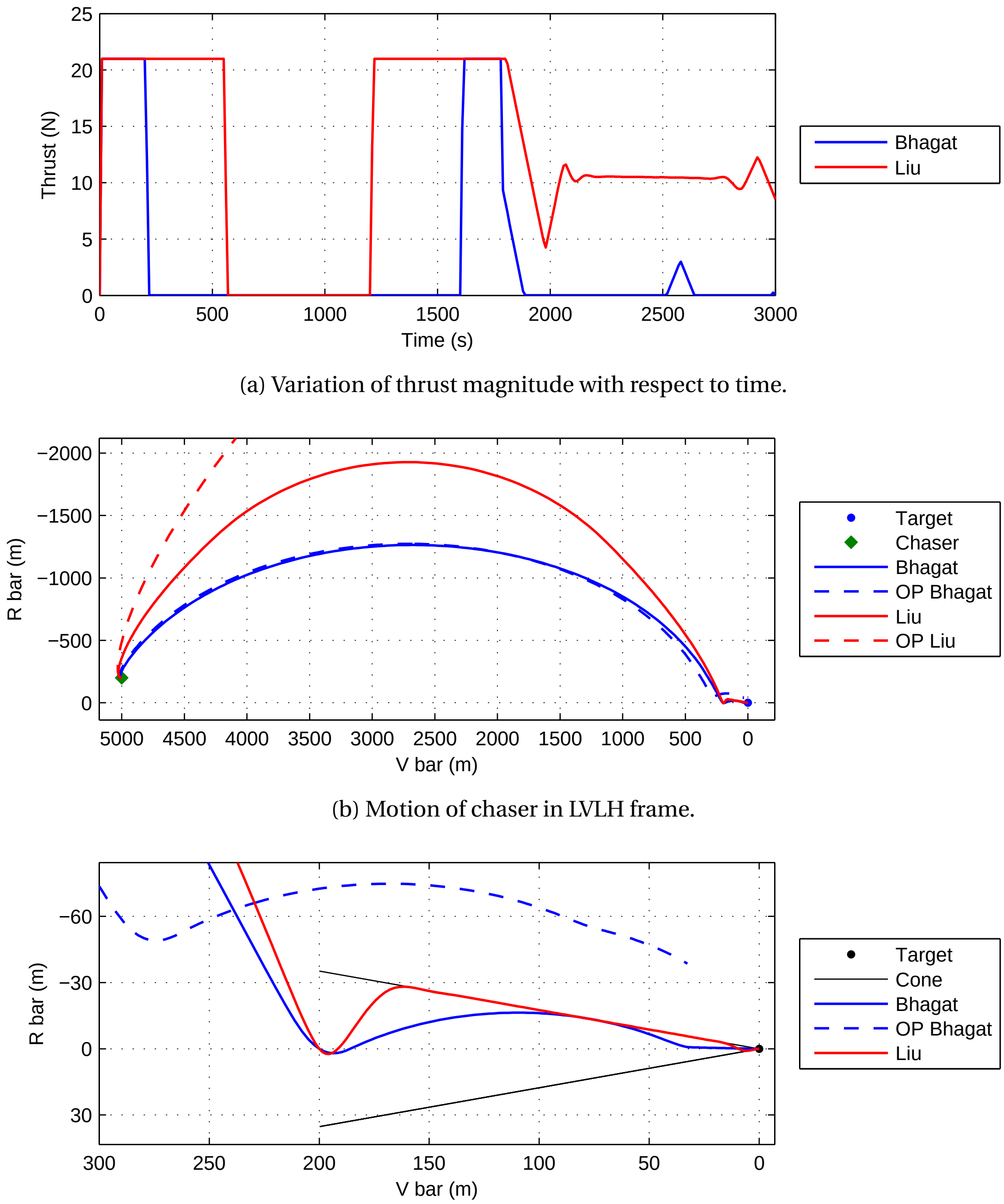
(a) Variation of thrust magnitude with respect to time.

(b) Motion of chaser in LVLH frame.



(c) Zoomed in view of motion of chaser in LVLH frame.

Figure 6.7: ME Test 1: Chaser is sent from LVLH [-100,0,100] km to [-5,0,0] km in 1800 s with maximum allowable thrust of 831 N



(a) Variation of thrust magnitude with respect to time.

(b) Motion of chaser in LVLH frame.

(c) Zoomed in view of motion of chaser in LVLH frame. The OP Liu trajectory is out of frame.

Figure 6.8: ME Test 2: Chaser is sent from LVLH [5000,0,-200] m to [-5,0,0] km in 3000 s. It has to pass through hold point [200,0,0] at 1800 s and remain within approach cone of half-angle 10°.

# 7

# Numerical Analysis of ConGAL

This chapter explores the numerical capabilities of the Convex Guidance Algorithm by Liu (ConGAL) method and analyses the numerical results provided by Liu [2013]. It is found that ConGAL suffers from problems of convergence and accuracy. Some methods to improve its performance are demonstrated. In the process, the capabilities of solvers (ECOS, MOSEK) and intermediate modelling softwares (CVX, YALMIP) are also compared.

## 7.1 Initial guidance tests on ConGAL

This section focusses on exploring the capability of ConGAL for rendezvous guidance. The specific numerical examples shown by Liu [2013] are concerned with showcasing the ConGAL method's capability for handling specialized constraints on rendezvous flight. However, he does not establish ConGAL's capability against the solutions from Clohessy-Wiltshire equations for simple rendezvous problems. Within its scope of application, the Clohessy-Wiltshire equations give globally optimal solutions, thus providing a easy and reliable benchmark to compare ConGAL's solution. In this regard, two tests are considered and their results are presented below:

1. ConGAL Test 1: Transfer along V-bar using
   (a) Radial impulse transfer
   (b) Tangential impulse transfer
2. ConGAL Test 2: Station keeping on V-bar

These tests cover the scope of the capabilities we would like to test, namely generation of relevant guidance commands and relative orbit propagation. The constants and parameters that are used in the tests, are listed in Table 7.1. For brevity, only the results from MOSEK (without an intermediate modelling software) are discussed. However, the conclusions one can draw from the results of the other solvers and modelling software are found to be the same. Unless specified otherwise, the constraints that are active during the tests are: C0 (initial state); C1 (orbit propagation); C2 andC3 (for maximum thrust); and Cf (final state).

### 7.1.1 ConGAL Test 1a: Transfer along V-bar with radial burns

A test to replicate the radial impulse transfer along V-bar is considered - the chaser is kept at a point along the V-bar and within half-orbital period (*i.e.*, $T/2$ s), the chaser's position and velocity should match that of the target. The input parameters for the test are as shown in Table 7.2. As seen in Section 3.4.3, the expected guidance result is a two-burn manoeuvre with the thrust vector in the R-bar direction.

Table 7.1: Parameters used for ConGAL verification tests

| Parameter | Value | Units |
|---|---|---|
| Earth gravitational parameter | $3.986004415 \times 10^{14}$ | $m^3s^{-2}$ |
| Earth equatorial radius | 6378136 | m |
| Spacecraft initial mass | 500 | kg |
| Specific Impulse | 320 | s |

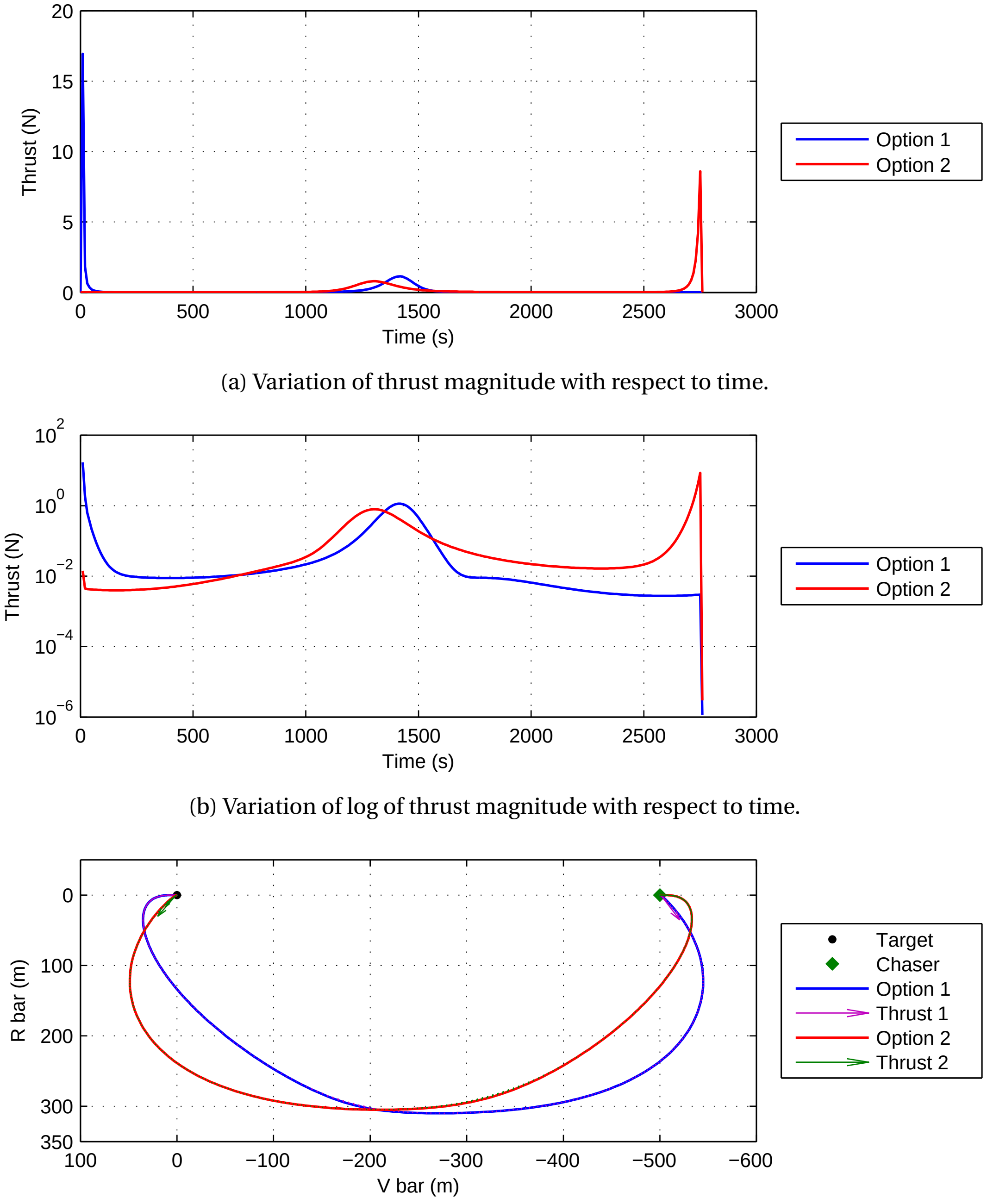
Table 7.2: Input to the ME

| Parameter | Value | Units |
|---|---|---|
| | Mission inputs | |
| Time of flight | 2760 | s |
| Step size | 10 | s |
| Convergence | [ 10 1 ] | [m,m/s] |
| | Target inputs | |
| Initial orbit | 370.00 × 370 | km |
| | Chaser inputs | |
| Initial LVLH position | [ −500 0 0 ] | m |
| Initial LVLH velocity | [ 0 0 0 ] | m/s |
| Final LVLH position | [ 0 0 0 ] | m |
| Final LVLH velocity | [ 0 0 0 ] | m/s |
| Maximum thrust | 500 | N |

The results of the ConGAl method, terminated at 30 iterations, are shown in Figure 7.1. Surprisingly, the method fails to converge to a solution but instead oscillates between two possible options (legend entries, Option 1 and Option 2). In Figure 7.1a, the thrust profile is plotted against time. We see that Option 1 has two major burns - the first burn (peaking at about 16 7N) is carried out immediately at the start of flight and a second burn (peaking at 1.1 N) near 1400 s (which is about $T/4$ s). In Figure 7.1b, which plots the thrust in logarithmic scale, we see that between the two burns and upto the end of flight for Option 1, the thrust never falls to zero but remains in the order of tens of millinewtons. This is a matter of concern, because over the time of flight, the combined effect of millinewton force can be substantial. Moving on to Option 2, we see that it too has two burns, however, the first major burn only occurs a little before 1400 s while the second burn at the very end of flight. Here as well, the thrust remains in the order of a few millinewtons between the two burns. Neither of these two options are similar to the expected R-bar burn, from the CW-equations. The $\Delta V$ of Option 1 is 0.851 m/s and Option 2 is 0.866 m/s, both of which are approximately 3 times more than the the analytical optimal of 0.285 m/s. Figure 7.1c shows the motion of the chaser spacecraft in the LVLH frame. Since the solution has not yet converged, the result cannot be sent to the `Orbit Propagator` for further verification.

Numerous similar tests, but with varying target altitude, initial points, step size and maximum thrust value were carried out as well. The conclusions which one could draw from them are as follows:

1. For a general problem to replicate the radial impulse transfer along V-bar, the ConGAL method fails to give a sensible result. The results oscillate indefinitely between two or more options and thus, do not converge within acceptable tolerance value (as defined in Table 7.2).



(a) Variation of thrust magnitude with respect to time.

(b) Variation of log of thrust magnitude with respect to time.

(c) Motion of chaser in LVLH frame.

Figure 7.1: ConGAL Test 1a: Chaser is sent from LVLH [-500,0,0] m to [0,0,0] km in 2760 s. The solution does not converge but oscillates indefinitely between Option 1 and 2.

2. Upon increasing the step size of the problem, the results are qualitatively similar. Below a step size of 5 s, the solver is unable to find an optimal solution to the problem within acceptable primal feasibility tolerance value (which is set to 1e-8, by default in MOSEK). Such behaviour is seen even when other solvers and intermediate modelling softwares are used. This could be seen as a limit to the capabilities of currently available primal-dual interior point method solvers, rather than a problem with the ConGAL method.

3. Upon varying the initial position along the V-bar, ConGAL method still doest not provide a

solution. In some cases, instead of oscillating between two options, the algorithm oscillates between more options.

4. Upon varying the target orbit altitude, the results are qualitatively similar.
5. As long as the maximum allowable thrust value is kept above 25 N, there is no effect on the solutions. Below 25 N, which corresponds to 0.005 m/s$^2$, an impulse burn is not possible for some of the options.
6. For most cases, the two major thrust burns do not occur at the beginning and end of flight, but are instead separated by about $T/4$ s (where $T$ is the orbital period of the target).

The result of ConGAL Test 1a is that ConGAL is unable to pass it.

### 7.1.2 CONGAL TEST 1B: TRANSFER ALONG V-BAR WITH TANGENTIAL BURNS

A test replicating tangential impulse transfer along V-bar is considered. The input parameters for the test are similar to the ones shown in Table 7.2, with the only difference being that the time of flight is equal to the orbital period, $T$. As seen in Section 3.4.3, the expected guidance result is a two-burn manoeuvre with the thrust vectors in the V-bar direction,

The results of the test, for the first 6 iterations, are shown in Figure 7.2. It is found that the ConGAL method fails to converge to a solution for this test as well. Instead of converging, the results diverge rapidly which causes the solver to stall and eventually crash. Figure 7.2a shows the thrust profile for the flight and Figure 7.2b shows the chaser motion in LVLH. It is clear that none of the iterations match the expected analytical solution from CW equations, even remotely. Iterations 7 and above (upto iteration 25, when MOSEK stalls), also do not show any correspondence with the expected solution but instead increasingly diverge. In fact, the solution diverges to the extent that at iteration 24, the difference in distance between consecutive solutions is of the order $10^7$ m.

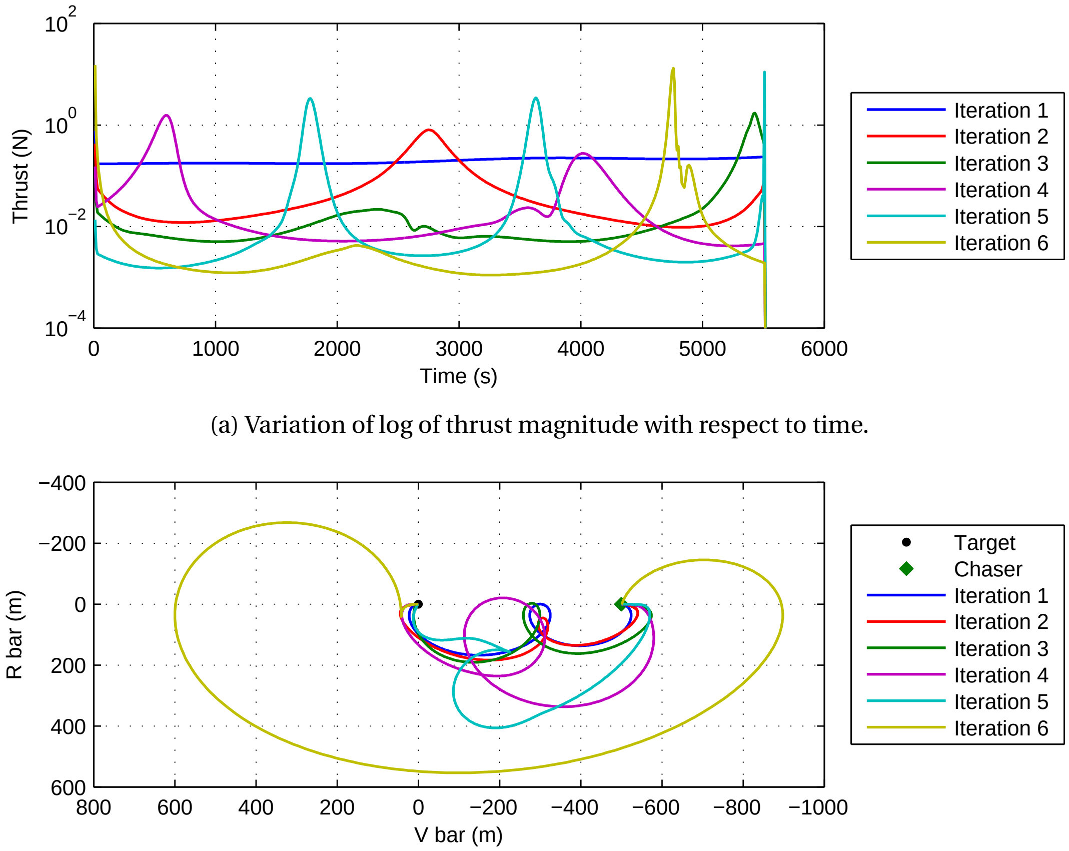
As a result, it is concluded that ConGAL method fails to pass ConGAL Test 1b as well.

### 7.1.3 CONGAL TEST 2: STATION KEEPING ON V-BAR

From the Clohessy-Wiltshire equations, it is known that if the chaser is at a short distance away from the target on the V-bar and has zero relative velocity, then the chaser's state shall remain unchanged in the LVLH frame. While this is only an approximation, it is very accurate over a small time period and if the chaser is not too far away from the target. If ConGAL's internal propagator is able to replicate such a behaviour, it shall pass this test. In addition, the state as propagated by the `Orbit Propagator` (OP) shall also be used as a reference. The inputs for the test are shown in Table 7.3. The constraint on final state (`Cf`) is no more active whereas the convergence tolerance for successive iterations is made tighter, to ensure that the solution is accurate.

We find that the ConGAL method converges within 2 iterations. As expected, ConGAL maintains the thrust profile to be zero for the length of the flight. The propagated state, as seen in the LVLH frame, is shown in Figure 7.3. For the given inputs, it is observed that the chaser drifts away substantially from its initial position (see legend entry, Run 1, h10). In 1000 s, the chaser has drifted nearly 90 m behind and about 3.5 cm above. While some drift is expected, as shown by the result from the OP (with legend entry, OP Run), the result from Run 1 is clearly not acceptable. To counter this, the problem is run again with reduced step sizes – 5, 2 and 1 s (with legend entries Run 2, h5; Run 3, h2; and Run 4, h1, respectively). We see a drastic reduction in the drift as the step size decreases. In Run 4, when the step size is 1 s, we see that the chaser drifts less than 1 m from its initial point, and thus matches reasonably well the with result from the OP Run. A further decrease in step size is not possible as it stalls the solver.

It is concluded that ConGAL passes Test 2, but the results are not fully satisfactory. In real applications, missions are much longer and thus, it might not be possible to have a step size as low as 1 s. When the time of flight is 1000 s, a step size of 1 s implies that the problem is discretized into



(a) Variation of log of thrust magnitude with respect to time.

(b) Motion of chaser in LVLH frame.

Figure 7.2: ConGAL Test 1b: Chaser is sent from LVLH [-500,0,0] m to [0,0,0] km in 5520 s. The solution diverges and thus, ConGAL fails the test.

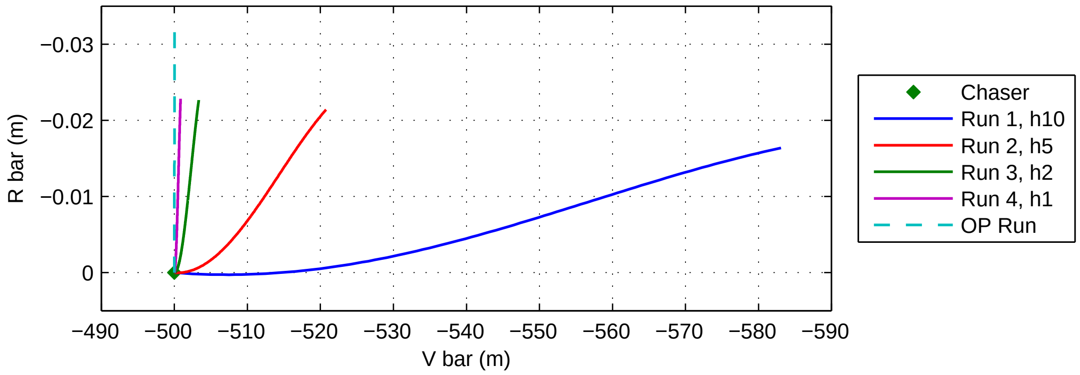


Figure 7.3: ConGAL Test 2: Chaser is allowed to drift from LVLH [-500,0,0] m for 1000 s. Results show that the step size affect the accuracy of the solution substantially.

1001 nodes. Experience with using the solver indicates that problems with more than 1200 nodes are likely to stall the solver. Thus, if the time of flight of a mission is 6000 s (which is approximately one orbital period in LEO), the lowest acceptable step size will be around 5 s. This draws attention to the limited accuracy of the the trapezoidal integrator used in ConGAL.

Table 7.3: Input to the ME

| Parameter | Value | Units |
|---|---|---|
| | Mission inputs | |
| Time of flight | 2000 | s |
| Step size | 10 | s |
| Convergence | $[1 \quad 0.1]$ | [m,m/s] |
| | Target inputs | |
| Initial orbit | 370.00 × 370 | km |
| | Chaser inputs | |
| Initial LVLH position | $[-100 \quad 0 \quad 0]$ | m |
| Initial LVLH velocity | $[0 \quad 0 \quad 0]$ | m/s |

## 7.2 ADDITIONAL GUIDANCE TESTS ON CONGAL

Seeing that ConGAL has not not passed ConGAL 1 (Sections 7.1.1 and 7.1.2), it is decided to carry out two additional tests to help isolate the cause of this problem and to determine ConGAL's capabilities. The following tests are considered:

1. ConGAL Test 3: Assisted transfer along V-bar
2. ConGAL Test 4: Transfer along V-bar in T/4 s.

### 7.2.1 CONGAL TEST 3: ASSISTED TRANSFER ALONG V-BAR

ConGAL's failure to pass Test 1 was communicated to Liu, who suggested that maybe ConGAL needs further assistance from the user to generate the required solution. In other words, additional constraints are to be enforced, such that the ConGAL method generates a solution that is similar to the solution provided by CW equations. Thus, we are assisting the ConGAL method to find the solution we need. It is possible that due to the non-linear nature of the problem, the successive approximations are unable to converge to the globally optimal solution.
In this regard, two new constraints are implemented:

1. C2a, constraint on varying magnitude of maximum thrust: This is a modification of C2, constraint on magnitude of maximum thrust. In C2, the magnitude of maximum thrust is kept constant throughout the flight. However, from the solutions of CW equations for transfer along V-bar, we know that the thruster is fired only at the initial and final instances of the flight. Thus, if the magnitude of maximum thrust is constrained to be 0 for the intermediate period, ConGAL could be forced to produce a thrust profile which is similar to the solution of CW equations and avoid the continuous force of few millinewtons we saw in Figure 7.1b. The equation for C2a is as follows:

$$0 \leq \sigma^{(k)}(t) \leq \widetilde{\mathbf{T}}_{\max} e^{-z^{(k-1)}(t)}\{1 - [z^{(k)}(t) - z^{(k-1)}(t)]\} \tag{7.1}$$

with

$$\widetilde{\mathbf{T}}_{\max} = \begin{cases} \mathbf{T}_{\max} & \text{if } t \leq 3h \\ 0 & \text{if } 4h \leq t \leq (N-3)h \\ \mathbf{T}_{\max} & \text{if } t \geq (N-2)h \end{cases} \tag{7.2}$$

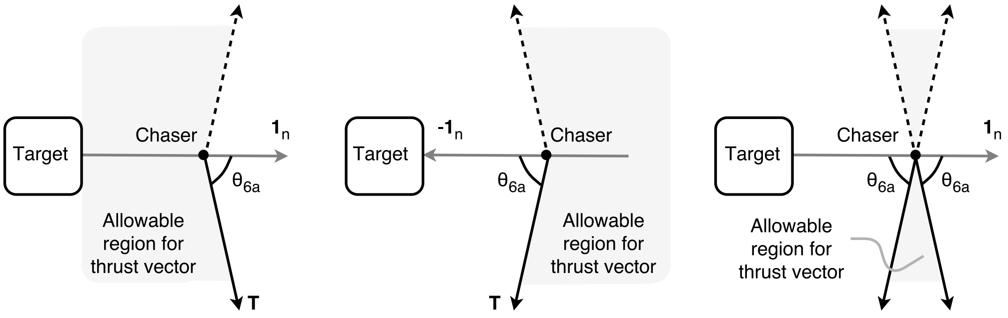
where, $N$ is the total number of nodes, $h$ is the (uniform) step size, and the remaining variables are as defined for constraint C2.

2. C6a, constraint on direction of thrust: This is a modification of C6, constraint on plume impingement. In Figure 7.4a, we see the general set up for the plume impingement constraint in radial direction. The thruster plumes are in a direction opposite to the thrust vector, hence, by constraining the thrust vector to the shaded region, one ensure little or no plumes are impinged on the target. This region can be be defined by $\theta_{6a}$, the minimum angle the thrust vector is allowed to make with respect to the docking axis vector, $\mathbf{1}_n$. In a similar manner, Figure 7.4b, shows a plume impingement constraint, but now the angle is defined with the negative docking axis vector. A combination of the two constraints gives rise to the narrow shaded region shown in Figure 7.4c. This is the basic definition of the constraint C6a. However, unlike the plume-impingement constraint, which is activated when the chaser is very near to the target, constraint C6a is activated in the beginning and in the end of flight. At each node, the constraint equations are as follows:

$$\mathbf{1}_n^\top \boldsymbol{\tau}^{(k)} \le \sigma^{(k)} \cos\theta_{6a} \tag{7.3}$$

$$-\mathbf{1}_n^\top \boldsymbol{\tau}^{(k)} \le \sigma^{(k)} \cos\theta_{6a} \tag{7.4}$$

where $\boldsymbol{\tau}^{(k)}$ is the thrust acceleration vector at iteration $k$ for that node. For constraining the thrust to some other direction, one can make use of a different value for $\theta_{6a}$ in each of the two Equations (7.3) and (7.4). Correspondingly, the orientation of the docking axis vector would also have to be modified so that the angles are measured appropriately.


Target
Chaser
1n
θ6a
Allowable region for thrust vector
T
Target
-1n
Chaser
θ6a
Allowable region for thrust vector
T
Target
Chaser
1n
θ6a
θ6a
Allowable region for thrust vector

(a) Allowable region as per constraint C6

(b) Allowable region as per constraint -C6

(c) Combined allowable region for constraint C6a

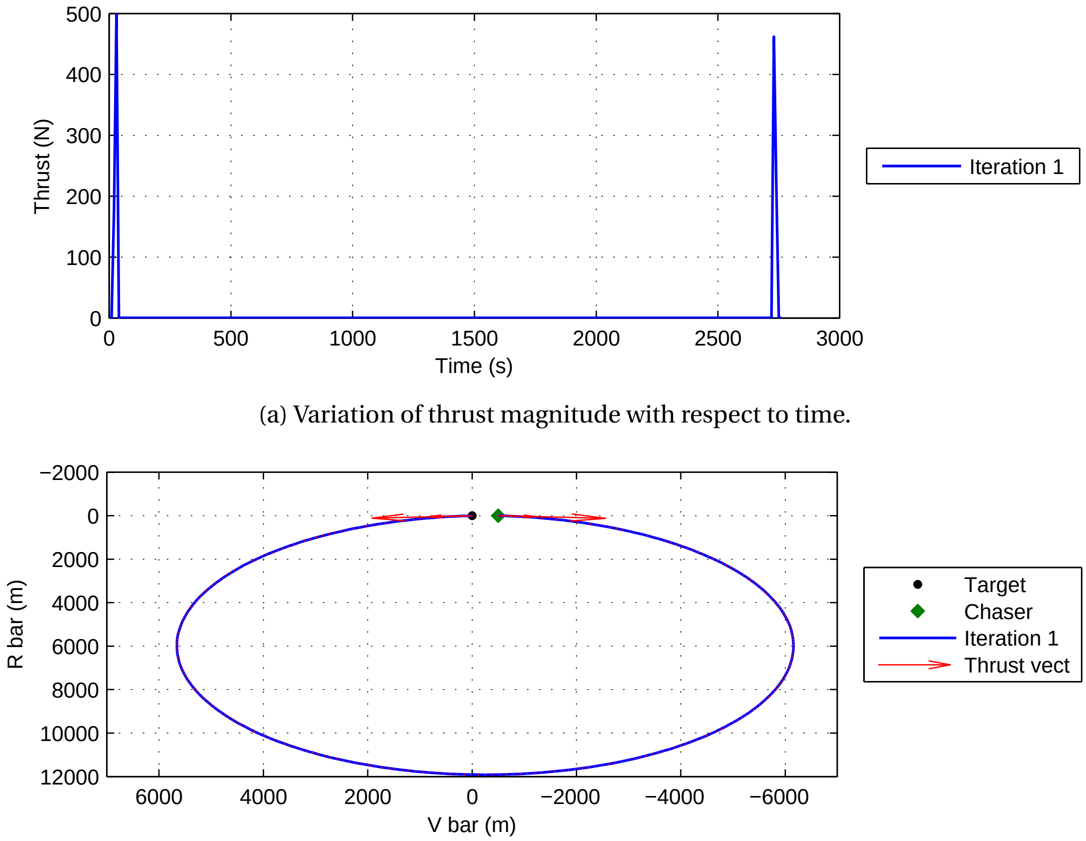
Figure 7.4: ConGAL Test 3 setup: Constraint C6a is a combination of two constraints C6. The figure on right constrains the thrust vector to be in radial direction in the LVLH frame.

