La Sapienza
Università degli Studi di Roma

DIPARTIMENTO ME. MO. MAT.

Optimization Techniques for Satellites Proximity Maneuvers

Riccardo Bevilacqua

DOTTORATO IN METODI E MODELLI MATEMATICI
PER LA TECNOLOGIA E LA SOCIETÀ
CICLO XIX

Dipartimento di Metodi e Modelli Matematici
per le Scienze Applicate
Universita' degli Studi di Roma La Sapienza
Via A. Scarpa 16, I-00161, Roma, Italy
E-mail: bevilacqua@dmmm.uniroma1.it

Dottorando
Dottor Ingegner Riccardo Bevilacqua

Tutor
Prof. Guido De Matteis
Co-Tutor
Prof. Marcello Romano

Coordinatrice del Corso di Dottorato
Prof. Daniela Giachetti

To the memory of Chiara Valente.

Abstract

The main topic of this dissertation is the control optimization problem for satellites Rendezvous and Docking. Saving resources is almost as important as the mission safeness and effectiveness. Three different numerical approaches are developed. The first two techniques deal with real-time and sub-optimal control, generating a reliable control sequence for a chaser spacecraft which eventually docks to a target. The first approach uses dynamic programming to quickly generate a sub-optimal control sequence on a predetermined path to be followed by one of the two vehicles involved into the docking operations. The second method presents a fast direct optimization technique, which was previously validated on real aircraft for trajectory optimization. The third approach aims to take into account the limitations of space qualified hardware, in particular thrusters. The new technique fuses the use of a set of low thrust on-off engines with impulsive-high-thrust engines. The hybrid method here developed combines and customizes different techniques.

The relative motion in the above mentioned control strategies is represented by a linear dynamic model.

As secondary topic of this dissertation, the use of a genetic algorithm optimizer to find possible conditions under which spacecraft relative motion can be periodic, or at least bounded, is presented. This analysis takes into account the J_2 gravity perturbation and some drag effects. The importance of the obtained results directly apply to the problem of formation keeping, as natural dynamics can be exploited to reduce the amount of active control preventing the spacecrafts to drift apart along time.

Acknowledgements

First of all I wish to thank Professor Marcello Romano of the Naval Postgraduate School of Monterey, California, for his constant support, suggestions, technical help and for the freedom he has always given me during my research period in the United States.
Thank you to Professor Oleg Yakimenko of the Naval Postgraduate School, for the trust he showed me, for the very interesting topics and mathematical methods I learned from him and for his crucial support in the development of a multi-criterion multi-variable optimization routine based on Hooke-Jeeves pattern search algorithm.
Obviously thanks to my supervisor at the University of Rome, Professor Guido De Matteis, for being a guide in very difficult moments, for his suggestions and helpful comments especially on the Dynamic Programming approach.
I also wish to thank the ESA Advanced Concepts Team for having funded Contract 18141/04/NL/MV, Project Ariadna AO04/4104.
And last, but not the least, thanks to all the people who always supported me, especially my family.

— *Riccardo Bevilacqua*

Contents

List of Figures

List of Tables

List of Symbols

The following list reports, in alphabetical order, the most significant symbols used in this dissertation. Note that the over right arrow indicates a vector, the symbols in bold are matrices:

a	Semi-major Axis
a_{ik}	Coefficients of Reference Polynomials
$\mathbf{B}$	Universal Transformation Matrix for Cubic B-Splines
ds	Path Section Length
$\frac{d^2(.)}{dt^2}$	Time Derivatives w.r.t. Inertial Frame
$\frac{\delta(.)}{\delta s}$	Partial Derivative with respect to s
e	Eccentricity
g	Acceleration due to Gravity
i	Inclination
J, J_{sc}	Performance Index (Cost Functional) and Scaled Performance Index
J_2	Earth Flattening Term in Gravity Field Expansion
$\mathbf{K}$	Vector Positions Matrix of the Cubic B-Splines Control Points
m	Mass
n	Polynomial Order
N	Number of: Nodes (chapter 4) / Switching Points (chapter 5) / Impulses (chapters 3 and 6)
N_{gen}	Number of Generations
N_{ind}	Number of Individuals
N_L	Number of Levels in Dynamic Programming Tree
N_C	Number of Values for Control Discretization
$\hat{n}$	Path Normal Unit Vector
O_{sc}	Slaves Control Constraint Violation Scaling Factor
$P_k(\tau)$	Polynomial Reference Function for the Cartesian Coordinates
p	Primer Vector Magnitude
$\vec{p}$	Primer Vector
$\vec{p}_m$	Primer Vector Absolute Maximum

$\vec{r}$	Relative Position Vector
$\vec{r}_j$	Space Collocation of the j^{th} Impulse
$\underline{\underline{r}}$	Dyadic of Position Vector $\vec{r}$
$\vec{r}_c$	Chaser Position Vector in the Inertial Frame
$\vec{r}_t$	Target Position Vector in the Inertial Frame
$\vec{r}_{rel}$	Chaser-Target Relative Position Vector (both in the Inertial and LVLH Frames)
$\vec{r}^*$	$[r_x,\ r_y,\ 0]^T$
s	Curvilinear Abscissa
t	Time
t_m	Midcourse Time corresponding to Primer Absolute Maximum
t_{int}	Intermediate Time
t_j	Time Instant of the j^{th} Impulse Application
$T_{\max}$	Maximum Thrust
t_T^* (τ_T^*)	Thrust-off Instant (Arc)
t_T^{**} (τ_T^{**})	Thrust-on Instant (Arc)
$t_{scaling}$	Time Scaling Factor
$\vec{u}$	Acceleration Control Vector in LVLH frame
$\hat{u}$	Control Unit Vector
V	Velocity
V_{sc}	Velocity Scaling Factor
w	Cost Index Weighting Coefficient
w_p	Penalty Weighting Coefficient
$w_{Overflow}$	Slaves Control Constraint Violation Penalty Weighting Coefficient
w_{Speed}	Speed Penalty Weighting Coefficient
x, y, z	Cartesian Coordinates in LVLH frame
$\underline{\underline{1}}$	Unitarian Dyadic
$(.)_0$, $(.)_f$	Value at Initial and Final Time
$\dot{(.)}, \ddot{(.)}$	Time Derivatives in the LVLH Frame
$(.)_x,\ (.)_y,\ (.)_z$	Components along $x,\ y,\ z$ axis in LVLH Frame
$(.)_j$	Quantity pertaining to the j^{th} Time Node
$(.)', (.)'', (.)'''$	Arc Derivatives
$\overline{(.)}$	Relative Parameter

List of Greek Symbols

δ_T	Throttle Position
Δ, Δ_{sc}	Penalty Function, Scaled Penalty Function
$\delta\left(t - t_j\right)$	Dirac's Delta Centered on Time t_j
ΔV	Velocity Variation Magnitude (Impulse)
ΔV_m	Midcourse Additional Impulse
$\Delta \vec{V}_j$	j^{th} Impulse Vector
$\Phi(t)$	State Transition Matrix for *CW* Equations
Γ	Control Magnitude (Impulse or Continuous)
$\vec{\gamma}$	Cubic B-splines Parameter
λ	Virtual Velocity along the Virtual Arc
$\vec{\lambda}_r$	Position Adjoint Vector
$\vec{\lambda}_V$	Velocity Adjoint Vector
Λ	Whole Adjoint Vector
μ	Earth Gravitational Constant
ρ	Local Curvature of Path in Chapter 4, Air Density in Chapter 7
τ	Virtual Arc
$\hat{\tau}$	Path Tangent Unit Vector
$\Delta\tau, \Delta t_j$	Sampling Period
ω	Angular Velocity (Argument of Perigee only in Chapter 7)
Ω	RAAN (Right Ascension of Ascending Node)
$\vec{\omega}_{LVLH}$	LVLH Frame Angular Velocity
$\Psi(t)$	Convolution Integral Matrix due to Optimal Control for *CW* Equations
Ξ	Vector of Optimization Parameters
θ	On Orbit Anomaly

List of Acronyms

The following list reports, in alphabetical order, the most significant abbreviations used in this dissertation:

ACS	Attitude Control System
AOCS	Attitude and Orbit Control System
ATV	Automated Transfer Vehicle
CSI	Constant Specific Impulse
CW	Clohessy-Wiltshire
DARPA	Defense Advanced Research Projects Agency
DMRP	direct method for rapid prototyping
DOF	Degrees Of Freedom
ESA	European Space Agency
FF	Formation Flying
GA	Genetic Algorithm
GEO	GEOstationary or GEOsynchronous Orbit
HCW	Hill-Clohessy-Wiltshire
ISS	International Space Station
LEO	Low Earth Orbit
LTV	Linear Time Varying
LVLH	Local Vertical Local Horizontal
LQR	Linear Quadratic Regulator
MEO	Medium Earth Orbit
NASA	National Aeronautics and Space Administration
NLP	Non Linear Programming
NPS	Naval PostGraduate School
OP	optimization parameter
PMP	Pontryagin maximum principle
PROXOP	Proximity Operations
PWM	Pulse Width Modulation
RDD	Rendezvous and Docking
SRP	Solar Radiation Pressure
STS	Space Transportation System (NASA Space Shuttle)
VSI	Variable Specific Impulse

Chapter 1

Introduction

1.1 Motivation and Scope

More and more demanding missions are planned for the near future for military, civil and commercial purposes: a short list includes ESA missions Proba, LISA, XEUS, Darwin and SMART-3, NASA mission ST5.
At the same time the costs for launching and provide services to the orbiting structures have become a key point in mission design as the space agencies have to face economical limitations. Improved technology has led to the possibility for the International Space Station (ISS) to be built on orbit, likewise large antennas, telemetry and interferometry vehicles can be substituted by smaller agents flying in formation and communicating with each other. Furthermore, the use of smaller satellites can bring better performances for the mission; think about the possibility of covering a much larger portion of the Earth with telecommunication small vehicles at a certain relative distance, with respect to a single, range limited, big satellite. The very last and the most futuristic applications of space flight have clearly shown that large payloads are not convenient to be launched at once, while the distribution into smaller agents and tasks seems to be an important advantage in terms of money (up to 80% according to [1]; there is actually a strong debate still going on regarding this percentage), mission lifetime, damage repairing, substitution of elements, reconfiguration, mission re-planning and so on. Construction of such platforms and the overall set of possible close distance operations, among two or more flying satellites, are usually referred to as Rendezvous and Docking (RDD) and Satellites Formation Flying (FF).
Formation Flying deals with maintaining and reconfiguring a formation of vehicles intended to fly respecting some relative position and velocity constraints. While RDD is a more challenging, or rather dangerous activity, as it drives two satellites towards each other in order to physically connect them. The danger which resides behind these two operations is obvious as the possibility of collisions menaces the satellites' integrity. While maneuvering a formation, attention has to be paid in avoiding this kind of risk. For the docking procedures the challenge is also more evident as the relative distance has to reach zero. A small error in controlling the approaching platforms could mean to abort the entire mission. The NASA Dart Satellite recently crashed into its target while approaching to it ([2]), due to a lack of propellant related to the

on-board algorithm errors, which brought the system to spend more fuel than it was supposed to. This kind of accidents cannot be allowed when humans are on-board, and when bigger and more important structures are involved into the proximity operations. The American Space Shuttle has docked several times to the International Space Station, in order to re-fuel it and perform various supply services (see Figure 1.1).

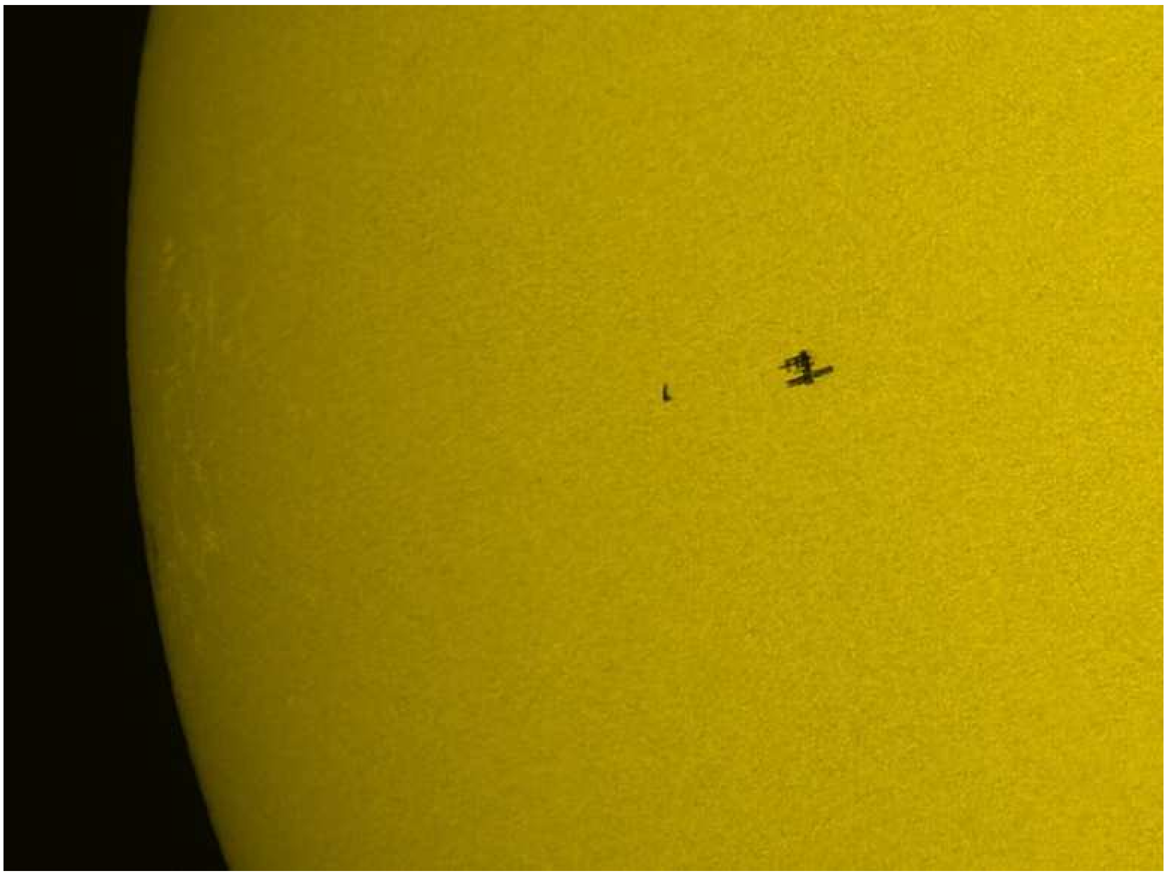
Figure 1.1: Astronomy Picture of the Day, recorded on September 17 2006 ([3]). Atlantis (left) has just undocked and moved about 200 meters away from the space station

The first examples of spacecraft rendezvous and docking are the manned US Gemini and Apollo programs and the unmanned Russian Cosmos missions of the late 1960's. Americans always included human intervention during the manoeuvres, while Russians performed the first automatic rendezvous and docking with the astronauts as supervisors.
The ESA Automated Transfer Vehicle (ATV) is planned for launch in 2007 to provide servicing to the ISS. Numerous companies are involved into the ATV project at European level (Alenia Spazio (Italy), CNES (France), GMV (Spain), etc.). The ATV is the first fully automated spaceship of its kind. It will be launched every 15 months to supply the crew of the International Space Station with equipment, fuel, food, water and air. During the NASA New Millennium Program, in 2000, the Earth Orbiter One automatically performed a rendezvous to fly in formation with the already orbiting Landsat-7 (*leader-follower* configuration). Autonomy is fundamental, especially for small formations and docking, where the vehicles are very close to each other, and a fast response is required for keeping and maneuvering the system. Down-link and up-link communications with the Earth could result in an non effective way to control the agents, due to the unavoidable delays.
The era of routine servicing of space stations may be close. Maintaining big orbiting constructions, assembling them, reconfiguration, and other RDD related activities, could soon become a recurrent set of operations. The ISS is the current base-point for outer space investigations, next will probably be the Moon, towards farther natural and/or artificial strategic collocations.
Is it then clear how the study of the relative dynamics and of the related guidance, navigation and control techniques represent a mandatory step for the scientific community, looking at the present and future applications of this particular kind of space flight. A deep knowledge of the involved dynamics and the related development of effective control strategies are under intense study. The European Space Agency encourages academic research in this field also ([4]).
One of the key aspects, which still represent open points for research and development, is the automation of RDD combined with a reliable optimization technique. The capability to perform this task basically resides into the algorithm reliability and velocity.
Optimization of resources, mainly fuel for manoeuvering and electrical power, affects the mission time length. For missions such as telecommunication FF it is fundamental to reduce the propellant consumption to keep the agents operative during years.
The main motivations of this thesis are then to give a contribution in the area of RDD optimal control and in formation flying dynamics. Two of the presented approaches focus the attention on the algorithm capability of being real-time implemented (chapters 4 and 5), while the third one (chapter 6) deals in particular with the real possibility of exerting a thrust of variable magnitude: now-a-days space qualified engines are basically on-off limited, and this feature cannot be left ignored while developing a control strategy.
The importance of the research field addressed in this thesis can be found in several applications, one above all can be the Automated Transfer Vehicle developed by the European Space Agency. On-the-ground experimentation of algorithms and hardware for satellites' relative control is also under a strong development in the last years ([5]), demonstrating the actuality of the issue. Rendezvous and Docking optimization has been studied at the "Naval Postgraduate School" of Monterey, California, USA. The interest in this particular field has been particularly fed by the possibility of testing algorithms on real hardware through on-the-ground experiments. In fact at

the Spacecraft Robotics Laboratory of the Naval Postgraduate School ([5]) a test bed has been recently set up for testing the integration of control strategies with sensors, actuators, etc. (see appendix E).
Though the real orbital dynamics cannot be simulated on-the-ground, such test beds have a huge potential in terms of low cost-high turnout tests. Furthermore real space qualified hardware can be used to establish whether the developed algorithms can be integrated with space systems.
Cooperation with the Naval Postgraduate School is still going on, future work will include the real-time implementation and test of the here proposed optimization techniques.
The last work (chapter 7) contributes to a deeper knowledge of the dynamics governing the relative motion between bodies in orbit around a master planet. This ESA funded research ([6]) brought to the discovery of particular inclinations for J_2 perturbed orbits.