Using C2a and C6a, it is now possible to re-run ConGAL Test 1a. The inputs to the test are as defined in Table 7.2. In addition, we define $\theta_{6a} = 89°$ in Equations (7.3) and (7.4), which restricts the thrust direction to within 1° of the ±R bar. The docking vector $\mathbf{1}_n$ is defined in the direction of $[\,-1 \quad 0 \quad 0\,]$ in the LVLH frame. The results are discussed as follows:

1. If only C2a is applied, the solver finds the problem to be infeasible in the second iteration. One is prevented from making any sensible conclusion from the obtained result as it is still the first iteration. The result is shown in Figure 7.5. As expected, the constraint C2a restricts the thrust burns to initial and final nodes.
2. If only C6a is applied, the problem oscillates between two options as seen in ConGAL Test 1a (Section 7.1.1). The problem is terminated at iteration 30 and the results are plotted in Figure 7.6. It is clear that due to constraint C6a, the thrust direction is radial in the LVLH frame.

3. If both C2a and C6a are applied, the solver finds the problem to be infeasible! Clearly, a solution exists, but for some reason ConGAL is unable to consider it as a valid solution. Varying other parameters like initial LVLH state, orbital altitude of target and step size, does not improve the situation.

The solutions are just as dismal for the tangential impulse transfer as well. It is concluded that the ConGAL method is unable to find solutions for radial and tangential transfers along V-bar, even when assisted in doing so. Thus, it does not pass ConGAL Test 3.



(a) Variation of thrust magnitude with respect to time.

(b) Motion of chaser in LVLH frame.

Figure 7.5: ConGAL Test 3 (with C2): Chaser is sent from LVLH [-500,0,0] m to [0,0,0] km in 2760 s. Solver finds the problem to be infeasible after iteration 1. Constraint C2 ensures thrust is available only at initial and final nodes.

### 7.2.2 CONGAL TEST 4: TRANSFER ALONG V-BAR IN $T/4$ S

Looking at the results from ConGAL Test 1, specifically Section 7.1.1 where the time between the two major burns was approximately $T/4$ s, it is decided to run the test for a time of flight of $T/4$ s (where $T$ is the orbital period of the target). This is not a standard rendezvous transfer like the radial and tangential impulse transfers along V-bar. However, its analytical solution can be found by solving the Clohessy-Wiltshire equations (see Section 3.4.3). For a V-bar transfer in $T/4$ s, we can already define a few parameters:

$$\begin{gathered} y_0, z_0 = 0; \qquad x(t_f), y(t_f), z(t_f) = 0 \\ t_f = \frac{T}{4} = \frac{\pi}{2n} \end{gathered} \tag{7.5}$$

Substituting the above in Equation (3.38), we can find the initial velocity required for commencing the transfer. Unlike the radial and tangential impulse transfers, this particular transfer is expected to have velocity components in both, R-bar and V-bar directions.

$$
\begin{aligned}
z(t_f) &= \left(\frac{2\dot{x}_0}{n} - 3\cdot 0\right)\cos(\frac{\pi}{2}) + \frac{\dot{z}_0}{n}\sin(\frac{\pi}{2}) + \left(4\cdot 0 - \frac{2\dot{x}_0}{n}\right)\\
0 &= \frac{\dot{z}_0}{n} - \frac{2\dot{x}_0}{n}\\
\Rightarrow \dot{z}_0 &= 2\dot{x}_0 && (7.6)\\
x(t_f) &= \left(\frac{4\dot{x}_0}{n} - 6\cdot 0\right)\sin(\frac{\pi}{2}) - \frac{2\cdot 0}{n}\cos(\frac{\pi}{2}) + (6n\cdot 0 - 3\dot{x}_0)\frac{\pi}{2n} + \left(x_0 + \frac{2\dot{z}_0}{n}\right)\\
0 &= x_0 + \left(\frac{4}{n} - \frac{3\pi}{2n}\right)\dot{x}_0 + \frac{2\dot{z}_0}{n}\\
\Rightarrow \dot{x}_0 &= \frac{2nx_0}{(3\pi - 16)} && (7.7)
\end{aligned}
$$

We see that the magnitude of required velocity depends only on the initial position of the chaser along V-bar, $x_0$ and the mean motion of the target, $n$. The direction of the velocity vector is always $-26.57°$ with the R-bar axis, independent of the initial parameters. As the flight progresses, and the chaser approaches the target at $t_f$, a second impulse will be required to cancel any remaining velocity. This can be calculated by substituting the above parameters in Equation (3.45). Upon simplification, we get:

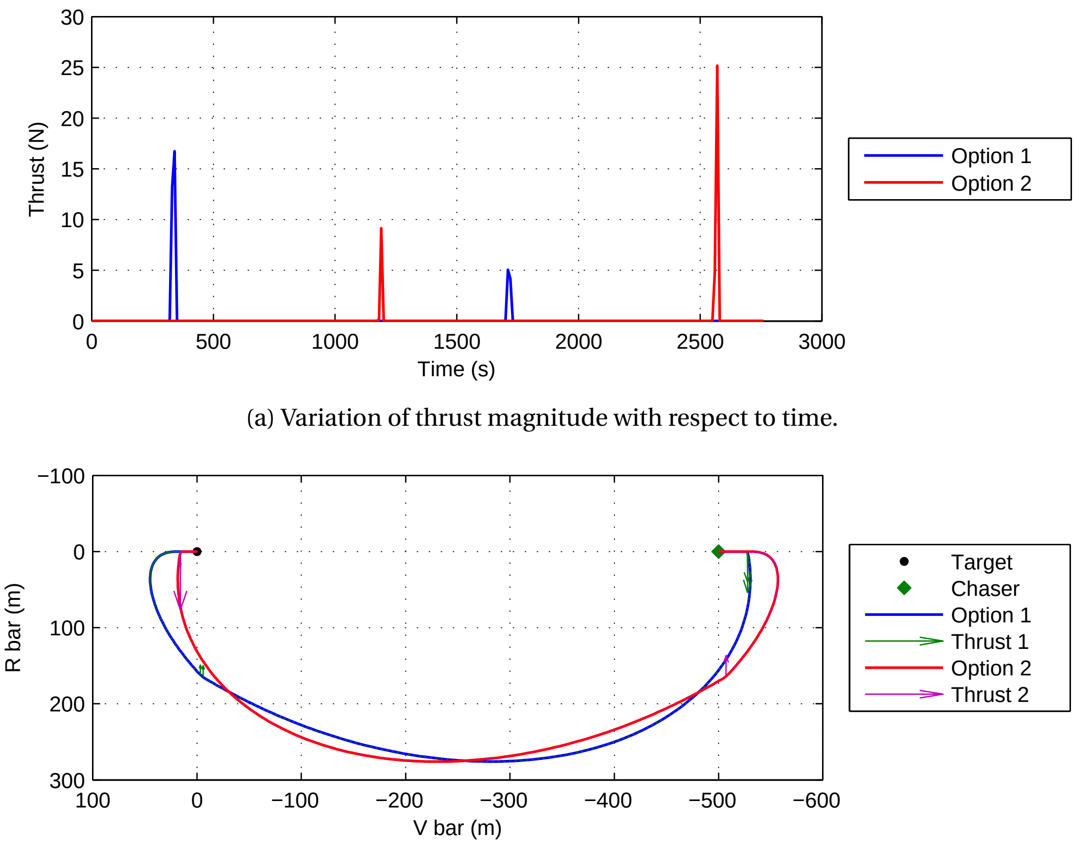
$$
\begin{aligned}
\dot{z}(t) &= 3n\sin(nt)z_0 + \cos(nt)\dot{z}_0 + 2\sin(nt)\dot{x}_0\\
\Rightarrow \dot{z}(t_f) &= 2\dot{x}_0 && (7.8)\\
\dot{x}(t) &= -6n(1-\cos(nt))z_0 - 2\sin(nt)\dot{z}_0 + (4\cos(nt) - 3)\dot{x}_0\\
\Rightarrow \dot{x}(t_f) &= -\dot{x}_0 && (7.9)
\end{aligned}
$$

This shows that second impulse has the same magnitude as the first one, but is in a different direction (making an angle 26.57° with the R-bar axis). The total $\Delta V$ for the transfer is given as:

$$
\begin{aligned}
\Delta V_{T/4} &= 2\sqrt{(\dot{x}_0)^2 + (\dot{z}_0)^2} && (7.10)\\
&= \left|\frac{4\sqrt{5}}{(3\pi - 16)}nx_0\right| && (7.11)\\
&\approx \left|1.3603nx_0\right| && (7.12)
\end{aligned}
$$

As was expected, this is much larger than the $\Delta V$ that is required for a radial impulse transfer ($\Delta V_{T/2} = \left|0.5nx_0\right|$), and tangential impulse transfer ($\Delta V_T \approx \left|0.1061nx_0\right|$). Now that we have the analytical expression, we can proceed with our test. The inputs for this test are same as the ones defined in Table 7.2, with the exception that the time of flight is now 2760 s. The required $\Delta V$, as per Equation (7.11) is 0.7746 m/s. The results from the test are shown in Figure 7.7 and are discussed below:

1. Run 1, step size = 10s: The first run (with legend entry, Run 1) is able to converge to a solution within 36 iterations. The thrust profile, as seen in Figure 7.7a, shows two near-impulsive burns at the beginning and end of flight. The chaser trajectory in LVLH frame is shown in Figure 7.7b. The thrust vector for the two burns (with legend entry, Thrust 1) are in the direction as expected for this transfer. The $\Delta V$ for the entire transfer is 0.9588 m/s, which is nearly 24% more than what is required. Feeding the results to the `Orbit Propagator`, shows a slight variation in the trajectory (with legend entry, OP Run 1). The difference between the final points is about 98.6 m. This variation in propagated state could be an explaination for the difference in the $\Delta V$ between ConGAL and the analytical solution. To check this, the problem was re-run with a lower step size, in Run 2.



(a) Variation of thrust magnitude with respect to time.

(b) Motion of chaser in LVLH frame.

Figure 7.6: ConGAL Test 3 (with C6a): Chaser is sent from LVLH [-500,0,0] m to [0,0,0] km in 2760 s. The solution does not converge but oscillates indefinitely between Option 1 and 2. Constraint C6a ensures thrust direction is radial in LVLH frame.

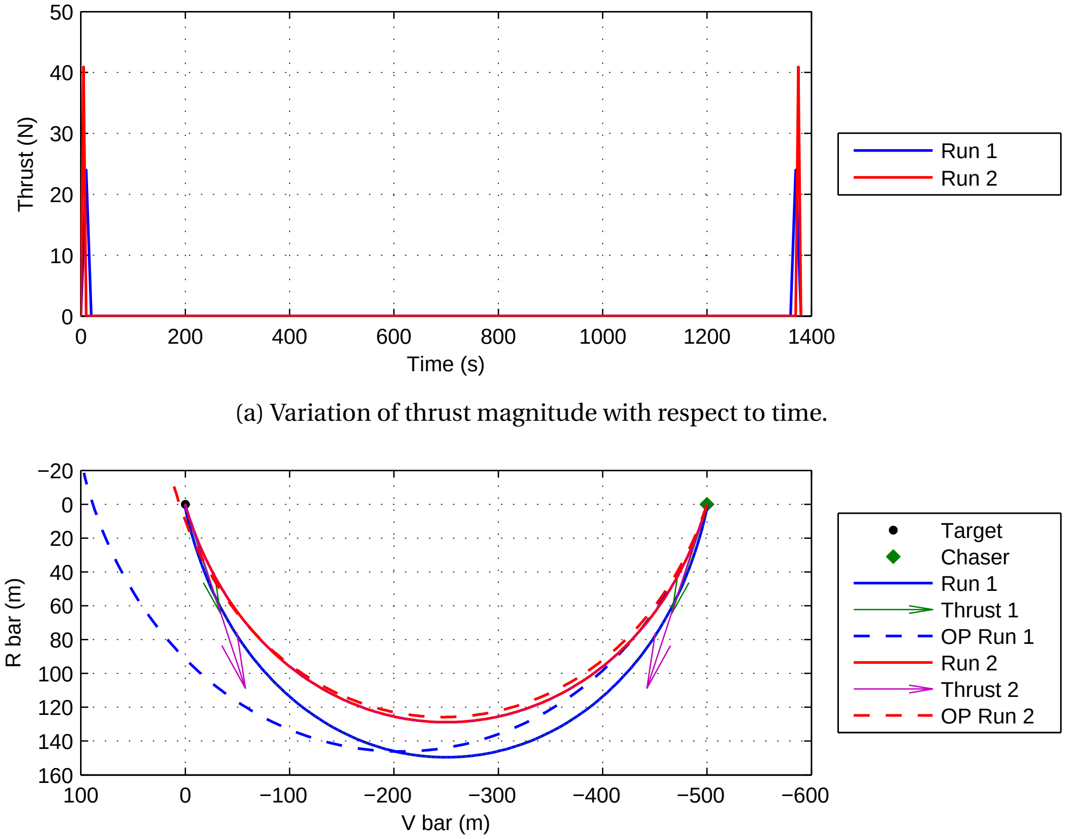
2. Run 2, step size = 5s: Its results are plotted in Figure 7.7 as well (with legend entry, Run 2). It takes 42 iterations to converge, and the results are more accurate - the $\Delta V$ for the transfer is 0.8165 m/s, while the difference between final points of the OP trajectory and ConGAL trajectory is less than 22 m. A further reduction in step size to 2s, further improves the solution - the $\Delta V$ is 0.7791 m/s and difference with OP is about 11m. It is not possible to reduce the step size any further as the solver stalls.

To conclude, the ConGAL method successfully passes Test 4. The solution is able to converge and match very well with the derived analytical value. A matter of concern is that ConGAL requires more than 45 iterations to converge. This is time consuming and can prohibit its application to cases where a quicker solution is required.

This brings to end the test campaign of ConGAL's capability for rendezvous transfer. Out of the four tests, ConGAL is able to pass two of them. The results are satisfactory but one needs to use small step sizes for moderately accurate results as well. In the failed tests, ConGAL has issues with convergence, raising questions about the reliability of Method of Successive Approach (discussed in Section 5.3).

## 7.3 METHODS TO IMPROVE CONGAL'S PERFORMANCE

The two major problems faced by ConGAL are with respect to convergence and accuracy of propagated state. It is also possible that some issues arise from the limitations of currently available



(a) Variation of thrust magnitude with respect to time.

(b) Motion of chaser in LVLH frame.

Figure 7.7: ConGAL Test 4: Chaser is sent from LVLH [-500,0,0] m to [0,0,0] km in 1380 s. The solution converges, but takes up more does not converge but oscillates indefinitely between Option 1 and 2. Constraint C6a ensures thrust direction is radial in LVLH frame.

solvers. Of these, the most importantly concern is the tendency of solvers to stall as the problem size increases. To combat these problems, two methods / modifications are developed for use with the ConGAL method and one re-formulation of constraint is suggested.

### 7.3.1 MEAN-SCHEME

The mean-scheme is a method to improve the rate of convergence of the ConGAL method. Despite the claim by Lu and Liu [2013], the ConGAL method does not show "convergence in just a few iterations". In fact, barring ConGAL Test 2, the converged solutions discussed in this chapter and the previous one (Chapter 6) required nearly 10, and sometimes over 40 iterations to converge. This is also in sharp contrast to the specific examples provided by Liu [2013] and Lu and Liu [2013], which almost always converged in less than 10 iterations (a possible reason for this contrast is further discussed in section 7.4).

The ConGAL method employs a method of successive approximations (as discussed in Section 5.3), which allows iterations to take updated values of gravitational acceleration and spacecraft mass, from their immediately previous iteration. Since all other input parameters remain constant between the successive iterations, the difference in solution of the next iteration is determined by these updated values alone. This update in values can make the problem to converge (as seen in ConGAL Test 4), diverge (as seen in ConGAL Test 1b), or fall into an indefinite loop (as seen in ConGAL Test 1a).

Often the trajectory of the upcoming iteration, due to method of successive approximation, lies in the vicinity of the trajectory of the current iteration. So, if a problem falls into a repeated loop of iterations between two (or more) options, then the trajectories of upcoming iterations will all be similar to any one of the current dead-end options. This restrict them to the specific region as seen in LVLH frame, and proves to be an obstacle, as updates to gravitational acceleration values are not fresh any more. The expected converged solution (if it exists) will be in the middle of these extreme trajectories, but since there are no new values being provided for gravitational acceleration, the deadlock of iterations does not break. Mean-scheme breaks away from this duopoly of options by taking the mean of the solution of these repeating iterations, and feeding it as a new iteration into the method of successive approximation. This update in values forces a different set of solution to occur in the next iteration, and has the potential to bring about a convergence.

The key challenge of mean-scheme is determining when and how to implement it. Obviously, such a method can be easily implemented in hindsight, after a problem has failed to converge, by analysing each of the trajectories of the various iterations. Then the problem can be re-solved by applying the mean-scheme at the appropriate iteration. In real applications, this would not be an attractive method at all, because the spacecraft would have to spend valuable time awaiting a slow or a failed convergence attempt, following which the problem would be re-solved. The focus, therefore, should be on a strategy that can decide impromptu whether or not the mean-scheme should be applied to the iteration for better convergence rates.

The foremost idea would be to compare the original decision variable vector (i.e., $\mathbf{W}^{(k)}$) of current iteration with previous iterations while the problem is running. If the difference in state between a set of non-consecutive trajectories is close to the some pre-determined value, it may call upon the mean-scheme. However, comparing solutions of all iterations with each other is time consuming as it involves comparing a combination of tens of vectors, each consisting of many hundreds of elements. The results may not be so clear as well, as the difference between the closest trajectories might still be pretty large as compared to the pre-determined value. A faster and more reliable alternative way is to compare objective values of each iteration. It is observed that iterations with similar objective values often have similar trajectories (even if some of the values in the solution might be preventing the convergence criterion to be met). Comparing objective values is also computationally cheaper, as each iteration has only one objective value. The algorithm for calling and implementing the mean scheme is as follows:

1. Mean-scheme pre-check: The mean-scheme can be allowed only when the iteration number, $k$ falls in the open interval $(3, k_{\text{max}} - 1)$, and the convergence criteria is not yet met.
2. Mean-scheme condition: Check if objective value of iteration $k$, is within $\epsilon_{\text{ms}}$% of iteration $(k - j)$, where $j \in \{2, 3, \ldots, j_{\text{max}}\}$. At the first $j$ that satisfies the condition, proceed to step 3.
3. Mean-scheme begin: The solution of iteration $k+1$ is calculated:

$$\mathbf{y}^{(k+1)} = \frac{\mathbf{y}^{(k)} + \mathbf{y}^{(k-j+1)}}{2} \tag{7.13}$$

4. Update $k = k+1$. End of mean-scheme.

Here, $k_{\text{max}}$ is the maximum allowable iterations for method of successive approximation, $\epsilon_{\text{ms}}$ is the mean-scheme tolerance for objective values, and $j_{\text{max}}$ is the maximum number of iterations that will be used for comparision. By experience, it is found $\epsilon_{\text{ms}} = 1$, is sufficient to allow the mean-scheme to work well. For most problems, the repeating loop oscillates between two options, thus, when $j = 2$ the mean-scheme is called. (In some problems, it can be upto 4 or 5, so setting $j_{\text{max}} = 6$ is more than sufficient). Step 3 calculates the solution for the new iteration, although it is not explicitly solved by ConGAL. It is a mean of previous solutions, and thus, only a pseudo-iteration of ConGAL. After the call, the newly calculated solution is used as the input for the the next real iteration.

The results of the numerical implementation of the mean-scheme are illustrated in Figure 7.8 and discussed below:

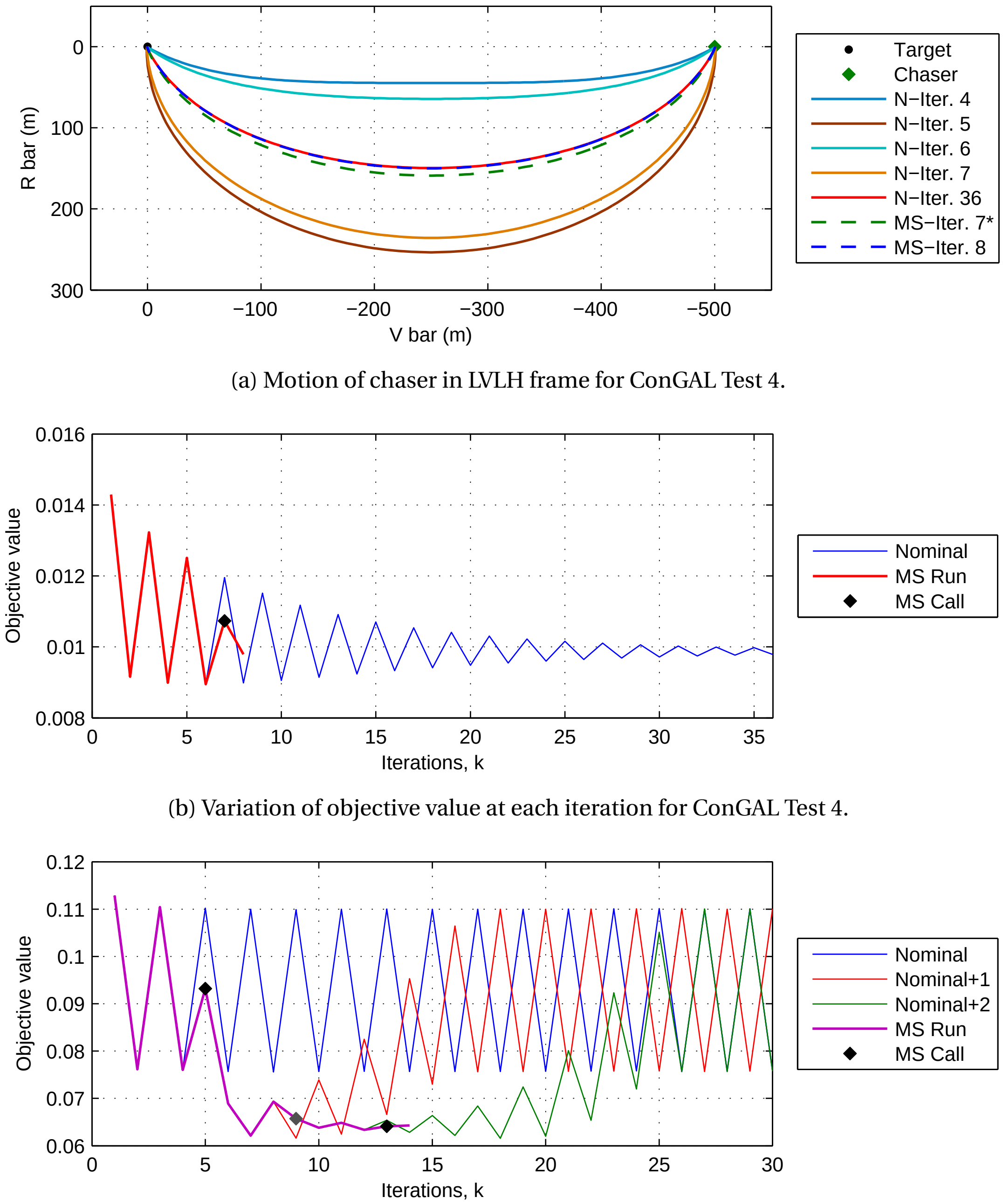
1. In ConGAL Test 4 (discussed in Section 7.2.2), the first run (with step size 10 s) took 36 iterations to converge. Upon implementing the mean-scheme, it is found that the problem converges in 8 iterations! Figure 7.8a shows the chaser state in LVLH frame for different iterations of the nominal run (with legend entries, N-Iter. X, where X is iteration number) and a run with the mean-scheme applied (with legend entries, MS-Iter. X). Nominal iterations 4 and 5 are two distinct options of the slow converging loop that ConGAL falls into. The trajectories of upcoming iterations, fall in the vicinity of either of these two options. For instance, N-Iter. 6 is close to N-Iter. 4, and so would be iterations 8, 10, etc. (which are not shown to avoid cluttering). The same can be said for odd numbered iterations with N-Iter. 5. The upcoming iterations slowly move towards each other and finally converge at N-Iter. 36. When the mean-scheme is available for use, iteration 6 qualifies the conditions because its objective value is within 1 % of the iteration 4 (as seen in 7.8b), even though the maximum difference between their trajectories is substantially large as compared to convergence criterion. The mean-scheme generates iteration 7, a pseudo-iteration, (with legend entry MS-Iter. 7*, where the * represents that it is not explicitly calculated by ConGAL). Using MS-Iter. 7*, ConGAL now proceeds to generate MS-Iter. 8, which passes the convergences criteria and is a perfect match with N-Iter. 36. Figure 7.8b shows the variation of objective value with respect to iterations. The nominal run of the problem (with legend entry, Nominal) is in a slowly converging oscillation that appears to settle just below 0.01 at iteration 36. The mean-scheme run (with legend entry, MS Run) follows the Nominal run, until iteration 6, after which the mean-scheme is called, which changes the flow of the iterations.

2. A peculiar case is the one mentioned in ME Test 2 (in section 6.5.4). Without the mean-scheme, it is unlikely that the problem would converge as seen by the variation of objective value in Figure 7.8c (with legend entry, Nominal). The problem appears to be in an unending deadlock between two possible outcomes. To make the problem converge, the mean-scheme has to be applied thrice, which generates solutions at pseudo-iterations 5, 9 and 13. The variation in objective value of the converged solution is shown by the plot with legend entry MS Run. If the second and third call for the mean-scheme are ignored, the problem falls back into a similar rhythm as was seen by the Nominal run (plots with legend entries Nominal+1 and Nominal+2).

The above examples show the benefit and capability of applying the mean-scheme: ConGAL Test 4 converges in 8 iterations (with 1 mean-scheme call) as compared to 36 iterations nominally; ME Test 2 converges in 14 iterations (with 3 mean-scheme calls) as compared to a non-converging deadlock. However, not all problems are suitable for mean-scheme application: ConGAL Test 1a (section 7.1.1) fails to converge even with the mean-scheme. Further, if a problem is naturally converging at a fast pace, for example, ME Test 1 (in section 6.5.4), then the mean-scheme hinders the progress (9 iterations nominally, as compared to 12 with the mean-scheme). Although, it is possible that these problems could be solved if other, better parameters are used for defining the mean-scheme. Nevertheless, the mean-scheme is definitely not applicable for problems where the expected converged trajectory is not in between trajectories of iterations that lead to it. This is a drawback of the suggested scheme, and needs to be overcome.

### 7.3.2 CONGAL INTEGRATOR

As discussed in Chapter 5, the ConGAL method generates the control vector and simultaneously propagates the state of the spacecraft. The state equations (Equation 5.36) are included as a system of linear constraints (constraint C1) in the optimization problem. Thus, the choice of integrator



(a) Motion of chaser in LVLH frame for ConGAL Test 4.

(b) Variation of objective value at each iteration for ConGAL Test 4.

(c) Variation of objective value at each iteration for ME Test 2.

Figure 7.8: Mean-scheme: Graphs showing results of nominal runs compared to results with mean-scheme.

not only affects the accuracy of the propagated state, but also influences the control vector solution provided by the solver. ConGAL employs the trapezoidal rule, a second-order, implicit method of integration. It is a simple, linear implementation that is easily accommodated within the ConGAL framework. In the numerical examples provided by Liu [2013] and Lu and Liu [2013], a step size of 10 s is considered as acceptable. However, as seen in ConGAL Test 2 (section 7.1.3) and ConGAL Test 4 (section 7.2.2), a step size of 10 s is not enough for accurate results. Reducing the step size is not an alternative for all cases because solvers have a tendency to stall (or crash) as the problem size increases beyond a certain limit. With these thoughts in mind, it is considered beneficial if a higher order integrator is made available for use with ConGAL.

Consider a system of first-order, ordinary differential equations:

$$\frac{d\mathbf{y}}{dt} = \dot{\mathbf{y}}(t) = f(t,\mathbf{y}), \quad \mathbf{y}(t_0) = y_0 \tag{7.14}$$

The numerical integration of the above equation looks to find an approximate solution at discrete points in time

$$\mathbf{y}(t) \approx \mathbf{y}_n = \int_{t_0}^{t_0+nh} f(t,\mathbf{y})\, dt; \quad t \in [t_0, t_0 + Nh], \; n = 1,2,3,\dots,N \tag{7.15}$$

where $h$ is the step-size, $N$ is the number of intervals. It constructs an array of dependent variables $\mathbf{y}$, in a stepwise process from values of the function evaluated at discrete values of the independent variable $t$. For ConGAL, the state equation is given by Equation 5.36. It is re-written in a simplified form as follows:

$$\begin{aligned} f(t_n,\mathbf{y}_n) &= \dot{\mathbf{y}}_n^{(k)} = \mathbf{A}(r^{(k-1)}(t_n))\mathbf{y}_n^{(k)} + \mathbf{B}\mathbf{u}_n^{(k)} \\ f_n &= \dot{\mathbf{y}}_n = \mathbf{A}_n\mathbf{y}_n + \mathbf{B}\mathbf{u}_n \end{aligned} \tag{7.16}$$

where time $t_n = t_0 + nh$ and $k$ is iteration of ConGAL. It is to be noted that the independent variable $t_n$ does not explicitly appear in the differential equation. Further, there are two dependent variables $\mathbf{y}_n$ and $\mathbf{u}_n$ that affect the derivative at a given state. Thus this is a multivariable, autonomous ordinary differential equation. However, we expect the control values to be constant between individual nodes of the discretized problem. This approximates $\mathbf{u}(t)$ to be piecewise constant and is a commonly used assumption for control problems. This allows us to carry out the integration of Equation (7.16) as if it were a uni-variable, autonomous ordinary differential equation.

The key question now is, *which integrator best serves the purpose?* To have a substantial improvement over the trapezoidal rule, it is decided to look for fourth-order integrators. A stringent requirement for ConGAL is that the discretized system of state equations, as set up by the integrator, should be linear. These linear relations get passed on to the optimization problem as a system of linear equality constraint (as seen while setting up constraint C1). Further, an integrator with adaptive step size is not acceptable because the optimization problem needs its constraints to be set up before calling the solver, and not during the solver run. Amongst the explicit methods, fourth-order Runge-Kutta and Adams-Bashforth are attractive options. For implicit methods, Adams-Moulton method of stage three and Simpsons rule are considered. Implicit Runge-Kutta method is discarded because of the substantially larger computational cost as compared to the two methods considered [Süli and Mayers, 2003]. The following list provides the necessary background for each of the selected methods, as presented by Montenbruck and Gill [2000], and [Süli and Mayers, 2003]. More importantly, the linear formulations suitable for use with ConGAL are set up as well.

1. **Fourth-order Runge-Kutta method**:
   A commonly encountered integrator is the fourth-order Runge-Kutta, often termed as *the* Runge-Kutta (RK) method. It has 4 stages and its equations are set up as follows [Montenbruck and Gill, 2000]:

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \frac{h}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4) \tag{7.17}$$

   where,

$$\begin{aligned} \mathbf{k}_1 &= f(t_n,\mathbf{y}_n), \\ \mathbf{k}_2 &= f(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{h}{2}\mathbf{k}_1), \\ \mathbf{k}_3 &= f(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{h}{2}\mathbf{k}_2), \\ \mathbf{k}_4 &= f(t_n + h, \mathbf{y}_n + h\mathbf{k}_3) \end{aligned} \tag{7.18}$$

It is a single step method that evaluates expressions $\mathbf{k}_i$ at various intermediate points within the integration step, and uses their weighted mean to estimate the value at next step. Each step requires four evaluations of the derivative function, thus it is considered moderately expensive. In its current form, the above equations are not suitable for use with ConGAL. They have to be restructured and expressed as a linear constraint. We start by introducing Equation (7.16) into the expressions for $\mathbf{k}_i$ and expanding them:

$$\mathbf{k}_1 = \mathbf{A}_n\mathbf{y}_n + \mathbf{B}\mathbf{u}_n, \tag{7.19}$$

$$\begin{aligned}
\mathbf{k}_2 &= \mathbf{A}_n\left(\mathbf{y}_n + \frac{h}{2}\mathbf{k}_1\right) + \mathbf{B}\mathbf{u}_n \\
&= \mathbf{A}_n\mathbf{y}_n + \frac{h}{2}\mathbf{A}_n(\mathbf{A}_n\mathbf{y}_n + \mathbf{B}\mathbf{u}_n) + \mathbf{B}\mathbf{u}_n \\
&= \left[\mathbf{A}_n + \frac{h}{2}\mathbf{A}_n^2\right]\mathbf{y}_n + \left[\mathbf{B} + \frac{h}{2}\mathbf{A}_n\mathbf{B}\right]\mathbf{u}_n,
\end{aligned} \tag{7.20}$$

$$\begin{aligned}
\mathbf{k}_3 &= \mathbf{A}_n\left(\mathbf{y}_n + \frac{h}{2}\mathbf{k}_2\right) + \mathbf{B}\mathbf{u}_n \\
&= \mathbf{A}_n\mathbf{y}_n + \frac{h}{2}\mathbf{A}_n\left(\left[\mathbf{A}_n + \frac{h}{2}\mathbf{A}_n^2\right]\mathbf{y}_n + \left[\mathbf{B} + \frac{h}{2}\mathbf{A}_n\mathbf{B}\right]\mathbf{u}_n\right) + \mathbf{B}\mathbf{u}_n \\
&= \left[\mathbf{A}_n + \frac{h}{2}\mathbf{A}_n^2 + \frac{h^2}{4}\mathbf{A}_n^3\right]\mathbf{y}_n + \left[\mathbf{B} + \frac{h}{2}\mathbf{A}_n\mathbf{B} + \frac{h^2}{4}\mathbf{A}_n^2\mathbf{B}\right]\mathbf{u}_n,
\end{aligned} \tag{7.21}$$

$$\begin{aligned}
\mathbf{k}_4 &= \mathbf{A}_n\left(\mathbf{y}_n + h\mathbf{k}_3\right) + \mathbf{B}\mathbf{u}_n \\
&= \mathbf{A}_n\mathbf{y}_n + \frac{h}{2}\mathbf{A}_n\left(\left[\mathbf{A}_n + \frac{h}{2}\mathbf{A}_n^2 + \frac{h^2}{4}\mathbf{A}_n^3\right]\mathbf{y}_n + \left[\mathbf{B} + \frac{h}{2}\mathbf{A}_n\mathbf{B} + \frac{h^2}{4}\mathbf{A}_n^2\mathbf{B}\right]\mathbf{u}_n\right) + \mathbf{B}\mathbf{u}_n \\
&= \left[\mathbf{A}_n + h\mathbf{A}_n^2 + \frac{h^2}{2}\mathbf{A}_n^3 + \frac{h^3}{4}\mathbf{A}_n^4\right]\mathbf{y}_n + \left[\mathbf{B} + h\mathbf{A}_n\mathbf{B} + \frac{h^2}{2}\mathbf{A}_n^2\mathbf{B} + \frac{h^3}{4}\mathbf{A}_n^3\mathbf{B}\right]\mathbf{u}_n
\end{aligned} \tag{7.22}$$

Gathering the above equations into Equation (7.17), we see:

$$\mathbf{y}_{n+1} = \left[\mathbf{I} + h\mathbf{A}_n + \frac{h^2}{2}\mathbf{A}_n^2 + \frac{h^3}{6}\mathbf{A}_n^3 + \frac{h^4}{4}\mathbf{A}_n^4\right]\mathbf{y}_n + \left[h\mathbf{B} + \frac{h^2}{2}\mathbf{A}_n\mathbf{B} + \frac{h^3}{6}\mathbf{A}_n^2\mathbf{B} + \frac{h^4}{24}\mathbf{A}_n^3\mathbf{B}\right]\mathbf{u}_n \tag{7.23}$$

where $\mathbf{I}$ is an identity matrix. A new variable, $\mathbf{K}_n$, is introduced:

$$\mathbf{K}_n = \mathbf{I} + \frac{h}{2}\mathbf{A}_n + \frac{h^2}{6}\mathbf{A^2}{}_n + \frac{h^3}{24}\mathbf{A}_n^3 \tag{7.24}$$

Equation (7.23) now becomes:

$$\mathbf{y}_{n+1} = \left[\mathbf{I} + h\mathbf{K}_n\mathbf{A}_n\right]\mathbf{y}_n + \left[h\mathbf{K}_n\mathbf{B}\right]\mathbf{u}_n \tag{7.25}$$

This linear equation is a new, elegant formulation of RK4 and can be easily passed on to ConGAL. The state equations constraint C1, at each node, is now written as:

$$-\mathbf{y}_{n+1} + \left[\mathbf{I} + h\mathbf{K}_n\mathbf{A}_n\right]\mathbf{y}_n + \left[h\mathbf{K}_n\mathbf{B}\right]\mathbf{u}_n = 0 \tag{7.26}$$

2. **Adams-Bashforth method of stage 4**:
The Adams-Bashforth method is a linear, multi-step integrator which takes advantage of previous steps by interpolating a polynomial through them and extrapolating it to predict the

value at the next step. A major advantage over Runga-Kutta methods of similar order is that it require fewer evaluations of the derivative function. However, they require a single step integrator to start-up and any initial errors are more likely to propagate through the problem.

An Adams-Bashforth (AB) method of stage size $s$, has an accuracy of order $s$. Thus, a fourth order AB method requires 4 stages. A commonly used formulation is [Montenbruck and Gill, 2000]:

$$\mathbf{y}_{n+4} = \mathbf{y}_{n+3} + \frac{h}{24}\left(55 f_{n+3} - 59 f_{n+2} + 37 f_{n+1} - 9 f_n\right) \tag{7.27}$$

By substituting Equation (7.16) and re-arranging the terms, we have constraint C1 as:

$$\begin{aligned} -\mathbf{y}_{n+4} + \left(\frac{55}{24} h\mathbf{A}_{n+3} + \mathbf{I}\right)\mathbf{y}_{n+3} - \frac{59}{24} h\mathbf{A}_{n+2}\mathbf{y}_{n+2} + \frac{37}{24} h\mathbf{A}_{n+1}\mathbf{y}_{n+1} - \frac{9}{24} h\mathbf{A}_{n}\mathbf{y}_{n} + \\ \frac{55}{24} h\mathbf{B}\mathbf{u}_{n+3} - \frac{59}{24} h\mathbf{B}\mathbf{u}_{n+2} + \frac{37}{24} h\mathbf{B}\mathbf{u}_{n+1} - \frac{9}{24} h\mathbf{B}\mathbf{u}_{n} = 0 \end{aligned} \tag{7.28}$$

3. **Adams-Moulton method of stage 3**:
   The Adams-Moulton (AM) method is an implicit, linear multistep method that is an extension on the Adams-Bashforth method. A stage $s$ integrator has an accuracy of $s+1$. The fourth order formulation is as follows [Montenbruck and Gill, 2000]:

   $$\mathbf{y}_{n+3} = \mathbf{y}_{n+2} + \frac{h}{24}\left(9 f_{n+3} + 19 f_{n+2} - 5 f_{n+1} + 1 f_n\right) \tag{7.29}$$

   This translates into constraint C1 as follows:

   $$\begin{aligned} \left(\frac{9}{24} h\mathbf{A}_{n+3} - \mathbf{I}\right)\mathbf{y}_{n+3} + \left(\frac{19}{24} h\mathbf{A}_{n+3} + \mathbf{I}\right)\mathbf{y}_{n+2} - \frac{5}{24} h\mathbf{A}_{n+1}\mathbf{y}_{n+1} + \frac{1}{24} h\mathbf{A}_{n}\mathbf{y}_{n} + \\ \frac{9}{24} h\mathbf{B}\mathbf{u}_{n+3} + \frac{19}{24} h\mathbf{B}\mathbf{u}_{n+2} - \frac{5}{24} h\mathbf{B}\mathbf{u}_{n+1} + \frac{1}{24} h\mathbf{B}\mathbf{u}_{n} = 0 \end{aligned} \tag{7.30}$$

4. **Simpson's rule**:
   Simpson's rule (SR), like the trapezoidal rule, is a member of the Newton-Cotes family of formulae used for numerical integration. It is an implicit, linear two-step method with order of accuracy four. Like all other multi-step methods, it too requires a single-step integrator to start up. Compared to the Adams family of integrators, it requires marginally fewer evaluation of the derivative function. Its general form is Süli and Mayers [2003]:

   $$\mathbf{y}_{n+2} = \mathbf{y}_{n} + \frac{h}{3}\left(f_{n+2} + 4 f_{n+1} + f_n\right) \tag{7.31}$$

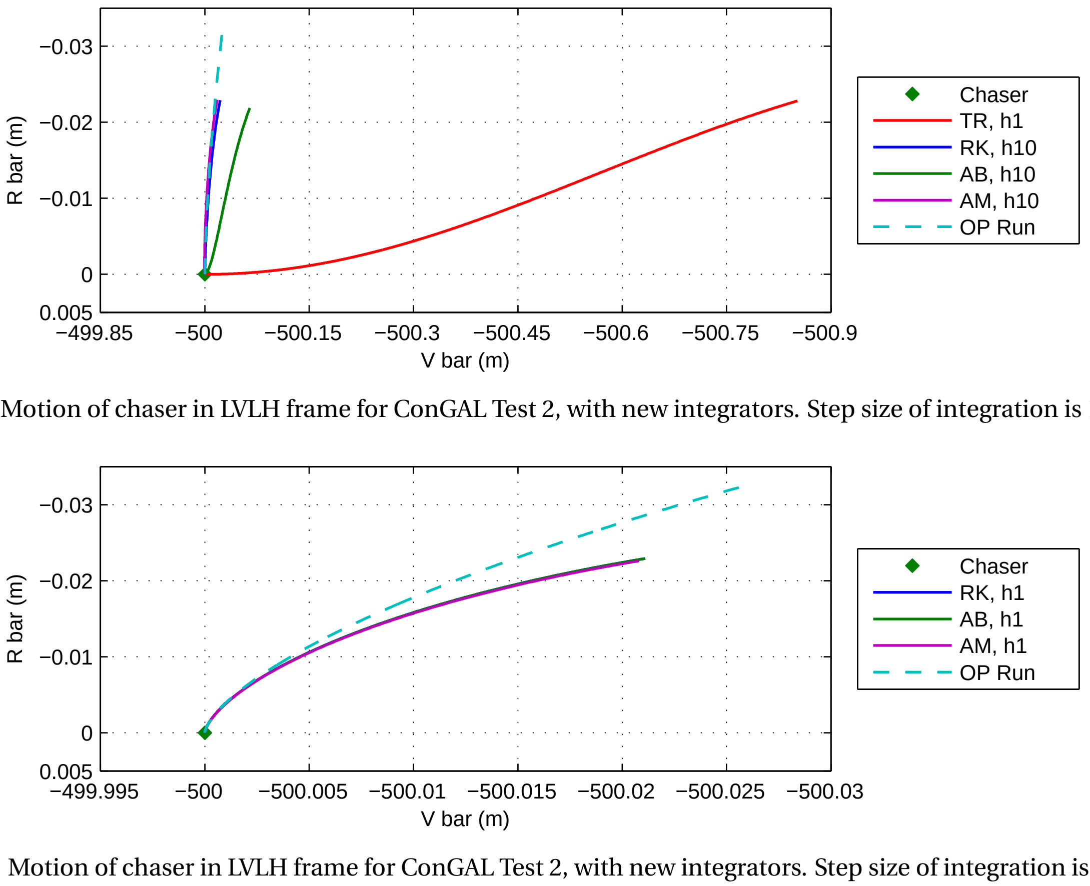
   The constraint C1 can be written as:

   $$\left(\frac{1}{3} h\mathbf{A}_{n+2} - \mathbf{I}\right)\mathbf{y}_{n+2} + \frac{4}{3} h\mathbf{A}_{n+1} + \left(\frac{1}{3} h\mathbf{A}_{n} + \mathbf{I}\right)\mathbf{y}_{n} + \frac{1}{3} h\mathbf{B}\mathbf{u}_{n+2} + \frac{4}{3} h\mathbf{B}\mathbf{u}_{n+1} + \frac{1}{3} h\mathbf{B}\mathbf{u}_{n} = 0 \tag{7.32}$$

The above integrators are put to test by re-solving ConGAL Test 2, and ConGAL Test 4. For the multi-step integrators, RK is used to start them up. The results are discussed as follows:

1. ConGAL Test 2: This test checks station-keeping on V-bar. The time of flight is 1000 s, and chaser is initially at rest, 500 m behind the target. For further details on the test setup refer Section 7.1.3. The results are as follows:

- For a step size of 10 s: Integrators RK, AB and AM are able to converge within 2 iterations. Integration by SR also converges, but the trajectory it produces has large oscillations and is thus ignored for the time-being. Figure 7.9a compares the trajectories from the new integrators with that of the OP. The three integrators are nearly match the results of the OP, although AB is a little off. Also shown in the figure is the results form the Trapezoidal rule (TR), but of step size 1 s. The improvement in accuracy is evident as the fourth-order integrators outperform TR, despite having a step size that is 10 times as big.
- For a step size of 1 s: Again, the integrators RK4, AB and AM are able to converge within 2 iterations. The run with SR stalls and thus, its results are not available. Figure 7.9b compares the trajectories from the new integrators with that of the `Orbit Propagator`. The result of the integrators overlap, and it is not possible to discern which integrator is better. The resulting trajectory is very close to the result from the OP.



(a) Motion of chaser in LVLH frame for ConGAL Test 2, with new integrators. Step size of integration is 10 s.

(b) Motion of chaser in LVLH frame for ConGAL Test 2, with new integrators. Step size of integration is 1 s.

Figure 7.9: ConGAL Integrators: Results from ConGAL Test 2 with fourth-order integrators.

2. ConGAL Test 4: The test consists of two-impulse transfer along V-bar, in a quarter orbital period. The chaser is 500 m behind the target, step size is 10 s and maximum thrust is 500 N. See Section 7.2.2 for more details about the setup. Amongst the fourth-order integrators, only RK and AM are able to converge to a solution. AB makes the solver to stall, whereas SR makes it crash - thus, they are ignored for the time-being. The results of the test are shown in Figure (7.10). RK and AB take only 8 iterations to converge (with the help of mean-scheme), as compared to 13 of TR. Figure (7.10a) shows the trajectory of chaser in LVLH frame. Comparision with corresponding OP Runs show that RK is accurate to 17 m, whereas AM is accurate

to 20 m. This is comparable to results from TR run of step size 5 s. Figure (7.10b) shows thrust profile against time. As expected, the profile is a two-burn solution. A strange observation is that the first burn of AM run does not commence until 20 s are passed. This delay is expected to make the transfer marginally inefficient. Comparing the $\Delta V$ of the transfer, we see that RK fares better at 0.7809 m/s as compared to 0.7915 m/s from the AM run. Both these values are very close to the analytical value of 0.7746 m/s. Figure (7.10c) shows the mass profile of the spacecraft over the flight. The AM plot shows a kink at about 40 s (which corresponds to end of first burn). This is unusual because at no point should the mass of the spacecraft increase. This numerical anomaly could be attributed to the fact that multi-step integrators are slow to respond to fast changes in values.

In conclusion, four fourth-order integrators were compared. For unknown reasons, Simpsons rule is unable to provide a meaningful solution with ConGAL. The Adams-Bashforth method showed promise in results of ConGAL Test 2, but was unable to generate any result in ConGAL Test 4. The fourth-order Runge-Kutta and Adams-Moultons methods were able to pass the tests, and provided substantially better results than the Trapezoidal rule. Between the two, the Runge-Kutta method performs marginally better in terms of accuracy. Being a multi-step method, the Adams-Moulton integrator is prone to numerical anomalies as seen in the chaser mass profile during ConGAL Test 4. Keeping these points in mind, it is concluded that the fourth order Runge-Kutta method is the most suitable choice for replacing the trapezoidal rule.

Through numerical experience, it is found that although the fourth-order methods provide increased accuracy for a given step size, their full benefit cannot be realized. For a given problem and integrator, the solver has a tendency to stall below a particular step-size. For the trapezoidal rule, this was around 2 s, and for Runge-Kutta, it is around 5 s. This could be seen as a limitation of the currently available solvers.

### 7.3.3 Rate of change of thrust constraint

As discussed in Section 5.5.1, the ConGAL method is able to incorporate constraints on rate of change of thrust vector magnitude and direction. It does so by adding two new equations to the dynamics[14] of the problem and two new constraints. For convenience, those equations are replicated here:

$$\begin{aligned} \dot{\boldsymbol{\tau}} &= \boldsymbol{\nu}, & \dot{\sigma} &= \lambda \\ \|\boldsymbol{\nu}\| &\leq \xi, & \left|\lambda\right| &\leq \zeta \end{aligned} \tag{5.51 revisited}$$

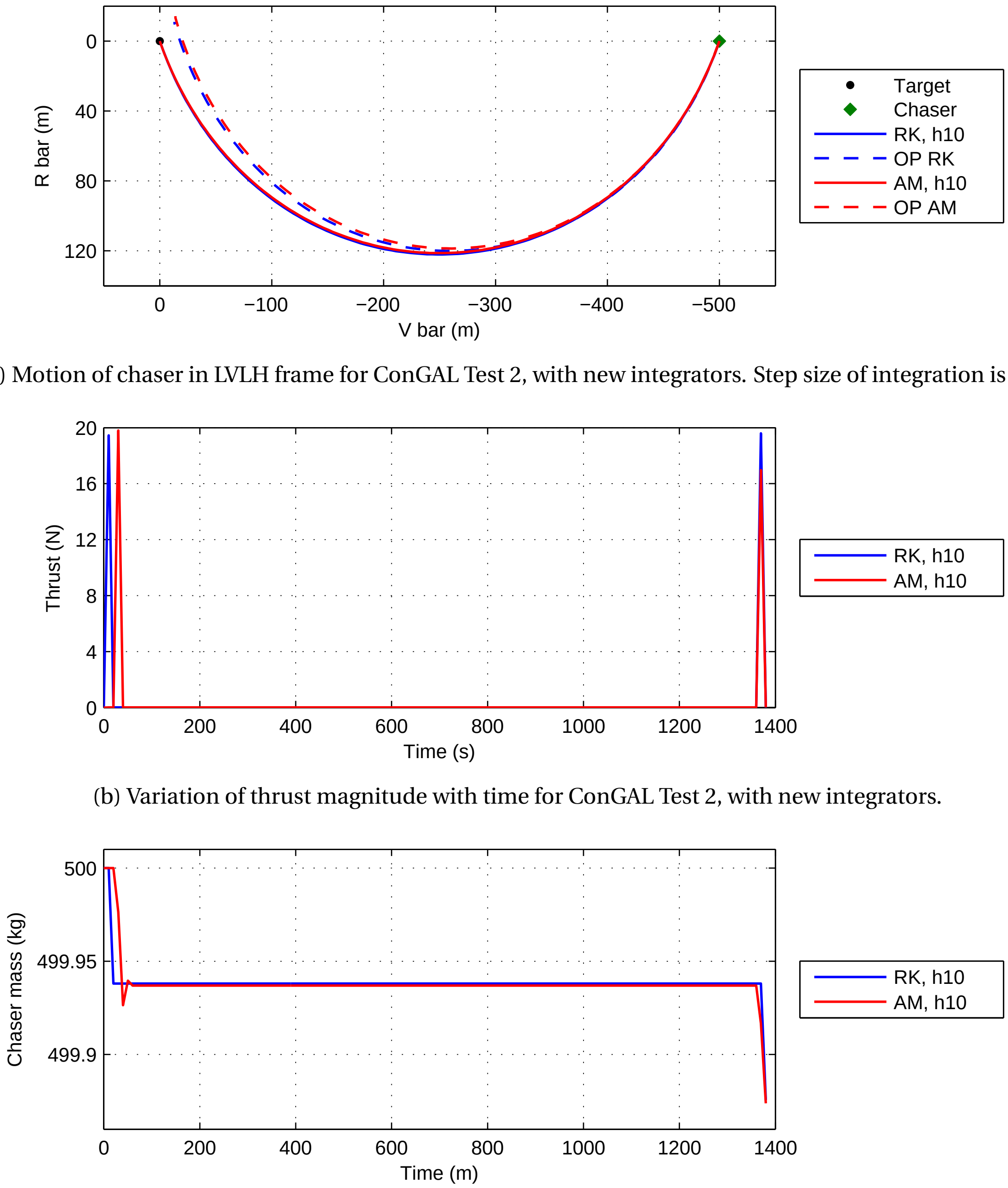
where $\xi$ is the maximum rate of change of thrust direction and $\zeta$ is the maximum rate of change of thrust magnitude. The state vector gets redefined from $\mathbf{y} = [\mathbf{r}; \mathbf{V}; z]$ to $\mathbf{y} = [\mathbf{r}; \mathbf{V}; z; \boldsymbol{\tau}; \sigma]$ and the control vector changes from $\mathbf{u} = [\boldsymbol{\tau}; \sigma]$ to $\mathbf{u} = [\boldsymbol{\nu}; \lambda]$. It also causes the inputs to the state equation (Equation 5.36) to change from:

$$\mathbf{A}(r) = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} \\ -\frac{1}{r^3}\mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & 0 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & -\frac{1}{v_{\text{ex}}} \end{bmatrix} \tag{5.27 revisited}$$

to the larger formulation:

$$\mathbf{A}(r) = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} \\ -\frac{1}{r^3}\mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & 0 & \mathbf{0}_{1\times3} & -\frac{1}{v_{\text{ex}}} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & 0 & \mathbf{0}_{1\times3} & 0 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & 0 \\ \mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix} \tag{5.52 revisited}$$

[14] By dynamics, we mean the equations of motion that relate the state and control vectors



(a) Motion of chaser in LVLH frame for ConGAL Test 2, with new integrators. Step size of integration is 10 s.

(b) Variation of thrust magnitude with time for ConGAL Test 2, with new integrators.

(c) Variation of chaser mass with time for ConGAL Test 2, with new integrators.

Figure 7.10: ConGAL Integrators: Results from ConGAL Test 4 with fourth-order integrators.

This appears to be a substantial variation in dynamics of the problem, just to be able to include rate of change in thrust magnitude and direction constraints. Two problems are seen with this elaborate setup:

1. The problem size increases substantially. The state vector **y** increases from being a $(7 \times 1)$ vector to an $(11 \times 1)$ vector. Combined with the control vector of size $(4 \times 1)$, ConGAL has to now solve for 11 unknowns at each node instead of 11. Matrix **A** increases from being a $7 \times 7$ to $11 \times 11$ vector. Correspondingly, the size of all other constraints and matrices also increases. Not only does this increase the time for solving the problem, it also brings us closer to the

maximum problem size that can be handled by currently available solvers.

2. A full-scale re-haul of the MATLAB code is required to incorporate the change in dynamics (and thus all constraints). And if for some reason, one wants to de-activate the rate of change of thrust vector constraint, one cannot merely return to using the smaller formulation of the problem. Instead, one would need the older version of the code, where all constraints are built for relevant state and control vectors.

To overcome these problems, a slight but far reaching variation in the way the rate of thrust change constraints is included is suggested. It is based on the assumption that more often than not, translational motion in rendezvous problems do not need a constraint on rate of change of thrust direction. In the absence of such a constraint, the maximum possible change in thrust direction is 180°, which physically means that RCS engines from one side of the chaser are shut down while the ones on diametrically opposite side are fired. For a chaser like the one defined in Section 2.3.3, this would not be a problem as it has RCS engines on all six directions.

With this assumption, we need not have to worry about adding the new equations to the dynamics of the problem. We can now focus only on the rate of change of thrust magnitude constraint. It is written as:

$$|\dot{\sigma}| \leq \lambda \tag{7.33}$$

and incorporated as

$$\begin{aligned} \sigma_i - \sigma_{i-1} &\leq h\lambda \\ \sigma_{i-1} - \sigma_i &\leq h\lambda \end{aligned} \tag{7.34}$$

where index $i = 1, 2, \ldots, N$ refers to node number in discretized formulation and $h$ is the step size. This can be easily appended to the list of constraints.

To demonstrate its capability and the associated benefits, we solve ME Test 2 (described in Section 6.5.4) exactly as it was carried out in the work by Liu [2013] (*i.e.*, the extended formulation, with YALMIP as interfacing software and using of scaling factors described ahead in Section 7.4) and compare it with our suggested *reduced* formulation. The results are as follows:

1. The extended formulation solves 6100 linear constraints and 423 conic constraints for 6329 variables. On the other hand, the reduced formulation solves 4896 constraints and 324 cones for 5125 variables. The difference is substantial and made larger, because the complexity of the problem scales with the square of number of variables.

2. The time taken to solve the extended formulation is 71.0 seconds (taken as the mean of 10 runs) as compared to 51.4 seconds for the reduced formulation.

Clearly, the reduced formulation is a much better choice for cases where there is no rate of change of thrust direction constraint. There is nearly 40% savings in computation time, and 20% fewer variables to solve. The values can differ based on the ratio of number of linear and conic constraints of the problem at hand.

## 7.4 SCALING FACTORS

Liu [2013] states that ConGAL requires the parameters used in its formulation to be normalized (or made dimensionless). Although, no specific reason is provided it is thought that by doing so, one reduces numerical error. Take for instance, matrix **A** that is used to define the equations of motion (constraint C1); it involves $r^{-3}$ in its elements, whose absolute value is in the range of 1e-21. This is much smaller than the tolerance values (in the range of 1e-8) used in solvers (to check for optimality, feasibility, etc.) and likely to cause untimely convergence. Further, any manipulation of such parameters is likely to see significant loss of data due to rounding-off errors. Normalizing the parameters ensures that they are workable within the limits of numerical errors.

The scaling factors for normalization, as suggested by Liu [2013], are shown in Table 7.4. Also, shown are the new, alternative scaling factors used in this thesis. Here, $R_0$ refers to Earth's (mean equatorial) radius, $g_0$ is the acceleration due to gravity at $R_0$, $\mu_E$ is Earth's gravitational parameter, and $m_0$ is the initial mass of the spacecraft.