1.2 Literature Review

1.2.1 Satellites Relative Motion Models

Clohessy and Wiltshire ([7]) developed a linear model for representing spacecraft relative motion when the reference is in circular orbit around a master body (linear models are obtained through a series expansion of the gravitational field with a reference satellite as base point). However, the approach of using the rotating reference frame for relative motion dates back to the 1800's to the work by Hill ([8]) in his development of the lunar theory, hence the other name of Hill's equations that is often used.
The Hill-Clohessy-Wiltshire model will be extensively used in this dissertation. Since then the modeling of this particular flight has known numerous efforts for improving the mathematical representation of the involved dynamics.
Neglecting aero-elastic effects and induced vibrations a satellite's motion can be approximated by the two cardinal equations (i.e. considered as a rigid body). In absence of drag forces, which relate the center of mass motion with the platform orientation, the two differential equations result to be uncoupled, that is the center of mass motion can be separately integrated with respect to the attitude. The overall control problem of an orbiting platform (AOCS) obviously includes both aspects. For relative motion among various agents the relative positions, orientations and their variation rates are crucial variables to be controlled. The positional part of the AOCS, when dealing with formation flying, is the most studied. One reason for this predominant interest among the others: collision avoidance. According to the application one can prefer complete models or linearized ones, orbital parameters or cartesian coordinates, in relation to the control technique we want to use as well.
There are basically two main reasons for studying the way to model this particular branch of space flight:

- Search for particular conditions generating advantageous relative trajectories (closed or quasi-closed relative orbits ([4, 9]): in the case of a controller required to maintain the formation within limited dimensions the natural motion can be conveniently exploited for reducing the control effort);

- Develop mathematical models helping the generation of simple control logics, such as linear regulators (LQR, [10, 11]). Typically space applications require optimal (fuel and/or time) control strategies.

The immediate improvement of the linear equations of motion for circular reference orbit can be found in [12] and [13], dealing with the effects of eccentricity; in particular in [12] the authors derive a set of LTV equations capable to represent the linear dynamics when the reference point is in elliptical orbit.
In [14, 9] mean orbital elements are used in order to find the so called J_2-invariant orbits. A Lyapunov based control is then applied to nullify possible deviations from the original closed orbits. In these two works approximated first order conditions are deduced for J_2-invariant orbits, by writing the Hamiltonian function for the system. Brouwer theory ([15]) for calculating mean orbital elements from cartesian coordinates (passing through obsculator orbital elements) is also used.
The master planet flattening is again faced in [16] and [17]. The second work also aims writing a linear model capable to include the J_2 effects in the relative dynamics. The strong will of obtaining constant coefficients linear equations brought the authors to operate a not very clear J_2 time average on the orbit, leading errors in the representation of the reality. The main point in this average is assuming a symmetry not proper of the J_2 perturbation not taking into account effects out of the orbital plane. The result basically brings to an accelerated reference orbit with respect to the classical circular unperturbed case.
The need to improve the reference orbit's attitude rate modeling brought the authors of [18] to use linear equations with a time dependant angular velocity in them. The instant by instant value of the attitude rate for the frame, with respect to which the linearization is performed, is obtained after observation of the position, inclination and anomaly behaviors on the reference J_2 orbit. This technique brought to a very satisfactory accuracy. One possibility would be to directly integrate the Gauss' equations ([19]) to obtain a more precise representation of the angular velocity, but, as shown in [18], this leads to a non linear system.
In [20] mean orbital elements are used together with a series expansion with respect to the eccentricity ([19]) to obtain explicit time functions for the radius and the other parameters associated with the reference frame.

1.2.2 Control Techniques for Formation Flying

Talking about control techniques for formation flying, the very last studies suggest alternatives like the *natural language* approach, based on fuzzy approach and evolutionary neurocontrollers ([21]), intended to deal with the problems of a very large formation, involving many agents, and often contrasting constraints ([22]). The fuzzy approach can be briefly explained as that experience that allows us to push the break pedal in our car after a visual estimation of the relative distance and velocity with respect to the one in front of us. From the engineering point of view we do not know any parameter involved in the dynamics of the two cars, but still we are able (hopefully!) to break in time. This is what the fuzzy technique wishes to do: control a system when a reduced knowledge of its parameters is given and especially when the system is very

complex. As just mentioned, experience is the basis for this controllers, so, neural networks-like functions have to be trained. The Sendai underground transportation system, in Japan, is an example of fuzzy application. The use of evolutionary neurocontrollers for space applications can be found in [23]. The principle of species selection, i.e., that the best individuals in a population statistically generate better offspring, is used to construct the neurocontrollers.
The disadvantage of these methods is that no very high precision is achieved, not being then suitable for operations such as very close maneuvers and docking.
In [24] a virtual structure methodology is presented. The formation is treated as a rigid body, controlling, at a virtual level, its center of mass dynamics and its attitude dynamics according to the two cardinal equations. The real vehicles are then in movement with respect to this imaginary platform. According to the required manoeuvre the virtual structure and the agents' relative motion have to be assigned, the control law is generated trough a Lyapunov approach. This kind of controllers can be useful for particular rigid reconfigurations, very effective from a theoretical point of view. Lyapunov global asymptotic stability has been proven for them, and they basically allow for a feedback. The drawback is that no limitations can be guaranteed on the amount of thrust to be exerted in order to perform the required operations, i.e. no optimization is possible.
In [25] a very elegant and efficient technique is presented for fuel or time optimization of formation flying manoeuvres. A set of nonlinear equations, easy to be numerically solved, is obtained. This method allows to compute the switching time instants for on-off thrusters, in order to bring a vehicle from an initially stable relative orbit to another final stable state. This technique also works for null final conditions, i.e., for the docking case. The obvious main limitations is that no generic initial conditions can be used, working only between two stable relative orbits.
For more references on formation flying control techniques refer, for example, to [26] and [1]. In particular the use of low thrust engines is cited in [1]. Such motors, as the Pulsed Plasma Thrusters, or the Electrostatic Thrusters, represent the new frontier for space flight, especially for long term missions and when a very low fuel consumption is required. These kind of engines allow for continuous very low thrust, having very high specific impulses. Thanks to this last feature, they present the lowest values for consumption among the space qualified engines. For formation flying and rendezvous manoeuvres electric propulsion can represent an interesting mean for saving fuel during the operations.

1.2.3 Control Techniques for Rendezvous and Docking

Physically, to connect two satellites means to satisfy some initial and final (zero) conditions in terms of position and velocity. An optimization method attempts to solve this two boundary value problem by minimizing some cost function. Typically some constraints arise as physical limitations on controls and so on. There are various ways to face the problem of optimizing the manoeuvres for RDD. The most common approach, which will be followed in this work as well, is to generate a control based on a linearized form of the relative dynamics. This simplifies the problem but still remains a very good approximation when dealing with relatively low distances between the docking agents. In other words the so obtained optimal or sub-optimal controls are very effective when applied to the real problem or anyway they can be a guideline for a

feedback control which compensates the disturbances with respect to the linearized model. As already mentioned at the beginning of this introduction, the new trends in relative space flight bring to the idea and realization of small satellites. Small vehicles, equipped with very low thrust engines, are already a reality ([27]). For such satellites and levels of thrust a very rich literature can be found.
In [28] Palmer presents a very elegant analytical solution for the fuel optimal problem by representing the low-thrust vector through Fourier series with a period equal to the manoeuvre required time. By doing this the author can apply the Lagrange principle for constrained minimization, the constraints being the boundary conditions. No bounds are imposed on the thrusters other than adjusting the final time for the manoeuvre thus reducing the peaks in the control magnitude, and the engines are supposed to be able to generate a time varying thrust. The dynamic is that of HCW. The same assumptions of thrust magnitude adjustability are made by Guelman in [29], while the thrust vector maximum magnitude is limited. In this paper a numerical solution is proposed for finding the adjoint variables initial conditions capable to satisfy the final boundary conditions, imposing an upper limit to the engines. Linear dynamic is assumed here too.
Carter deeply studied the optimization problem for rendezvous. In [30] the problem for the linear equations of motion with upper and lower limits on the thrusters is analyzed, allowing for intermediate states of thrust, i.e. continuous variation. In [31] the power limited problem is faced from a theoretical point of view, while in [32] an analysis is performed on wether the fuel efficiency can be improved or not by varying the number of on board thrusters. The thrusters are saturated, capable of variable magnitude.
In [33] the impulsive approach to the in-plane problem is implemented through the numerical solution of a quadratic expression capable to give at most four impulses for rendezvous. Solving the problem by assuming impulsive thrust capabilities means supposing relatively high-thrust engines on board, the impulse being a very rapid exertion of the thrust (the time extension for the impulse is negligible with respect to the manoeuvre required time). The way of collocating the impulses in time and space comes from the Lawden Primer Vector Conditions ([34]), reported later on (section 6). Additional work on impulsive control for HCW equations can be found in [35] where some impulses are pre-fixed, the remaining, if needed, are optimized. The solution passes through numerical solution of a set of non linear equations. Also a numerical approach on a set of non linear equations is needed in [36] for the bounded thrust problem.
The most used methodology for solving the impulsive problem is inherited from the interplanetary transfers optimization, i.e. from the non linear general matter of driving a probe between two influence spheres of different planets. The main idea of the numerical solution is to calculate a reference two impulses trajectory and then linearize with respect to it. By doing this several first order conditions can be derived for additional impulses insertion, re-collocation, coasting phases and so on. Representative works are the ones of Lion ([37]), Jezewski ([38]), Prussing ([39], [40], [41]), Edelbaum ([42], [43]). A complete development for an optimization algorithm of such impulses for non linear manoeuvres can be found in [44]. In [45] the first order conditions above mentioned are applied to the linear equations of relative motion between satellites.
In [46] Neutstadt demonstrated a fundamental theorem for linear systems: the optimal number

of impulses cannot exceed the number of final conditions to be met. Such theorem is used in chapter 6 and in all the mentioned papers dealing with impulsive control on HCW equations (see also [47]).
Let us also mention a relatively old but still actual survey on issues and developments of rendezvous and docking technologies: [48]. In particular the routine manoeuvres performed by the Space Shuttle to dock to a target in circular orbit are described. No optimization is considered, as currently done when the STS docks to the ISS, the approach is basically impulsive.
To conclude this last section of the literature review note that neither real-time capabilities nor the reliability of the software are analyzed in the above cited works.

1.3 Thesis Outline

In chapter 2 the classical variational formulation and the necessary conditions are given for functional optimization problems.
In chapter 3 an overview on the available mathematical models for satellites proximity flight is presented, focusing the attention on the Hill-Clohessy-Wiltshire linear model. The chapter also specializes what generally described in chapter 2, dealing with the optimization of RDD problem when continuous or impulsive motors are mounted on the controlled agent.
Main issue, dealt with in the optimization algorithms here developed, is the routines reliability and computation time scale needed for achieving the solution. For this reasons two direct methods have been studied at first.
In chapter 4 a dynamic programming based direct method is developed for optimizing the RDD control sequence on a predetermined trajectory. The approach acts on the curvilinear acceleration along the path, avoiding the curse of dimensionality proper of the dynamic programming technique. This allows the algorithm to be tested in real time and the hardware test is also described.
Being the previous technique limited to one pre-chosen path, more flexibility is introduced for the trajectory in chapter 5, through a direct method for rapid prototyping of sub-optimal trajectories based on high order polynomials for path representation. The methodology was previously applied to the atmospheric flight world and has been tested on real aircraft. Adapted to the dynamics involved in RDD, the algorithm shows high velocity and reliability which make it a good candidate for the on-board software.
Considerations on the obtained results with the first two techniques, especially the actual possibility of modulating rockets thrust, bring to face, in chapter 6, the fundamental issue of the space thrusters limitations, by introducing an hybrid optimization technique. The chaser satellite is supposed to be equipped with both impulsive high-thrust and continuous low-thrust engines. The way the low-thrust engines can modulate the thrust is limited by actually considering them simply on-off capable, and assuming a set of motors mounted on-board. Thus the set of admissible magnitudes for the low thrust segment is finite.
Optimizing fuel is important not only when a certain maneuver has to be performed, as in the RDD case. When thinking about formation flying satellites, exposed to disturbances such as air drag and high order harmonics of the gravitational field, it is important to minimize the con-

trol routine of formation keeping, which is needed to keep the spacecraft close to each other, counteracting the natural drift. For this reason in 7 the work developed under the ESA Ariadna Project is described with its main results. The application of a genetic optimizer gives the possibility of widely exploring the possibilities for having periodic, or at least bounded motion for formation flying satellites, when considering non linear dynamics, J_2 effects and air drag.
Chapter 8 closes the dissertation with comments on the obtained results and possible future developments.
Appendix A reports the state transition matrices and the convolution matrices for unbounded optimal control for two forms of the HCW linear equations.
Appendix B deals with the problem of having a cubic B-spline passing on the desired initial and final points.
Appendix C describes the algorithm translated in software for the Hooke-Jeeves first order minimization routine used in chapter 6.
Appendix D briefly describes the software developed for this dissertation.
In appendix E a brief description of the Autonomous Docking Test Bed, set up at the Naval Postgraduate School of Monterey, is reported.
Appendix F is a survey on currently space qualified (or in qualification process) low-thrust thrusters, developed to better understand the hardware features and the issues arising when dealing with spacecraft approaching to each other.
Appendix G reports some notes on Genetic Algorithms.

1.4 Main Contribution

Works of chapters 4 and 5 have been presented at the 7th "Dynamics and Control of Systems and Structures in Space" conference 2006, in Greenwhich, UK (see [49] and [50]).
A technical note based on the technique of chapter 6 has been submitted for publication to the "Journal of Guidance, Control and Dynamics" (see [51]).
Results reported in chapter 7 have been presented at the "American Astronautical Society" conference 2006, in Tampa, Florida, USA. A previous work which used GA for studying only non linear effects was accepted for publication on the "Journal of Non Linear Dynamics and Systems" ([52]). The same work was presented at the "4th Workshop on Satellite Constellations and Formation Flying", from which it was selected by the organizers for publication.
Main original contributions of this thesis are:

1. Real-time and online autonomous control during the very last phases of satellites docking procedure. Autonomy resides in the algorithm capability of taking decisions without external intervention. A new versatile strategy is derived to optimize the approaching stage on a suitably chosen path, in a quick way;

2. a direct method is also adapted to the problem in order to optimize the rendezvous trajectory;

3. to assume an hybrid continuous-impulsive propulsion considering the continuous thrust to be generated by a cluster of independent low-thrust engines, each one operating either

at the maximum or at zero thrust (on-off). The far-away rendezvous manoeuvres are optimized with low-thrust motors, the last phase until docking is precisely controlled by pulses. This makes the method applicable in principle to current electric thruster technology;

4. the application of genetic algorithms to the non linear J_2 an drag perturbed model for relative motion gives interesting and new results. The possibility of enriching the set of particularly good-natured conditions for reducing the formation keeping control effort arises.

Chapter 2

Calculus of Variations

This chapter reports a general mathematical formulation for functional optimization. We here follow most of the notation and style of [53].
Functional optimization means searching for unknown functions which minimize (maximize) a cost functional.
Given a dynamical system in the form:

$$\dot{\overrightarrow{X}} = \overrightarrow{f}(\overrightarrow{X}, \overrightarrow{U}, t) \tag{2.1}$$

with $\overrightarrow{f}$ a column vector of n components, $\overrightarrow{U}$ the m components control vector and $\overrightarrow{X}$ obviously the state vector, the Mayer problem consists in searching for the optimal control law $\overrightarrow{U}^*$ such that the functional of the initial and final states:

$$J = J(\overrightarrow{X}_0, t_0, \overrightarrow{X}_f, t_f) \tag{2.2}$$

has a stationary value. $\overrightarrow{X}_0, t_0, \overrightarrow{X}_f, t_f$ indicate the initial and final state vectors with the corresponding time instants. They generally have to satisfy the constraints:

$$\overrightarrow{\phi}(\overrightarrow{X}_0, t_0, \overrightarrow{X}_f, t_f) = 0 \tag{2.3}$$

Meanwhile, the Lagrange problem consists in defining the cost functional as an integral:

$$J = \int_{t_0}^{t_f} f_{n+1}(\overrightarrow{X}, \overrightarrow{U}, t)dt \tag{2.4}$$

The problem represented by eq. 2.4 can be stated in the Mayer form by defining an additional state x_{n+1} to the vector $\overrightarrow{X}$, such that:

$$x_{n+1} = \int_{t_0}^{t_f} f_{n+1}(\overrightarrow{X}, \overrightarrow{U}, t)dt \tag{2.5}$$

having the new cost:

$$J = x_{n+1}(t_f) \tag{2.6}$$

and an additional state equation:

$$\dot{x}_{n+1} = f_{n+1}(\overrightarrow{X}, \overrightarrow{U}, t) \tag{2.7}$$

Eq. 2.6 is a particular case of eq. 2.2.

2.1 Necessary Conditions for Optimality

Having written the problem of Lagrange in the Mayer form, we can use the theory of ordinary maxima or minima. The dynamic constraint will be $\dot{\overrightarrow{X}} - \overrightarrow{f} = 0$, to be handled while minimizing (maximizing) eq. 2.6 through the use of the Lagrange multipliers $\overrightarrow{\Lambda}$. Then, let us introduce the augmented cost functional:

$$I = J - \int_{t_0}^{t_f} \overrightarrow{\Lambda} \cdot (\dot{\overrightarrow{X}} - \overrightarrow{f})dt = J - \int_{t_0}^{t_f} (\overrightarrow{\Lambda} \cdot d\overrightarrow{X} - H)dt \tag{2.8}$$

being $H = \overrightarrow{\Lambda} \cdot \overrightarrow{f}$ the Hamiltonian function.
Imposing $\delta I = 0$, in order to find the stationarity, leads to:

$$\begin{aligned} \delta I = & \frac{\delta J}{\delta \overrightarrow{X}_0} \delta \overrightarrow{X}_0 + \frac{\delta J}{\delta t_0} \delta t_0 + \frac{\delta J}{\delta \overrightarrow{X}_f} \delta \overrightarrow{X}_f + \frac{\delta J}{\delta t_f} \delta t_f + \\ & - \int_{t_0}^{t_f} \left[\overrightarrow{\Lambda} \cdot \delta(d\overrightarrow{X}) - \left(\frac{\delta H}{\delta \overrightarrow{X}} \delta \overrightarrow{X} + \frac{\delta H}{\delta \overrightarrow{U}} \delta \overrightarrow{U} \right) \right] dt + \\ & - (\overrightarrow{\Lambda} \cdot \dot{\overrightarrow{X}} - H)_f \delta t_f + (\overrightarrow{\Lambda} \cdot \dot{\overrightarrow{X}} - H)_0 \delta t_0 = 0 \end{aligned} \tag{2.9}$$

integration by parts gives:

$$\begin{aligned} \int_{t_0}^{t_f} \overrightarrow{\Lambda} \cdot \delta(d\overrightarrow{X}) = \int_{t_0}^{t_f} \overrightarrow{\Lambda} \cdot d(\delta \overrightarrow{X}) = \left[\overrightarrow{\Lambda} \cdot \delta \overrightarrow{X} \right]_{t_0}^{t_f} - \int_{t_0}^{t_f} \dot{\overrightarrow{\Lambda}} \cdot \delta \overrightarrow{X} dt = \\ = \overrightarrow{\Lambda}_f \cdot (\delta \overrightarrow{X})_f - \overrightarrow{\Lambda}_0 \cdot (\delta \overrightarrow{X})_0 - \int_{t_0}^{t_f} \dot{\overrightarrow{\Lambda}} \cdot \delta \overrightarrow{X} dt \end{aligned} \tag{2.10}$$

By representing $\delta \overrightarrow{X}_0 = (\delta \overrightarrow{X})_0 + \dot{\overrightarrow{X}}_0 \delta t_0$ and $\delta \overrightarrow{X}_f = (\delta \overrightarrow{X})_f + \dot{\overrightarrow{X}}_f \delta t_f$ (see [53]), we can further calculate:

$$\int_{t_0}^{t_f} \overrightarrow{\Lambda} \cdot \delta(d\overrightarrow{X}) = \overrightarrow{\Lambda}_f \cdot (\delta \overrightarrow{X}_f - \dot{\overrightarrow{X}}_f \delta t_f) - \overrightarrow{\Lambda}_0 \cdot (\delta \overrightarrow{X}_0 - \dot{\overrightarrow{X}}_0 \delta t_0) - \int_{t_0}^{t_f} \dot{\overrightarrow{\Lambda}} \cdot \delta \overrightarrow{X} dt \tag{2.11}$$

which, substituted into eq. 2.9 gives:

$$\delta I = \delta J - \left[\overrightarrow{\Lambda} \cdot \delta \overrightarrow{X} - H \delta t \right]_0^f + \int_{t_0}^{t_f} \left[\left(\frac{\delta H}{\delta \overrightarrow{X}} + \dot{\overrightarrow{\Lambda}} \right) \cdot \delta \overrightarrow{X} + \frac{\delta H}{\delta \overrightarrow{U}} \cdot \delta \overrightarrow{U} \right] dt = 0 \tag{2.12}$$

Introduction of the Lagrange multipliers leads independency between the vectors $\delta \overrightarrow{X}$ and $\delta \overrightarrow{U}$. Then, the necessary conditions for optimality:

$$\begin{aligned} \frac{\delta H}{\delta \overrightarrow{U}} &= 0 \\ \dot{\overrightarrow{\Lambda}} &= - \frac{\delta H}{\delta \overrightarrow{X}} \end{aligned} \tag{2.13}$$

What remains of eq. 2.12 is:

$$\delta I = \delta J - \left[\overrightarrow{\Lambda} \cdot \delta \overrightarrow{X} - H \delta t \right]_0^f = 0 \tag{2.14}$$

to be satisfied by the variations of initial and final time instants and states, which also respect eq. 2.3.

2.2 Transversality Conditions

As just done in section 2.1, the constraint $\overrightarrow{\phi} = 0$ can be handled by augmenting once more the functional I, introducing an additional Lagrange multipliers vector $\overrightarrow{\nu}$:

$$K = I + \overrightarrow{\nu} \cdot \overrightarrow{\phi} \tag{2.15}$$

As previously mentioned, the multipliers lead arbitrary variation of initial and final time instants and state vectors. In other words, using condition 2.14, the necessary condition $\delta K = 0$ becomes:

$$\delta K = \delta J - \left[\overrightarrow{\Lambda} \cdot \delta \overrightarrow{X} - H \delta t \right]_0^f + \overrightarrow{\nu} \cdot \overrightarrow{\phi} = 0 \tag{2.16}$$

explicitly we have the transversality conditions:

$$\begin{aligned} \overrightarrow{\Lambda}_0 &= -\frac{\delta J}{\delta \overrightarrow{X}_0} - \overrightarrow{\nu} \cdot \frac{\delta \overrightarrow{\phi}}{\delta \overrightarrow{X}_0} \\ \overrightarrow{\Lambda}_f &= \frac{\delta J}{\delta \overrightarrow{X}_f} + \overrightarrow{\nu} \cdot \frac{\delta \overrightarrow{\phi}}{\delta \overrightarrow{X}_f} \\ H_0 &= \frac{\delta J}{\delta t_0} + \overrightarrow{\nu} \cdot \frac{\delta \overrightarrow{\phi}}{\delta t_0} \\ H_f &= -\frac{\delta J}{\delta t_f} - \overrightarrow{\nu} \cdot \frac{\delta \overrightarrow{\phi}}{\delta t_f} \end{aligned} \tag{2.17}$$

2.3 Solving the Optimization Problem

It is straightforward to deduce from the Hamiltonian definition that $\dot{\overrightarrow{X}} = \frac{\delta H}{\delta \overrightarrow{\Lambda}}$. Together with the second of eq. 2.13, it constitute the system to be integrated, using the conditions 2.3 and the transversality conditions. For the most of the practical problems the solution requires a numerical integration.

NOTE 1: for the case of bounded ($\overrightarrow{U} \in \bar{U}$) controls it can be demonstrated that the previous theory holds and it is sufficient to replace the first condition of eq. 2.13 with:

$$\overrightarrow{U}^* = arg_{\overrightarrow{U} \in \bar{U}} \left(sup(H) \right) \tag{2.18}$$

NOTE 2: The necessary conditions here reported are also sufficient if the system 2.1 is linear (see [53]).

Chapter 3

Optimal Rendezvous and Docking: The Problem

3.1 Introduction

The present chapter describes rendezvous and docking from the mathematical point of view. First, a dynamic model for relative motion between two satellites in close orbits is derived. Before addressing the optimization problem for two classes of rocket engines, i.e. continuous and impulsive, a short list of currently space qualified and under study thrusters is presented.

3.2 Dynamic Model for Relative Motion

This dissertation uses for the most part a classical linear LVLH referred (Figure 3.2) model is used to represent the relative motion between two satellites flying in very close orbits ([7]). The equations will be per unit mass, not considering mass variation of the system, i.e. not making distinction between acceleration and thrust. The satellite's center of mass motion differential equation, in an inertial or quasi-inertial (the frame's acceleration effects are negligible within the phenomenon observation time) attraction body centered frame, is:

$$\frac{d^2\overrightarrow{r}_{sat}}{dt^2} = -\frac{\mu}{r_{sat}^3}\overrightarrow{r}_{sat} + \overrightarrow{f} \tag{3.1}$$

where $\overrightarrow{r}_{sat}$ is the object position vector with respect to the attraction body, $\mu = 3.98601 \cdot 10^5 \frac{km^3}{s^2}$ is the Earth gravitational constant, the first term on the right-hand side is the keplerian force per unit mass and $\overrightarrow{f}$ represents the set of remaining forces (per unit mass) different from the attraction of a perfectly spherical, uniform mass distribution planet (high order harmonics of the gravitational field, air drag, SRP, engines thrust, etc.).
The x_I, y_I, z_I quasi-inertial frame used, in the Earth case, is showed in Figure 3.1:

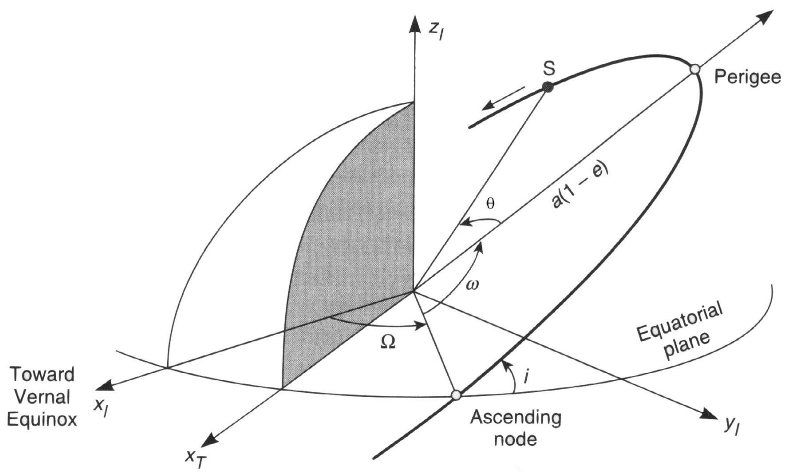

Figure 3.1: Earth-Centered Quasi-Inertial Reference Frame

In order to consider the reality of non perfectly shaped attraction body it is common practice to consider the flattening effect (called J_2) as one of the major issues when describing the dynamic of Earth orbiting space vehicles. J_2 is measured as $\frac{C-A}{MR_e^2}$ where C and A represent the two different inertia moments of the Earth, approximated as a circular ellipsoid (in the perfectly spherical hypothesis $C = A$), M is the Earth's mass, R_e the equatorial radius of the planet. Here is the complete center of mass differential equation of motion including the J_2 effect:

$$\frac{d^2\overrightarrow{r}_{sat}}{dt^2} = -\frac{\mu}{r_{sat}^3}\overrightarrow{r}_{sat} + \overrightarrow{J}_2(\overrightarrow{r}_{sat}) + \overrightarrow{f'} \tag{3.2}$$

, having called $\overrightarrow{f'}$ the set of forces mentioned in eq. 3.1 a part from $\vec{J_2}$. For the sake of completeness let us express the perturbation force deriving from the flattening along the orbit radius (r), the radius-normal in-plane direction (θ), and a component perpendicular to the orbital plane (z):

$$f_r = -\frac{3\mu R_e^2 J_2}{2r_{sat}^4}\left(1 - 3\sin^2 i \sin^2 u\right)$$
$$f_\theta = -\frac{3\mu R_e^2 J_2}{2r_{sat}^4}\sin^2 i \sin 2u$$
$$f_z = -\frac{3\mu R_e^2 J_2}{2r_{sat}^4}\sin 2i \sin u$$

where i is the orbit inclination and $u = \theta + \omega$, being ω the argument of perigee, θ the on orbit anomaly. In order to linearize the relative motion between two satellites let us begin expressing eq. 3.2, by means of the relative motions theorem, in the LVLH frame:

$$\overrightarrow{\ddot{r}_{sat}} + 2\overrightarrow{\omega} \times \overrightarrow{\dot{r}_{sat}} + \overrightarrow{\omega} \times (\overrightarrow{\omega} \times \overrightarrow{r_{sat}}) + \overrightarrow{\dot{\omega}} \times \overrightarrow{r_{sat}} = -\frac{\mu}{r_{sat}^3}\overrightarrow{r_{sat}} + \overrightarrow{J_2}(\overrightarrow{r_{sat}}) + \overrightarrow{f'} \tag{3.3}$$

where, for brevity, $\overrightarrow{\omega} = \overrightarrow{\omega}_{LVLH}$ is the LVLH angular rate; this frame remains associated with a real or virtual satellite orbiting around the planet: usually x points from the attraction body to the virtual satellite, y as the velocity vector of the satellite, z completes the frame:

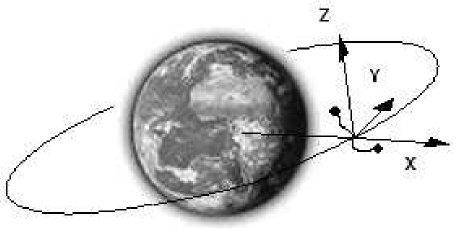

Figure 3.2: LVLH Reference Frame

A very rough way to compute the relative position and velocity between two objects would be propagating eq. 3.1 for each vehicle and subtract the obtained vectors. This would not work as the distances from the attraction body are obviously orders of magnitude higher than the relative ones, easily restituting erroneous values for the relative vectors (numerical errors could give a completely wrong result) or anyway requiring machines with a very high precision. A part from the numerical reasons behind the direct propagation and subtraction, we should take into account that, when controlling a formation, especially with autonomous systems, it is much easier to directly measure the relative vector onboard the vehicles than using external measure sources to obtain the absolute positions to subtract. The preceding justifications support the necessity of models capable to describe the relative dynamics. The literature is very rich on linearization of the relative dynamics model as this allows, even if in an approximate form, to better understand the phenomenon and design controllers. Just as example the study of simplified equations gives the possibility of searching for periodic or pseudo-periodic relative orbits.
Deriving a linear model requires first of all a reference orbit on which the virtual agent moves on, then the gravitational field series expansion up to the first order is performed with respect to this given motion. After having linearized every satellite's motion it is easy to calculate the relative dynamics among more than two satellites, by simply subtracting the obtained equations. For one satellite, indicated by 1, calling 0 the reference virtual satellite, it is:

$$\frac{d^2\overrightarrow{r}_1}{dt^2} = -\frac{\mu}{r_0^3}\overrightarrow{r}_0 + \overrightarrow{J}_2(\overrightarrow{r}_0) + \mathbf{G}\cdot(\overrightarrow{r}_1 - \overrightarrow{r}_0) + \nabla \mathrm{J}_2|_{\overrightarrow{r}_0}(\overrightarrow{r}_1 - \overrightarrow{r}_0) + \overrightarrow{f_1'} \qquad (3.4)$$

where $\mathbf{G}$ is the gravity gradient for spherical uniform mass Earth, ∇J_2 represents the gradient associated with the Earth's flattening.
Being 2 a second vehicle, writing for it the same expression of eq. 3.4, subtracting the two

obtained formulas and projecting in the LVLH frame according to eq. 3.3:

$$\ddot{\vec{r}} + 2\vec{\omega} \times \dot{\vec{r}} + \vec{\omega} \times (\vec{\omega} \times \vec{r}) + \dot{\vec{\omega}} \times \vec{r} = \mathbf{G} \cdot \vec{r} + \nabla J_2|_{\vec{r_0}} \vec{r} + \vec{f_2'} - \vec{f_1'} \tag{3.5}$$

where $\vec{r} = \vec{r_2} - \vec{r_1}$ and x, y, z are the $\vec{r}$ components in LVLH. For the remaining of this dissertation we will refer to the relative position between two satellites by simply calling it $\vec{r}$, without any confusion, not needing the absolute position with respect to the Earth.
The open point is now the correct representation of the virtual orbit motion, i.e. the LVLH frame motion, keeping in mind the will to maintain the linearity of the equations. As showed independently by Sabatini [10] and Ricelli [11], who based their works on [17], the choice for the angular velocity $\vec{\omega}_{LVLH}$ of this frame is of fundamental importance for a correct representation of the reality. For example, choosing an angular velocity proper of a keplerian orbit with no consideration of the effect of J_2 on it, using eq. 3.5, will lead to consider the differential effects of J_2 for the relative motion but having the linearization point moving differently, i.e. the linearization looses its significance.
From here the general importance of correctly representing $\vec{\omega} = \vec{\omega}_{LVLH}$. In [17] an average on J_2 is performed, with no deep explanation of its calculation, to obtain an expression for $\vec{\omega}$ accelerated with respect to the keplerian case.
This model is not capable anyway to consider some consequences of the Earth's flattening, like the fact that the orbital plane has no constant inclination in space, in other words the angular rate shows a component on $\vec{r}$ as well.
It is possible to write, in the frame $[\hat{r}, \hat{\theta}, \hat{h}]^T$ ([10]):

$$\vec{\omega} = \begin{bmatrix} \dot{\gamma} \\ 0 \\ \frac{h}{r_{sat}^2} \end{bmatrix} \tag{3.6}$$

where:

$$\dot{\gamma} = -\frac{3}{2} J_2 \mu \frac{R_e^2}{r_{sat}^3 h} \sin\theta \sin 2i,$$

and h is the orbit constant angular momentum ($\vec{h} = \vec{r} \times \vec{V}$). The authors of [18, 10], showed how the integration of eq. 3.5, where $\vec{\omega}$ is obtained by integrating the Gauss equations ([19]) gives a very high accuracy. Obviously this cancels all the efforts in obtaining a linear model.
Then the only option is to obtain at least LTV equations, where the angular velocity depends on time.
As already mentioned, in [17] an average on J_2 is performed obtaining constant coefficients linear equations. This assuming, as the authors themselves recognized and reported in [11] also, for J_2 a symmetry which does not consider the orbital plane tumbling.
In [18] and [10] time functions for r and h are suggested. These laws are derived from the behavior of r, inclination and θ due to J_2, they are approximated forms which try to take into account as many effects as possible.