Table 7.4: Scaling factors for normalizing parameters

| Parameter | Scaling factors | |
|---|---|---|
| | Liu | New |
| Radius vector | $R_0$ | $R_0$ |
| Velocity vector | $\sqrt{g_0 R_0}$ | $\sqrt{\mu_E / R_0}$ |
| Time | $\sqrt{R_0 / g_0}$ | $\sqrt{R_0^3 \big/ \mu_E}$ |
| Mass | $m_0$ | $m_0$ |
| Thrust vector | $m_0 g_0$ | $m_0 g_0$ |

To explain the difference in the scaling factors, we first derive our system of scaling factors. We start by defining $R_0$ as the scaling factor for radius vector. Thus,

$$\underline{\mathbf{r}} = \frac{\mathbf{r}}{R_0} \tag{7.35}$$

Normalized parameters are underlined, to mark the difference with absolute parameters. Further, we define the scaling factor for time to be the inverse mean motion of a satellite whose semi-major axis is equal to Earth's radius:

$$\underline{t} = \frac{t}{T_0} = t \Bigg/ \sqrt{\frac{R_0^3}{\mu_E}} \tag{7.36}$$

Using the above values, we find the normalized velocity vector:

$$\begin{aligned} \underline{\mathbf{V}} &= \frac{\mathrm{d}\underline{\mathbf{r}}}{\mathrm{d}\underline{t}} = \frac{\mathrm{d}\underline{\mathbf{r}}}{\mathrm{d}t} \cdot \frac{\mathrm{d}t}{\mathrm{d}\underline{t}} \\ &= \frac{1}{R_0} \cdot \frac{\mathrm{d}\mathbf{r}}{\mathrm{d}t} \cdot \sqrt{\frac{R_0^3}{\mu_E}} \\ &= \mathbf{V} \Bigg/ \sqrt{\frac{\mu_E}{R_0}} \end{aligned} \tag{7.37}$$

Thus, the scaling factor for velocity is $\sqrt{\frac{\mu_E}{R_0}}$. Using this, we can derive the normalized equation for acceleration acting on the chaser as:

$$\begin{aligned}
\underline{\mathbf{a}} &= \frac{\mathrm{d}\underline{\mathbf{V}}}{\mathrm{d}\underline{t}} = \frac{\mathrm{d}\underline{\mathbf{V}}}{\mathrm{d}t} \cdot \frac{\mathrm{d}t}{\mathrm{d}\underline{t}} \\
&= \left(\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t} \cdot \sqrt{\frac{R_0}{\mu_E}}\right) \cdot \left(\sqrt{\frac{R_0^3}{\mu_E}}\right) \\
&= \left(-\frac{\mu_E}{r^3} \cdot \mathbf{r} + \frac{\mathbf{T}}{m}\right) \cdot \frac{R_0^2}{\mu_E} \\
&= \left(-\frac{\mu_E}{\underline{r}^3 R_0^3} \cdot \underline{\mathbf{r}} R_0 + \frac{\mathbf{T}}{m}\right) \cdot \frac{R_0^2}{\mu_E} \\
&= -\frac{1}{\underline{r}^3} \cdot \underline{\mathbf{r}} + \frac{\mathbf{T}}{m} \cdot \frac{R_0^2}{\mu_E} \\
&= -\frac{1}{\underline{r}^3} \cdot \underline{\mathbf{r}} + \mathbf{T} \cdot \frac{R_0^2}{\mu_E} \cdot \frac{1}{m} \cdot \frac{m_0}{m_0} \\
&= -\frac{1}{\underline{r}^3} \cdot \underline{\mathbf{r}} + \frac{\mathbf{T}}{m_0 \frac{\mu_E}{R_0^2}} \bigg/ \frac{m}{m_0} \\
&= -\frac{1}{\underline{r}^3} \cdot \underline{\mathbf{r}} + \frac{\underline{\mathbf{T}}}{\underline{m}}
\end{aligned} \tag{7.38}$$

We now have the normalized acceleration vector written in terms of normalized parameters. This is equal to the equation derived by Liu as well (see Equation (5.3)). In the process, we choose that the spacecraft mass $m$, is normalized by the initial mass $m_0$, leading to the thrust vector $\mathbf{T}$ being normalized by $m_0 \frac{\mu_E}{R_0^2}$. Since $\frac{\mu_E}{R_0^2}$ is the acceleration due to gravity at $R_0$ for spherical earth, the thrust vector is actually normalized by the initial weight of the chaser spacecraft.

Using $R_0$ = 6378136 m and $\mu_E$ = 3.986004415e14 m$^3$/s$^2$, we find $g_0 = \frac{\mu_E}{R_0^2}$ = 9.7983 m/s$^2$. Using these values, if we have a spacecraft at unit radius and unit velocity, then in unit time the spacecraft would have completed one orbit around Earth. The resulting orbit would have a semi-major axis of unit radius. However, Liu [2013] makes use of $g_0$ = 9.807 m/s$^2$ to calculate the scaling factors. Using them, the spacecraft's parameters do not remain equal to unity at the end of unit time period. This small but important difference in defining $g_0$ results in inconsistent scaling factors.

To demonstrate the difference, we let ConGAL propagate the state of chaser using both Liu's scaling factors and the new scaling factors. The state of chaser after one orbital period is then compared with that from OP. The initial state corresponds to a 370 km, circular orbit; the orbital period is ≈5520 s. The results are plot in Figure 7.11, which shows orbital radius with respect to time of flight. The orbit with Liu's scaling factors is no more circular, but has a perigee of 358.42 km (or eccentricity of 0.00172). The orbit with the new scaling factors remains circular and matches exactly with that from OP.

The repercussions of using inconsistent scaling factors extend beyond inaccuracies in state propagation. It also affects the guidance solution generated by the solver and the ability of MOSA to converge. Such a behaviour was observed in ME Test 2 (see Section 6.5.4), when the solution with Liu's scaling factors was *falsely* able to converge within 4 iterations, while infact, the convergence was elusive until the mean-scheme was applied. This raises questions about the credibility of numerical examples provided by Liu [2013], Lu and Liu [2013] and Liu and Lu [2013]. In their work, it is shown that convergence is reached "in just a few iterations", while our numerical experience contrasts it.

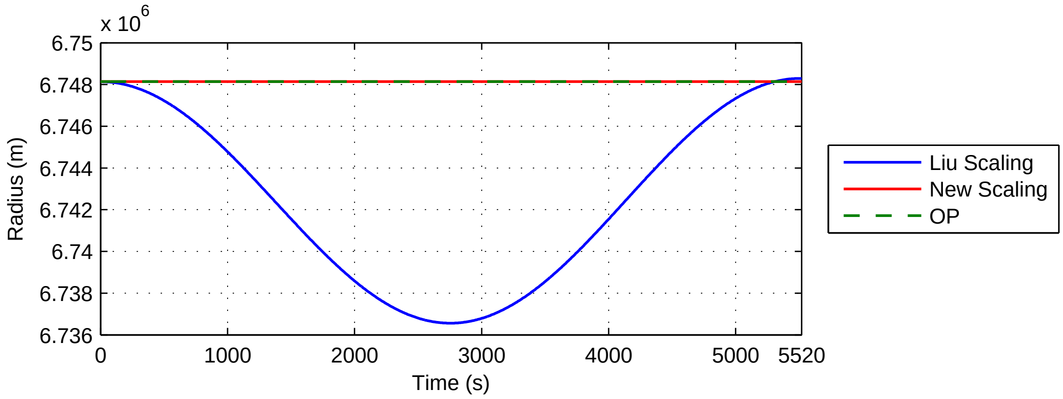


Figure 7.11: Scaling Test: ConGAL results for variation of orbital radius over one orbital period; Satellite is initially in a 370 km, circular orbit.

## 7.5 INFLUENCE OF SOFTWARE

Besides the issue with scaling factors, there was another source that did not allow initial verification of results with those presented by Liu [2013]. Solvers and the intermediate software used for interfacing with the solver, affect the accuracy and the convergence of ConGAL, sometimes to the extent of finding a solvable problem to be unsolvable.

To find the influence and identify the best solver/software for our use, we carry out ME Test 1 and ME Test 2 (see Section 6.5.4) for three step-sizes each. There are six possible combinations of solvers and interfacing software: Solver MOSEK is called twice (individually and with YALMIP), solver ECOS is called thrice (individually, YALMIP and CVX) and solver SDPT3 is called once (with CVX). Any other combination is not available and hence, not considered. Details about MOSEK are available in Chapter 6, while brief descriptions about other solvers/software is given in Appendix B.

The results of the tests are shown in Figure 7.12. Solid lines represent acceptable results, while dashed lines indicate some constraints were violated.

1. ME Test 1, Time of flight = 1800 s, step sizes = 5, 10 and 20 s.

   (a) All six combinations are able to generate an acceptable solution (with help of mean-scheme).

   (b) There is nearly linear relation between computation time and number of nodes. This is expected because solutions to conic programs is found in polynomial time.

   (c) ECOS is marginally quicker than MOSEK, however, the difference reduces as the problem size increases. This is similar to the behaviour described by Domahidi et al. [2013] during his benchmark tests.

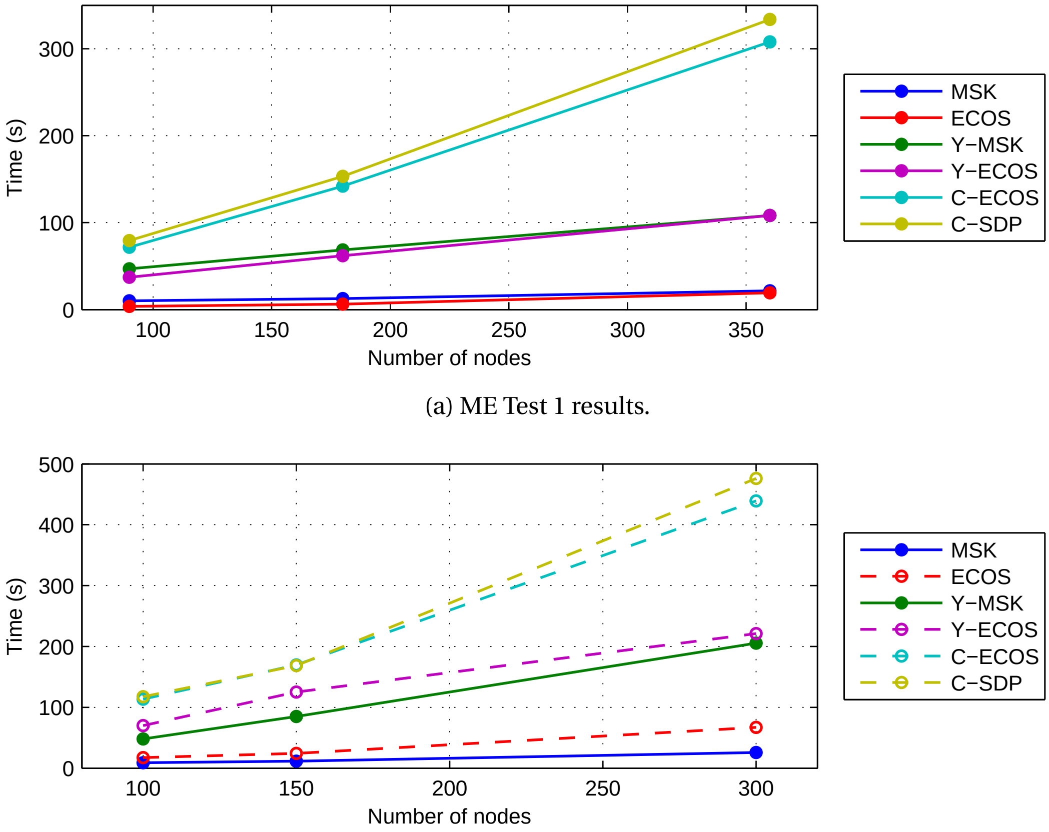
   (d) CVX takes the highest time to solve the problem, and also has the steepest rise in time as the problem size increases.

2. ME Test 2, Time of flight = 3000 s, step sizes = 10, 20, 30 s.

   (a) Barring MOSEK, no other solver is able to find a solution to the problem. Both ECOS and SDPT3 violate some constraints / run out of solver iterations (not iterations of MOSA, but internal iterations of the primal-dual interior point method).

   (b) CVX shows non-linear relation with time, and a slight bulge is also observed for Y-ECOS. This is probably because of running out of solver iterations.

Clearly, MOSEK is the preferred choice of solver, as it is able to solve both the problems. The influence of using YALMIP with MOSEK is substantial - upto 9 times higher cost. This is bound to increase as the problem size increases, as the slope of Y-MSK is much higher than MSK entry.



(a) ME Test 1 results.

(b) ME Test 2 results.

Figure 7.12: Influence of software on ConGAL results: Variation of computational time with problem size. Solid lines are acceptable results.

## 7.6 SUMMARY

A quick summary of the results from the numerical analysis of ConGAL is presented below:

1. ConGAL's fails to produce guidance commands for simple / lightly constrained rendezvous problem (despite user-provided assistance), indicating that its application is limited to specific, highly constrained rendezvous problems.

2. One of the primary cause for its failure to generate a solution is attributed lack of convergence / slow convergence of MOSA. The mean-scheme was developed to combat this issue and it has been shown to be moderately successful.

3. Another concern is accuracy, as the current integrator used in ConGAL is only second-order accurate. A trade-off between four integrators found the fourth-order Runge-Kutta integrator to be the most suitable replacement.

4. The constraint on rate of change of thrust vector was replaced with a reduced formulation, to avoid unnecessary manipulation of the dynamics of the problem.

5. The credibility of the numerical examples provided by Liu [2013] and his collaborators is contested because of inconsistency in the scaling factors used for normalization. Such an error can result in inaccurate trajectories and false convergence of solutions. It could also be the cause for their overall positive numerical experience with ConGAL.

6. The choice of solver influences the solution generated substantially. MOSEK was found to be the most reliable solver for this study. Interfacing software, for calling the solver and easing the problem formulation, are found to be computationally very expensive.

The variation of ConGAL, with the improvements suggested in this chapter, is termed as modified ConGAL (or mConGAL).

# 8

# ENVISAT RENDEZVOUS

Moving past the extended detour that was Chapter 7, we now focus upon the main problem at hand - to perform rendezvous with Envisat. The mission poses significant challenges that need to be modelled for use with convex guidance. To this end, we shall make use of mConGAL, the modified version of ConGAL, that we have been building in the last two chapters. Owing to unknown exact orientation of Envisat's docking axis, numerous representative cases are considered for the mission. It is followed by an extended analysis of results, including a sensitivity analysis. A path-tracking PID controller and a work-around to accommodate perturbed dynamics are also demonstrated.

## 8.1 ENVISAT PROBLEM MODELLING

In Section 2.3.4, the scope of mission scenario, based on a study by ESA [2012], was defined. It was expected that ConGAL would be able to generate guidance commands for all the three phases of the mission. However, based on our experience with ConGAL, it is concluded that while it might be able to generate solutions for highly constrained rendezvous missions (as seen in ME Test 2 for example), it is unable to produce guidance commands for unconstrained (or mildly constrained) transfers between hold points on V-bar (based on results of ConGAL Test 1, 3 and 4, from Sections 7.1 and 7.2). This calls for redefining the scope of the mission scenario so that ConGAL (and our modification, mConGAL) can focus on the final, highly constrained part - the final approach phase. The phase is also slightly modified to start from initial point (denoted by IP) at distance of -100 m on V-bar, and end when the chaser reaches a final point (denoted by FP) at distance of 3 m from the target along the docking axis. Due to public unavailability of certain mission requirements, the work by Deloo [2014] is used as a source for missing data. The mission poses numerous challenges which are modelled with the following constraints:

1. **Hold Point (HP1)**:
   A hold point, at a distance of 50 m from the target, is defined to mark the point from where the chaser should follow the docking axis of Envisat, and serves as an entry point to the keep-out-sphere and approach cone (which are defined below). Its exact location in the LVLH frame is dependent on the orientation of the docking axis vector, $\mathbf{1}_n$. At the hold point, the chaser relative velocity is zero. It is modelled using constraint C4.

2. **Keep-Out-Sphere (KOS)**:
   As the chaser carries out the final approach to the target, it should ensure that its trajectory does not come too close to Envisat or its appendages except along a pre-defined approach cone. In this regard, a keep-out-sphere of radius 50 m is defined at the start of the mission. This is modelled by constraint C9, and is active until the chaser reaches HP1 (which falls on

the KOS boundary). A second KOS of radius 3m was found to be necessary when the chaser is inside the approach cone. The reason for this is discussed at the end of this list. Unless explicitly stated, all mentions of the acronym KOS refer to the larger sphere of radius 50 m.

3. **Docking axis (DA)**:
The docking axis is defined along the spinning axis of Envisat (i.e., the $Z$-axis of the $\mathcal{B}$ frame). Due to Envisat's tumbling motion, three representative Attitude Scenarios (AS) were defined (see Table 2.4), with anti-clockwise motion (normal right-handed system). AS 1 has the docking axis aligned with the angular momentum vector, **h**, and thus appears to be fixed in the LVLH frame (always along negative H-bar axis). In AS 2 and 3, the docking axis precesses while making angles $\frac{\pi}{4}$ and $\frac{\pi}{2}$ with the angular momentum vector, respectively. Thus, its orientation changes with time in the LVLH frame, as well as in the Inertial frame ($\mathcal{I}$). It is used for defining HP1, FP, approach cone and plume impingement constraints, and thus is an important function. mConGAL allows a time-varying docking axis vector, $\mathbf{1}_n(t)$ but it should be defined in the inertial frame. It can be calculated as follows:

   (a) Define the initial orientation of $\mathcal{B}$ frame with respect to LVLH frame using precession angle, $\theta_p$, which is the angle between $Z_{\mathcal{B}}$ and **h**; and the rotational angle, $\theta_r$, which is the angle the plane defined by $Z_{\mathcal{B}}$ axis and **h** vector, makes with the cross-track plane (*i.e.,* $XY$-plane in LVLH frame) at a given point in time. See Figure 8.1 for illustration. As time passes, $\theta_p$ remains constant while $\theta_r$ changes in accordance with the precession rate. The spin rate and the exact initial orientation of $X_{\mathcal{B}}$ and $Y_{\mathcal{B}}$ do not affect the instantaneous orientation of the docking vector axis, and thus can be arbitrary. For sake of uniformity in the tests, the initial orientation of $X_{\mathcal{B}}$ axis shall be presumed to lie in the rotating plane defined above. The rotation matrix for frame transformation (see Section 3.3 for details) is given as follows:

      i. Attitude Scenario 1: At all point in time, $\theta_p = 0$, and $\theta_r$ is undefined. The rotation matrix is constant:

$$\mathbf{R}_{\mathcal{B}}^{\mathcal{V}} = \mathbf{R}_x(-\frac{\pi}{2})\mathbf{R}_y(0)\mathbf{R}_z(0) \tag{8.1}$$

      ii. Attitude Scenario 2 and 3: For any value of $\theta_p \notin \{0, \pi\}$, the rotation matrix is:

$$\mathbf{R}_{\mathcal{B}}^{\mathcal{V}} = \mathbf{R}_z(\frac{\pi}{2})\mathbf{R}_x(\theta_p - \frac{\pi}{2})\mathbf{R}_y(\theta_r + \frac{\pi}{2}) \tag{8.2}$$

      The choice of the uncommon $YXZ$ rotation sequence allows a singularity free transformation for any value $\theta_r \in [0, 2\pi]$.

   (b) Convert the initial rotation matrix defined above to its quaternion form, $\{\mathbf{q}_{\mathcal{V}}^{\mathcal{B}}\}_0$. Switching to quaternions allows easier kinematic propagation, as discussed in Section 3.3.1. Using Equation (3.54), we have:

$$\{\mathbf{q}_{\mathcal{V}}^{\mathcal{B}}\}_{i+1} = \left[\cos\left(\omega_{\mathcal{B}}\frac{\Delta t}{2}\right)\mathbf{I}_{4\times4} + \frac{1}{\omega_{\mathcal{B}}}\left(\sin\left(\omega_{\mathcal{B}}\frac{\Delta t}{2}\right)\right)\mathbf{\Omega}_{\mathcal{B}}\right]_i \{\mathbf{q}_{\mathcal{V}}^{\mathcal{B}}\}_i \tag{8.3}$$

   where $i$ refers to the time instant, $\Delta t$ is the time interval between successive instances, $\omega_{\mathcal{B}}$ is the magnitude of the instantaneous angular velocity, and $\mathbf{\Omega}_{\mathcal{B}}$ is the $(4 \times 4)$ matrix of relative angular velocity components defined in Equation (3.53). The angular velocity vector $\boldsymbol{\omega}_{\mathcal{B}}$, is time-varying because of the precessing motion. It is given by:

$$\boldsymbol{\omega}_{\mathcal{B}} = \boldsymbol{\omega}_s + \boldsymbol{\omega}_p(t_i) \tag{8.4}$$

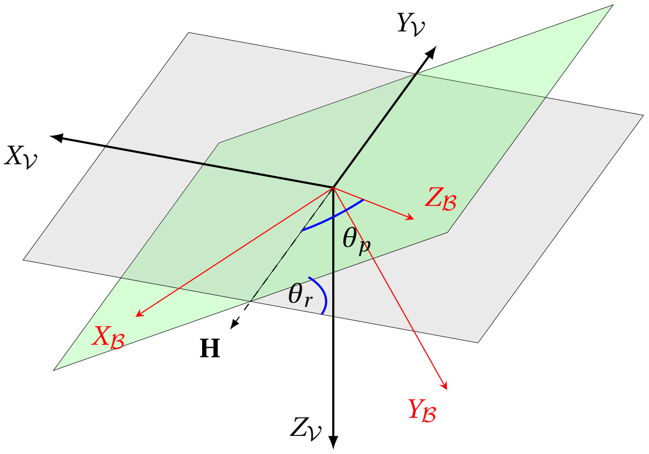


Figure 8.1: Initial orientation of Envisat body frame ($\mathcal{B}$) with LVLH frame ($\mathcal{V}$) is defined by precession angle $\theta_p$ and rotational angle $\theta_r$. For a given Attitude Scenario, $\theta_p$ remains constant while $\theta_r$ changes at 0.15°/s.

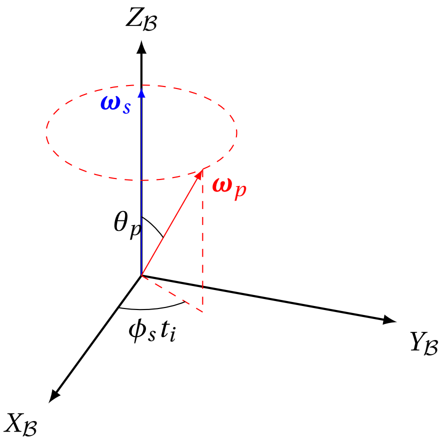


Figure 8.2: Angular velocity of frame $\mathcal{B}$ with respect to frame $\mathcal{V}$ as seen in $\mathcal{B}$ frame. It consists of spinning and precessing motion, as shown by angular velocity vectors $\boldsymbol{\omega}_s$ and $\boldsymbol{\omega}_p$.

where indices $s$ and $p$ refer to the spinning and precessing motion. These vectors can be derived geometrically, with the help of Figure 8.2.

$$\boldsymbol{\omega}_s = \begin{bmatrix} 0 \\ 0 \\ \phi_s \end{bmatrix}, \qquad \boldsymbol{\omega}_p = \begin{bmatrix} \phi_p \sin\theta_p \cos(\phi_s t_i) \\ \phi_p \sin\theta_p \sin(\phi_s t_i) \\ \phi_p \cos\theta_p \end{bmatrix} \tag{8.5}$$

Note that $\boldsymbol{\omega}_s$ is constantly aligned along $Z_{\mathcal{B}}$, and its magnitude is equal to the spin rate, $\phi_s$. On the other hand, $\boldsymbol{\omega}_p$ depends on the precession rate, $\phi_p$, the precession angle, $\theta_p$ for $Z_{\mathcal{B}}$ component value, and additionally dependent on spin rate, $\phi_s$ and time since epoch, $t_i$ for values in the remaining components[15]. The magnitude of $\boldsymbol{\omega}_p$ is equal to the precession rate.

(c) The quaternion propagation is sampled at a much higher rate (*i.e.*, lower time step size) than mConGAL for accurate results. Filter away the extra sampled results to end up with a set of quaternions $\mathbf{q}_{\mathcal{V}}^{\mathcal{B}}$, one for each node of the mConGAL problem. Take their conjugate, $\mathbf{q}_{\mathcal{B}}^{\mathcal{V}}$, and convert them to corresponding rotation matrices, $\mathbf{R}_{\mathcal{B}}^{\mathcal{V}}$. It can then be used

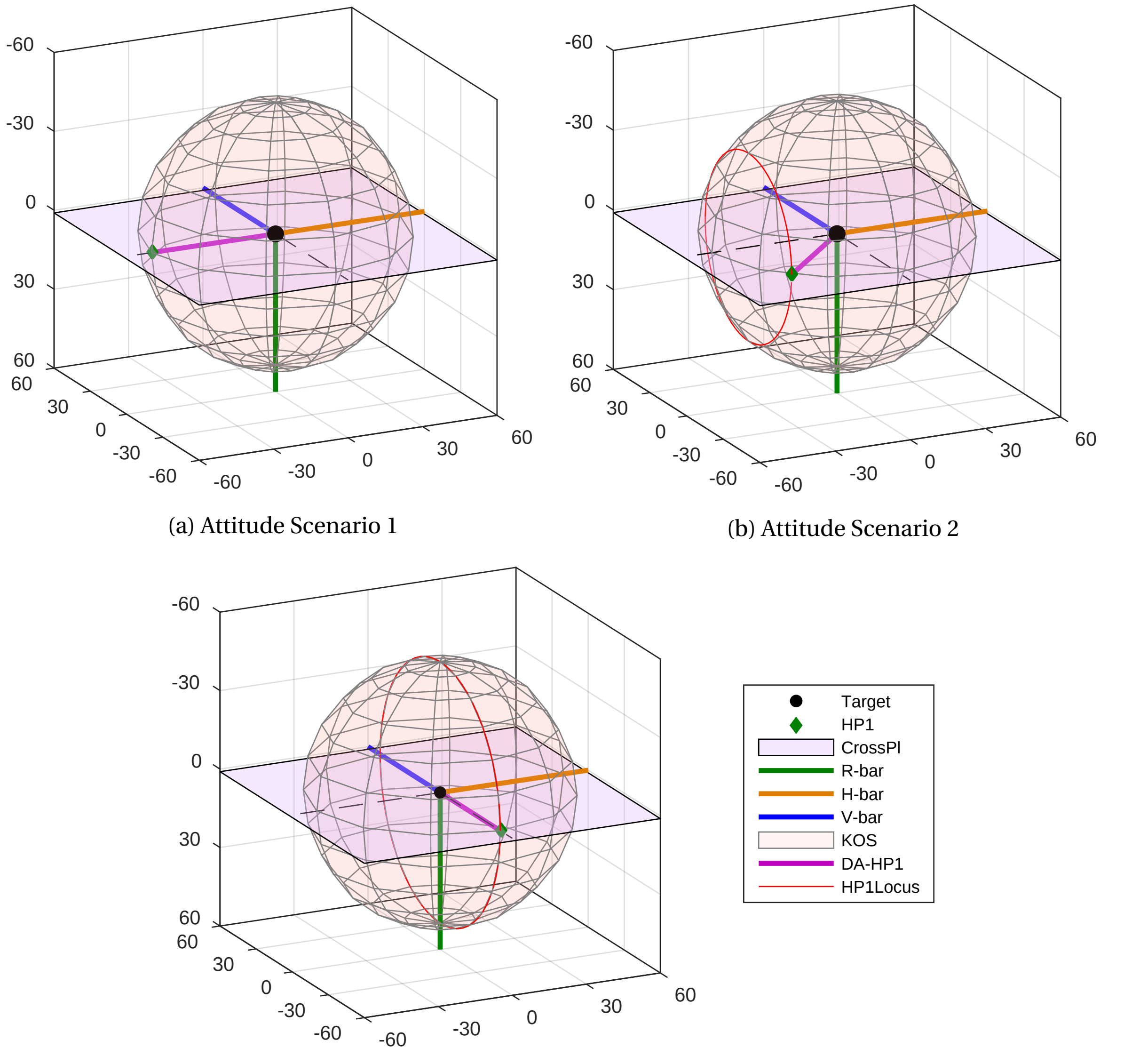
[15] The equations are valid for the (arbitrary) initial orientation of $X$ and $Y$ axis of $\mathcal{B}$-frame as defined in point 3a of this list

to convert docking axis from $\mathcal{B}$ frame to $\mathcal{I}$ frame as follows:

$$\mathbf{1}_n(t) = \mathbf{R}_{\mathcal{V}}^{\mathcal{I}} \mathbf{R}_{\mathcal{B}}^{\mathcal{V}} \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}' \tag{8.6}$$

Note that both the rotation matrices in the above equation are time-varying.

Using the above equations, we can calculate the docking axis vector in the inertial frame. To verify this model, it is plotted back in the LVLH frame to check if the locus of the docking axis vector follows expected values. The results are shown in Figure 8.3. The locus starts from a hold point selected on the cross-track plane, marked HP1 in the figure, and follows a circle on the KOS, marked HP1Locus. It is concluded that the model passes the test. Notice that the HP1Locus for AS 3, lies in orbital (R-bar/V-bar) plane, while for AS 2, it is offset towards the negative H-bar direction.



(a) Attitude Scenario 1

(b) Attitude Scenario 2

(c) Attitude Scenario 3

Figure 8.3: Orientation of Docking Axis (DA) when the chaser reaches Hold Point (HP1), for the three Attitude Scenarios. KOS: Keep-Out-Sphere, CrossPl: Cross plane with R-bar as normal to it.

4. **Approach Cone**:
   The region inside KOS, within which the chaser is allowed to move. The cone's vertex lies at the target and its central axis moves along with the docking axis. It is decided to fix the approach-cone half angle to 10° throughout. Constraint C5 is used for modelling it.

5. **Plume impingement**:
   Once within the approach cone, the chaser cannot fire its thrusters freely in all directions as the plume might impinge the target, imparting impulse or causing damage. In this regard, a plume impingement angle of 85° is enforced using constraint C6 when the chaser is inside the KOS.

6. **Forced Motion**:
   Another requirement for the mission is that within the KOS, the chaser should have forced motion[16]. It means, that there should be no impulse shot manoeuvres and the trajectory should follow a particular set path with continuous thrusting. An attempt to define a realistic forced motion constraint in mConGAL failed, forcing us to apply a slightly compromised set of constraints instead. The failure could be attributed to the high requirement for sampling rate for forced motion constraint and a need for solvers that can handle higher tolerance values for the solution.

   The suggested work-around is put in place by reducing maximum available thrust and enabling a cap on rate of thrust magnitude change. This ensures that there are no impulse transfers and for most part, the chaser requires continuous thrusting. Using results by Deloo [2014] as a benchmark, the maximum available thrust is reduced to 2 N when within the KOS (enforced by constraints C2), and the rate of thrust magnitude is constrained to 0.1 N/s (enforced through constraint C7). The method has a disadvantage that there can be some intervals when there is no thrust acting on the chaser, making the motion to be forced only partially. However, it also has the advantage that one need not manually specify the *set path* for the chaser to force its motion along. As seen in work by Deloo [2014], the path can be complex, non-optimal and in violation of other constraints like thrust impingement. The suggested alternative automatically finds a fuel optimal, well constrained solution as shall be seen later.

7. **Final Point (FP)**:
   The final point is defined at a distance of 3 m from the target at end of flight. It is located along the docking axis and thus, its exact position in LVLH frame changes with time. Also, the chaser relative velocity with the target should be zero. It can be modelled as a modified final state constraint (Cf) as follows:

   (a) Convert inertial docking axis vector to LVLH frame: $\{\mathbf{1}_n(t)\}_{\mathcal{V}} = \mathbf{R}_{\mathcal{I}}^{\mathcal{V}}\mathbf{1}_n(t)$

   (b) Define LVLH final state: $\{\mathbf{x}_f\}_{\mathcal{V}} = \begin{bmatrix} 3\cdot\{\mathbf{1}_n^{\intercal}(t)\}_{\mathcal{V}} & 0 & 0 & 0\end{bmatrix}'$

   (c) Convert LVLH final state to Inertial state, $\mathbf{x}_f$, as discussed in Section 3.3.2.

   (d) Normalize the state and replace in constraint Cf, defined in Section 6.5.2.

   A rotating approach cone defined from LVLH origin along with an FP that is offset from the LVLH origin, gives the chaser a chance to reach the LVLH origin (and in the process crash with the target). This is because the region between LVLH origin and FP is not explicitly stated to be inaccessible. By using an inner KOS of 3 m, centred at LVLH origin, we can solve this problem.

[16] There is no specific definition of forced motion or an explicit requirement that can be used to define forced motion, in the documents by ESA. However, based on statements by Deloo [2014] and personal communication with him, it is concluded that there is a forced motion requirement once inside the KOS, where the chaser should follow a set path with continuous thrusting.

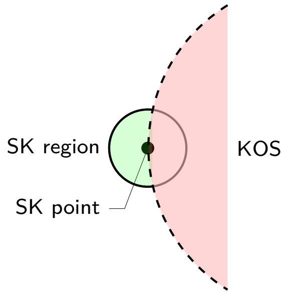


Figure 8.4: Station Keeping (SK) constraint: A small sphere (green region) is defined around the station keeping point within which the chaser is allowed to drift. The region overlapping with the Keep-Out-Sphere is out of bound for the chaser. This reduces the available SK region to about half a sphere. Figure not to scale.

8. **Station Keeping (SK)**:
   Taking cue from work by Deloo [2014], two station keeping intervals of 300 s each at HP1 and FP, are considered for the mission. Tasks such as awaiting further instructions before entering KOS, matching attitude spin before clamping and awaiting docking axis alignment can be incorporated into the mission by considering such station keeping intervals. The time intervals are merely representative, and the primary objective is to see if such manoeuvres could be included in the mission.

   At first glance, it appears, this constraint could be modelled as an extension of the hold point constraint - instead of having the hold point constraint (C4) at one node, one could have it enabled for a set of consecutive nodes. However, mConGAL is unable to reliably solve the problem with such a constraint. In fact, it finds the problem to be infeasible in most cases. A possible cause for such behaviour could be that in the time interval between two consecutive nodes, the chaser drifts away substantially, and this cannot be corrected in time if the state of the second node is to be kept the same as the first node. One way to tackle the problem would be to reduce the time step, but as seen in Chapter 7, that is not an option. Another way to solve it involves defining a drifting tolerance region around the station keeping state. If the region is sufficiently small, it can mimic the station keeping that we seek. The equations are as follows:

   $$\begin{aligned} \left\|\mathbf{r}(t) - \mathbf{d}_{10\mathrm{a}}(t)\right\| &\leq \epsilon_r \\ \left\|\mathbf{V}(t) - \mathbf{d}_{10\mathrm{b}}(t)\right\| &\leq \epsilon_V \end{aligned} \tag{8.7}$$

   where, $\mathbf{d}_{10\mathrm{a}}(t)$ and $\mathbf{d}_{10\mathrm{b}}(t)$ refer to the inertial position and velocity we wish to maintain, and $\epsilon_r$ and $\epsilon_V$ are the drifting tolerances on position and velocity. Through numerical experience, it is decided to set the values of drifting tolerance to 0.2 m and 0.05 m/s. In some cases, mConGAL finds the problem to be infeasible if the values are smaller. The position drifting tolerance is made tighter by the presence of KOS constraints at HP1 and FP, as illustrated in Figure 8.4.

These are some of the main challenges of the Envisat rendezvous mission with respect to guidance. With respect to the initial problem this thesis sought to solve, the final approach mission is smaller in terms of the time of flight. However, it still incorporates all the constraints that make Envisat as a challenging rendezvous mission. Note that some other challenges, like those pertaining to passive safety, collision avoidance manoeuvres, are beyond the scope of this thesis work.

## 8.2 ENVISAT PROBLEM FORMULATION

Having defined the challenges of the Envisat rendezvous mission, we now move on to formulating it for mConGAL. The mission starts at $t_0$ and ends at $t_f$, with a total of $N+1$ nodes where $N = t_f/h$. The chaser is expected to reach HP1 at $t_{\mathrm{HP}}$ and perform station keeping (SK1) until $t_c$, after which the chaser enters the approach cone. It reaches FP at $t_{\mathrm{FP}}$ and maintains that position (SK2) until $t_f$. The discretized optimization problem we are looking to solve is shown in Problem 7 on Page 122.

For ease of reader and to allay any confusion in notations, some variables are re-defined here. At each point in time, the optimization problem solves for a (11 × 1) vector $\mathbf{w} = \begin{bmatrix}\mathbf{y}; & \mathbf{u}\end{bmatrix}$, where $\mathbf{y} = \begin{bmatrix}\mathbf{r}; & \mathbf{V}; & z\end{bmatrix}$ is the state vector consisting of inertial radius vector, velocity vector and logarithm of spacecraft mass. The control vector $\mathbf{u} = \begin{bmatrix}\boldsymbol{\tau}; & \sigma\end{bmatrix}$ consists of thrust acceleration vector and norm of thrust acceleration. The problem seeks to minimize the cost function $J$, which is the sum total of impulse delivered to the chaser. The state of the target is referred to by the index $t$. The docking axis vector is $\mathbf{1}_n$, approach cone half-angle is $\alpha$, minimum plume impingement angle is $\beta$, maximum thrust rate change is $\zeta$. The problem shall be solved repeatedly until it converges (with the help of mean-scheme discussed in Section 7.3). All variables which are not explicitly associated with iteration $(k-1)$ in Problem 7, belong to iteration $k$. The $\mathbf{d}$ vector refers to inertial state (or only position or velocity, as required) that is required to define each of the constraints, the details of which can be found in Section 6.5.2. Lastly, $d_{9\mathrm{a}}$ refers to KOS radius and $d_{9\mathrm{a}}$ refers to trust region, which helps solves the KOS constraint. Note that in constraint C1, we are making use of the Runge-Kutta integrator and constraint C7 makes use of the reduced formulation of rate of thrust change (discussed in Section 7.3). The constraint on forced-motion is applied by a combination of constraints C2 and C7 when the chaser is inside the approach cone.

Figure 8.5 shows the timeline of the mission, and the constraints that are active during each section. One can clearly make out that the the mission is divided into two major parts: Part 1 from $t_0$ to $t_c$ (during which the chaser travels from IP to HP1, and performs station-keeping) and Part 2 from $t_c$ to $t_f$ (during which the chaser travels from HP1 to FP, and performs station-keeping).

At this point, we still have to define the exact position of HP1 and values of $t_c$ and $t_f$ for the mission. These are discussed as follows:

1. Position of HP1: For AS 2 and 3, the position of HP1 is dependent on the initial orientation of Envisat (when the chaser is at IP) and the time it takes to reach from IP to reach HP1. In a real mission, the position of HP1 could be anywhere on its locus and the chaser should be capable of handling all or most of such cases. Since it is not possible to try out each of these HP1 positions, it is decided to try a range of typical HP1 positions for AS 2 and 3. These points are listed in Table 8.1. As mentioned before, the HP1 points for AS 3 are in the orbital plane, while for AS 2, they are in a plane along that is offset in the negative H-bar direction.

Table 8.1: List of position of HP1 for analysis of Envisat Mission (EM)

| Attitude Scenario | EM id | $\theta_p$ (°) | $\theta_r$ (°) | LVLH position (m) |
|---|---|---|---|---|
| AS 1 | 1a | 0 | - | $[0 \quad -50 \quad 0]$ |
| AS 2 | 2a | 45 | 0 | $[-50/\sqrt{2} \quad -50/\sqrt{2} \quad 0]$ |
| | 2b | | 90 | $[0 \quad -50/\sqrt{2} \quad -50/\sqrt{2}]$ |
| | 2c | | 180 | $[50/\sqrt{2} \quad -50/\sqrt{2} \quad 0]$ |
| | 2d | | 270 | $[0 \quad -50/\sqrt{2} \quad 50/\sqrt{2}]$ |
| AS 3 | 3a | 90 | 0 | $[-50 \quad 0 \quad 0]$ |
| | 3b | | 90 | $[0 \quad 0 \quad -50]$ |
| | 3c | | 180 | $[50 \quad 0 \quad 0]$ |
| | 3d | | 270 | $[0 \quad 0 \quad 50]$ |

**Problem 7:**
Minimize for the given $t_f$:

$$J = \sum_{t_0}^{t_f} h\sigma(t_i) \tag{8.8}$$

subject to
C0: Initial state

$$\mathbf{y}(t_0) - \mathbf{d}_0 = 0 \tag{8.9}$$

C1: Inertial equations of motion

$$-\mathbf{y}(t_{i+1}) + \left[\mathbf{I} + h\mathbf{K}(t_i)\mathbf{A}(t_i)\right]\mathbf{y}(t_i) + \left[h\mathbf{K}(t_i)\mathbf{B}\right]\mathbf{u}(t_i) = 0 \tag{8.10}$$

C2: Maximum thrust value

$$0 \le \sigma(t_i) \le \mathbf{T}_{\max} e^{-z^{(k-1)}(t_i)}\{1 - [z(t_i) - z^{(k-1)}(t_i)]\} \tag{8.11}$$

C3: Norm of thrust less than maximum thrust value

$$\left\|\boldsymbol{\tau}(t_i)\right\| \le \sigma(t_i) \tag{8.12}$$

C4: Hold point

$$\mathbf{r}(t_{\mathrm{HP}}) - \mathbf{d}_{4\mathrm{a}} = 0 \tag{8.13}$$

$$\mathbf{V}(t_{\mathrm{HP}}) - \mathbf{d}_{4\mathrm{b}} = 0 \tag{8.14}$$

C5: Approach cone

$$\left\|\mathbf{r}(t_i) - \mathbf{r}_t(t)\right\| \cos\alpha \le \mathbf{1}_n^{\mathsf{T}}(t) \cdot (\mathbf{r}(t) - \mathbf{r}_t(t)) \tag{8.15}$$

C6: Plume impingement

$$\mathbf{1}_n^{\mathsf{T}}(t_i)\boldsymbol{\tau}(t_i) \le \sigma(t_i)\cos\beta \tag{8.16}$$

C7: Maximum rate of thrust change

$$\left|\sigma(t_i) - \sigma(t_{i-1})\right| \le \zeta \tag{8.17}$$

C9: Keep-Out-Sphere

$$\frac{[\mathbf{r}^{(k-1)}(t_i) - \mathbf{r}_t(t_i)]^{\mathsf{T}}}{\left\|\mathbf{r}^{(k-1)}(t_i) - \mathbf{r}_t(t_i)\right\|}[\mathbf{r}^{(k-1)}(t_i) - \mathbf{r}(t_i)] \le \left\|\mathbf{r}^{(k-1)}(t_i) - \mathbf{r}_t(t_i)\right\| - d_{9\mathrm{a}}(t_i) \tag{8.18}$$

$$\left\|\mathbf{r}^{(k-1)}(t_i) - \mathbf{r}(t_i)\right\| \le d_{9\mathrm{b}}(t_i) \tag{8.19}$$

C10: Station keeping

$$\left\|\mathbf{r}(t_i) - \mathbf{d}_{10\mathrm{a}}\right\| \le \epsilon_r \tag{8.20}$$

$$\left\|\mathbf{V}(t_i) - \mathbf{d}_{10\mathrm{b}}\right\| \le \epsilon_V \tag{8.21}$$

Cf: Final State

$$\mathbf{r}(t_{\mathrm{FP}}) - \mathbf{d}_{\mathrm{fa}} = 0 \tag{8.22}$$

$$\mathbf{V}(t_{\mathrm{FP}}) - \mathbf{d}_{\mathrm{fb}} = 0 \tag{8.23}$$

For a given attitude scenario and HP1 case, the chaser reaches the LVLH position defined in Table 8.1 at $t_{\mathrm{HP}}$, and maintains that LVLH position until $t_c = t_{\mathrm{HP}} + 300$ s, at which point the docking axis is exactly orientated towards that location. This is the end of Part 1 of the mission.

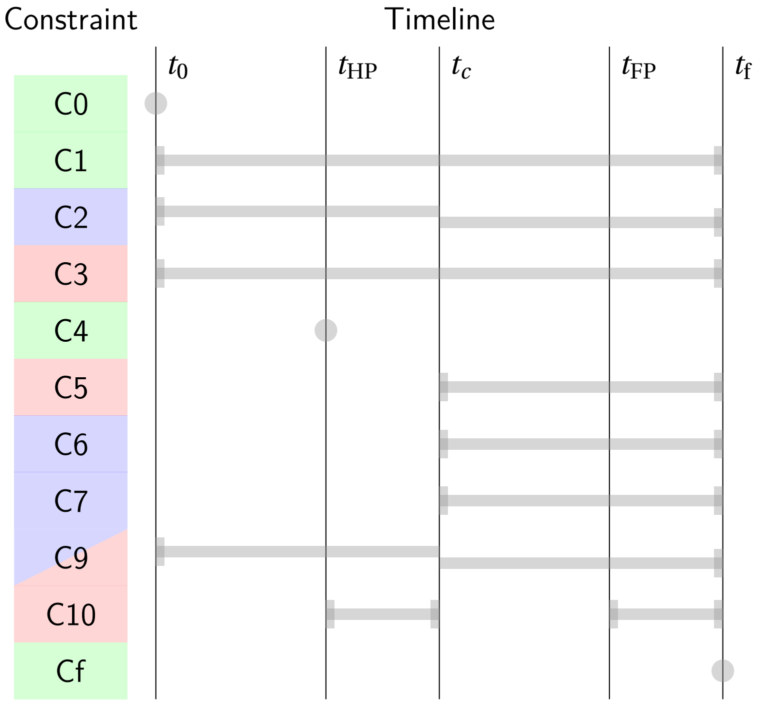


Figure 8.5: Timeline of active constraints: Green refers to linear equality constraints, blue for linear inequality constraints, red for conic constraints. Constraints C2 and C9 appear to break mid-way at $t_c$, because their parameters change.

From $t_c$ onwards, the chaser moves along with the docking axis vector, and continues Part 2 of the mission until $t_f$.

2. Time of flight: For a given time of flight, mConGAL is expected to produce fuel optimal trajectories. However, it is unknown for which value of $t_f$ and $t_c$, will we have the most fuel optimal solution. This calls for setting up an outer-loop to the optimization problem, which varies the time of flight so that a solution with minimal propellant consumption is found. This is expressed as:

**Problem 8:**

$$J_T = \min_{t_f} J(t_f) \tag{8.24}$$

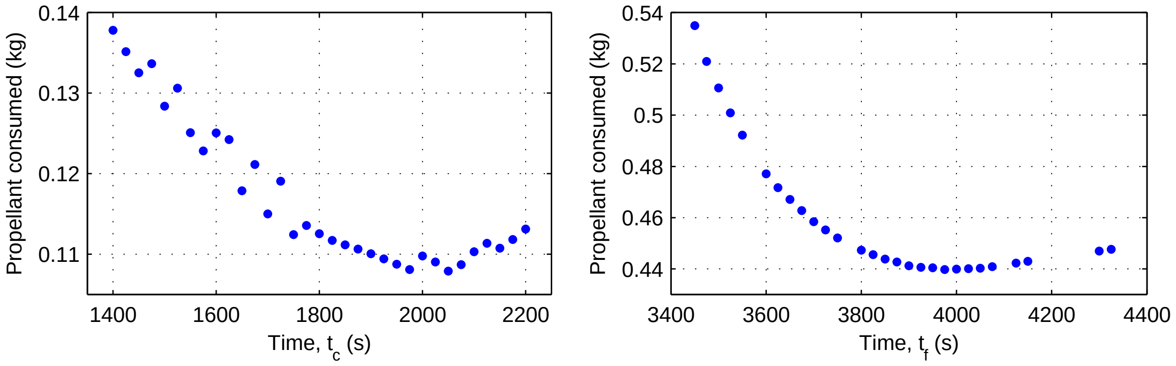
subject to

$$\{t_f\}_{\text{min}} \leq t_f \leq \{t_f\}_{\text{max}} \tag{8.25}$$

A similar problem set up can be shown for $t_c$. Note that the presence of a hold point (and the corresponding station-keeping) allows us to break our mission into two separate problems, and solve them individually to find the optimal time. For a given choice of HP1 (from Table 8.1), we shall solve the Problem 8 over $t_c$ first, and using the optimal value found, $t_c^*$, we shall re-solve the problem over $t_f$. This means that in the process of finding the optimal $t_f^*$, we are also finding the optimal the position of FP.

One may notice that Problem 8 is univariate, depending only $t_f$ (or $t_c$), as other variables will be kept constant. Due to the nature of Problem 7, one can expect Problem 8 to be non-linear as well. Further, we do not have any analytical expression and / or derivative information for the problem. These conditions restrict the available algorithms for searching the optimal time of flight. Seeing that we have a total of 9 mission cases to handle, each requiring two separate searches (each for $t_c$ and $t_f$), it is imperative that the algorithm is quick and reliable. This rules out stochastic methods as they often require tens or even hundreds of computations of the cost function, to reliably provide a solution. We are left with only one-dimensional search algorithms.

The most obvious choice would be to use a grid search algorithm, which samples the problem uniformly within the given range of time. It guarantees that a global minimum would be found but it is inefficient at finding it. Another option is the golden-section search, which successively narrows the range and finds the minimum with much fewer number of computations of the cost function. But, it requires that the problem be continuous and have only one minimum (*i.e.,* unimodal) between the bounds of the range. At this point in time, it is hard to ascertain if Problem 8 qualifies the conditions of golden-section search, as there is no mathematical proof to support / refute it. Although, taking hints from work by Gerth [2014], we can hope that the problem shall qualify the conditions. Through numerical examples, Gerth [2014] successfully demonstrates that such a problem can be solved using a golden-section search based method. However, one should also note that his and our convex optimization problem are substantially different in terms of intended application, underlying dynamics, nature and amount of constraints, and problem size. At best, his approach and results can be used as an indicator for our problem.



(a) Grid-search for Part 1 of mission.

(b) Grid-search for full mission.

Figure 8.6: Grid Test results for Envisat Mission 1a, sampled at 25 s. In (a), there is a general region where the minimum occurs but there are local variations, while in (b) not all results converge satisfactorily, resulting in gaps in the grid.

To further clarify the nature of the problem at hand, a full-scale grid search is carried out for one case of Envisat problem (with id: EM 1a from Table 8.1). The grid is sampled at 25 s, and a small section of the results for Part 1 of the mission and full mission are shown in Figure 8.6. It is clear that there is a general region where a (global) minimum occurs. However, there are numerous irregularities that make the problem far from ideal for use with golden-section search. In Figure 8.6a, $t_c^*$ is observed at 2050 s, but we see there are many local fluctuations which make the function non-unimodal. There could be two reasons for such behaviour: (1) the real solution of the problem shows such fluctuations (2) the iteration at which mConGAL converges shows fluctuation. Since we already saw minor fluctuations in results while discussing mean-scheme (see Section 7.3.1), it is likely that the behaviour is primarily, if not completely, due to mConGAL (and the solver used). In Figure 8.6b, which shows the result for the full mission, we see a more uniform behaviour with an undisputed minimum at 3975 s. However, there are gaps within the grid, where the solution fails to converge or a suitable solution is not found without violating constraints. These discontinuities can throw off the golden-section method to reach the exact minimum we are looking for.

In view of the above behaviour, and in the expectation that the remaining cases of Envisat Mission will show similar behaviour, it is decided to undertake a *combined* approach to finding the optimum time. Since there is a general trend, we shall start with a golden-section search that will help find a small region (of range 300 s) where a minimum is likely to occur. Then, to combat local fluctuations in that region, we shall make use of a grid-search, sampled at 25 s. This appears to be the best trade-off between effort and results, as compared to using either of those methods individually.

We now have all the information we need to solve the problem. The commonly used parameters for the problem are enlisted in Table 8.2. The chaser mass is updated to 1440 (instead of the 1588 kg defined initially for the 3-phase mission). For each mission case defined in Table 8.1, we shall run a combined approach search for an optimal time to Problem 8. In the process, we shall have a series of solutions for Problem 7. These solutions are accepted only if they converge within the specified tolerance, under 30 iterations, without violating any constraints. The golden-section search will cover a range of [600,3300] s for $t_c$ and narrow it down to a range of 300 s (give or take 100 s) for the Grid search. For $t_f$, the bounds are $[t_c^*+300, t_c^*+3300]$. The bounds are empirically chosen and are flexible, depending on the nature of solutions obtained.

Table 8.2: Input to the ME

| Parameter | Value | Units |
|---|---|---|
| | Mission inputs | |
| Step size | 5 | s |
| Convergence | $[5 \quad 0.1]$ | [m,m/s] |
| Maximum iterations | 30 | - |
| | Target inputs | |
| Initial orbit | 749.29 × 770.71 | km |
| Inclination | 98.15 | ° |
| Arg. of perigee | 90.00 | ° |
| | Chaser inputs | |
| Initial LVLH position | $[-100 \quad 0 \quad 0]$ | m |
| Initial LVLH velocity | $[0 \quad 0 \quad 0]$ | m/s |
| Maximum thrust ($t \leq t_c$) | 44 | N |
| Maximum thrust ($t > t_c$) | 2 | N |

## 8.3 ENVISAT MISSION RESULTS

On an average, 48 computations of Problem 7 had to be carried out for finding $t_c^*$ and $t_f^*$ for each of the remaining 8 mission cases (EM 2 a *to* d and EM 3 a *to* d). Barring the $t_c^*$ search for case EM 3b, which required a full scale grid-search due to large discontinuities, all cases were solved for successfully using the combined approach discussed above. The results of the final grid-search for all cases are presented in Figure 8.7. The graph compares propellant consumption against time of flight and has a section of results from EM 1a (in Figure 8.6) for comparision. The lower left quadrant shows results for $t_c$ and the upper right quadrant for $t_f$. One can draw the following conclusions from that figure, regarding the EM 2 group (marked with dots) and the EM 3 group (marked with circles):

1. Lower left quadrant / Part 1 of mission:

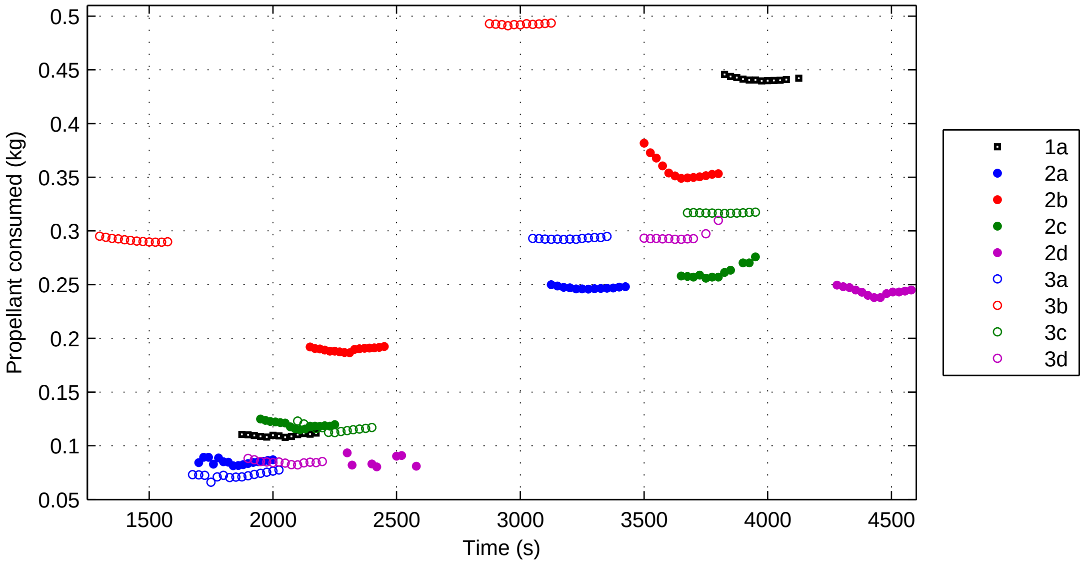


Figure 8.7: Part of grid-search solutions to 9 cases of Envisat Mission (EM) problem. Lower left shows results for optimization of $t_c$ and upper right for $t_f$

(a) The pair EM 2b and 3b (red markers in the figure) consume the most propellant, compared to other cases in their respective groups. This is expected because the motion from IP to HP1 in their cases is against the natural tendency of the chaser. However, the amount of propellant consumed is nearly twice of the other cases, hinting that there could be other factors at play.

(b) Compared to other cases, EM 3b takes much lesser time for the transfer, at 1500 s. This could be a possible reason for its high propellant consumption. But, then the question arises, why is the optimal found at 1500 s and not later in time. To answer this, we look at the full scale grid-search results (sampled at 50 s) in Figure 8.8. Results which did not pass as an acceptable solution (due to non-convergence / infeasibility) are shown at the bottom of the graph. Beyond 1500 s, we see nearly all results are unacceptable, although theoretically, these solutions should exist. This shows mConGAL's inability to find solutions for this case. We have no choice but to pick a solution from the first 1500 s, albeit an interpolation through the acceptable solutions points out that the minimum is probably in the first 1500 s.

2. Upper right quadrant / Full mission:

(a) The time interval for Part 2 of the mission ($t_f - t_c$) is within 1300 to 1600 s for most cases. This means, that the docking axis has precessed more than 180° once the chaser entered the approach cone. An outlier is EM 2d, where the chaser spends nearly 2000 seconds inside the approach cone, corresponding to a 300° change in orientation.

(b) Over entire time of flight, $t_f$, the EM 3 group's cases consume more propellant than corresponding EM 2 cases. This indicates that the constrained motion in orbital plane is more expensive than cross-plane for this mission!

(c) The EM 3 group take lesser or at most, equal time as compared to cases from the EM 2 group. This could further be a factor as to why the EM 3 group requires more propellant.

(d) In the range of solutions shown, the EM 3 group show near-constant propellant consumption over time of flight, showing some robustness in solution. The same cannot be said for the EM 2 group missions.

(e) The HP1 for 'a' cases (blue) lies in the cross plane along with IP, and is closest to IP, as compared to other cases. We expect that they would have the lowest propellant consumption. However, EM 1a takes one of the highest amount of propellant, 2a is less optimal than 2d and 3a is about the same as 3d. Since 'd' cases (purple) have their HP1 at farther distance (and lower semi-major axis) than 'a' cases, they should not have same or better propellant consumption.

(f) EM 2c consumes marginally more than EM 2a - this is qualitatively what one would expect, but considering that HP1 in 'c' cases (green dots/circles) are the farthest away from IP, the propellant consumption should have been higher. A similar point can be made for EM 3a and 3c.

(g) For cases EM 2b and EM 3b (red), where the hold point is located above the cross plane (farther from Earth), the propellant consumption is higher than others in the group. In contrast, cases EM 2d and 3d (purple), where the hold point is located below the cross plane, the propellant consumption is lowest.

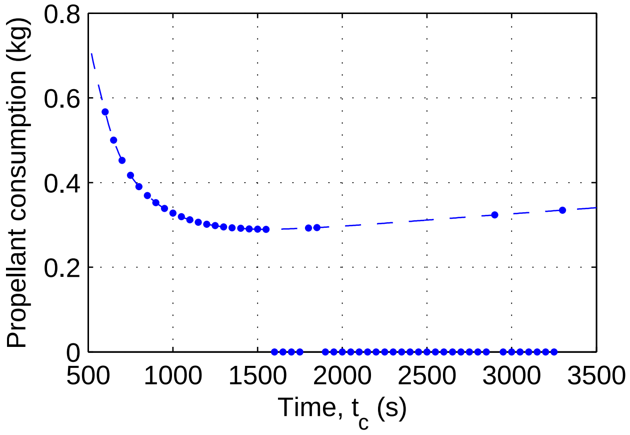


Figure 8.8: Full-scale grid-search solution to find $t_c^*$ for EM 3b. After 1500 s, there is a large discontinuity where mConGAL does not find an acceptable result.

The above conclusions are based on the set of results of the 9 different cases. Now, we shall delve into the specific, optimal solution we found and analyse it further. The results are presented in the following sub-sections and a summary is presented in Table 8.3.

### 8.3.1 Attitude Scenario 1

The results are shown in Figure 8.9. It consists of 6 plots, with the top four showing different views of the LVLH trajectory. The points IP, HP1 and FP are marked on the trajectory as well. As expected, the transfer from HP1 to FP happens along the negative H-bar axis. The bottom left plot shows the thrust profile for full mission, and the bottom right plot shows a zoomed-in version for Part 2 of the mission (where the thrust is constrained to 2 N). We already see a disadvantage of using the work-around for forced-motion constraint - there are numerous parts in time (between 2025 to 3200 s) where the thrust is zero or near-zero, thus allowing the chaser to freely drift towards the target (although, within the approach cone).