The modeling improvement goes beyond the scope of the present dissertation and here the very simple Hill-Clohessy-Wiltshire constant coefficients equations are used for the controllers design in chapters 4, 5, 6. This means to consider the reference virtual agent on a circular orbit, approximating the Earth to be perfectly spherical and uniform mass distribution. In other words the angular velocity is constant $\overrightarrow{\omega}_{LVLH} = \sqrt{\frac{\mu}{r_{sat}^2}}\hat{z}$, where $\hat{z}$ is the unit vector of z in the LVLH frame.
Eq. 3.5 remains the same but without the J_2 term. The gravity gradient, after some calculation, is:

$$\mathbf{G} = \begin{pmatrix} 2\omega^2 & 0 & 0 \\ 0 & -\omega^2 & 0 \\ 0 & 0 & -\omega^2 \end{pmatrix} \tag{3.7}$$

finally giving the three differential equations of motion projected in LVLH:

$$\begin{cases} \ddot{x} - 2\omega\dot{y} - 3\omega^2 x = u_x \\ \ddot{y} + 2\omega\dot{x} = u_y \\ \ddot{z} + \omega^2 z = u_z \end{cases} \tag{3.8}$$

with:

$$\begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} = \overrightarrow{f_2'} - \overrightarrow{f_1'}$$

the relative acceleration due to non keplerian forces acting on the vehicles.
Dealing with linear constant coefficients equations, i.e. a system in the form $\dot{X} = \mathbf{A}X + \mathbf{B}U$, being X the state (position and velocity: x, y, z, V_x, V_y, V_z), U the control vector (u_x, u_y, u_z) and:

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 3\omega^2 & 0 & 0 & 0 & 2\omega & 0 \\ 0 & 0 & 0 & -2\omega & 0 & 0 \\ 0 & 0 & -\omega^2 & 0 & 0 & 0 \end{pmatrix} \mathbf{B} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.9}$$

the linear systems theory ([54]) can be applied to deduce analytical expressions for the uncontrolled and forced evolution of the state (mathematically speaking: the solution of the associated homogeneous system of 3.8 and the complete one).
At a time instant t:

$$X(t) = e^{\mathbf{A}(\mathrm{t}-\mathrm{t}_0)} \cdot X(t_0) + \int_{t_0}^{t} e^{\mathbf{A}(\mathrm{t}-\tau)} \cdot \mathbf{B} \cdot \mathrm{u}(\tau)\mathrm{d}\tau \tag{3.10}$$

The use of a symbolic calculator, such as the Matlab® one (used for the most of the software developed for this dissertation) helps in finding the explicit solution of the free evolution part. The forcing term can be analytically solved in particular cases, here remains indicated in the general form:

$$X(t) = \Phi(t - t_0) \cdot X(t_0) + \int_{t_0}^{t} e^{\mathbf{A}(\mathrm{t}-\tau)} \cdot \mathbf{B} \cdot \mathrm{u}(\tau)\mathrm{d}\tau \tag{3.11}$$

the expression for the state transition matrix $\Phi(t)$, in normalized form, is reported in the appendix A.

NOTE: some authors (e.g. [29]) use a different LVLH frame where y points from the attraction body to the reference satellite, x is in the $-\overrightarrow{V}$ direction. The HCW equations become:

$$\begin{cases} \ddot{x} - 2\omega\dot{y} = u_x \\ \ddot{y} + 2\omega\dot{x} + 3\omega^2 y = u_y \\ \ddot{z} + \omega^2 z = u_z \end{cases} \tag{3.12}$$

In chapter 6 this modified LVLH system is used, following the notation of [29]. The transition and convolution matrices for the set of eq. 3.12 are reported in appendix A (obviously they slightly differ from the classical LVLH ones).

3.3 Rendezvous and Docking Optimization: Problem Statement

In this section a general formulation for the mathematical problem is given for optimization of rendezvous and docking between two satellites. Two cases are addressed: continuous and impulsive control.

For the continuous case the notation follows that of chapter 2, some over right arrows for vector variables have been omitted for brevity, leaving anyway clarity in the exposition of the results. For the impulsive case, refer to [34] for the notation and the results here briefly reported.

The physical system is shown in detail in Figure 3.3, where a generic manoeuvre for the chaser agent is represented too. The target vehicle is assumed to be on a circular orbit, i.e. in the origin of the LVLH frame. It is also assumed that some attitude control devices, such as reaction wheels, are used on the chaser, which is the only controlled agent (passive target), to maintain its angular velocity to be equal to that of the target, ω_{LVLH}=ω, i.e. the chaser does not change its attitude in the LHLV frame. In other words the attitude is neglected and we assume to be capable of generating the required controls (thrust) along the three axis of the LVLH frame. Therefore, as far as we are concerned, the chaser has only three translational degrees of freedom, whose evolution is given in eq. 3.8.

Chapters 4 and 5 assume the chaser onboard thrusters to be capable of thrust modulation. The reality of space qualified devices is that it is still very difficult to have rockets with a high modulation bandwidth, working in a stable way at all admissible thrust levels.

After the presentation of the two approaches in chapters 4 and 5, the strong practical limitations on thrusters are taken into account in chapter 6, assuming the low-thrust engines to be capable of stay on for long time intervals, but with no modulation.

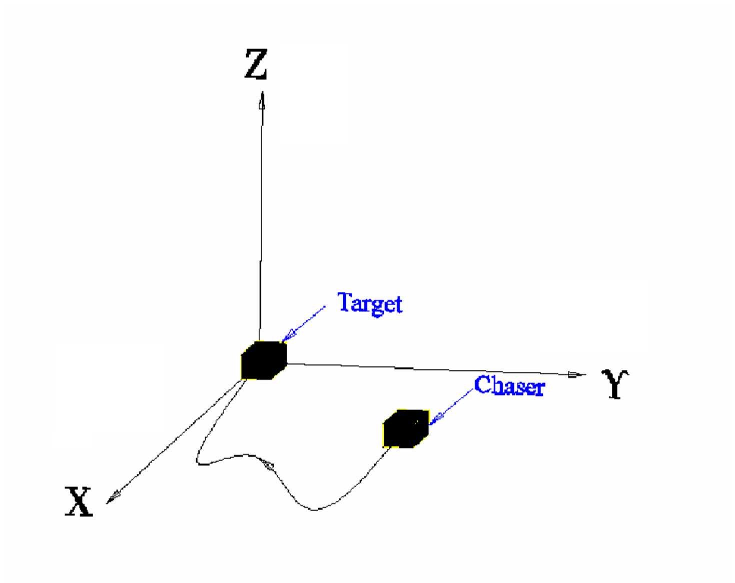

Figure 3.3: Chaser and Target in the LVLH coordinate system

Section 3.3.2 shows the way to set up the problem when impulsive engines are mounted on-board.
A survey of current technology for space engines is given in appendix F.

3.3.1 Continuous Control

The system of linear equations driving the chaser dynamics, written in LHLV coordinate frame, has been derived in eq. 3.8 and is expressed again as:

$$\begin{array}{ll} \dot{x} = V_x & \dot{V}_x = 2\omega V_y + 3\omega^2 x + u_x = f_1(x, V_y) + u_x \\ \dot{y} = V_y & \dot{V}_y = -2\omega V_x + u_y = f_2(V_x) + u_y \\ \dot{z} = V_z & \dot{V}_z = -\omega^2 z + u_z = f_3(z) + u_z \end{array} \tag{3.13}$$

In the vector form system (3.13) looks as follows:

$$\dot{X} = \mathrm{f}(X, \mathrm{U}). \tag{3.14}$$

Again the six states $X = [x, y, z, V_x, V_y, V_z]^T$ are three coordinates and three components of the velocity vector. The chaser's relative position can be controlled by multiple thrusters that can produce independent control inputs in each direction: u_x, u_y and u_z ($\mathrm{U} = [u_x, u_y, u_z]^T$). The magnitudes of these control inputs can be, depending on the hardware, limited by a chosen value u_{max}.
In general it might be required to satisfy the following sets of boundary conditions at the initial

and final points (time instants):

$$\begin{array}{llllll}
x(t_0)=x_0 & \dot{x}(t_0)=\dot{x}_0=V_{x0} & \ddot{x}(t_0)=\ddot{x}_0 & x(t_f)=x_f & \dot{x}(t_f)=\dot{x}_f=V_{xf}\approx 0 & \ddot{x}(t_f)=\ddot{x}_f\approx 0 \\
y(t_0)=y_0 & \dot{y}(t_0)=\dot{y}_0=V_{y0} & \ddot{y}(t_0)=\ddot{y}_0 & y(t_f)=y_f & \dot{y}(t_f)=\dot{y}_f=V_{yf}\approx 0 & \ddot{y}(t_f)=\ddot{y}_f\approx 0 \\
z(t_0)=z_0 & \dot{z}(t_0)=\dot{z}_0=V_{z0} & \ddot{z}(t_0)=\ddot{z}_0 & z(t_f)=z_f & \dot{z}(t_f)=\dot{z}_f=V_{zf}\approx 0 & \ddot{z}(t_f)=\ddot{z}_f\approx 0
\end{array} \tag{3.15}$$

In some cases the accelerations can be considered also, see chapter 5. The chaser starts from whatever current condition it has and should maneuver itself precisely into the docking position with near-zero velocity.
In general constraints on states and controls can be included if required by the particular case under study. These are in general expressed by:

$$g^L \leq g(X, U, t) \leq g^U \tag{3.16}$$

g^L, g^U being respectively the lower and upper limits. The Bolza ([55]) formulation of optimization problem for system (3.13) looks like follows. The problem is to choose the control input U to minimize:

$$J = E\left(X(t_f),\, t_f\right) + \int_{t_0}^{t_f} f_0\left(X(t),\, U(t),\, t\right) dt \tag{3.17}$$

subject to 3.15 (and 3.16) which, for brevity, is:

$$X(t_0) = X_0, \quad X(t_f) = X_f. \tag{3.18}$$

The performance index J of eq. 3.17 includes both the Mayer and Lagrange cases in general (see chapter 2). The cost to minimize can be either the time required to perform the docking maneuver or, much more important in space applications, the overall amount of propellant spent to produce thrust (or their combination). Therefore in Bolza formulation (3.17) $f_0 = 1$ for the time minimum problem, $f_0 = u_x^2 + u_y^2 + u_z^2$ for the propellant minimum problem and $f_0 = 1 + w(u_x^2 + u_y^2 + u_z^2 \ -1)$ in the most general case where w is the weighting coefficient. For the case under study it will be $E\left(X(t_f),\, t_f\right) = 0$.
The transversality conditions (eq. 2.17) do not give any help being the final state represented by the function $e\left(X(t_f),\, t_f\right) = X(t_f) - X_f = 0$. In fact the final value for the costate is obtained deriving the End-Point Lagrangian $\bar{E}\left(\nu,\, X(t_f),\, t_f\right) = E\left(X(t_f),\, t_f\right) + \nu^T e\left(X(t_f),\, t_f\right) = \nu^T\left(X(t_f) - X_f\right)$ with respect to $X(t_f)$. In other words $\Lambda(t_f) = \nu$, which is the lagrangian multiplier associated with the final condition $e\left(X(t_f),\, t_f\right)$, which is unknown.
Let us write the Hamiltonian (see eq. 2.8 and following definition of it) of the system (3.13). For the moment consider no constraints on states and control variables:

$$\begin{aligned}
&H = \lambda_x V_x + \lambda_y V_y + \lambda_z V_z + \lambda_{V_x} f_1(x, V_y) + \lambda_{V_y} f_2(V_x) + \lambda_{V_z} f_3(z) + \\
&+ \lambda_{V_x} u_x + \lambda_{V_y} u_y + \lambda_{V_z} u_z + f_0 = \\
&\lambda_x V_x + \lambda_y V_y + \lambda_z V_z + \lambda_{V_x}(2\omega V_y + 3\omega^2 x + u_x) + \lambda_{V_y}(-2\omega V_x + u_y) + \lambda_{V_z}(-\omega^2 z + u_z) + \\
&+ 1 + w(u_x^2 + u_y^2 + u_z^2 - 1)
\end{aligned} \tag{3.19}$$

The set of adjoint differential equations for the costate vector λ_X then looks like follows ([56], see also the second of eq. 2.13):

$$\begin{aligned}
\dot{\lambda}_x &= -\frac{\delta H}{\delta x} = -3\omega^2\lambda_{V_x} & \dot{\lambda}_{Vx} &= -\frac{\delta H}{\delta V_x} = -\lambda_x + 2\omega\lambda_{V_y} \\
\dot{\lambda}_y &= -\frac{\delta H}{\delta y} = 0 & \dot{\lambda}_{Vy} &= -\frac{\delta H}{\delta V_y} = -\lambda_y - 2\omega\lambda_{V_x} \\
\dot{\lambda}_z &= -\frac{\delta H}{\delta z} = \omega^2\lambda_{V_z} & \dot{\lambda}_{Vz} &= -\frac{\delta H}{\delta V_z} = -\lambda_z
\end{aligned} \tag{3.20}$$

that is the adjoint vector shows a linear dynamics very similar to that of HCW equations. To be more precise:

$$\begin{bmatrix} \dot{\lambda}_x \\ \dot{\lambda}_y \\ \dot{\lambda}_z \\ \dot{\lambda}_{V_x} \\ \dot{\lambda}_{V_y} \\ \dot{\lambda}_{V_z} \end{bmatrix} = -\mathbf{A}^{\mathrm{T}} \cdot \begin{bmatrix} \lambda_x \\ \lambda_y \\ \lambda_z \\ \lambda_{V_x} \\ \lambda_{V_y} \\ \lambda_{V_z} \end{bmatrix} \tag{3.21}$$

likewise eq. 3.11:

$$\begin{bmatrix} \lambda_x(t) \\ \lambda_y(t) \\ \lambda_z(t) \\ \lambda_{V_x}(t) \\ \lambda_{V_y}(t) \\ \lambda_{V_z}(t) \end{bmatrix} = \Phi_\lambda(t-t_0) \cdot \begin{bmatrix} \lambda_x(t_0) \\ \lambda_y(t_0) \\ \lambda_z(t_0) \\ \lambda_{V_x}(t_0) \\ \lambda_{V_y}(t_0) \\ \lambda_{V_z}(t_0) \end{bmatrix} \tag{3.22}$$

Refer to appendix A for the normalized Φ_λ. Deriving eq. 3.19 with respect to the controls and imposing the derivatives to be zero gives the optimal control expression (first necessary condition of eq. 2.13; it is also sufficient in this case of linear dynamic):

$$\begin{bmatrix} u_x^* \\ u_y^* \\ u_z^* \end{bmatrix} = -\frac{1}{2w} \begin{bmatrix} \lambda_{V_x} \\ \lambda_{V_y} \\ \lambda_{V_z} \end{bmatrix} \tag{3.23}$$

Keep searching for the solution with no constraints, the linear systems theory helps again in obtaining an analytical expression for the optimal trajectory. In fact, substituting eq. 3.22 (the only part which is needed, i.e. the adjoint velocity) in eq. 3.23 and finally in eq. 3.11, leads to (again a symbolic calculation):

$$X(t) = \Phi(t-t_0)\cdot X(t_0) + \frac{1}{2w}\Psi(t-t_0)\cdot\Lambda_0 \tag{3.24}$$

being Ψ reported in appendix A. The initial condition on the costate can be found matching the boundary conditions at final time:

$$\Lambda_0 = \Psi^{-1}(t_f - t_0)2w\left(X(t_f) - \Phi(t_f - t_0)\cdot X(t_0)\right) \tag{3.25}$$

But what if some constraints are introduced on the states or controls or both of them? Eq. 3.16 has to be taken into account by defining an augmented Hamiltonian with respect to eq. 3.19, or Lagrangian of the Hamiltonian (augmentation of the Hamiltonian function, as done in chapter 2):

$$\begin{aligned}&\bar{H} = \lambda_x V_x + \lambda_y V_y + \lambda_z V_z + \lambda_{V_x} f_1(x, V_y) + \lambda_{V_y} f_2(V_x) + \lambda_{V_z} f_3(z) + \\ &+ \lambda_{V_x} u_x + \lambda_{V_y} u_y + \lambda_{V_z} u_z + f_0 = \\ &\lambda_x V_x + \lambda_y V_y + \lambda_z V_z + \lambda_{V_x}(2\omega V_y + 3\omega^2 x + u_x) + \lambda_{V_y}(-2\omega V_x + u_y) + \lambda_{V_z}(-\omega^2 z + u_z) + \\ &+ 1 + w(u_x^2 + u_y^2 + u_z^2 - 1) + \mu^T g(X, U, t)\end{aligned} \tag{3.26}$$

where $\mu^T = [\mu_1, \mu_2, ..., \mu_{N_{con}}]$ are the Lagrangian multipliers, $g(...) = [g_1(...), ..., g_{N_{con}}(...)]^T$, being N_{con} the number of constraints to be met. The condition for a control sequence to be optimal is still the derivatives with respect to U equaling zero, the Lagrangian variables respecting the Karush-Kuhn-Tucker (KKT) conditions:

$$\mu_i = \begin{cases} \leq 0, & g_i(X, U, t) = g_i^L \\ = 0, & g^L < g_i(X, U, t) < g_i^U \\ \geq 0, & g_i(X, U, t) = g_i^U \\ unrestricted, & g^L = g_i^U \end{cases} \tag{3.27}$$

It is now obvious how the reality of existing limitations on engines, forbidden regions on the state space, etc., do not leave the possibility for an analytical solution such as in eq. 3.24. In fact there is no closed form solution for the Lagrangian multipliers, as there is for the costate (eq. 3.22). Depending on the particular problem one numerical method can be preferred among the others.

One last observation for the case of limited thrust magnitude: looking at eq. 3.23 one can recognize the well-known bang-unconstrained-bang structure for the fuel optimal case ($w = 1$), the bang-bang one typical of time optimal manoeuvres, after having re-written U^* as (k=$\{x,y,z\}$):

$$u_k^* = sign \lambda_{V_k} \min\left(\frac{|\lambda_{V_k}|}{2w}, u_{max}\right) \tag{3.28}$$

In chapter 4 the here illustrated results are not used, being the approach particularly intended to reduce fuel consumption on a predetermined trajectory.

In chapter 5 the variational results are partly used, in particular only for pre-shaping one of the controls for a direct sub-optimization algorithm.

While, in chapter 6 the variational results are the base for deducing the optimal control sequence. As above mentioned, constraining the problem means numerical solution. Then, in chapter 6, the combination of the variational principles with ad-hoc developed algorithms, leads to the problem solution.

3.3.2 Impulsive Control

When considering high-thrust engines a common approximation is to assume the burn intervals to be very short compared to the manoeuvre duration. Lawden's work ([34]) on impulsive optimal space trajectories is the milestone for this approach. An impulse at time t_j is an instantaneous variation of the chaser vehicle velocity; using the Dirac's delta notation: $\Delta\vec{V}\delta\left(t-t_j\right)$. In eq. 3.11 only the free evolution of the state can be considered, updating the boundary conditions at the impulse time any time an impulse occurs, varying the velocity vector at that instant, not the spatial position of the chaser. Impulsive controls can be then translated into reality, considering the maximum thrust of the engines u_{max} and calculating a burn time interval which gives the same velocity variation. During this Δt the engines run at maximum thrust. Δt has to be as small as possible in order to not loose the accuracy in reaching the desired final condition. This operation is called PWM:

$$\Delta t = \frac{\Delta V}{u_{max}} \tag{3.29}$$

The control vector of eq. 3.12 is here written, for following convenience (see chapter 6), as $\Gamma\hat{u}$, having partitioned it into its magnitude Γ ($= \Delta V$ for impulsive case) and its direction $\hat{u}$. On the whole manoeuvre it is $\Gamma\hat{u} = \sum_{j=1}^{N} \Delta\vec{V}_j\delta\left(t-t_j\right)$.