Also shown in the top four plots is the trajectory estimated by the Orbit Propagator (OP). It very quickly goes out of the view of the plots, with the final point nearly 600 m away. This is an enormous deviation, keeping in mind that the general scale of the rendezvous problem is less than 100 m. This is unexpected behaviour, especially because the integrators was upgraded to RK4 to avoid

Table 8.3: Primary results for 9 EM cases

| Mission id | Time (s) $t_c^*$ | $t_f^*$ | Iterations required | Propellant used (kg) | Docking axis precess (°) | Difference with OP (m) |
|---|---|---|---|---|---|---|
| 1a | 2050 | 3975 | 6 | 0.440 | 273 | 594 |
| 2a | 1900 | 3275 | 4 | 0.246 | 206 | 407 |
| 2b | 2300 | 3650 | 4 | 0.349 | 202 | 496 |
| 2c | 2075 | 3750 | 4 | 0.256 | 251 | 509 |
| 2d | 2400 | 4450 | 19* | 0.240 | 307 | 604 |
| 3a | 1875 | 3175 | 3 | 0.292 | 195 | 384 |
| 3b | 1550 | 2950 | 5 | 0.491 | 210 | 314 |
| 3c | 2250 | 3825 | 7 | 0.316 | 236 | 541 |
| 3d | 2100 | 3650 | 12* | 0.293 | 232 | 489 |

* Required mean-scheme to converge

such large deviations. But, using the other three integrators or reverting back to trapezoidal rule, provides worse or no result. This appears to be a characteristic of the problem at hand, rather than a shortcoming of the new integrators.

### 8.3.2 ATTITUDE SCENARIO 2

The four cases considered for AS 2, are shown in Figures 8.10 and 8.11 (and their sub-figures). We shall use the same arrangement of plots used for EM 1a. In many ways, the results are similar between the cases in this group: In Part 1, there is a clear, two-burn manoeuvre between IP to HP1 (with occasional corrections in between to avoid the KOS), followed by small burns to sustain and exit the SK1 manoeuvre. In Part 2, there are two large burns at $t_c$ and $t_{\text{FP}}$, with a near-continuous burn in between these two points. This burn is to ensure that the chaser is able to curve its trajectory around Envisat, making an arch that subtends nearly 160°, before it makes initial contact with FP. Lastly, there are some minor burns to ensure the rotating FP is followed for the last 300 s of flight (which adds another 45° to the arch of the trajectory). By the end of the flight, HP1 and FP are nearly diametrically opposite to each other and the chaser's trajectory resembles a question-mark that is oriented in different directions for the four cases (see top-right plot of Figures 8.10a, 8.10b, 8.11a).

EM 2b behaves slightly differently, as instead of a nearly impulse burn upon reaching HP1, we see a 600 s long burn of low magnitude (1 - 1.5 N). This results in a rather peculiar shape for the transfer from IP to HP1 (see middle-right plot of Figure 8.10b). It is the primary cause for the high propellant consumption, thus answering the concern of point (1a) in the list of conclusions drawn from Figure 8.7 on Page 126. It is unclear why such an inefficient solution is found to be the optimal for this case.

EM 2d (Figure 8.11b) also requires a special remark: It was unable to converge on its own, and even with mean-scheme (discussed in Section 7.3.1), it required nearly 20 iterations to converge. Its docking axis precesses over 300 °, thus HP1 and FP are in adjacent quarters (see top-right plot in the said figure). The final difference with OP is nearly 600 m, the highest in the EM 2 group. Despite these issues, it consumes the lowest propellant in the group, which conflicts with our expectation for EM 2a.

### 8.3.3 ATTITUDE SCENARIO 3

The results of the four cases are shown in Figures 8.12 and 8.13 (and their sub-figures). Here too, we shall use the same arrangement of plots used for EM 1a. There are some common characteristics among the four cases, and with the results of the EM 2 group as well: The thrust profile for Part 1 includes a two-burn manoeuvre from IP to HP1, followed by burns for sustaining and exiting SK1 at

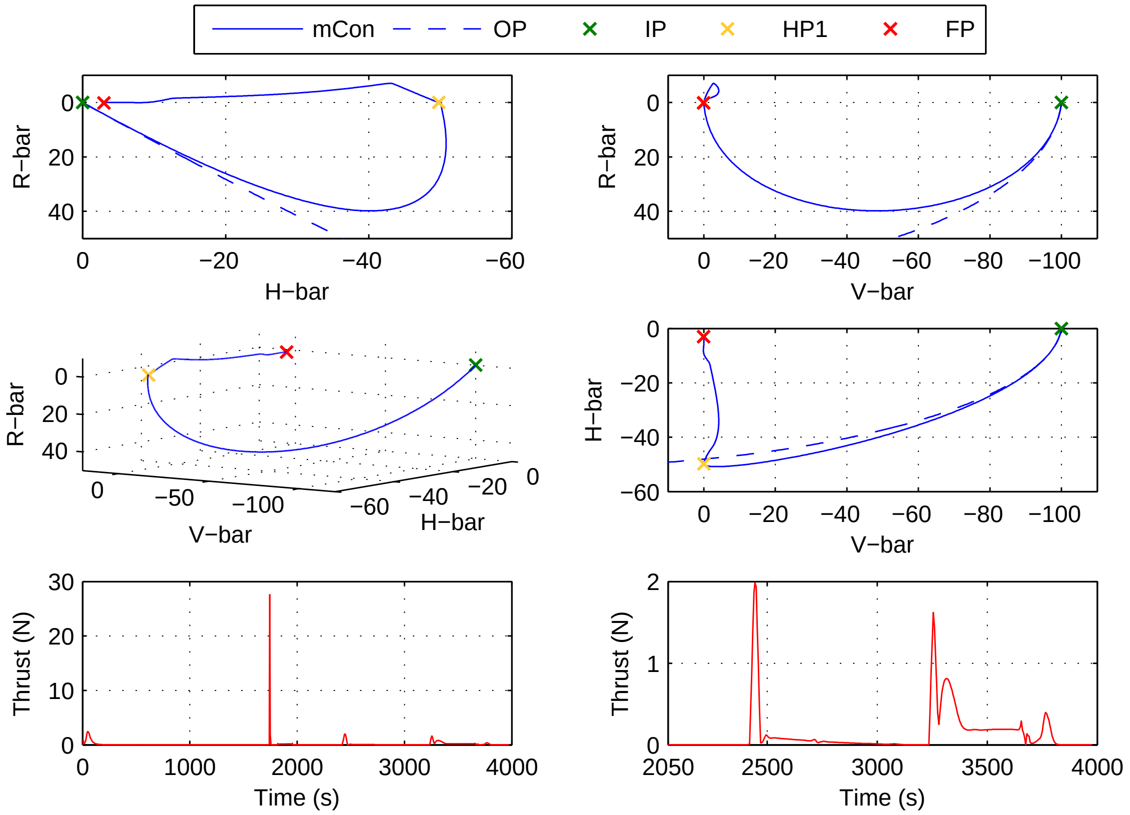
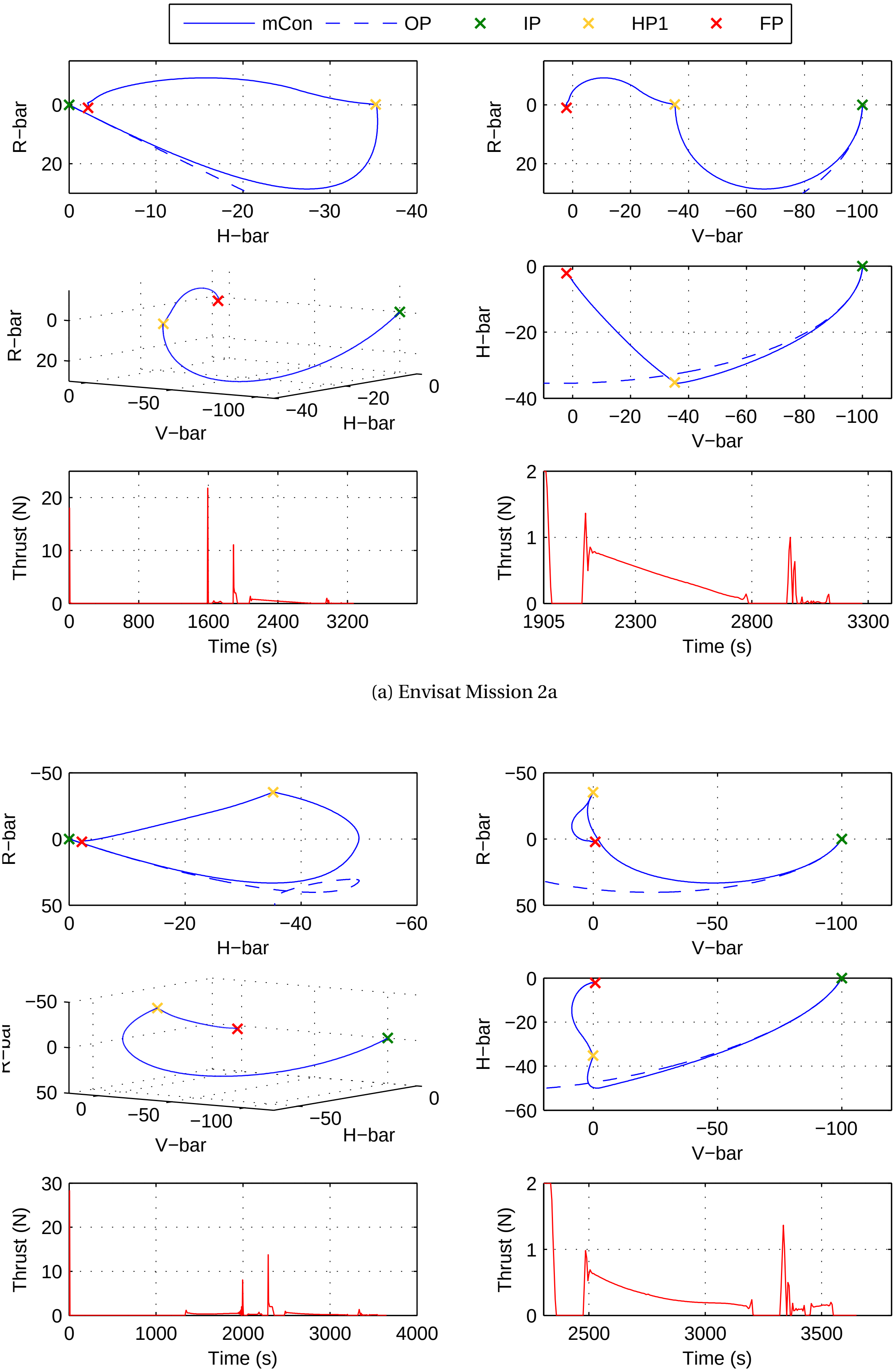


Figure 8.9: Attitude Scenario 1: Results for Envisat Mission 1a.

$t_c$. Part 2 has two burns, each at $t_c$ and $t_{\text{FP}}$, with a near-continuous burn in between. The last 300 s upto $t_f$ sees the chaser make minor burn to sustain the rotation motion of FP. Overall, the chaser travels nearly 200° around the target, in a question-mark resembling trajectory (see top-right plot of Figures 8.12a, 8.12b, 8.13a, 8.13b).

The resemblance with the EM 2 group results extends to the peculiarities as well: EM 3b requires a 600 second continuous burn of low magnitude before reaching HP1, which is the prime cause for its high propellant consumption. Further, EM 3d does not converge on its own and requires 12 iterations (with help of mean-scheme). It also has the lowest propellant consumption of the group, although it is equal to EM 3a's performance.

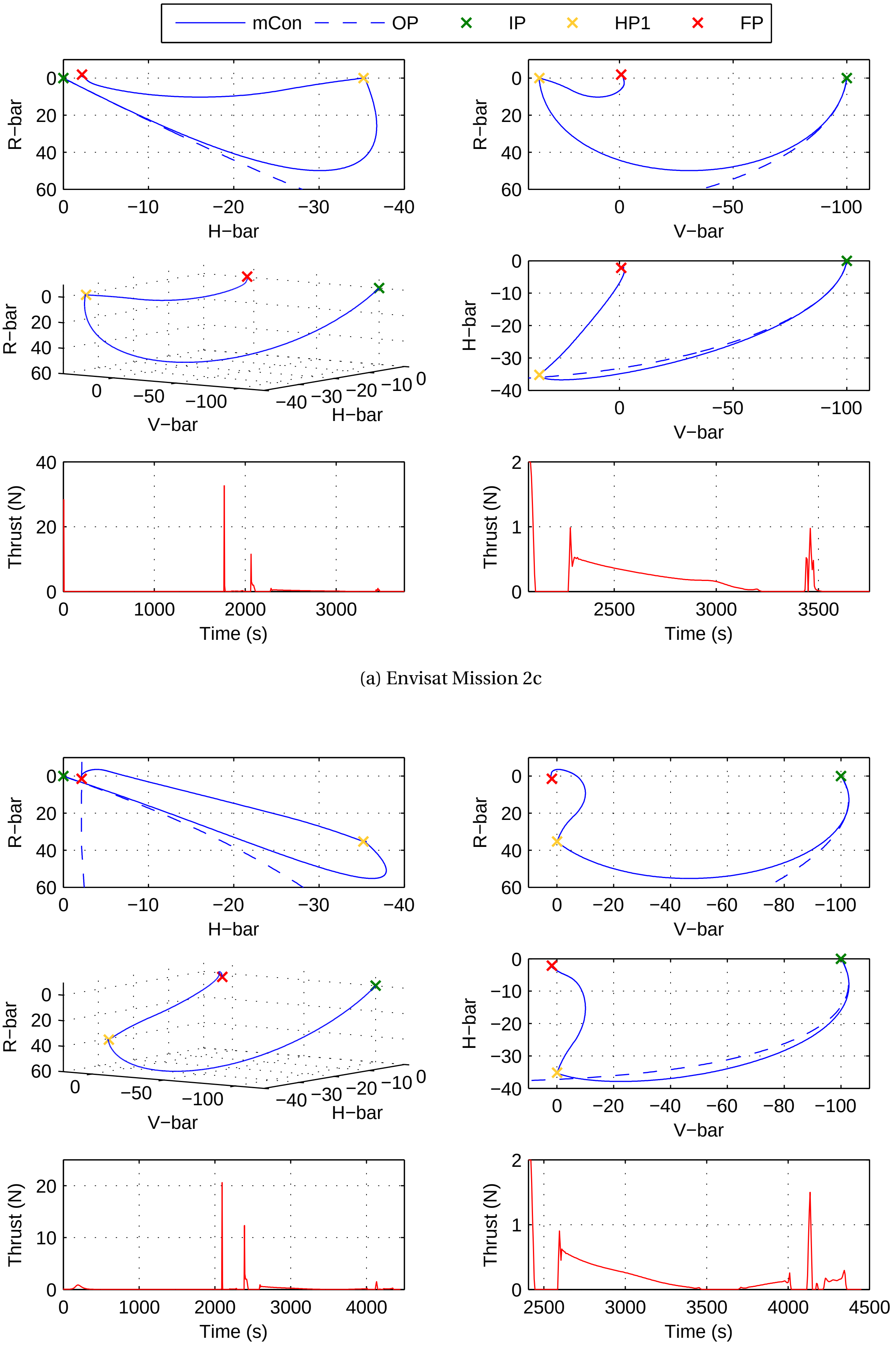
There are also some substantial differences with the EM 2 group. There is no motion in the cross-plane, as IP, HP1, FP and the rotating docking axis are all placed in the orbital plane. Additionally, the burns in Part 2 are consistently more expensive than corresponding cases in EM 2 (in the neighbourhood of 0.05 kg mostly). We already saw the effect of this in Figure 8.7 (refer point (2b) in the list of conclusions on Page 126). Analysing the LVLH trajectories of EM 3 group, one sees that for Part 2 of the mission, the chaser is moving in the opposite direction to its natural tendency. See the top-right plot of Figure 8.12a for instance: The natural tendency of the chaser positioned in negative R-bar quadrants is to move farther away from the target, and in the direction of negative V-bar. However, the chaser is being forced to do the exact opposite - move towards the origin along positive R-bar and positive V-bar. On the other hand, when the chaser is in positive R-bar quadrants (see Figure 8.12a), the tendency is to move in positive R-bar and positive V-bar direction. This is away from the target and in opposite direction of docking axis precession (which is anti-clockwise from this direction of viewing). These *unnatural* motions require more effort than if the chaser were moving in the natural direction. Now, focussing back on the motion of the EM 2 group, we see



(a) Envisat Mission 2a

(b) Enivsat Mission 2b

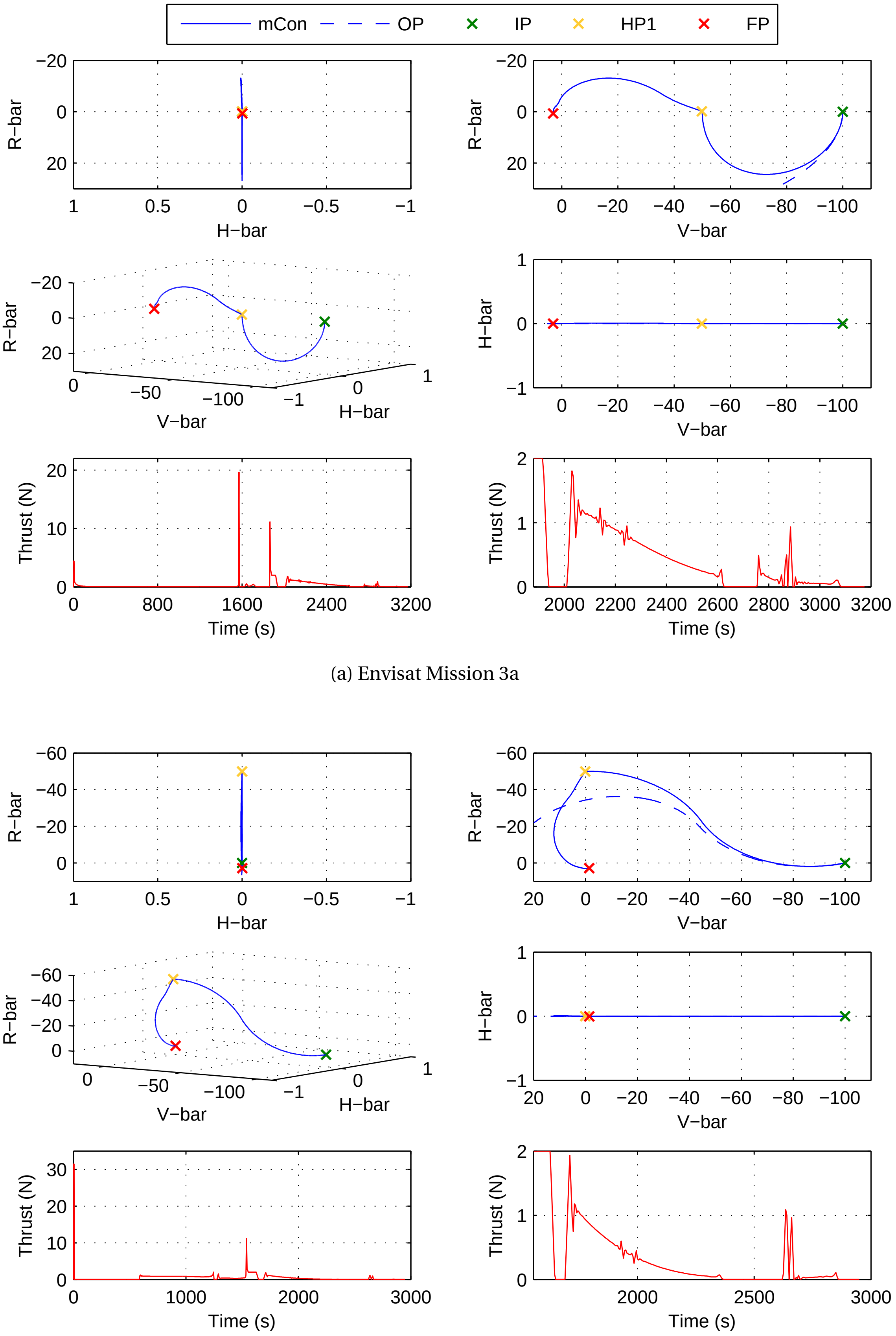
Figure 8.10: Attitude Scenario 2: Results for Envisat Mission 2a and 2b.



(a) Envisat Mission 2c

(b) Enivsat Mission 2d

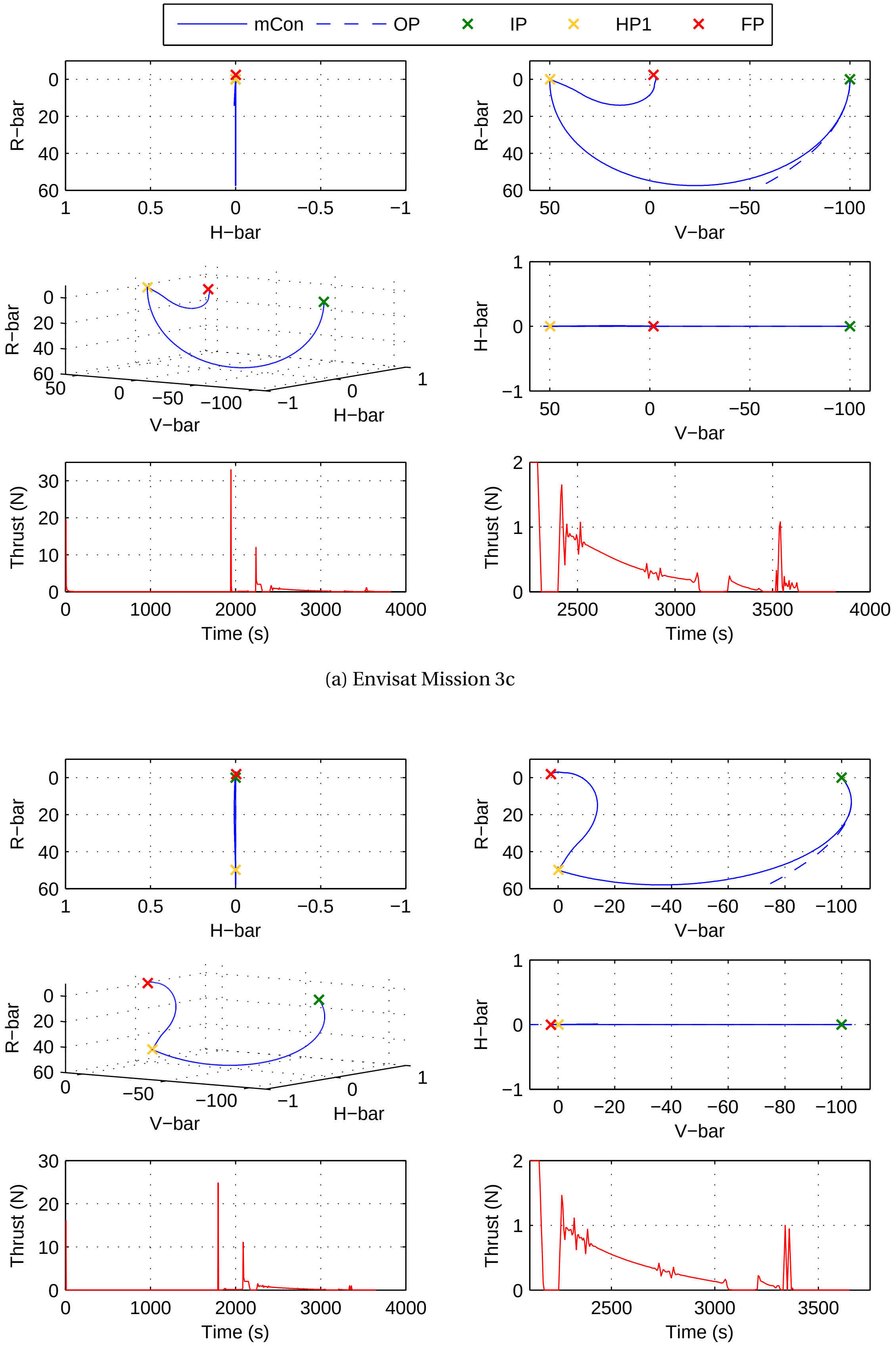
Figure 8.11: Attitude Scenario 2: Results for Envisat Mission 2c and 2d.



(a) Envisat Mission 3a

(b) Enivsat Mission 3b

Figure 8.12: Attitude Scenario 3: Results for Envisat Mission 3a and 3b.



(a) Envisat Mission 3c

(b) Enivsat Mission 3d

Figure 8.13: Attitude Scenario 3: Results for Envisat Mission 3c and 3d.

the same behaviour of motion against the natural tendency - however, in addition to it, there is also the effect of the chaser's cross-plane motion. A body in cross-plane has a natural tendency to move towards the origin (and the orbital plane), which is favourable for our mission. This partially reduces the cost for the transfer in EM 2 missions, resulting in lower propellant consumption than EM 3.

Another difference with the EM 2 group is that the thrust profile for Part 2 of the mission (bottom right plot in the said figures) appears to have a lot more variations (or fluctuations, so to say). These variations are within the acceptable rate of change of thrust magnitude. However, the presence of these variations indicates that the transfer is more complicated to maintain than compared to those in the EM 2 group.

## 8.4 Further analysis of Envisat results

The results of Problem 7 and 8 for the 9 cases raise some important questions that need answering, if mConGAL is to be used for real-life applications. Firstly, mConGAL's results show substantial divergence from OP results, even though an improved integrator is used. Also, on numerous calls, an acceptable solution was not found - either due to slow convergence (despite help from mean-scheme) or infeasible solutions (despite common belief that a solution should exist, especially for Part 1 of the mission). To check robustness of the method and to gauge the effect of input parameters on the solution, a sensitivity analysis is carried out. Also, a path-tracking algorithm is developed and implemented for improving accuracy of results.

To avoid redundant analysis and owing to time constraints, the prime focus of the upcoming sections is the EM 3c case. The reason being that the EM 3 group, where the docking axis precesses in the orbital plane, has a more complex thrust profile than the EM 2 group, where there is cross-plane motion as well. It also allows easy visualization on paper, as the chaser trajectory is restricted to one plane. Where it is deemed necessary, final results of the remaining cases are also presented.

### 8.4.1 Sensitivity analysis

Sensitivity analysis is the computation of the effect of changes in input values or assumptions on the outputs [Morgan and Henrion, 1990]. By investigating the relative *sensitivity* of model parameters, the user becomes knowledgeable of the relative importance of parameters in the model. A robust method is one where the sensitivity is minimal.

The general process of sensitivity analysis involves running the model for a range of randomly dispersed inputs and analysing the results mathematically, graphically or statistically for insight. A preliminary analysis showed that, mathematical analysis is best suited for linear models with little or no interactions between inputs, and thus, can be ruled out for use with our current problem. Statistical methods require uncertainty models for inputs, but due to their unavailability, we are restricted to graphical analysis. The advantage of graphical analysis is that it gives a quick visual assessment of the input's influence. It is often recommended as a first step in sensitivity analysis as it helps in identifying potentially complex dependencies, and gives the user a feel of the model. Its disadvantage is that it fails to give concrete relations between parameters and makes it hard to rank them in order of influence.

Table 8.4 lists the six input parameters considered for the test. These parameters are most likely to be uncertain for the mission, as information about Envisat's state and rotational motion is still not fully known. Also shown in the table, are their nominal values and the range considered for the test. To account for interactions between the inputs, they are grouped into three sets and are to be varied simultaneously. The selected range is substantially large for lack of more accurate ranges.

The setup for the test involves solving Problem 7 using nominal values of remaining parameters (including $t_c^*$ and $t_f^*$) while varying the selected inputs. For each set, 50 samples (uniformly distributed over the range) are computed and their results (effect on propellant consumption) are plot in Figure 8.14. The nominal value is a scatter point marked with a red-circle at the centre of the cor-

Table 8.4: Sensitivity analysis inputs

| Set | Parameter | Unit | Nominal | Range |
|---|---|---|---|---|
| S1 | Position of IP along V-bar | m | -100 | [-80,-120] |
| | Position of HP1 along DA | m | 50 | [40,60] |
| S2 | Precession angle ($\theta_p$) of DA at $t_c$ | deg | 90 | [80,100] |
| | Rotation angle ($\theta_r$) of DA at $t_c$ | deg | 180 | [170,190] |
| S3 | Approach cone half-angle ($\alpha$) | deg | 10 | [7,13] |
| | Precession rate ($\phi_p$) of DA | deg/s | 0.15 | [0.1,0.2] |

responding sub-figure, while the results from the sampling are shown by the remaining plot marks. Unacceptable results are marked with a green cross. The following conclusions can be drawn from the results:

1. Set 1: In Figure 8.14a, we see that a solution is found for nearly all the sampled points. There is a strong direct relation between propellant consumption and position of HP1, as the propellant consumed varies nearly ±14%, over the range of HP1. On the other hand, there is a weak direct relation between IP and propellant consumed, as there is very little variation (about 0.6%) horizontally in the plot.

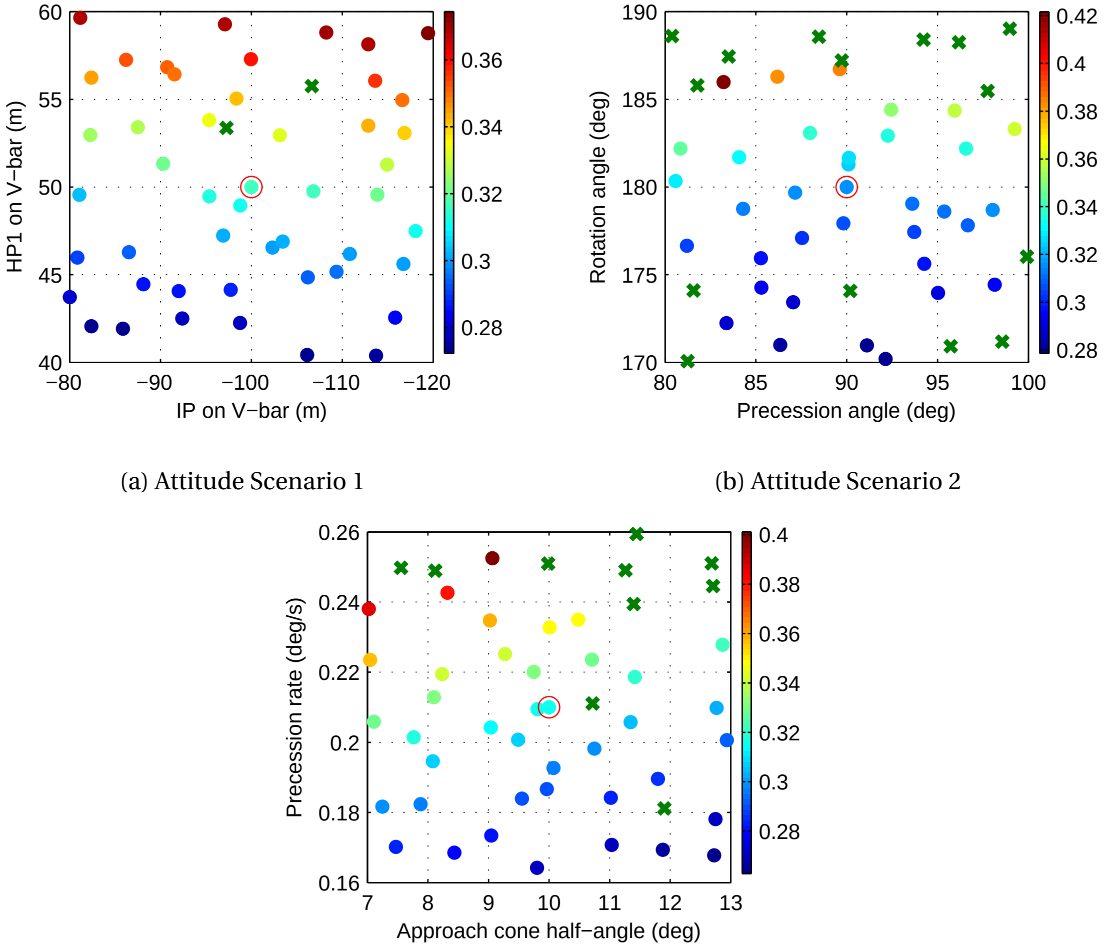
2. Set 2: In Figure 8.14b, there are a lot more unacceptable results as compared to the other sets. These appear more frequently as one moves farther away from the nominal value. There is a strong direct relation between propellant consumed and rotation angle. The relation also appears to be non-linear as the effect is a lot more pronounced for higher values of rotation angle. As for precession angle, there is a weak relationship that is symmetric around the nominal value. This is expected because ideally there should be no preference of positive or negative H-bar for the docking axis - it should cost the same if the axis were on either side of the orbital plane.

3. Set 3: In Figure 8.14c, using nominal $t_c^*$ and $t_f^*$, there are numerous unacceptable results at the top of the plot, where precession rate is high. On the other hand, nearly all samples have an acceptable solution at lower precession rate. Further, looking at the range of propellant consumed, one sees a strong direct relation with precession rate. With respect to the approach cone half-angle, there is a weak inverse relation that is observed. The presence of two unacceptable results not too far away from the nominal value is inexplicable at the moment.

Over all, we can conclude that solution by mConGAL is moderately robust to variations in nominal values. There are occasional unacceptable results close to nominal values, which are hard to explain. As the inputs vary substantially from nominal, mConGAL is not able to provide acceptable solutions. Although, most of this could be attributed to infeasibility of problem for given $t_c^*$ and $t_f^*$. There is a strong effect of HP1, rotational angle and precession rate on the propellant consumed. For these inputs, the smaller the value the lesser propellant is consumed. There is weak direct correlation with IP and precession angle, while a weak inverse relation exists with approach cone half-angle.

### 8.4.2 PATH-TRACKING

The equations of motion used in mConGAL are only an approximation of the actual dynamics that affect the spacecraft, and hence there is always a deviation between the mConGAL computed trajectory and OP trajectory. The improvement in integrator, discussed in Section 7.3, is a means to reduce this deviation. However, it is surprising to see a large deviation for the EM cases, reaching



(a) Attitude Scenario 1

(b) Attitude Scenario 2

(c) Attitude Scenario 3

Figure 8.14: Sensitivity analysis: Effect of variation in mission parameters on propellant consumption for EM 3c. Nominal case is marked at centre with red circle; unacceptable results are shown by green crosses.

nearly 600 m at the end of flight. This behaviour will probably get worse if one also includes perturbations in the system. This calls for a path-tracking algorithm which reduces the deviations to within acceptable limits. Two methods are considered and implemented in the following subsections.

### RE-COMPUTING ALGORITHM

This method involves frequently re-solving Problem 7 on-board with the updated chaser state as the new initial state. Consider that a guidance solution is computed when the chaser spacecraft has an initial state of $\mathbf{x}_{0,i}$. As the spacecraft proceeds towards the chaser, it drifts from the computed trajectory by an amount $\epsilon_{\mathrm{RC}}$. This is set to trigger a re-computation of the optimization problem with its current state as the new initial state $\mathbf{x}_{0,i+1}$. This re-computation process continues till the end. Figure 8.15 illustrates the concept.

Note that the actual trajectory which the chaser ends up taking will not be the same as the initially computed optimal trajectory or the trajectory by OP, but somewhere in between. If a solution is found from the updated initial state, one can be assured that all relevant constraints shall be met. We use the term *relevant* because, based on the updated initial state, a slightly modified set of constraints may be acting on the chaser.

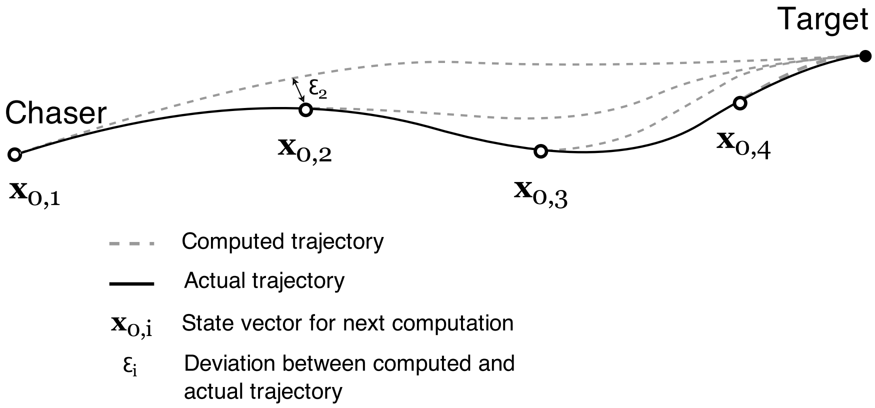


Figure 8.15: The deviation between the computed trajectory and actual trajectory calls for a re-computation of the solution from an updated initial state.

The actual implementation of the Re-Computing Algorithm (RCA) is carried out by an outer loop to Problem 7 that compares computed trajectory with trajectory from OP, finds the exact state where the decided drift tolerance is violated, passes that state to Problem 7 with relevant modifications to constraints for re-computation. The resulting actual trajectory would be a stitch-work of numerous runs of Problem 7 such that eventually the OP trajectory matches with it.

Four values for drift tolerance are chosen for analysis: [10, 1, 0.1, 0.01] m; their results are plot in Figure 8.16. It is found that none of the four options are able to make the computed and OP trajectories to converge. With varying amounts of success, all four options succumb to non-feasibility of problem from one of the updated initial states. They are discussed as follows:

1. Consider Figure 8.16a, where a drift tolerance of 1 m is applied. The RCA computed trajectories are shown in solid lines, while the corresponding OP trajectories are shown in dotted line. As usual, the target is shown as a black dot and the initial chaser state as green dot. We see that the RCA is able to make one re-computation (notice the point where the red-dotted line starts) after which it stops, as it finds the problem to be infeasible. Small variations ($\pm 0.1$) to drift tolerance in the hopes of bypassing the in-feasible region did not give better results.

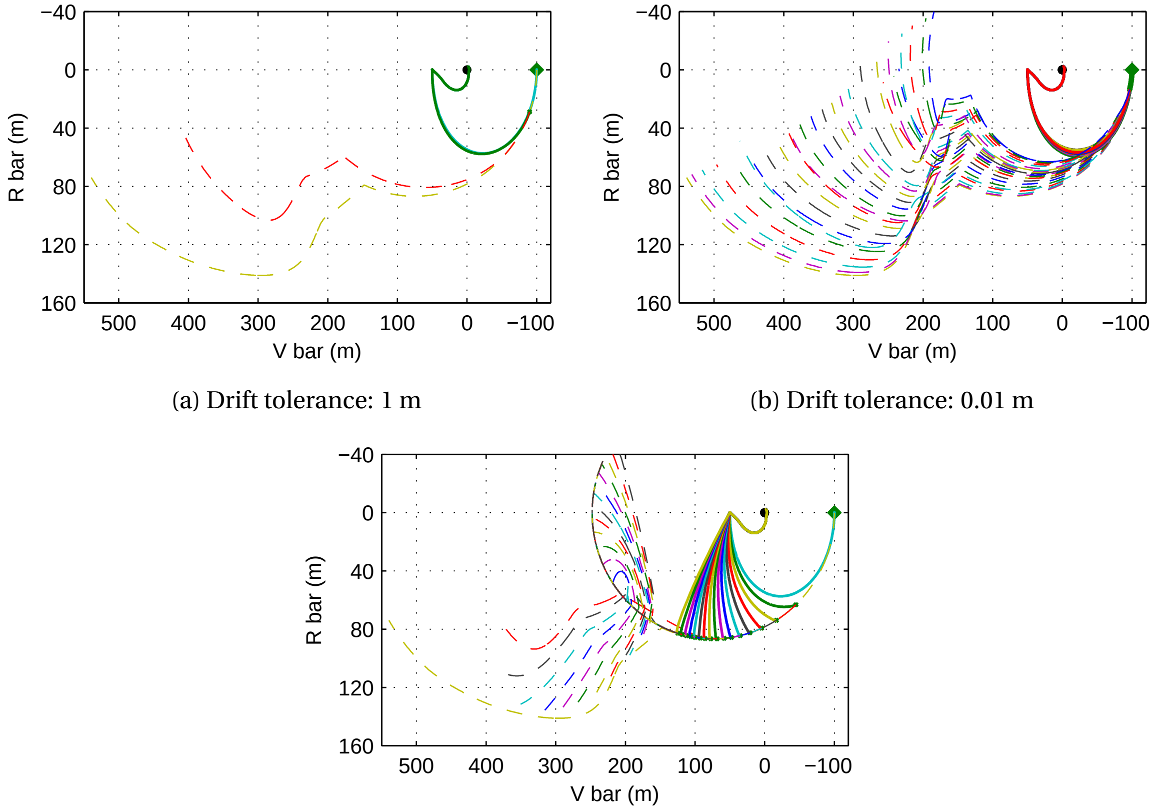
2. A similar behaviour is noticed in Figure 8.16b which uses a drift tolerance of 1 cm. Some 27 re-computations are made but due to the small drift tolerance, the chaser's updated initial state does not travel too far away from the original initial state. A re-computation was required every 15 seconds of flight time, which might not be an option for real on-board applications (as, right now it takes nearly 45-130 seconds for mConGAL to solve the problem).

3. Figure 8.16c shows the result for drift tolerance of 10 m. Compared to the previous two results, we see a distinct behaviour as the updated initial states lie almost entirely on the very first OP trajectory. This shows that for large drift tolerances, RCA has little or no effect to direct the chaser towards computed trajectory. Once again, numerous updated states are found to be acceptable but eventually RCA hits a (seemingly normal) state from which it is unable to find a feasible solution.

4. Plot for drift tolerance of 0.1 m is not shown, because the RCA is unable to carry out even one re-computation! Clearly, this is an issue because

A similar analysis was carried out on the remaining 8 cases of Envisat Mission, but the results are just as poor and RCA is unable to solve the problem. This behaviour is surprising as the points at which RCA (and thus, mConGAL) finds the problem to be infeasible show no special characteristic that could cause infeasibility. We have noticed some inexplicable in-feasibilities in Chapter 7 and Sections 8.3 and 8.4.1 as well, so RCA's failure could be just another manifestation of ConGAL's unreliable nature. Further analysis with RCA is abandoned as no leading insights are found.

This grossly reduces mConGAL's ability for autonomous use because it is unable to produce a reliable result in case of contingency. RCA does not look like the way to take in achieving closed-loop guidance.



(a) Drift tolerance: 1 m

(b) Drift tolerance: 0.01 m

(c) Drift tolerance: 10 m

Figure 8.16: Results from re-computing algorithm for path tracking. OP results (dotted line) get progressively closer to computed trajectory (solid line)

### PID CONTROLLER

The other method considered for path-tracking is a feedback controller, that generates corrective burns such that the actual trajectory remains as close as possible to the computed trajectory. The burns are generally a function of the deviation between the current state and the computed state. This method ensures that the resulting trajectory matches the original computed solution, thus meeting all state constraints. However, the corrective burns are not subjected to thrust constraints, thus it is possible that those constraints are not met. Figure 8.17 illustrates the concept.

The choice of the controller is dictated by numerous factors; however, owing to non-linearity of the problem and keeping in mind that this is a brief, proof-of-concept analysis, it is decided to employ a manually tuned Proportional-Integral-Derivative (PID) controller. The control value, that is used to correct the trajectory, is a weighted sum of the position error of the system, its integral over time and current derivative. A general schematic of the feedback mechanism is given in Figure 8.18, where **e** is the position error between computed state (from mConGAL) and state as propagated by the OP. In response, the controller generates a control value **u** that, along with the thrust profile

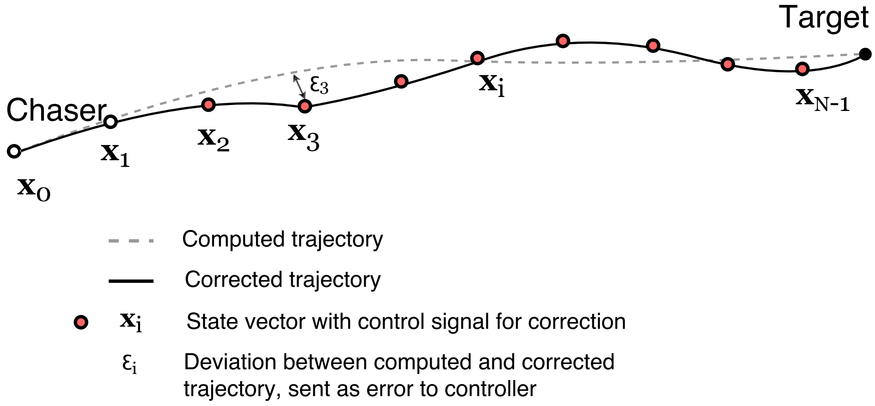

Target
Chaser
ε3
x0
x1
x2
x3
xi
xN-1
Computed trajectory
Corrected trajectory
xi State vector with control signal for correction
εi Deviation between computed and corrected trajectory, sent as error to controller

Figure 8.17: The deviation from computed trajectory is reduced by making small corrective burns along the way.

(from mConGAL), is fed to the OP. Note that this does not guarantee an optimal control solution. The choice of notation is kept consistent with standard practices, to avoid confusion. The reader is suggested the work by Aström and Murray [2010] for further information about PID controllers.

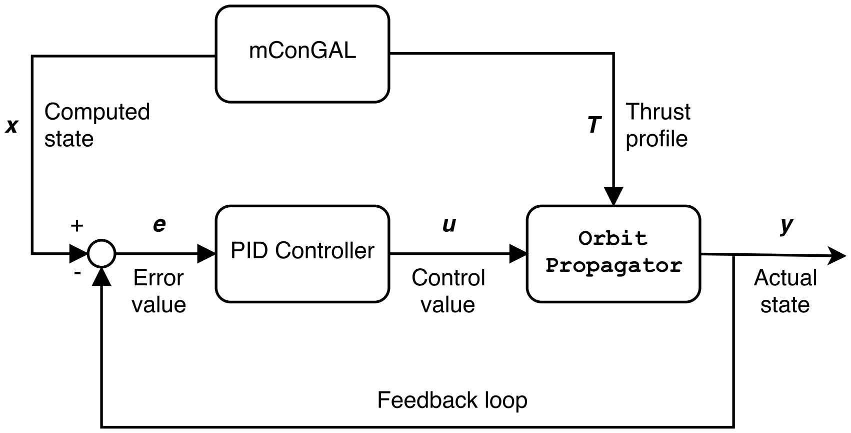
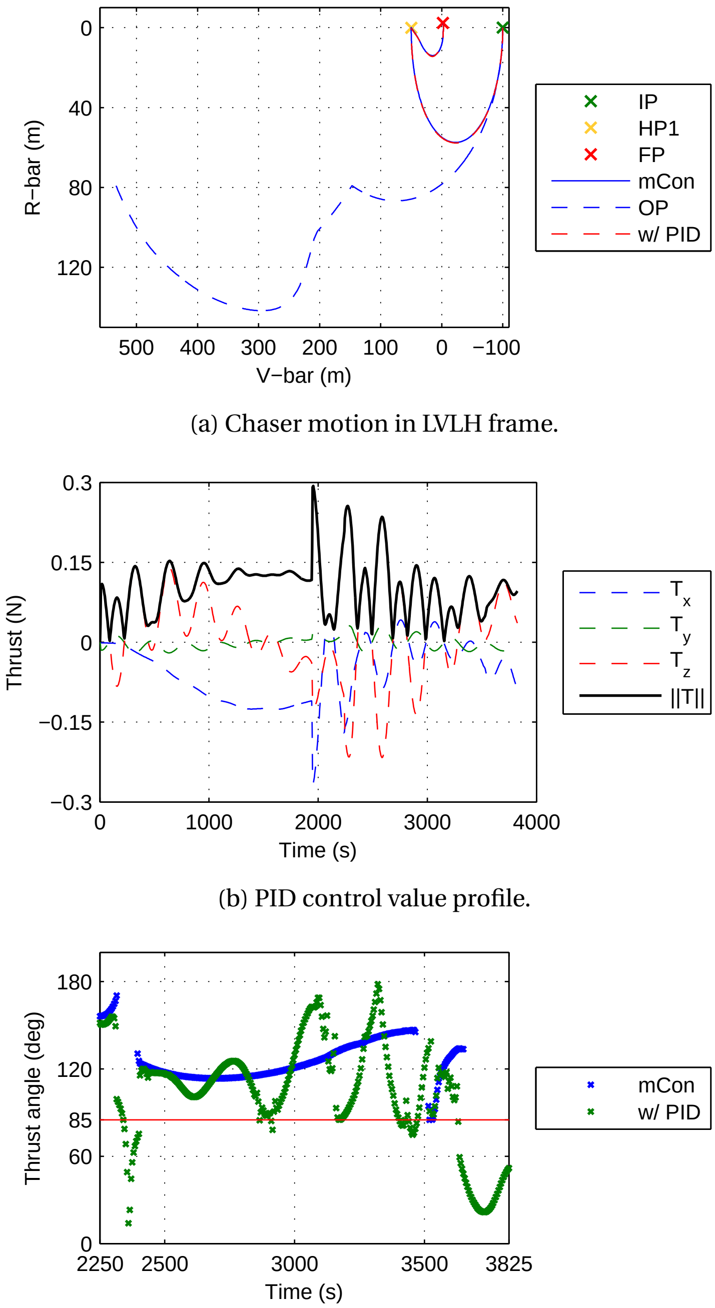


Figure 8.18: Schematic of PID control feedback mechanism.

The controller is tuned to ensure that the maximum position error at any given point is less than 1 m. This is achieved by setting the PID gains to [0.4, 0.0001, 20]. It is found that using these gains, a local minimum is achieved in terms of position error and the propellant expense for the corrective burns. The results are shown in Figure 8.19.

In Figure 8.19a, we see that the trajectory by OP deviates from mConGAL trajectory, but with the help of PID, the two trajectories overlap. In fact, the maximum deviation between the two is 0.5368 m, much less than the 1 m we sought to achieve. Figure 8.19b shows the variation in control value that is used to achieve the result. Note that the magnitude of the corrective burns does not exceeds 0.3 N, indicating that the corrections are minute. However, they are present through the entire mission, so their effect on propellant consumption is substantial - an additional 0.0939 kg of propellant is consumed, which is a 30% increase over mConGAL result! The PID results are able to satisfy all constraints of the mission except the constraint on plume impingement angle. It is required that the thrust be directed atleast 85° away from the target during Part 2 of the mission. Figure 8.19c shows the plume angle with respect to docking axis, and we can see that with use of PID controller, the angle drops below 85° at a few places. Barring this, it can be concluded that use



(a) Chaser motion in LVLH frame.

(b) PID control value profile.

(c) Variation of thrust plume angle; values above red line are acceptable as per constraint C6.

Figure 8.19: Results from PID control to EM 3c.

of PID controller is a success that fares much better than RCA. Similar results are observed for the remaining 8 cases of Envisat mission.

We now shift focus to the effect of perturbations. As discussed in Chapter 3, only $J_2$ and atmospheric drag are expected to have any notable effect on the results. Now that we consider only the final approach phase, the effect of these perturbations will be minute as well. Although Liu [2013] suggests ways to include both of these perturbations in the equations of motion of ConGAL, we are unable to replicate the results with satisfactory results. Primarily, it is found that in the presence of even small values of eccentricity, the results diverge from expected results. Due to time constraints, it is not possible to foray into its details, subjecting it to same validations tests and analysis as shown in Chapter 7. However, a work around has been improvised that allows us to compare and correct the effects of perturbations, despite propagating the chaser only in spherical gravity.

The work around is based on the assumption that the guidance solution and the chaser state as seen in the LVLH frame is nearly identical for cases when only spherical gravity is acting and when perturbations are present too. This is a valid assumption to start with, because perturbations affect Envisat and the chaser nearly equally (especially for small time of flight and small distance between the chaser and target). There will be a substantial difference from the nominal state as seen in the inertial frame (for example, changes in semi-major axis due to $J_2$ can be as high as 27 km), but since both the chaser and target experience these perturbations, their effect is barely visible in the LVLH frame. Thus, the guidance solution should be similar for the two cases of dynamics. Using this assumption, we undertake the following steps:

1. Convert nominal mConGAL solution (both thrust vector and chaser state) from inertial to LVLH frame. Here, nominal refers to spherical gravity field.
2. Generate perturbed trajectory of Envisat using OP.
3. Using the LVLH state and thrust from Step 1 and Envisat state from Step 2, we back-calculate the inertial values. This is the approximate solution in perturbed dynamics.
4. The obtained state and thrust is fed as input to the feedback mechanism. And the OP is set to include the perturbations in its dynamics.

In other words, we dress-up the nominal solution as an approximate solution to perturbed dynamics. Then with the help of OP and the feedback mechanism, we strive to match this solution while subjecting the chaser to perturbed dynamics. At best, the nominal solution would be an optimal solution to perturbed dynamics, and at worst, it would be one of the many possible solutions in perturbed dynamics. For minute perturbations, it is likely the nominal solution is the optimal solution as well. A schematic of the process is shown in Figure 8.20

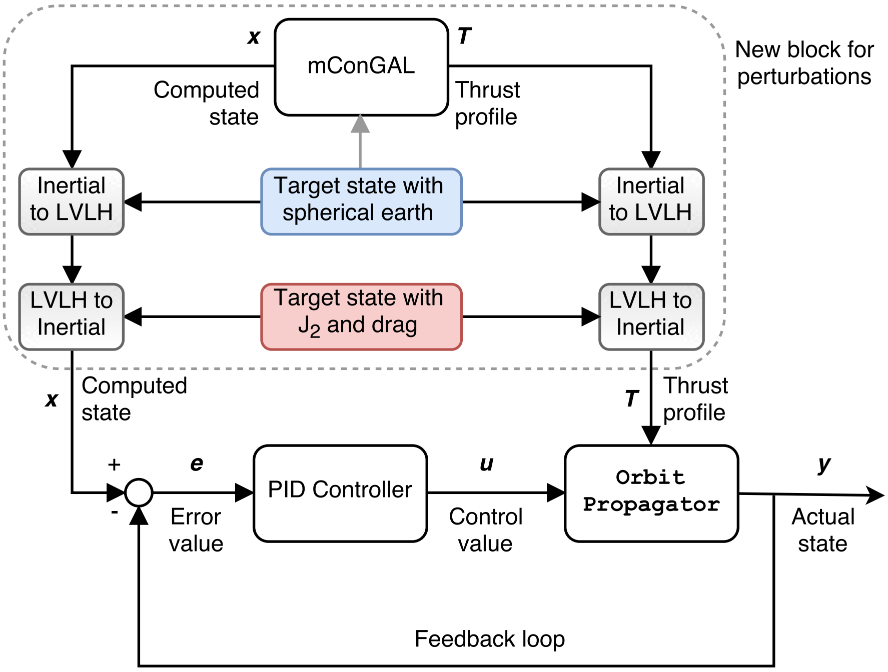


Figure 8.20: Schematic of PID control feedback mechanism for including perturbations.

The controller is again tuned to ensure that the maximum position error at any given point is less than 1 m. Not too surprisingly, this is achieved by setting the PID gains to [0.4, 0.0001, 20], the same as for spherical gravity. The results from the run are identical to the the ones presented in Figure 8.19 (with differences in fourth significant digit after decimal). But, the difference now is that solution holds despite the effects of perturbations. For instance, see Figure 8.21 which plots cross-plane motion of the chaser. Notice that there is nearly no cross-plane motion for nominal OP run and its PID solution (as one would expect for EM 3c). However, in the presence of perturbations, the OP run (with legend entry J2 OP) shows about 12 cm drift by the end of flight. This is corrected by the PID controller (legend entry J2 w/ PID). The overall propellant cost is now 0.0932 kg, marginally lower than nominal cost.

This successfully demonstrates the capability of the work-around for perturbations. We have had to solve mConGAL only for spherical gravity field, but with the work-around its results hold good in perturbed dynamics as well. An advantage of this method is that any perturbation can be easily included and accounted for. A disadvantage is that the solution need not be optimal for perturbed dynamics (but likely to be very close to optimal).

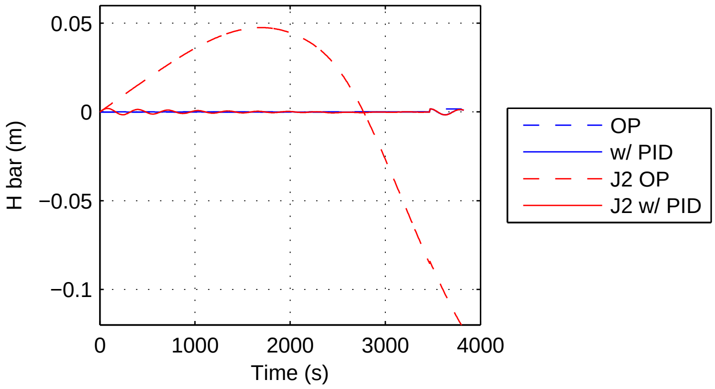


Figure 8.21: Cross-plane motion of chaser using PID control feedback mechanism for including perturbations.

### 8.4.3 NET EFFECT ON ENVISAT MISSION

Using the PID controller and including the effect of $J_2$ and atmospheric drag, we can now have another look at the results of the 9 mission cases of Envisat. The workaround is able to generate solutions for all the 9 cases and the additional cost incurred due to it is shown in Table 8.5.

Table 8.5: Updated results for 9 EM cases with additional cost due to perturbations and PID controller

| Mission id | Normal cost (kg) | Additional cost (kg) |
|---|---|---|
| 1a | 0.440 | −0.1616 |
| 2a | 0.246 | 0.0511 |
| 2b | 0.349 | 0.0227 |
| 2c | 0.256 | 0.1013 |
| 2d | 0.240 | 0.0956 |
| 3a | 0.292 | 0.0380 |
| 3b | 0.491 | 0.0058 |
| 3c | 0.316 | 0.0932 |
| 3d | 0.293 | 0.0836 |

This leads us back to the analysis of propellant consumption based on Figure 8.7 for the 9 cases. To compare with it, we plot updated propellant consumption for the grid-search set of the 9 cases in Figure 8.22. Between the said figures, one can observe many similarities: EM 3 group still consumes more propellant than the EM 2 group; EM 2b and 3b remain the most expensive in their groups. On the other hand, two major differences are observed: EM 1a becomes the most optimal mission case as it sees a reduction in propellant cost; EM 2d and 3d are not as optimal as EM 2a and 3a. Both these changes confirm with our expectations and put to rest the anomaly discussed in point (2e) on Page 126. Thus, the PID controller helps us gain a better insight on actual propellant consumption, which is otherwise under/over estimated by mConGAL.

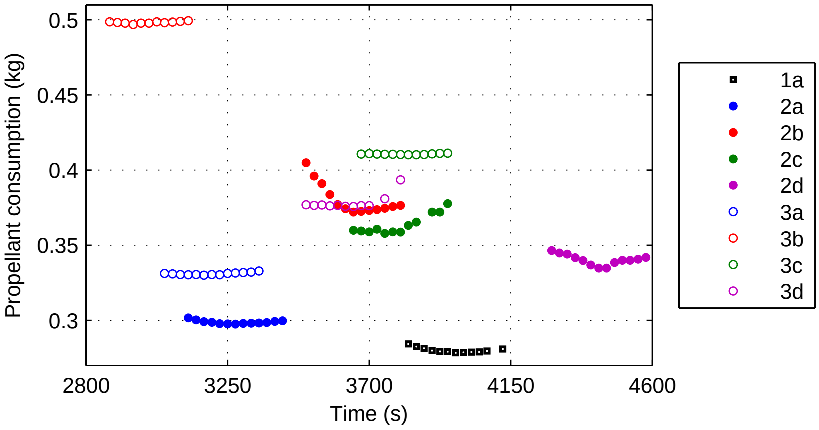


Figure 8.22: Grid-search solutions for 9 EM cases updated with additional cost due to PID controller and perturbations,

## 8.5 SUMMARY

The Envisat problem was modelled for use with the convex guidance algorithm, mConGAL. Most constraints were successfully modelled, except the forced motion constraint. An alternative constraint, consisting of lower thrust magnitude and restricted thrust rate change was applied instead, and was found to be moderately successful. 9 cases, each of which indicative of the position of HP1 due to precession of docking axis, were considered and analysed. mConGAL was able to find a solution for all of them. The optimal time of flight was found through a combined-search (inclusive of grid-search and golden section search). Detailed analysis of the results for the 9 cases showed that the EM3 group consumes more propellant and has a slightly more complex thrust profile than EM2 group. This is counter-intuitive as EM3 group only has motion in orbital plane as compared to EM2 group, which also has cross plane motion. However, it can be explained by keeping in mind the natural tendency of chaser's relative motion in LVLH frame.