For linear systems optimization problems the Neutstadt ([46]) theorem limits the number of impulses to be at most equal to that of the final conditions to be met. For the RDD case it means maximum six impulses to reach zero relative position and velocity between the two satellites. The minimization of fuel consumption means to find the number N, the time instants t_j, space collocation $\vec{r}_j$, and values $\Delta\vec{V}_j$ of the impulses to be applied for the dynamics in 3.12, in order to minimize the cost function:

$$J = \int_{t_0}^{t_f} \Gamma dt = \sum_{i=1}^{N} \Delta V_i,\ N = 2..6 \tag{3.30}$$

Not considering the possibility of initial and final coasting phases, that is the first impulse occurs at t_0, the last one at t_f, the number of mid-course impulses is limited to at most four ([33]). The optimization problem just stated is now translated into the so-called primer vector conditions for impulsive trajectories. The primer vector $\vec{p}$ is the adjoint velocity $\vec{\lambda}_V$ of eq. 3.20, re-named after Lawden, who derived the necessary conditions it has to satisfy ([34]). The control Hamiltonian can be written after a quick, intuitive consideration. An impulse can be considered as an infinite control acceleration (its effect is instantaneous), looking at eq. 3.26 and 3.27 it can be imagined as a constrained control always working at the maximum admissible limit: ∞. This intuitively explains the minus in the following expression (refer to [34] for the details):

$$\begin{aligned}\bar{H} = &-\Gamma + \lambda_x V_x + \lambda_y V_y + \lambda_z V_z + p_x f_1(x, V_y) + p_y f_2(V_x) + p_z f_3(z) + \\ &+ p_x \Gamma \hat{u}_x + p_y \Gamma \hat{u}_y + p_z \Gamma \hat{u}_z = \\ &-\Gamma + \Gamma(\vec{p}\cdot\hat{u}) + \lambda_x V_x + \lambda_y V_y + \lambda_z V_z + p_x(2\omega V_y + 3\omega^2 x) + \\ &p_y(-2\omega V_x) + p_z(-\omega^2 z)\end{aligned} \tag{3.31}$$

Note how, for a lighter notation, the sum of the ΔVs is not showed in eq. 3.31 and in the following. This obviously does not affect the result.

As showed in [40] the Hamiltonian can be maximized by choosing the thrust direction $\hat{u}$ parallel to the primer vector, leading to the maximum scalar product in eq. 3.31. This gives:

$$\bar{H} = (p-1)\Gamma + \lambda_x V_x + \lambda_y V_y + \lambda_z V_z + p_x(2\omega V_y + 3\omega^2 x) + p_y(-2\omega V_x) + p_z(-\omega^2 z) \tag{3.32}$$

At this point the Lawden's necessary (and sufficient, see observation on linear systems, chapter 2) conditions of optimality can be listed, (see [34] for their derivation and more details):

1. $p \in C^1$;
2. during coasting ($\Gamma = 0$) $p < 1$;
3. at an impulse $p = 1$, tangent to 1 from below;
4. at an impulse time $\hat{u} = \vec{p}$;
5. $\dot{p}(t_0) < 0$, otherwise initial coast is needed;
6. $\dot{p}(t_f) > 0$, otherwise final coast is needed.

Conditions on primer are also sufficient to give the optimum, being the cost function independent from the state ([57]).
In the approach of chapter 6 a numerical technique is used for collocating the optimal sequence of impulses for RDD. In the same section the criteria to reduce the cost by adding new impulses to a previously determined sequence, and moving their position in time and space, are presented too. The numerical way of solving the problem is inherited by previous literature ([37], [38], etc.).

Chapter 4

Real-Time Time/Fuel Optimal Control on Fixed Path

4.1 Introduction

In this chapter the Clohessy-Wiltshire equations 3.8 are taken as dynamic model and inverted, after a variable change, in order to be used by a control algorithm to drive the chaser spacecraft along a specified path. Path parameterization is performed through cubic B-splines having the curvilinear abscissa as parameter. The proposed optimization algorithm uses dynamic programming to find the time or fuel quasi-optimal controls, and mitigate the "curse of dimensionality", issue of Bellman's approach [58], by working only on the acceleration component along the vehicle trajectory. Therefore, the number of optimization parameters is drastically reduced and it is possible to constrain the tangential acceleration value. One can choose the shape of the path according to the specific manoeuvre requirements and, if needed, modify it onboard by varying the splines control points. The optimization algorithm is split into a trajectory planner which generates the best tangential acceleration sequence through backward exploration of a tree of possible policies, and a control generator which inverts the parameterized dynamics in order to get the thrusters commands sequence. The optimization algorithm has been coded in Simulink as a library of Embedded Functions and has been experimentally proved to run in Real Time on a Pentium II machine with a sample time of 0.2 seconds for the planner. The approach considers no limitation on the thrusters, i.e. unbounded control, and thrust modulation ability.
The proposed strategy is a direct optimization method similar to the one used in [59] and in [60] for the control of robotic manipulators. The basic idea is to parameterize the trajectory in a way that allows for independent choice of path and velocity profile. In order to obtain such feature, the curvilinear abscissa is used as in [59].
Dynamic Programming ([58]) is here used as the optimization method. This approach has been proved to be suitable for real-time implementation, especially when a sub-optimal solution is searched through a step-by-step optimization of the trajectory ([61, 62]).
It is worth remind the reader that the variational results shown in chapter 3 are here not used. The methodology here developed aims planning the control sequence on a fixed trajectory with an ad-hoc direct technique, in order to minimize (optimize or rather sub-optimize) the fuel con-

sumption and/or time.

4.2 Inverting the Dynamic

The proposed approach is based on the search for a suboptimal policy to drive the chaser vehicle along a specified docking path.
The objective of the method is twofold. First, we want to be able to impose a certain path in order to guarantee a priori the safety of the manoeuvre and track it with minimum propellant consumption ([29]). Second, we want the solution to be suitable for real time implementation.
The search for the optimal or sub-optimal control on a specified path can be seen as the constrained problem discussed in chapter 3.3.1. The constraint of eq. 3.16 is represented by eq. 4.1 where the specified path is parameterized in terms of the curvilinear abscissa as follows ([59]):

$$\vec{r}_{rel} = \vec{r}\,(s) \quad = \quad [x\,(s)\,,y\,(s)\,,z\,(s)] \tag{4.1}$$

Noteworthy one may eliminate s and deduce the path shape as $y = y\,(x)\,,\ z = z\,(x)$. Differentiation with respect to time and substitution of this relation into (3.5), with no J_2 effects, gives:

$$\frac{\delta^2\vec{r}}{\delta s^2}\dot{s}^2 + \frac{\delta\vec{r}}{\delta s}\ddot{s} + 2\vec{\omega}_{LVLH} \times \frac{\delta\vec{r}}{\delta s}\dot{s} - \omega_{LVLH}^2\vec{r}^{*} = \frac{\mu}{r_t^5}\left(3\underline{\underline{r_t}} - r_t^2\underline{\underline{1}}\right)\vec{r} + \vec{u} \tag{4.2}$$

The tangential acceleration profile $\ddot{s}$ to be tracked by the chaser spacecraft is the free parameter for the optimization. The actual controls (accelerations along the three axis) can be then easily determined by inverting the dynamics:

$$\vec{u} = -\frac{\mu}{r_t^5}(3\underline{\underline{r_t}} - r_t^2\underline{\underline{1}})\vec{r} + \frac{\delta^2\vec{r}}{\delta s^2}\dot{s}^2 + \frac{\delta\vec{r}}{\delta s}\ddot{s} + 2\vec{\omega}_{LVLH} \times \frac{\delta\vec{r}}{\delta s}\dot{s} - \omega_{LVLH}^2\vec{r}^{*} \tag{4.3}$$

where $\overrightarrow{r}_t$ is the target radius on circular orbit in LVLH frame and the dyadic notation is used (given a vector $\overrightarrow{a} = [a_x,\ a_y,\ a_z]$, the associated dyadic is $\underline{\underline{a}} = \begin{pmatrix} a_xa_x & a_xa_y & a_xa_z \\ a_ya_x & a_ya_y & a_ya_z \\ a_za_x & a_za_y & a_za_z \end{pmatrix}$). In equation (4.3) it is not straightforward to impose a limit on the controls when one works in term of acceleration.
By choosing a "sufficiently smooth" trajectory (i.e. limiting ρ, the local curvature of the path) and limiting $\ddot{s}$ one can avoid undesired peaks in the control vector $\vec{u}$.
Once the control history on the chosen path, relative to a certain $\ddot{s}$ sequence, is determined via eq. 4.3, the cost function (eq. 3.17) is computed. The sequence giving the lowest value of the cost is the optimal one. In the following simulations fuel is optimized ($w = 1$ in eq. 3.17).

4.3 Geometric representation of the path

Cubic B-spline curves are used to represent the trajectory $\vec{r}_{rel} = \vec{r}\,(s)$. B-splines have the advantage of narrowly propagating the local changes [63]. Given n control points, a first and second order continuous curve which fits them is univocally determined by a composition of (n-1) B splines. As the positions of control points change, the curve shape changes consequently. Each spline is defined by four control points and has the parametric representation:

$$\vec{r}(\vec{\gamma}) = \vec{\gamma}\mathbf{B}\mathbf{K} \tag{4.4}$$

Where $\vec{r}(\vec{\gamma})$ is the position vector of a generic point of the spline, $\vec{\gamma}$ is the parameter vector, defined as:

$$\vec{\gamma} = \begin{bmatrix} \gamma^3 & \gamma^2 & \gamma & 1 \end{bmatrix}, \; 0 \leq \gamma \leq 1 \tag{4.5}$$

K contains the vector positions of the control points:

$$\mathbf{K} = \begin{bmatrix} \vec{r}_0 & \vec{r}_1 & \vec{r}_2 & \vec{r}_3 \end{bmatrix}^T \tag{4.6}$$

and **B** is the universal transformation matrix, containing the same numerical values, obtained imposing continuity, for every B spline [63]:

$$\mathbf{B} = \frac{1}{6}\begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{pmatrix} \tag{4.7}$$

Accordingly, the partial derivatives of the path with respect to the arc length, needed in (4.2) and (4.3), are given by:

$$\begin{aligned} \frac{\delta\vec{r}}{\delta s} &= \frac{\frac{\delta\vec{r}}{\delta\gamma}}{\left|\frac{\delta\vec{r}}{\delta\gamma}\right|} \\ \frac{\delta^2\vec{r}}{\delta s^2} &= \frac{\frac{\delta^2\vec{r}}{\delta\gamma^2}}{\left|\frac{\delta\vec{r}}{\delta\gamma}\right|^2} - \frac{\delta\vec{r}}{\delta\gamma}\frac{\frac{\delta\vec{r}}{\delta\gamma}\cdot\frac{\delta^2\vec{r}}{\delta\gamma^2}}{\left|\frac{\delta\vec{r}}{\delta\gamma}\right|^4} \end{aligned} \tag{4.8}$$

where:

$$\begin{aligned} \frac{\delta\vec{r}}{\delta\gamma} &= \begin{bmatrix} 3\,\gamma^2 & 2\gamma & 1 & 0 \end{bmatrix}\mathbf{BK} \\ \frac{\delta^2\vec{r}}{\delta\gamma^2} &= \begin{bmatrix} 6\,\gamma & 2 & 0 & 0 \end{bmatrix}\mathbf{BK} \end{aligned} \tag{4.9}$$

Since the B splines, by definition, do not pass through the end points, two additional artificial control points are added at the extremes of the curve (see appendix B). Remember that the denominator of (4.8) is always strictly positive when considering a regular curve. In order to convert s to a value of γ it is sufficient to find (numerically [64])), once the corresponding spline is selected, the zero of the function $h(\gamma) = s - \left(s_{left_{current\,spline}} + \int_0^\gamma \frac{ds}{d\gamma} d\gamma\right)$. For the sake of completeness we remind that: $\frac{ds}{d\gamma} = \sqrt{\left(\frac{dx(\gamma)}{d\gamma}\right)^2 + \left(\frac{dy(\gamma)}{d\gamma}\right)^2 + \left(\frac{dz(\gamma)}{d\gamma}\right)^2}$.

From eq. 4.6 it is clear that four points are needed for a single spline. The software developed is capable of parameterize the path with a number of control points bigger than 4, no matter if odd or even. This flexibility is achieved by overlapping the splines, shifting of one control point any time a new spline is generated, with no significant increase of computation time.

4.4 Optimization Approach: Trajectory Planner by Dynamic Programming

Dynamic programming ([58]) is very useful in problems where one needs to take subsequent decisions. The word "Dynamic" states that the decisions are sequentially taken and the future is influenced by the past. Sequenced problems can be solved according to the Bellman's Principle of Optimality:

"*An optimal strategy has the property that, no matter the initial state and decision, the future decision's set has to constitute an optimal strategy with respect to the state reached according to the decisions taken until that moment*".

Dynamic Programming takes the decisions one by one. At every step the optimal policy for the future is found, independently of the past decisions. The cost function is divided in the sum of elementary costs, one for each step of the trajectory. In the present work a similar approach to that used in [61] is considered.
The curvilinear acceleration on the specified curve is taken as the parameter for the optimization. In this way the issue of "curse of dimensionality" ([58]) is greatly mitigated, by limiting the number of variables the algorithm has to work with. Once the optimal $\ddot{s}$ profile is determined, the controls are calculated according to Eq. (4.3). On a certain section of the entire path, where the acceleration $\ddot{s}$ is kept constant, we know that the section length ds needs a time interval t to be run:

$$\begin{aligned} t &= -\frac{\dot{s}_0}{\ddot{s}} \pm \sqrt{\left(\frac{\dot{s}_0}{\ddot{s}}\right)^2 + \frac{2ds}{\ddot{s}}},\ where: \\ s(t) &= s_0 + \dot{s}_0 t + \ddot{s}\frac{t^2}{2} \\ ds &= \dot{s}_0 t + \ddot{s}\frac{t^2}{2} \rightarrow t^2 + \frac{2\dot{s}_0}{\ddot{s}}t - \frac{2ds}{\ddot{s}} = 0 \end{aligned} \tag{4.10}$$

Dividing the complete trajectory into sub-intervals of length ds, the algorithm tests the different possibilities, in terms of $\ddot{s}$, for the chaser to track the specified path. In this way one can run the algorithm until the final condition is reached: $\begin{cases} s(t_f) = s_f = trajectory\ length \\ \dot{s}(t_f) = 0 = final\ velocity \end{cases}$
The tree is a graph structure, with no cycles, in which every node generates different branches, every one connected to a subsequent node, called son-node. A son can generate other sons. Every node, apart from the root, has one and only one entering branch, belonging to the parental node (or father-node). The root does not have any father.
Starting from the initial condition, the tree of possible trajectories is built in an incremental

way, exploring the reachable states in order to find the optimal sequence. Every step adds new branches (so new nodes) to the tree, only if they are acceptable: in particular, only positive velocities are admitted. Therefore, nodes with zero velocity and zero acceleration are not taken into account. Moreover, the cases when Eq. (4.10) does not have real solutions are discarded, since this implies that the current section cannot be run with the current value of $\ddot{s}$. For instance, this occurs if the acceleration is negative and the square root in (4.10) returns a complex solution, then we are facing the just mentioned case.
A cost value is associated to every node and, once the stopping condition is achieved, the algorithm finds the node with the minimum cost. At this point it is necessary to run backwards the tree to find the optimal policy, the one which brings to the final, minimum cost, node. The cost is calculated through a trapezoidal integration formula (refer to Numerical Recipes [64]) of the control magnitude along with time.
The algorithm structure follows (refer to Figure 4.1):

1. start from the initial condition s_0, $\dot{s}_0$ (ROOT)

2. apply the chosen values of $\ddot{s}$ until the trajectory span ds is completed, obtaining the first generation;

3. calculate the cost function for each node of the present generation;

4. starting from every son, repeat the step 2 and obtain the second generation;

5. iterate step 2 and 3 until the end s_f, $\dot{s}_f$ is reached;

6. recognize the minimum cost node of the last generation. In Figure 4.1 at the last level the numbers at the top report, as an example, the total cost associated with every node;

7. individuate the optimal policy, by starting from the optimal son of the last generation and run the tree backward.

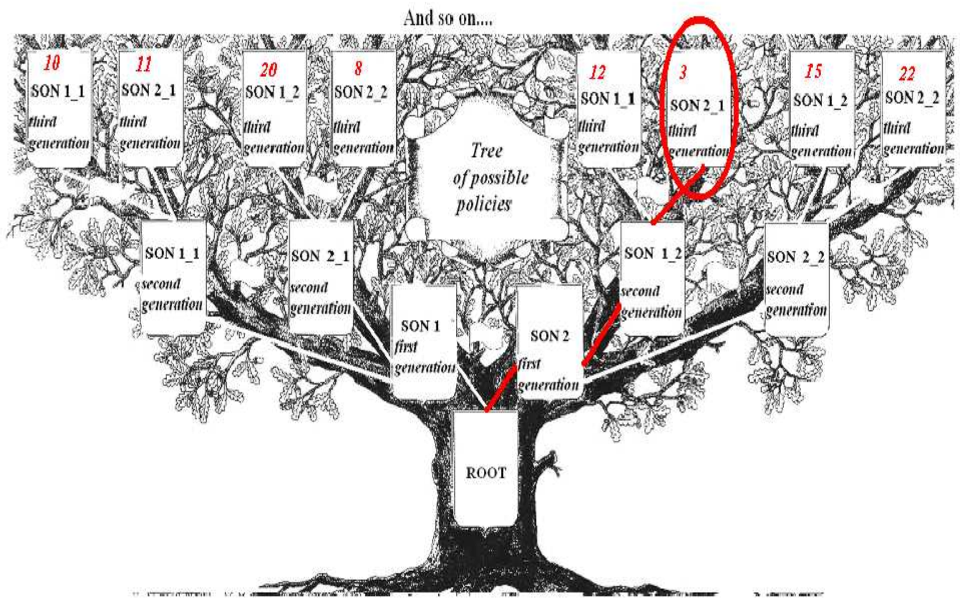

Figure 4.1: Tree of possible policies

From the software point of view each son-node is represented by a 7 elements array:

- value of the curvilinear abscissa s (this value is the same for every node of the same generation);

- value of the velocity along the path $\dot{s}$;

- value of the acceleration along the path $\ddot{s}$ (“control”);

- time to reach the node from the previous one;

- cost associated to the last branch;

- total time (from beginning of the path) to run until that point;

- index (identifier of the father).

By keeping memory of the father for every node it is possible to quickly run the tree backwards once the final condition is reached in order to find the best $\ddot{s}$ profile. The number of generations N_L (that is, the number of spans in which the path is divided) and the number N_C (and values) admissible for $\ddot{s}$ can be selected by the user, keeping in mind that the final level can reach a maximum of $N_C^{N_L}$ nodes, therefore the required memory and computation time increase very quickly with the number of levels. As for the $\ddot{s}$ testing values, the one giving zero velocity, starting from an $\dot{s}_0 \neq 0$ and requiring a ds space to bring $\dot{s}$ to zero, is included among the possibilities (with its multiples). It is easy to demonstrate that the mentioned value is $\ddot{s} = -\frac{\dot{s}_0}{2ds}$. By doing this, we have at least one node with zero velocity at the final level. In the case of a rest-to-rest manoeuvre the user can give any tangential acceleration value below a specified threshold.

4.5 Pruning

The tree exploration is the most time consuming task for the algorithm. Keeping all the branches with no check on their physical meaning could result in a waste of resources. Let us imagine to be at a generic level of the tree structure, at distance s (measured along the path) from the target position, at velocity $\dot{s}$. If the remaining length of trajectory to be run is not sufficient to brake down and reach the final point with zero velocity, there is no reason to keep generating new branches from that node. According to this criterion the new nodes are analyzed while building the tree and pruned if not useful, if nonsense.

4.6 Simulation Results: testing and comparing the algorithm

Two sample simulations are here presented as significant cases in order to show the features of the proposed approach.

4.6.1 Simulation test 1

This simulation shows the behavior of a chaser spacecraft toward a target spacecraft as a function of the number of path segments, in order to study its response and reliability. The manoeuvre is an example where the initial position is $500m$ from the target and a cubic trajectory $(y(x) = \frac{y_0}{x_0^3}x^3)$ has to be tracked until the origin of the reference frame is reached.
The numerical values used for the simulation are as follows:

Table 4.1: Numerical values for cubic manoeuvre

Parameter	**Units**	**Value**
Height above the Earth surface	km	480
Maximum (minimum) tangential acceleration $\ddot{s}_{\max}$ $(-\ddot{s}_{\min})$	$\frac{m}{s^2}$	$5 \cdot 10^{-4}$
Initial velocity $\dot{s}_0$	$\frac{m}{s}$	0.2
Initial position (x_0, y_0, z_0)	m	$(300, 400, 0)$
Number of levels N_L	-	5
Number of possible tangential acceleration values N_C	-	3

The same trajectory has been obtained for a different number of levels. This result is reported in Figure 4.2. No matter the number of levels, the dynamic is driven into the required path, as the nature of the algorithm implies. Figure 4.3 shows the fuel consumption and time required for

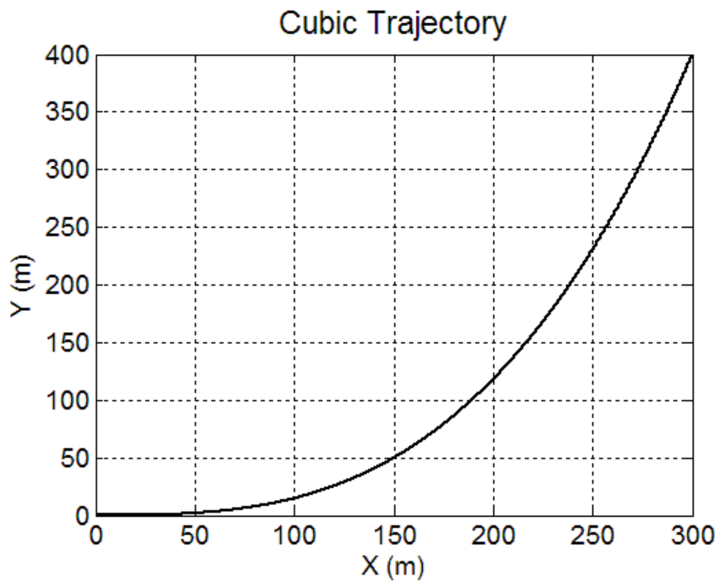

Figure 4.2: Cubic path tracked by the chaser

the manoeuvre, Figure 4.4 the CPU time (on a Pentium IV machine). The independent variable is the number of levels. It is apparent in Figure 4.4 that increasing the number of levels means a larger tree to build and explore, i.e. higher CPU resources. Figure 4.3 a) is worth some comment on the goal of the proposed technique. Dividing the trajectory in a finite number of segments, and imposing a limited set of values for the command acceleration, should bring to the well known bang-bang or bang-off-bang profile for $\ddot{s}$ (depending on what we are optimizing and the initial conditions). We do not know a priori where, along the length of the path, the switching points are positioned. We do not even know how many switches we should expect in general. The tree approach gives a near-optimal solution, imposing how many switches the policy has to

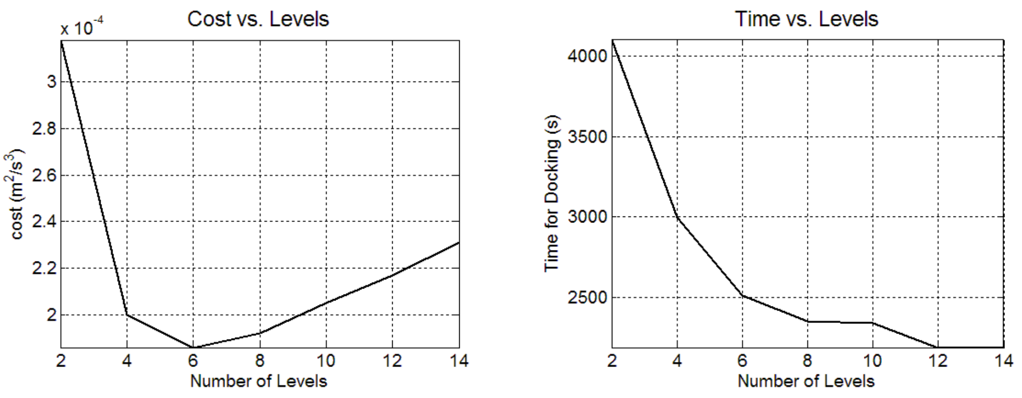

Figure 4.3: a) Fuel vs. levels; b) Time for manoeuvre vs. levels

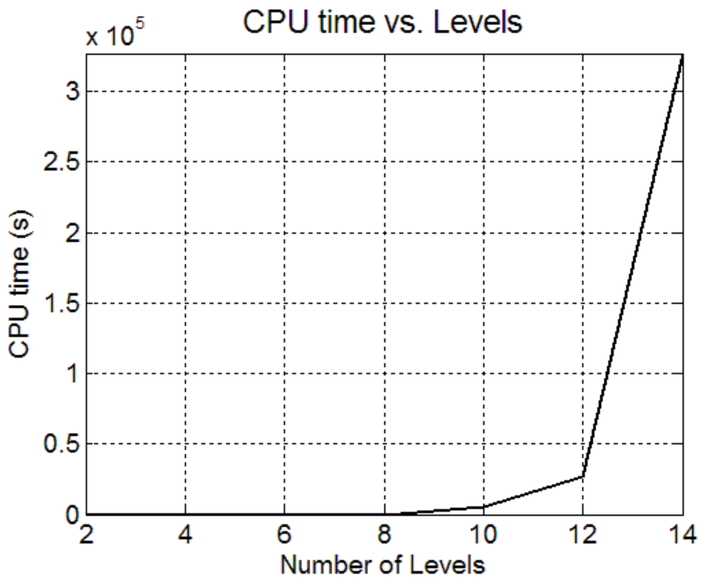

Figure 4.4: CPU time vs. n. of levels

have and where (that is, it imposes N_L); by doing this we do not know how far it is from exactly catching the optimal solution. For this reason, increasing N_L guarantees an higher probability of obtaining the optimal switching structure for the $\ddot{s}$, as long as the number of levels is not too high and there are more switches than in the optimum solution ($N_L \to \infty$ obviously is the ideal condition to obtain the optimum). Testing the algorithm for different manoeuvres it results, as indicated in Figure 4.3 a), that a reasonable number of levels is between 4 and 6. The cost is satisfying and the CPU time is reasonable. Figure 4.5 reports the controls for different number of levels. As expected, increasing N_L the controls tend to have a discrete time evolution, the time required for manoeuvring decreases, while the control magnitude increases, tending to the optimal solution.

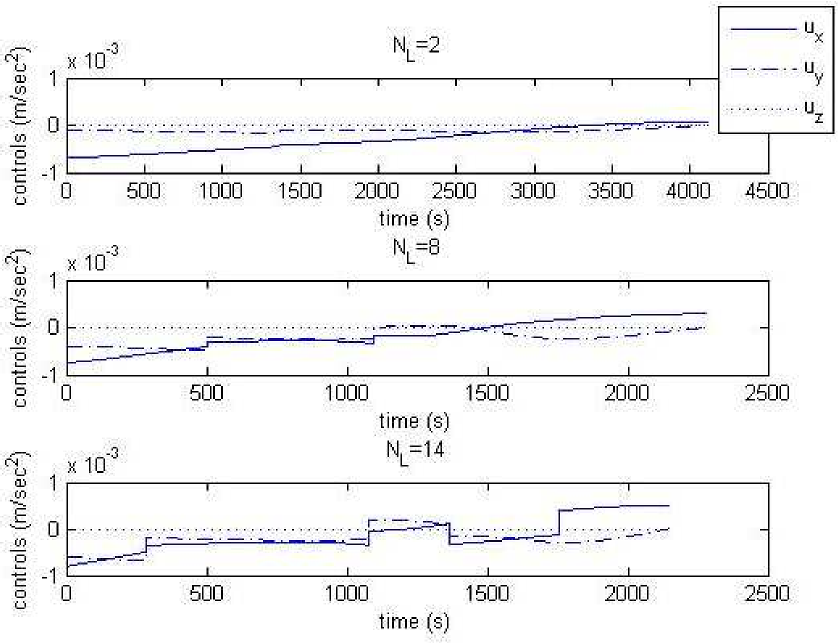

Figure 4.5: Controls behavior increasing N_L for cubic manoeuvre

4.6.2 Simulation test 2

As done in [29] by Guelman et al. we manoeuvre the chaser into a straight line approach along the y direction. We show here how the tree strategy is able to generate similar results but in a quicker way, being the velocity of the algorithm demonstrated and measured (see next paragraph). Initial and final conditions are exactly satisfied, not requiring the introduction of some small values to represent them, as done, for example, in [29]. This manoeuvre is the final stage bringing to docking. The initial conditions here adopted (table 4.2) are intermediate conditions (breakpoint between two stages) for the authors of [29].

Table 4.2: Numerical values for straight line manoeuvre

Parameter	**Units**	**Value**
Height above the Earth surface	km	480
Maximum (minimum) acceleration $\ddot{s}_{\max}$ $(-\ddot{s}_{\min})$	$\frac{m}{s^2}$	$5 \cdot 10^{-4}$
Initial velocity $\dot{s}_0$	$\frac{m}{s}$	0.2
Initial position (x_0, y_0, z_0)	m	(0, 300, 0)
Number of levels N_L	-	5
Number of possible tangential acceleration values N_C	-	3

Figure 4.6 shows that the trajectory is perfectly tracked. Controls, total acceleration and velocity

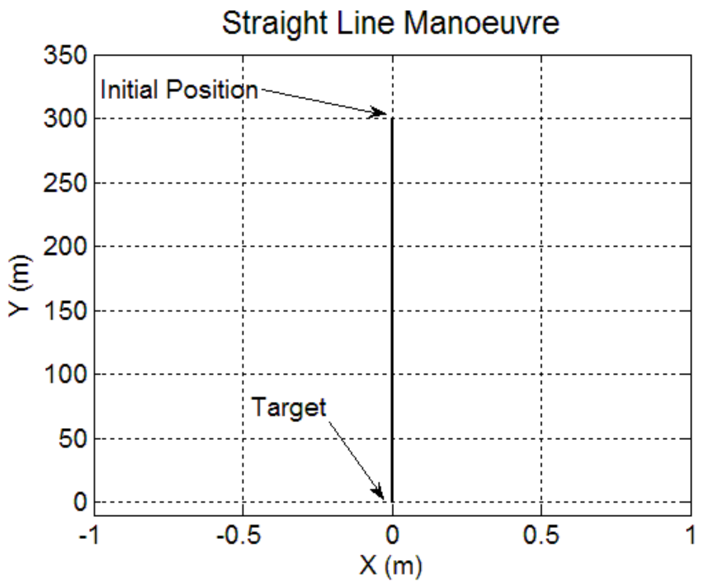

Figure 4.6: Straight line trajectory

profile are reported on Figure 4.7 and Figure 4.8. The time required to complete the docking

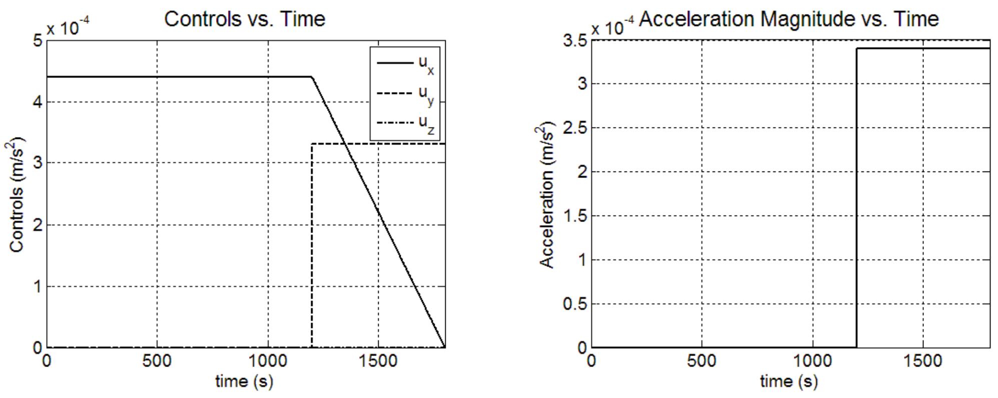

Figure 4.7: a) Controls vs. time; b) Acceleration vs. time

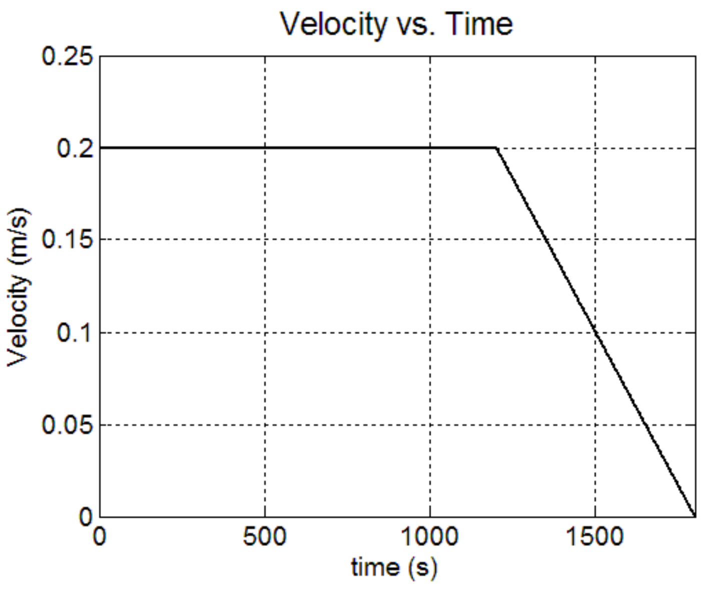

Figure 4.8: Velocity vs. time

is $1800\,s$, with a total cost of $1.566 \cdot 10^{-4}\,\frac{m^2}{s^3}$, while in [29] the corresponding values are: $1885\,s$, $1.538 \cdot 10^{-4}\,\frac{m^2}{s^3}$. Discrepancies in cost and time are due to the fact that in [29] small values are introduced for the zero final boundary conditions, that is, they are not exactly matched, whereas we are here manoeuvring between two specified and accurately achieved positions and velocities.

4.7 Real Time Validation

Real-Time systems are computer that process real-world events as they happen, under the constraint of a real-time clock, and that can implement algorithms in dedicated hardware. Examples include mobile telephones, test and measurement devices, and avionic and automotive control systems. Real-time validation of any software means a practical test of its capabilities of repeating the whole set of tasks it is composed by in an amount of time always lower than a fixed limit. The expression real-time derives from the real possibility of using that particular software with the operating machine it is supposed to command (the hardware). In other words real-time validation of a software on its particular target machine (the one it will drive) means validating what will be the final product. Imagine we need to control a system for which an update of information (control and other software computed quantities) is needed every x seconds. x will be the sample time. If the programs driving the system are capable of terminating their calculations within a time frame $< x$ then the real-time capability of the software has been proven. This fact obviously depends on the algorithm velocity and hardware velocity.
This kind of test is here performed assuming a simple pc as target hardware (its computation velocity would represent the velocity of the satellite onboard calculator), downloading the software on the pc through the Matlab® automatic C code generator and compiler Real-Time Workshop (Figure 4.8).