Low accuracy of results and unreliable convergence led us to perform a sensitivity analysis and create two path-tracking mechanisms. The sensitivity analysis showed that mConGAL is moderately robust to variations in parameters. There are some instances where it behaves erratically. Amongst the path tracking mechanisms, the RCA fails to generate reliable solutions for seemingly feasible problems. The cause for this is unknown, but likely to be related to the moderate reliability of mConGAL to produce results. On the other hand, the PID controller is successful in improving accuracy of trajectory (to within 1 m) if one is to relax the thrust plume constraint. Additionally, a novel work-around to include perturbations ($J_2$ and drag) without changing mConGAL's dynamics was suggested and successfully demonstrated for use with the PID controller.

# 9

# Conclusions and Recommendations

This chapter brings us to the very end of this thesis. The research question we set out to solve is:

*Can the challenges of the highly constrained Envisat rendezvous mission be handled by a convex optimization based guidance algorithm in a robust, optimal and autonomous manner?*

To answer in as few words as possible, we can say it is ’unlikely’. However, this does not do justice to the capabilities of ConGAL and its modification, mConGAL. The question is bettered answered by the following summary:

1. ConGAL, by itself, faces many problems with respect to convergence and accuracy. These issues were not presented or discussed by Liu [2013] and were discovered after subjecting it to ConGAL Tests 1 to 4 (discussed in Chapter 7). Analysing the results, three new methods / modifications to improve its performance were developed and demonstrated:

   (a) Mean-Scheme, an on-board method to improve the convergence rate. It works by comparing objective values of previous iterations and generates a pseudo-iteration to help converge. In the examples demonstrated, problems that take nearly 40 iterations were solved in 8 iterations.

   (b) Runge-Kutta Integrator, for ConGAL dynamics. Seeing that ConGAL's second-order integrator is not accurate enough, four fourth-order integrators were compared and RK4 was selected. It allows us to retain higher step sizes, and still provide slightly more accurate results.

   (c) Reduced formulation: A modification to the way rate of change of thrust constraint is implemented showed an improvement in computation time by 40%.

   Further, a bid to numerically verify the results from Liu [2013] led to further discoveries:

   (a) Scaling factors: An error in the way Liu [2013] uses scaling factors was discovered (and corrected). This discrepancy questions the credibility of most of the numerical examples presented by Liu [2013], Liu and Lu [2013] and his other paper with Lu and Liu [2013]. The use of wrong scaling factors not only affects the accuracy of results, but also the convergence of ConGAL (see Section 7.4 and ME Tests from Chapter 6 ).

   (b) Influence of solver: It was observed that the choice of solver substantially affects the convergence and accuracy of the solution. To identify a suitable solver, a comparison in terms of accuracy and computation time was carried out, resulting in the choice of

MOSEK (despite its unsuitability for on-board implementation). Intermediate software like YALMIP and CVX were found to take up nearly 9 times more expensive computationally, and were thus, not used.

These modifications and the choice of using MOSEK without an interfacing solver, was termed as modified ConGAL (or mConGAL).

2. The Envisat Mission (EM) had to be cut-short to include only the final-approach phase, as (m)ConGAL was unsuitable to generate commands for the initial two phases. Most of the challenges of the mission were modelled successfully, as seen in Section 8.1. The exception being the forced-motion constraint which could not be modelled directly; however, an alternative was suggested, and was found to be moderately successful. With the modelled equations, the Envisat problem was solved as follows:

   (a) 9 distinct mission cases, to cover the possible orientations of Envisat (1 for Attitude Scenario 1, and 4 each for scenario 2 and 3) were selected.

   (b) Optimal time of flight was found using a combined search (using both grid-search and golden-section search) algorithm. Using which, a detailed analysis of the 9 cases was carried out. It was found that mission cases of attitude scenario 3 (EM 3 group, where the docking axis precesses in orbital plane) are more complex and computationally expensive to solve than compared to others. This is counter-intuitive because attitude scenario 2 (EM 2 group) missions have the docking axis precess in all three axis. The cause for this was explained by considering the natural tendency of a satellite in relative motion.

   (c) There were numerous instances where mConGAL failed to find an acceptable solution. This called for a sensitivity analysis to check influence of parameters and to gauge robustness. The results indicated that mConGAL is moderately robust to variations in inputs, but it still shows erratic behaviour on seemingly normal problems.

   (d) The accuracy of the results was also found to be lower than expected. To correct it, two path tracking mechanisms were developed:

      i. Re-computing algorithm: This involved repeatedly solving the guidance problem based on updated state of chaser and relevant modifications to constraints. The method turned out to be a failure, as mConGAL is unable to consistently converge to a new solution from the updated initial states. The exact cause of the behaviour is inexplicable but most likely related with other instance of erratic behaviour shown by (m)ConGAL. This grossly restricts its ability to perform guidance autonomously, as contingency solutions are not generated reliably.

      ii. PID controller: A manually tuned PID controller was implemented for use with mConGAL and the orbit propagator. It was able to correct the deviations successfully for all the 9 cases.

   The influence of perturbations was corrected by a novel work-around that solves the mConGAL problem in spherical gravity but generates (near) optimal solutions for use in perturbed field with the help of the PID controller (see 8.4.2 for details). The need for this was substantiated because the block to include perturbations in mConGAL could not be reliably used despite suggestions from Liu [2013].

   (e) Based on the use of the PID controller and the work-around to include perturbations, an updated set of results for the 9 mission cases were provided and compared. This led to the correction in the results of some of the cases, notably EM 1a (For attitude scenario 1), where there was a reduction in propellant cost. The final result confirms well with expected outcome, thus indicating that mConGAL can sometimes over/under estimate results.

Overall, (m)ConGAL is able to generate solutions for specific highly constrained rendezvous missions. It has some strong advantages over using conventional Clohessy-Wiltshire equations based algorithm (which was used by Deloo [2014]):

1. Almost all constraints pertaining to rendezvous missions are incorporated successfully. There are some concerns with respect to forced motion constraint, but barring that the problem is able to generate optimal solutions that does not violate any constraint.

2. There is no need for the user to supply any guesses for the guidance solution. This is important because as more constraints get added to the rendezvous problem, it is hard to make a guess that is optimal and feasible. For example, the solution by Deloo [2014] for transfer within the KOS.

3. Hundreds of variations of the Envisat Mission could be solved easily (albeit with varying degree of success) by merely changing input parameters. The method is well suited for automation, especially because there is no user supplied initial guess for the guidance solution. Carrying out such a test campaign with CW-equations based algorithm is likely to be difficult and time-consuming.

As a result, it can find application in studies where a large number of solutions are needed for a variety of highly constrained problems.

However, (m)ConGAL is unsuitable for robust and autonomous use because of its moderate-to-low reliability. At best, its solutions can be used as a first-guess for further use with other algorithms. The numerical experience in this thesis contrasts with the numerical examples provided by Liu [2013] (which, as mentioned, use wrong scaling factors) and questions their claim that "...numerical experiences have always shown good convergence of the successive solutions".

The following recommendations can be made for future work:

1. Unless there is concrete mathematical proof to support convergence of method of successive approximations, all numerical experience merely presents a slice of the overall picture. Also, seeing that the problem set-up is elaborate and laborious, it would be helpful for the user if a thorough mathematical proof or a methodology can pre-check whether a particular type of problem shall converge or not.

2. In the absence of a definite proof of convergence, it is hard to ascertain whether the mConGAL method fails to provide a solution by itself or due to the solver. Improvements in solvers could possibly pave way for better results.

3. The mean-scheme can be improved by implementing a weighted average to calculate the pseudo-iteration. Currently, its internal parameters are empirically chosen to suit problems where the iterations are stuck between two options. A systematic method to choose (and vary) the weights has the potential to increase its scope.

4. The work-around to include perturbations with the PID controller, has not been sufficiently tested. Rigorous testing over long time of flights, and additional perturbations is suggested.

# A

# STATE VECTOR CONVERSIONS

Often one has to convert between the coordinate systems. For example, a need to convert between Cartesian representation and Keplerian elements arises when computing orbit propagation. While such computations are relatively easier in Cartesian coordinates, the results are better expressed in Keplerian elements.

## A.1 CARTESIAN TO SPHERICAL SYSTEM

The conversion between the two systems is simple and given by:

$$\begin{aligned} r &= \sqrt{x^2+y^2+z^2} \\ \phi &= \arctan(\frac{y}{x}) \\ \theta &= \arcsin(\frac{z}{r}) \end{aligned} \tag{A.1}$$

## A.2 SPHERICAL TO CARTESIAN SYSTEM

The equations for conversion are:

$$\begin{aligned} x &= r\cos\theta\cos\phi \\ y &= r\cos\theta\sin\phi \\ z &= r\sin\theta \end{aligned} \tag{A.2}$$

Conversions between Cartesian and spherical systems are a rather standard procedure and do not require further discussion.

## A.3 CARTESIAN SYSTEM TO KEPLERIAN ELEMENTS

As defined in equation (3.1), the Cartesian state vector consists of position $\mathbf{r}$ and velocity $\mathbf{V} = \dot{\mathbf{r}}$. We define the following from these:

$$r = \|\mathbf{r}\| \tag{A.3}$$

$$V = \|\mathbf{V}\| \tag{A.4}$$

$$\mathbf{h} = \mathbf{r} \times \mathbf{V} \tag{A.5}$$

$$\mathbf{N} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^{\mathsf{T}} \times \mathbf{h} \tag{A.6}$$

$$N_{xy} = \sqrt{N_x^2 + N_y^2} \tag{A.7}$$

where $\mathbf{h}$ is the specific angular momentum vector, and $\mathbf{N}$ is referred to as node vector. Note that $\|\cdot\|$ is the Euclidean norm of a vector. This is generally also known as the vector's magnitude or length. Using the above definitions, we can determine the Keplerian elements as follows:

$$a = \frac{1}{\frac{2}{r} - \frac{V^2}{\mu}} \tag{A.8}$$

$$\mathbf{e} = \frac{\mathbf{V} \times \mathbf{h}}{\mu} - \frac{\mathbf{r}}{r} \tag{A.9}$$

$$e = \|\mathbf{e}\| \tag{A.10}$$

$$i = \arccos(\frac{h_z}{\|\mathbf{h}\|}) \tag{A.11}$$

$$\Omega = \text{atan2}\left(\frac{N_y}{N_{xy}}, \frac{N_x}{N_{xy}}\right) \tag{A.12}$$

$$\omega = \omega_{\text{sign}} \cdot \arccos\left(\hat{\mathbf{e}} \cdot \hat{\mathbf{N}}\right) \tag{A.13}$$

$$\theta = \theta_{\text{sign}} \cdot \arccos\left(\hat{\mathbf{r}} \cdot \hat{\mathbf{e}}\right) \tag{A.14}$$

where $\omega_{\text{sign}} = 1$ if $(\hat{\mathbf{N}} \times \mathbf{e}) \cdot \mathbf{h} > 0$, else $-1$; and $\theta_{\text{sign}} = 1$ if $(\mathbf{e} \times \mathbf{r}) \cdot \mathbf{h} > 0$, else $-1$.

In some cases, we encounter singularities and thus, have to be handled in a special manner. These are:

- When $i = 0$ or $\pi$ The orbit is parallel to the equatorial plane, and thus, $N = 0$. This makes it difficult to calculate $\Omega$ and $\omega$. We make the following corrections to accommodate such cases:

$$\Omega = 0 \tag{A.15}$$

$$\omega = \begin{cases} \arccos(\frac{e_x}{e}) & \text{if } e_z \geq 0 \\ 2\pi - \arccos(\frac{e_x}{e}) & \text{otherwise} \end{cases} \tag{A.16}$$

- When $e = 0$ This makes the orbit circular and thus, any point could be the pericentre. The following corrections are made:

$$\omega = 0 \tag{A.17}$$

$$\theta = \begin{cases} \arccos(\hat{\mathbf{r}} \cdot \hat{\mathbf{i}}) & \text{if } (\hat{\mathbf{i}} \times \mathbf{r}) \cdot \mathbf{h} \geq 0 \\ -\arccos(\hat{\mathbf{r}} \cdot \hat{\mathbf{i}}) & \text{otherwise} \end{cases} \tag{A.18}$$

where $\hat{\mathbf{i}} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$ is a unit vector along the X-axis in vector space.

## A.4 Keplerian elements to Cartesian state vector

The Keplerian elements can also be converted back to Cartesian state vector in a few steps. We first determine the radius of the Kepler orbit, $r_K$ as follows:

$$r_K = \frac{a(1-e^2)}{1 + e\cos\theta} \tag{A.19}$$

It is followed by determining the position vector $\mathbf{r}$ as follows:

$$\mathbf{r} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} l_1 & l_2 \\ m_1 & m_2 \\ n_1 & n_2 \end{bmatrix} \begin{bmatrix} r_K\cos\theta \\ r_K\sin\theta \end{bmatrix} \tag{A.20}$$

where, the elements $l, m$ and $n$ are defined as follows:

$$\begin{aligned}
l_1 &= \cos\Omega\cos\omega - \sin\Omega\sin\omega\cos i \\
l_1 &= -\cos\Omega\sin\omega - \sin\Omega\cos\omega\cos i \\
m_1 &= \sin\Omega\cos\omega + \cos\Omega\sin\omega\cos i \\
m_1 &= -\sin\Omega\sin\omega + \cos\Omega\cos\omega\cos i \\
n_1 &= \sin\omega\sin i \\
n_2 &= \cos\omega\sin i
\end{aligned} \tag{A.21}$$

To obtain the velocity vector of the spacecraft, we make use of equation A.21 along with the specific angular momentum $\mathbf{h}$. The equations are:

$$\mathbf{h} = \sqrt{\mu a(1-e^2)} \tag{A.22}$$

$$\begin{aligned}
\dot{x} &= \frac{\mu}{\|\mathbf{h}\|}(-l_1\sin\theta + l_2(e+\cos\theta)) \\
\dot{y} &= \frac{\mu}{\|\mathbf{h}\|}(-m_1\sin\theta + m_2(e+\cos\theta)) \\
\dot{z} &= \frac{\mu}{\|\mathbf{h}\|}(-n_1\sin\theta + n_2(e+\cos\theta))
\end{aligned} \tag{A.23}$$

Thus, the Cartesian state vector is obtained from the Keplerian elements.

Conversions between spherical system and Keplerian elements can be done by an intermediate conversion to Cartesian systems.

# B

# Other Solvers

Besides MOSEK, the simulator was also developed in three other environments - ECOS, YALMIP and CVX - all of which are available freely and can be used directly with MATLAB. Below, we provide a brief background about them, and present the code to solve the optimization problem describe in Equation (6.13) (termed as example run, in this Appendix). Comparision of syntax shows the ease / difficulty of using a particular software over another.

## B.1 ECOS

ECOS or Embedded COnic Solver was developed by Domahidi et al. [2013] as a light yet capable solver for use in embedded systems. It is free, portable and compatible for use with MATLAB. It requires the problem to be formulated in the conic form (see Equation (4.12)), or an interfacing software.

The MATLAB code for example run is shown below in Code B.1. In terms of lines of code, it is the longest code. Its problem setup requires knowledge in convex theory as well. However, since there is no need for additional dummy variables, it is easier to use it than MOSEK.

MATLAB Code B.1: Example run in ECOS

```
%%ECOS example run
%Constraints
E_L1 = -[1 1 0 1];                          %Inequality constraints
E_B1 = -3;                                  %of the form Lx <= B
dimsL1 = length(E_B1);
E_L2 = -[0 3 1 0];
E_B2 = -2;
dimsL2 = length(E_B2);
E_L3 = -[0 0 -1 1];
E_B3 = -0;
dimsL3 = length(E_B3);
E_L4 = -[0 0 0 1];
E_B4 = -2;
dimsL4 = length(E_B4);
E_G1 = -[1 0 0 0; 0 1 0 0; 0 0 1 0];        %Conic constraints
E_h1 = [0;0;0];                             %Gx + s = h & s>=0
dimsG1 = 3;
E_G2 = -[2 0 0 0; 0 2 0 0; 0 0 0 1];
E_h2 = [1;0;-1];
dimsG2 = 3;
%Problem setup
E_G = sparse([E_L1; E_L2; E_L3; E_L4; E_G1; E_G2]); %Matrices to ECOS
E_h = [E_B1; E_B2; E_B3; E_B4; E_h1; E_h2];
```

```
E_dims = struct('l',dimsL1+dimsL2+dimsL3+dimsL4,'q',[dimsG1;dimsG2]);
E_c = [1 1 1 1]';                           %Objective function

%Solver call
opts = ecosoptimset('verbose',1,'maxit',100);%,'feastol',1e-4);
[x,y,info,s,z] = ecos(E_c,E_G,E_h,E_dims,opts);

%Solution
x
```

## B.2 YALMIP

YALMIP is a modelling language for convex optimization problems (amongst other applications), developed by Löfberg [2004]. Its main intent is rapid algorithm development, allowing the user to focus on high-level modelling. It generates the numerical models required by external solvers and thus, acts as an interfacing software. It supports both MOSEK and ECOS.

The example run shown below in Code B.2. Compared to Codes 6.1 and B.1, this is substantially easy to write as all constraints are directly input as relations.

MATLAB Code B.2: Example run in YALMIP

```
1  %%YALMIP example run
2  %Initial setup for solver
3  Opt = sdpsettings('solver','mosek');
4  %Problem setup
5  x = sdpvar(4,1);                            %Define decision variable
6  Obj = sum(x);                               %Objective function
7  Y_C1 = [x(1) + x(2) + x(4) >= 3];           %Constraints
8  Y_C2 = [3*x(2) + x(3) >= 2];
9  Y_C3 = [x(4) - x(3) >= 0];
10 Y_C4 = [x(4) >= 2];
11 Y_C5 = [x(1) >= norm([x(2),x(3)],2)];
12 Y_C6 = [2*x(1) + 1 >= sqrt(4*x(2)^2 + (x(4) - 1)^2)];
13
14 %YALMIP call
15 r = solvesdp([Y_C1, Y_C2, Y_C3, Y_C4, Y_C5, Y_C6],Obj,Opt);
16
17 %Solution
18 x = double(x)
19 Obj = double(Obj)
```

## B.3 CVX

CVX is another high-level modelling language for convex optimization problems, with a specific focus on disciplined convex programming [Grant et al., 2008]. The disciplined programming paradigm makes use of a set of (strict) conventions when constructing the problem, allowing a mix of high-level modelling and basic principles of convex analysis. It supports ECOS and SDPT3, but requires a licence for MOSEK.

Code B.3 shows the code for example run. It is definitely the easiest of the lot as there are no extra variables / functions that need to be called. The use of mathematical keywords as part of the syntax makes the program readable. Notice the slight difference in the way the last constraint (line 12) is formulated, as compared to expression Y_C6 (line 12) in Code B.2. This is a result of the disciplined convex programming approach.

MATLAB Code B.3: Example run in CVX

```
1  %%CVX example run
```

```
cvx_begin                                   %CVX call start
cvx_solver ecos                             %Select solver
variable x(4)                               %Define decision variable
minimize sum(x)                             %Objective function
subject to                                  %constraints
x(1) + x(2) + x(4) >= 3;
3*x(2) + x(3) >= 2;
x(4) - x(3) >= 0;
x(4) >= 2;
x(1) >= norm([x(2),x(3)],2);
2*x(1) + 1 >= norm([2*x(2),(x(4) - 1)],2);
cvx_end                                     %CVX call end

%Solution
x
```

# Bibliography

Acikmese, B., Carson, J. M., and Blackmore, L. (2013). "Lossless Convexification of the Soft Landing Optimal Control Problem with Non-Convex Control Bound and Pointing Constraints." *IEEE Transactions on Control Systems Technology*, 21(6):2104–2113.

Acikmese, B. and Ploen, S. R. (2007). "Convex Programming Approach to Powered Descent Guidance for Mars Landing." *Journal of Guidance, Control, and Dynamics*, 30(5):1353–1366.

Alexander, J. D. and Young, K. A. (1970). "Apollo lunar rendezvous." *Journal of Spacecraft and Rockets*, 7(9):1083–1086.

Aström, K. J. and Murray, R. M. (2010). *Feedback systems: an introduction for scientists and engineers*. Princeton university press.

Benninghoff, H., Boge, T., and Tzschichholz, T. (2012). "Hardware-in-the-loop rendezvous simulation involving an autonomous guidance, navigation and control system." *Advances in the Astronautical Sciences*, 145:953–972.

Bhagat, M. R. (2015). "Convex guidance for Envisat rendezvous: Literature Study." Technical report, Delft University of Technology.

Blackmore, L., Acikmese, B., and Scharf, D. P. (2010). "Minimum-Landing-Error Powered-Descent Guidance for Mars Landing Using Convex Optimization." *Journal of Guidance, Control, and Dynamics*, 33(4):1161–1171.

Boyd, S. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

Carter, T. and Humi, M. (1987). "Fuel-optimal rendezvous near a point in general Keplerian orbit." *Journal of Guidance, Control, and Dynamics*, 10(6):567–573.

Clohessy, W. H. and Wiltshire, R. S. (1960). "Terminal Guidance System for Satellite Rendezvous." *Journal of the Aerospace Sciences*, 27(9):653–658.

DARPA (2007). "Orbital Express: Fact sheet." Technical report, Defense Advanced Research Projects Agency.

Davis, T. M. and Melanson, D. (2004). "XSS-10 microsatellite flight demonstration program results." In *SPIE Defence and Security Symposium*, volume 5419, 16–25.

Deloo, J. A. F. (2014). *Analysis of the rendezvous phase of e.Deorbit*. Msc thesis, Delft University of Technology. URL `http://repository.tudelft.nl/view/ir/uuid%3A7f445d54-c758-45a8-b87e-a9c0eec61617/`.

Domahidi, A., Chu, E., and Boyd, S. (2013). "ECOS: An SOCP solver for embedded systems." In *European Control Conference*, 3071–3076. IEEE, Zurich, Switzerland.

ESA (2012). "e.Deorbit Assessment - CDF Study Report." Technical Report CDF-135(C), European Space Agency.

ESA (2014). "e.Deorbit mission analysis guideline." Technical Report HSO-GFA WP600, European Space Operations Centre / European Space Agency.

ESA (2015). "e.Deorbit phase B1 mission and system requirements document." Technical Report GSTP MSRD e.deorbit Issue 1, Revision 0, ESTEC / European Space Agency.

Fehse, W. (2003). *Automated Rendezvous and Docking of Spacecraft*. Cambridge University Press.

Flores-Abad, A., Ma, O., Pham, K., and Ulrich, S. (2014). "A review of space robotics technologies for on-orbit servicing." *Progress in Aerospace Sciences*, 68:1–26.

Gerth, I. (2014). *Convex Optimization for constrained and unified landing guidance*. Msc thesis, Delft University of Technology.

Goodman, J. L. (2006). "History of Space Shuttle Rendezvous and Proximity Operations." *Journal of Spacecraft and Rockets*, 43(5):944–959.

Goodman, J. L. (2011). "History of space shuttle rendezvous." Technical Report 2011/23479, Johnson Space Centre, National Aeronautics and Space Administration.

Grant, M., Boyd, S., and Ye, Y. (2008). "CVX: Matlab software for disciplined convex programming." Technical report, CVX Research, Inc.

Hill, G. W. (1878). "Researches in the Lunar Theory." *American Journal of Mathematics*, 1(1):5–26.

Hindi, H. (2004). "A tutorial on convex optimization." In *American Control Conference*, 3252–3265. IEEE.

IADC (2007). "IADC Space Debris Mitigation Guidelines." Technical Report IADC-02-01, Revision 1, Inter-Agency Space Debris Coordination Committee.

Jezewski, D. J. (1971). "An optimal, analytic solution to the linear- gravity, constant-thrust trajectory problem." *Journal of Spacecraft and Rockets*, 8(7):793–796.

Kerambrun, S., Despré, N., Frapard, B., Hyounet, P., Polle, B., Ganet, M., Silva, N., Cropp, A., and Philippe, C. (2008). "Autonomous rendezvous system: The HARVD solution." In *Proceedings of the 7th International ESA Conference on Guidance, Navigation & Control Systems. Tralee, Ireland.*

Labourdette, P., Julien, E., Chemama, F., and Carbonne, D. (2009). "ATV Jules Verne mission maneuver plan." In *International Symposium on Space Flight Dynamics*. Toulouse, France.

Liou, J. C. (2013). "Engineering and technology challenges for active debris removal." In *Progress in Propulsion Physics*, volume 4, 735–748.

Liou, J. C., Johnson, N. L., and Hill, N. (2010). "Controlling the growth of future LEO debris populations with active debris removal." *Acta Astronautica*, 66(5-6):648–653.

Liu, X. (2013). *Autonomous Trajectory Planning by Convex Optimization*. Phd thesis, Iowa State University. URL `http://lib.dr.iastate.edu/etd/13137`.

Liu, X. and Lu, P. (2013). "Robust Trajectory Optimization for Highly Constrained Rendezvous and Proximity Operations." In *AIAA Guidance, Navigation, and Control (GNC) Conference*. Boston, USA.

Löfberg, J. (2004). "YALMIP: A toolbox for modeling and optimization in MATLAB." In *IEEE International Symposium on Computer Aided Control Systems Design*, 284–289. IEEE.

Lu, P. and Liu, X. (2013). “Autonomous Trajectory Planning for Rendezvous and Proximity Operations by Conic Optimization.” *Journal of Guidance, Control, and Dynamics*, 36(2):375–389.

Lunney, G. (1967). “Summary of Gemini rendezvous experience.” In *Simulation and Support Conference*, Space Simulation Conference. American Institute of Aeronautics and Astronautics.

Luo, Y., Zhang, J., and Tang, G. (2014). “Survey of orbital dynamics and control of space rendezvous.” *Chinese Journal of Aeronautics*, 27(1):1–11.

Montenbruck, O. and Gill, E. (2000). *Satellite Orbits - Models, methods and Applications.* Springer Verlag.

Mooij, E. (2013). *AE4870B Re-entry Systems: Draft Lecture Notes.* Delft University of Technology.

Morgan, G. and Henrion, M. (1990). *Uncertainty: A Guide to Dealing With Uncertainty in Quantitative Risk and Policy Analysis.* Cambridge University Press, Cambridge.

Morin, D. (2008). *Introduction to Classical Mechanics: With Problems and Solutions.* Cambridge University Press.

MOSEK ApS (2015). “The MOSEK optimization toolbox for MATLAB manual. Version 7.1 (Revision 28).” Technical report, MOSEK ApS.

NASA (2005). “Demonstration of autonomous rendezvous technology: Press kit.” Technical Report 67214, National Aeronautics and Space Administration.

Nesterov, Y. and Nemirovskii, A. (1994). *Interior-point polynomial algorithms in convex programming*. Society for Industrial and Applied Mathematics.

NRC (2012). *NASA space technology roadmaps and priorities.* National Research Council, National Academy of Sciences, USA, Washington D.C.

Oda, M. (2000). “Summary of NASDA’s ETS-VII robot satellite mission.” *Journal of Robotics and Mechatronics*, 12(4):417–424.

Ohkami, Y. and Kawano, I. (2003). “Autonomous rendezvous and docking by engineering test satellite VII: a challenge of Japan in guidance, navigation and control—Breakwell memorial lecture.” *Acta Astronautica*, 53(1):1–8.

Pisacane, V. L. (2005). *Fundamentals of Space Systems.* Oxford University Press, New York.

Polites, M. E. (1998). “An assessment of the technology of automated rendezvous and capture in space.” Technical Report TP-1998-208528, Marshall Space Flight Centre, NASA.

Rockafeller, R. T. (1993). “Lagrange multipliers and optimality.” *Society for Industrial and Applied Mathematics Review*, 35(2):183–238.

Ruiz, J. and Hart, J. (2010). “A Comparison Between Orion Automated and Space Shuttle Rendezvous Techniques.” In *AIAA Guidance, Navigation, and Control Conference*. Toronto, Canada.

Sauceda, F. (2001). “Space Station Reference Coordinate Systems.” *International Space Station, Rept. TR-SSP-30219.*

Shuster, M. D. (1993). “A Survey of Attitude Representations.” *The Journal of the Astronautical Sciences*, 1(4):439–517.

Singla, P., Mortari, D., and Junkins, J. L. (2005). "How to avoid singularity for Euler angle set?" In *AAS/AIAA Space Flight Mechanics Conference*. Copper Mountain, Colarado, USA.

Süli, E. and Mayers, D. F. (2003). *An Introduction to Numerical Analysis*. Cambridge University Press.

Tan, A., Zhang, T. X., and Dokhanian, M. (2013). "Analysis of the Iridium 33 and Cosmos 2251 Collision Using Velocity Perturbations of the Fragments." *Advances in Aerospace Science and Applications*, 3:13–25.

Tütüncü, R. H., Toh, K. C., and Todd, M. J. (2003). "Solving semidefinite-quadratic-linear programs using SDPT3." *Mathematical programming*, 95(2):189–217.

Vallado, D. A. (2001). *Fundamentals of Astrodynamics and Applications*. Second edition. Springer Science & Business Media.

Virgili, B., Lemmens, S., and Krag, H. (2014). "Investigation on Envisat attitude motion." In *ESA e.Deorbit Symposium*.

Wakker, K. F. (2010a). "AE4874 Astrodynamics I: Lecture Notes." Technical report, Delft University of Technology.

Wakker, K. F. (2010b). "AE4874 Astrodynamics II: Lecture Notes." Technical report, Delft University of Technoloy.

Wertz, J. R. and Bell, R. (2003). "Autonomous rendezvous and docking technologies: status and prospects." In *AeroSense 2003*, 20–30. International Society for Optics and Photonics.

Wie, B. (2008). *Space Vehicle Dynamics and Control*. Second edition. American Institute of Aeronautics and Astronautics, Reston, Virginia, USA.

Wieser, D. (2014). "e.Deorbit Mission Phase A study." Technical report, OHB System AG, Bremen.

Woffinden, D. C. (2008). *Angles-only navigation for autonomous orbital rendezvous*. Dissertation, Utah State University. URL `http://digitalcommons.usu.edu/etd/12`.

Woffinden, D. C. and Geller, D. K. (2007). "Navigating the Road to Autonomous Orbital Rendezvous." *Journal of Spacecraft and Rockets*, 44(4):898–909.

Zimpfer, D., Kachmar, P., and Tuohy, S. (2005). "Autonomous Rendezvous, Capture and In-Space Assembly:Past, Present and Future." In *1st Space Exploration Conference: Continuing the Voyage of Discovery*. Orlando.