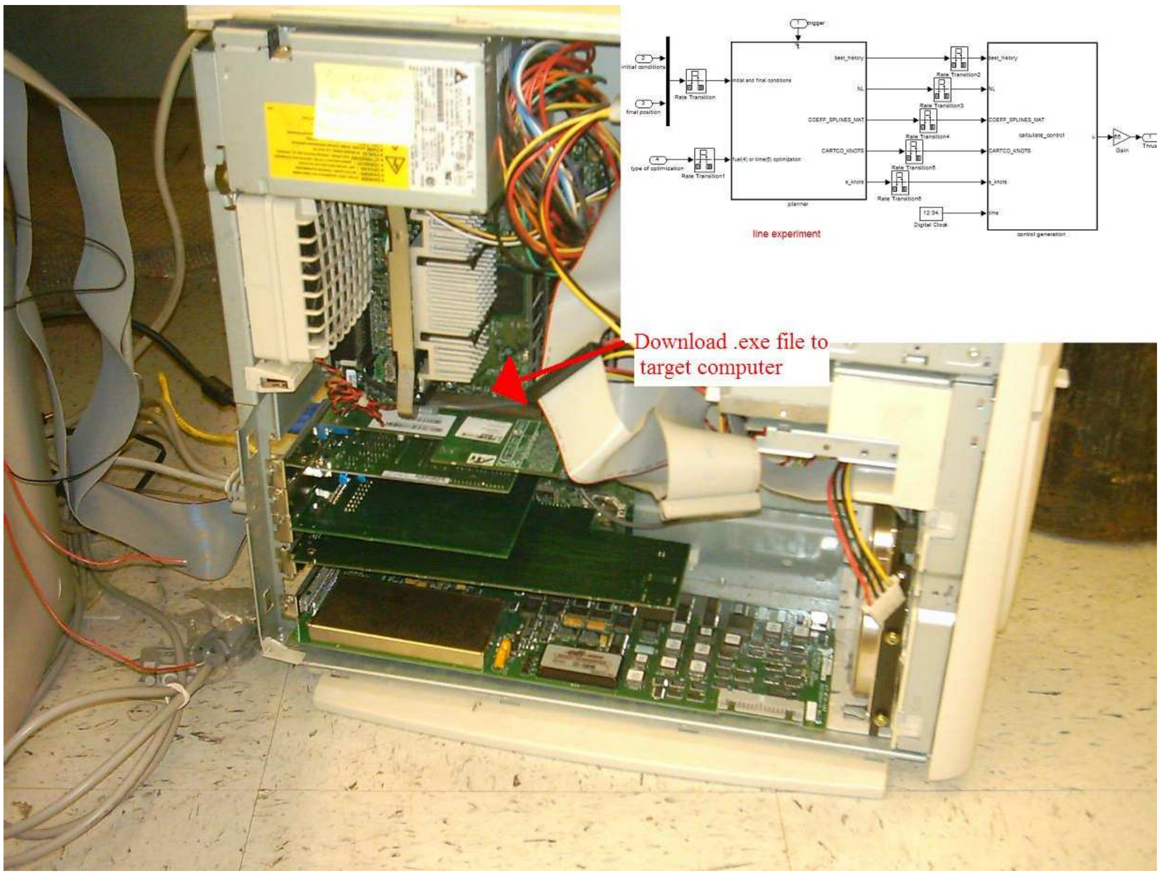

Figure 4.9: Downloading Compiled Software on Target Machine through Real-Time Workshop

Dynamic Programming appears a valid method also for real time implementation of near-optimal control. In 1997 Miles ([62]) applied it to a free flying robot required to avoid several obstacles during its manoeuvring. Figure 4.10 shows the Simulink model developed for the present work, composed by two main Embedded Functions. The Planner generates a sequence of nodes following the near-optimal $\ddot{s}$ profile. This sequence is feed forwarded to the dynamic inversion function (called "control generation"). The planner is triggered in order to run it several times in exact instants of time during the simulation. Compiled with Real Time Workshop and tested on a Pentium II 800 Mhz machine as target hardware, the algorithm showed its ability to control the system up to a time step of 0.2 s.
The executable file was also downloaded on the chaser agent in the laboratory described in appendix E. Again, it showed compatibility with the onboard PC calculation characteristics.

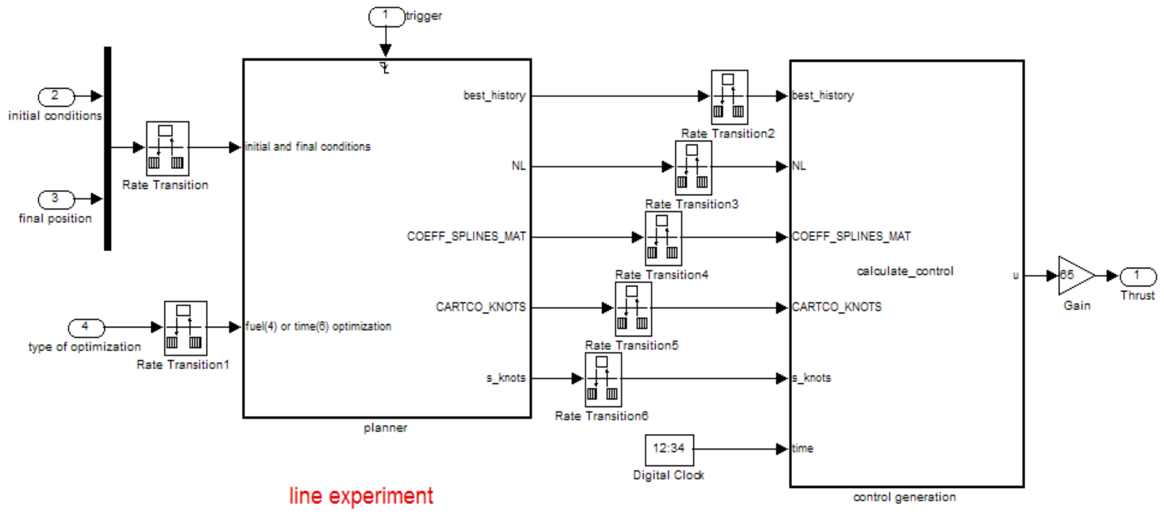

Figure 4.10: Simulink Model for Real Time testing

4.8 Comments and Future Improvements

The presented approach uses a direct method for real time sub-optimal control for spacecraft rendezvous and docking. Clohessy-Wiltshire equations have been parameterized through cubic B-splines having the curvilinear abscissa as parameter. Dynamic programming operating on the curvilinear acceleration has been proposed as optimization method to find the fuel and-or time near-optimal policy to drive the chaser spacecraft into a specified path towards the target spacecraft. The number of optimization parameters is thus drastically reduced and it is possible to constrain the acceleration value. By simply moving the splines control points, the designer (or an automatic system) can modify the shape of the path while manoeuvring.
The reliability of the algorithm has been assessed for different number of levels in the dynamic programming tree, and the results of a manoeuvre described in the recent literature have been compared.
Real Time implementation has been tested on a Pentium II machine with a sample time up to 0.2 seconds for the planner. The proposed method is very versatile. One, if required, can maneuver the chaser for a shorter portion of trajectory. By simply adjusting the final conditions to the specified maneuver it could be possible to optimize portions of the path. This part requires a minor modification of the software and it is under development. Speeding up the planner could bring the possibility of running the tree exploration several times in order to move the control points and searching for the optimal rendezvous and docking trajectory in real time or at least online.
In order to make the control bounded one of the future developments could be the analysis of the generated path before using it into the optimizer. The objective would be to maintain the curvature of the path limited. In this way the normal component of the vehicle acceleration is bounded, thus limiting the controls.
To conclude let us underline that non linear dynamics can be used with this method also. Virtually any mathematical model of relative motion can be used, if needed.

Chapter 5

Online Time/Fuel Optimal Control: Preshaped Thrust

5.1 Introduction

Following the idea of parameterizing the trajectory already used in chapter 4, in this section a direct method for a rapid generation of near-optimal RDD trajectories, with predetermined thrust history (based on the variational results of chapter 3) along a master direction, is presented. The new direct method, already implemented and tested onboard for the case of a real aircraft ([65]), is based on three concepts: high-order polynomials from the virtual arc as reference functions for the spatial coordinates, preset sequence of a master control, reduction of the optimization problem to the determination of a small set of parameters. By doing this the remaining controls act as slaves, guarantying the chaser to move along the desired path. The master thrust has an on-off structure. Seeking of the optimum strategy is transformed into a nonlinear programming problem, then numerically solved through an ad-hoc algorithm in accelerated time scale. Examples are reported in order to prove the how fast the approach is to generate a sub-optimal docking trajectory. Unlike chapter 4, here the algorithm velocity is not proved with a real-time test, the method is kept at an online level of validation. This simply indicates that it could be anyway used onboard for planning the guidance sequence, being relatively fast, but not at a specific sample time, not integrated with the hardware as it was for the dynamic programming approach.
The proposed strategy is a direct optimization method already used in for the control of aircraft (see [65]). The basic idea is to parameterize the trajectory in a way that allows for independent choice of path and velocity profile. The approach permits to reduce the functional problem into an NLP, with a rather small number of optimization parameters. Although this method obviously maintains the main disadvantage of all direct methods: it gives near optimal instead of optimal solution, its proven robustness makes it a good candidate for onboard real-time implementation.

5.2 Synthesis of the Optimal Control

The optimal control $U^* = [u_x^*,\ u_y^*,\ u_z^*]^T$ can be found from $U^* = \arg\max_U H(U)$. It has been already shown in chapter 3, eq. 3.28, that:

$$u_k^* = sign\lambda_{V_k} \min\left(\frac{|\lambda_{V_k}|}{2w}, u_{\max}\right) \tag{5.1}$$

From eq. 5.1 it is possible to deduce a first approximation structure for a sub-optimal control. It is worth mentioning that eq. 3.21 (time evolution of costate) here is not used. The only information that the control would result in a bang-unconstrained-bang (fuel optimization) or bang-bang (time optimization) is used. Singular control arcs (when $p_{V_k} \equiv \dot{p}_{V_k} = 0$) correspond to $u_k^* \equiv 0$. It is worth mentioning that from the standpoint of physical realization (construction of thrusters) bang-off-bang (on-off-on) control is also preferable (that actually casts the problem as a discrete optimization problem).

Now that the optimal control has been synthesized (although additional analysis needs to be performed to establish the rules for switching to/from the singular control arcs) the optimal control problem can be reduced to the problem of parameter optimization. In our particular case it means that we would guess on the final time t_f and initial values of co-states $\lambda_X(t_0)$, then knowing the structure of the optimal control integrate the π-system (3.13)+(3.20). At the end we compare the final values of states with the given ones (3.15) and, since in general they will not coincide, we repeat integration of the π-system trying to tune t_f and $\lambda_X(t_0)$ to match the required final conditions (3.15) and $\lambda_X(t_f) = 0$.

However, knowing how difficult it is to numerically solve the problem in real time it may be possible to define (parameterize) the controls time histories directly and then only integrate the original system (3.13). Figure 5.1 represents an example of such a profile (suggested by discretization of equation (5.1)) and defined by several (N) switching points t_n^k, $n = \overline{1,N}$ (this profile has to be established in each channel k=$\{x,y,z\}$). Varying the final time t_f and location of

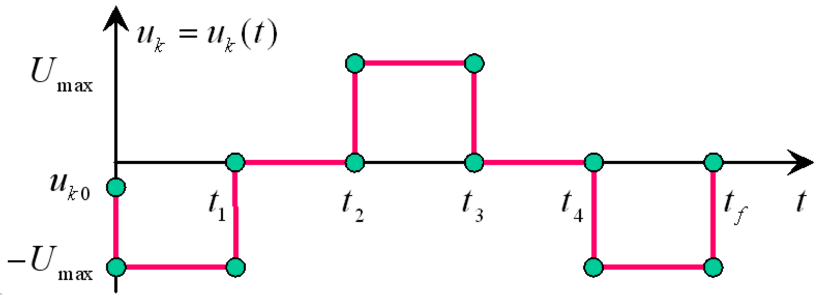

Figure 5.1: Parameterized control inputs time history.

the switching points t_n^k, $n = \overline{1,N}$,k=$\{x,y,z\}$ of the control-time history profiles it maybe possible to satisfy most of boundary conditions (3.15) (not guaranteed though).

Although there is no a priory hints on the number of switching points and the sequence of control inputs, we could start from some small number of switching points alternating all possible values of controls and then increase N if necessary to achieve more feasible solution. For instance, even with as little as four switching points shown on Figure 5.1 we may explore a wide variety of control profiles including pure bang-bang control as demonstrated on Figure 5.2. Although this latter approach may lead to the real-time algorithm, its robustness will not be

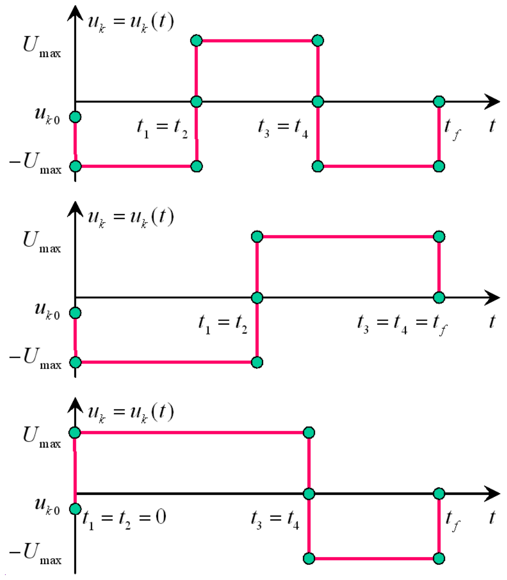

Figure 5.2: Pure bang-bang control profiles available with four switching points as defined on Figure 5.1.

guaranteed. The algorithm may diverge. No solution may exist. Not all boundary conditions can be satisfied. No predictions on the shape of the trajectory (its feasibility) can be made upfront even if solution exists. That is why the following introduces the direct method of calculus of variations as a mean to find a near-optimal solution to the boundary value problem in a fast way.

5.3 Introducing the Reference Trajectory

Let us start from the "end" defining the near-optimal trajectory we want the chaser to follow upfront. Moreover, to be able to separate the trajectory from the speed profile some artificial argument τ is introduced rather than time t([65]). It should be understood from the very beginning though, that in our particular case by doing this the independency in controls is lost (the spacecraft should fly along the trajectory, hence its speed vector should always be tangent to this trajectory and as it will be shown later that, in turn, implies certain relationships between u_x, u_y and u_z). However, introducing the reference trajectory allows satisfying the majority of the boundary conditions (3.15) upfront. It also excludes the possibility of "wild" unpredicted trajectories during the following parameter optimization.
As mentioned above, each of three chaser coordinates should be represented by some parameterized reference function versus virtual arc τ, $P_x(\tau)$, $P_y(\tau)$ and $P_z(\tau)$, respectively. Without

loss of generality further consider a single class of reference functions, namely polynomials to show how their unknown coefficients can be determined and what can be done to assure an additional flexibility to the reference trajectory (another class of reference functions might be trigonometric functions).
As stated by equalities (3.15) we could need to satisfy up to the second derivative of Cartesian coordinates at both ends of the trajectory. It's natural to require that the second derivative (proportional to accelerations) to be as smooth as at least third-order polynomial. Therefore, for each coordinate k=$\{x,y,z\}$ we may write the following:

$$P_k''(\tau) = a_{k2} + a_{k3}\tau + a_{k4}\tau^2 + a_{k5}\tau^3 = \sum_{l=2}^{5} a_{kl}\tau^{l-2} \tag{5.2}$$

Integrating equation (5.2) twice yields:

$$P_k'(\tau) = \sum_{l=1}^{5} \frac{a_{kl}\tau^{l-1}}{\max(1, l-1)}$$

and

$$P_k(\tau) = \sum_{l=0}^{5} \frac{a_{kl}\tau^{l}}{\max(1, l(l-1))}$$

so that six coefficients a_{kl} , $l = \overline{0,5}$ for each $k = \{\mathrm{x,y,z}\}$ can now be defined from the following linear matrix equation:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & \tau_f & \frac{1}{2}\tau_f^2 & \frac{1}{6}\tau_f^3 & \frac{1}{12}\tau_f^4 & \frac{1}{20}\tau_f^5 \\ 0 & 1 & \tau_f & \frac{1}{2}\tau_f^2 & \frac{1}{3}\tau_f^3 & \frac{1}{4}\tau_f^4 \\ 0 & 0 & 1 & \tau_f & \tau_f^2 & \tau_f^3 \end{bmatrix} \begin{bmatrix} a_{k0} \\ a_{k1} \\ a_{k2} \\ a_{i3} \\ a_{k4} \\ a_{k5} \end{bmatrix} = \begin{bmatrix} k_0 \\ k_0' \\ k_0 \\ k_f \\ k_f' \\ k_f \end{bmatrix} \tag{5.3}$$

To write this matrix equation the derivatives of coordinates in (3.15) were converted to the new argument using the so-called speed factor:

$$\lambda = \frac{d\tau}{dt} \tag{5.4}$$

so that:

$$k' = \lambda^{-1}\dot{k}$$

and

$$k = \lambda^{-2}(\ddot{k} - \dot{k}\lambda')$$

Once determined, the coefficients of the reference functions and the reference trajectory itself will depend on the only varied parameter τ_f. But what if additional flexibility is needed? Then we can increase the order of the reference polynomials and use the higher-order derivatives at both ends as additional varied parameters. For instance in our particular case it would make sense using 6^{th} or even 7^{th} order polynomials instead of 5^{th} order polynomials and then use the third derivatives of coordinates with respect to the virtual arc at the both ends of the trajectory as varied parameters.

5.4 Introducing Master Control Arc History

Let us first convert the system (3.8) to the new argument. Using the speed factor (5.4) it leads to:

$$\begin{array}{ll} x' = \lambda^{-1}V_x & V_x' = \lambda^{-1}f_1(x, V_y) + \lambda^{-1}u_x \\ y' = \lambda^{-1}V_y & V_y' = \lambda^{-1}f_2(V_x) + \lambda^{-1}u_y \\ z' = \lambda^{-1}V_z & V_z' = \lambda^{-1}f_3(z) + \lambda^{-1}u_z \end{array} \tag{5.5}$$

Combining the first three equations of (5.5) as

$$\sqrt{V_x^2 + V_y^2 + V_z^2} = |V| = \lambda\sqrt{x'^2 + y'^2 + z'^2} \tag{5.6}$$

specifically addresses the issue of independency of the trajectory and the velocity along it. Having the trajectory defined with respect to the virtual arc τ, i.e. having x', y' and z' defined, still leaves a possibility of varying the magnitude of the speed via varying the speed factor λ. However, the orientation of the speed vector with respect to the trajectory is completely determined by this trajectory (regardless its argument) and can be defined by two Euler angles as:

$$tg\varphi = \frac{x'}{y'} = \frac{V_x}{V_y}, \quad tg\theta = \frac{x'}{\sqrt{y'^2 + z'^2}} = \frac{V_x}{\sqrt{V_y^2 + V_z^2}} \tag{5.7}$$

That means that all three controls, u_x, u_y and u_z, cannot be longer varied independently. One master control should be defined (in the predominant direction), say u_x, and then define two others so that the direction of the velocity vector is tangent to the trajectory (meaning that equalities (5.7) hold).
Now, as suggested by the optimal control theory we assume the master control arc profile to be bang-singular-bang as shown on Figure 5.3 (which represents exactly the same control profile as on Figure 5.1 but with respect to the virtual arc τ rather than time t). Obviously two other

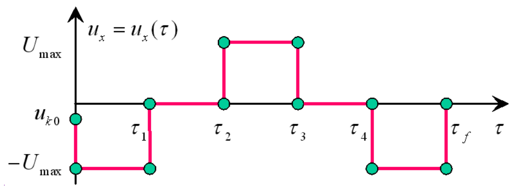

Figure 5.3: Suggested control profile for the master control.

controls, u_y and u_z, will not be bang-singular-bang anymore so that the PWM should be used to produce continuous accelerations in these two channels if the engines are not continuously magnitude adjustable.
The parameter optimization routine may be established as shown on Figure 5.4. Given the boundary conditions (3.15) we first define reference functions $P_x(\tau)$, $P_y(\tau)$ and $P_z(\tau)$, and compute their coefficients using these boundary conditions (and initial guesses on the third derivatives in case higher than 5^{th} order polynomials were employed for more flexibility). For the master control we also establish a bang-singular-bang arc profile defined by several switching

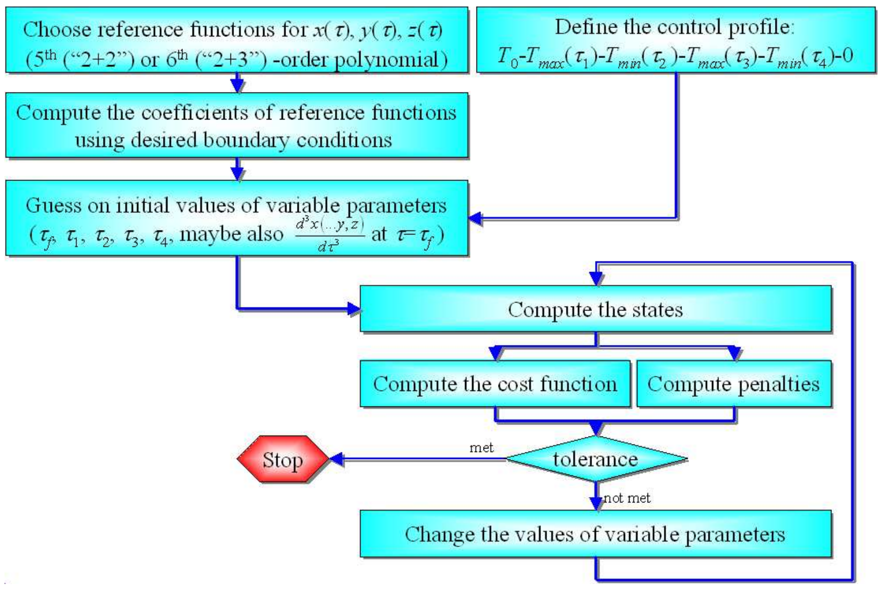
Choose reference functions for $x(\tau)$, $y(\tau)$, $z(\tau)$ (5th ("2+2") or 6th ("2+3") -order polynomial)
Define the control profile: T_0-$T_{max}(\tau_1)$-$T_{min}(\tau_2)$-$T_{max}(\tau_3)$-$T_{min}(\tau_4)$-0
Compute the coefficients of reference functions using desired boundary conditions
Guess on initial values of variable parameters (τ_f, τ_1, τ_2, τ_3, τ_4, maybe also $\frac{d^3x(\ldots y,z)}{d\tau^3}$ at $\tau=\tau_f$)
Compute the states
Compute the cost function
Compute penalties
tolerance
met
Stop
not met
Change the values of variable parameters

Figure 5.4: Parameter optimization flow chart.

points. These switching points, τ_i, $i = \overline{1,4}$, along with the length of the virtual arc τ_f (and possibly values of the higher-order derivatives of the coordinates at initial and/or final points) form the vector of variable parameters Ξ.
Next, we numerically solve the problem (integrating just one instead of all state equations and applying inverse dynamics for the rest of them). The transition between the virtual arc τ and time t is made using the speed factor

$$\lambda = \frac{\sqrt{x'^2 + y'^2 + z'^2}}{\sqrt{V_x^2 + V_y^2 + V_z^2}} \tag{5.8}$$

Then, we estimate the performance index J and compound the aggregated penalty Δ. The existence of this penalty is caused by the fact that the constraints on two slave controls are necessarily met and that the boundary conditions for the velocity vector components are not satisfied (since one equation was integrated and two others are related to it via dynamic constraints (5.7)).
Now, we apply any standard nonlinear constrained minimization routine to minimize the performance index keeping the penalty within a certain tolerance ε

$$\min_{\Xi} J \Big|_{\Delta \leq \varepsilon} \tag{5.9}$$

5.5 Computation of States, Performance Index and Penalty

We start from dividing the virtual arc τ_f onto N-1 equal pieces $\Delta\tau = \frac{\tau_f}{N-1}$ so that we have N equidistant nodes $j = \overline{1,N}$. All states and the master control at the first point $j = 1$ (correspond-

ing to $\tau_1 = \tau_0 = 0)$ are defined. Additionally we define $\lambda_1 = 1$.
Then, for each of the subsequent N-1 nodes $j = \overline{2,N}$ we do the following. We compute the current values of coordinates x, y and z using polynomials $x_j = P_x(\tau_j)$, $y_j = P_y(\tau_j)$ and $z_j = P_z(\tau_j)$ respectively. Next, knowing the master control from the predetermined arc history $u_{x,j-1} = u_x(\tau_{j-1})$ we integrate the fourth equation of system (5.5):

$$V_{x,j} = V_{x,j-1} + \lambda_{j-1}^{-1}\left(f_1(x_{j-1}V_{y,j-1}) + u_{x,j-1}\right)\Delta\tau. \tag{5.10}$$

To assure the correct direction of the velocity vector we apply relations (5.7) to obtain two other velocity components:

$$V_{y,j} = V_{x,j}\frac{y'_j}{x'_j} \textit{ and } V_{z,j} = \frac{z'_j\sqrt{V_{x,j}^2 + V_{y,j}^2}}{\sqrt{x'^2_j + y'^2_j}} \tag{5.11}$$

and therefore calculate the magnitude of speed:

$$|V|_j = \sqrt{V_{x,j}^2 + V_{y,j}^2 + V_{z,j}^2} \tag{5.12}$$

Now that we know the change in the chaser's position and the magnitude of speed we may compute the time interval between (j-1)th and j^{th} nodes:

$$\Delta t_{j-1} = 2\frac{\sqrt{\sum_{i=1}^{3}\left(\xi_{i,j} - \xi_{i,j-1}\right)^2}}{|V|_j \ + \ |V|_{j-1}} \tag{5.13}$$

and the current value of the speed factor:

$$\lambda_j = \frac{\Delta\tau}{\Delta t_{j-1}} \tag{5.14}$$

Current time then equals to:

$$t_j = t_{j-1} + \Delta t_{j-1} \ (t_1 = 0) \tag{5.15}$$

Finally, using the last two equations of the system (5.5) we find the values of two slave controls that yield speed components:

$$u_{x,j-1} = \frac{V_{y,j} - V_{y,j-1}}{\Delta\tau}\lambda_j + 2\omega V_{x,j-1}$$

and

$$u_{z,j-1} = \frac{V_{z,j} - V_{z,j-1}}{\Delta\tau}\lambda_j + \omega^2 z_{j-1}$$

Once all states and controls are computed, we may estimate the performance index:

$$J = (1-w)t_N + w\sum_{j=0}^{N-1}\left(u_{x,j}^2 + u_{y,j}^2 + u_{z,j}^2\right)\Delta t_j \tag{5.16}$$

and form the penalty as:

$$\Delta = w_p \sum_k (V_{k,N} - \dot{k}_f)^2 + (1 - w_p) \sum_k \max_j \left(0; |u_{k,j}| - U_{\max}\right)^2 \tag{5.17}$$

where k=$\{x,y,z\}$ and w_p is the penalty weighting coefficient.
For numerical convergence of the algorithm, time, the slaves control constraint violation w.r.t. the limit on maximum thrust and the discrepancy on final velocity, have been re-scaled obtaining the following modified performance index and penalty:

$$\begin{aligned} J_{sc} &= t_{sc}(1-w)t_N + w \sum_{j=0}^{N-1} \left(u_{x,j}^2 + u_{y,j}^2 + u_{z,j}^2\right) \Delta t_j \\ \Delta_{sc} &= w_{speed} V_{sc} \sum_k (V_{k,N} - \dot{k}_f)^2 + w_{overflow} O_{sc} \sum_k \max_j \left(0; |u_{k,j}| - U_{\max}\right)^2 \end{aligned} \tag{5.18}$$

Coefficients have been adjusted to bring the values to the same order.

5.6 Simulation Results

Two examples are reported in order to show how the algorithm is capable to generate in a short time the command sequence to drive the chaser towards the target in a sub-optimal way.
The boundary conditions are the same for the two test cases, only the weighting ratio between propellant and time is changed showing how the resulting trajectories differ from each other. In the first case propellant is more important to be saved than the time required for docking: $w = 0.9$. While in the second test the weighting coefficient is 0.1, giving more importance to execute the maneuver in a short time.
The numerical values used in both simulations are shown in the following table:

Table 5.1: Numerical values for the test cases.

Parameter	**Units**	**Value**
Height above the Earth surface	km	981.46
Initial relative position	m	(-60, -40, 0)
Initial relative velocity	m/s	(0.005, 0, 0)
Required relative final position	m	(0, 0, 0)
Required relative final velocity	m/s	(0.0005, 0, 0)
Initial and final relative accelerations	m/s^2	(0, 0, 0)
Maximum relative acceleration (thrust)	m/s^2	0.0138
Number of points for computation along arc τ	$-$	200
Time scaling factor	$1/s$	1/1000
Control constraint violation scaling factor	$(m/s^2)^{-1}$	40
Final discrepancy on velocity scaling factor	$(m/s)^{-1}$	200

Unfortunately the Matlab® *fmincon* function for constrained non-linear optimization to solve (5.9) failed to work (it is known to be quite unstable.). Therefore the Matlab® *fminsearch* function has been employed in both simulations as optimization means to work on five varied parameters, i.e., the virtual arc length and the master control switches. In this case the performance index J_{sc} and penalty function Δ_{sc} (5.18) were blended together using an appropriate weighting coefficient. It worth mentioning that in some sense the *fminsearch* function was more preferable because of another reason too. This reason is that it employs zero-order (non-gradient) rather than gradient algorithms (like Nelder-Mead downhill simplex algorithm) assuring unconditioned finding of at least a local minimum, i.e. certain reliability for the future on-line implementation given by a probability of solution equal to 1.
The software used in the modeling stage is based on a call to a Simulink® model every time an index and penalty evaluation is required. This obviously slows down the overall process w.r.t. a simple Matlab®script and even more w.r.t. a C code, but still demonstrates a very satisfactory relative CPU time (percentage of time required by the machine w.r.t. the complete maneuver required time). In what follows the relative CPU time is reported for two different machines: an AMD Athlon 2600 MHz processor, and a Pentium III 1200 MHz processor.

5.6.1 Simulation test case 1

An initial guess on virtual arc length and position of the four switching points was: $\tau_f = 5.5$, $\tau_1 = 0.007\tau_f$, $\tau_2 = 0.1\tau_f$, $\tau_1 = 0.33\tau_f$, and $\tau_1 = 0.4\tau_f$.
The resulting trajectory, with the corresponding optimized values of the OPs controls behavior, velocity history, fuel consumption and other significant parameters is shown on Figure 5.5.

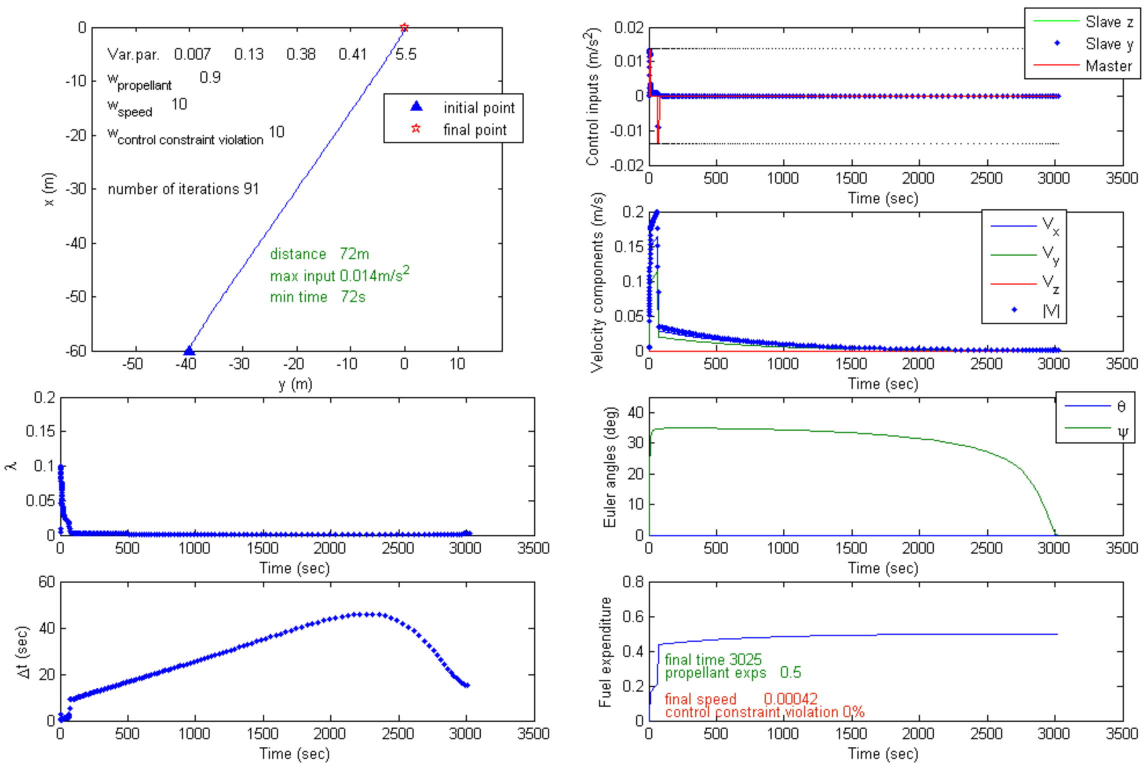

Figure 5.5: Results for simulation test case 1

The significant phase is zoomed for the controls in Figure 5.6. Number of iterations is rather small (< 100). Note how the final velocity is of the same order of the required one (0.0005 $\frac{m}{s}$) and the fact that there is no control constraint violation, i.e. the slaves are respecting the imposed bounds.
Relative CPU time came out to be 2.9% with the faster machine (AMD Athlon 2600 MHz) and 4.3% with the Pentium III, 1200 MHz.

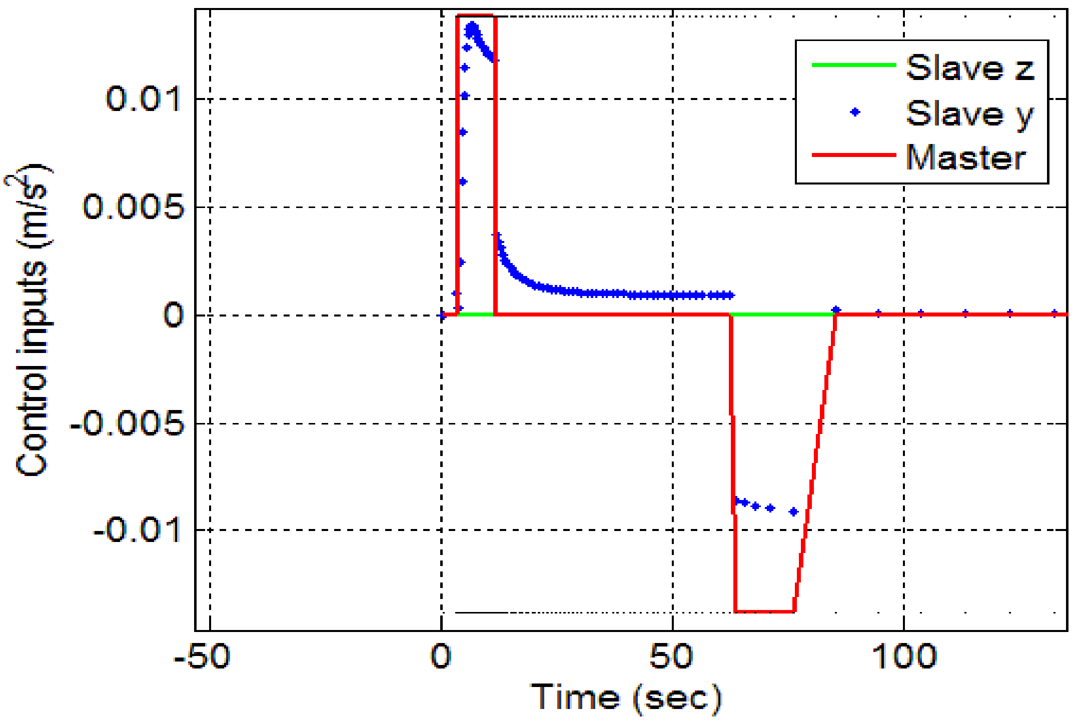

Figure 5.6: Zoom on the first 150 seconds for Test Case 1: Controls

5.6.2 Simulation test case 2

For this test case, where we still optimize a combination of fuel and time, but giving more importance to the rapidity of the maneuver execution, the initial guess was $\tau_f = 9$, $\tau_1 = 0.007\tau_f$, $\tau_2 = 0.2\tau_f$, $\tau_1 = 0.5\tau_f$, and $\tau_1 = 0.6\tau_f$.
The results are shown on Figure 5.7.

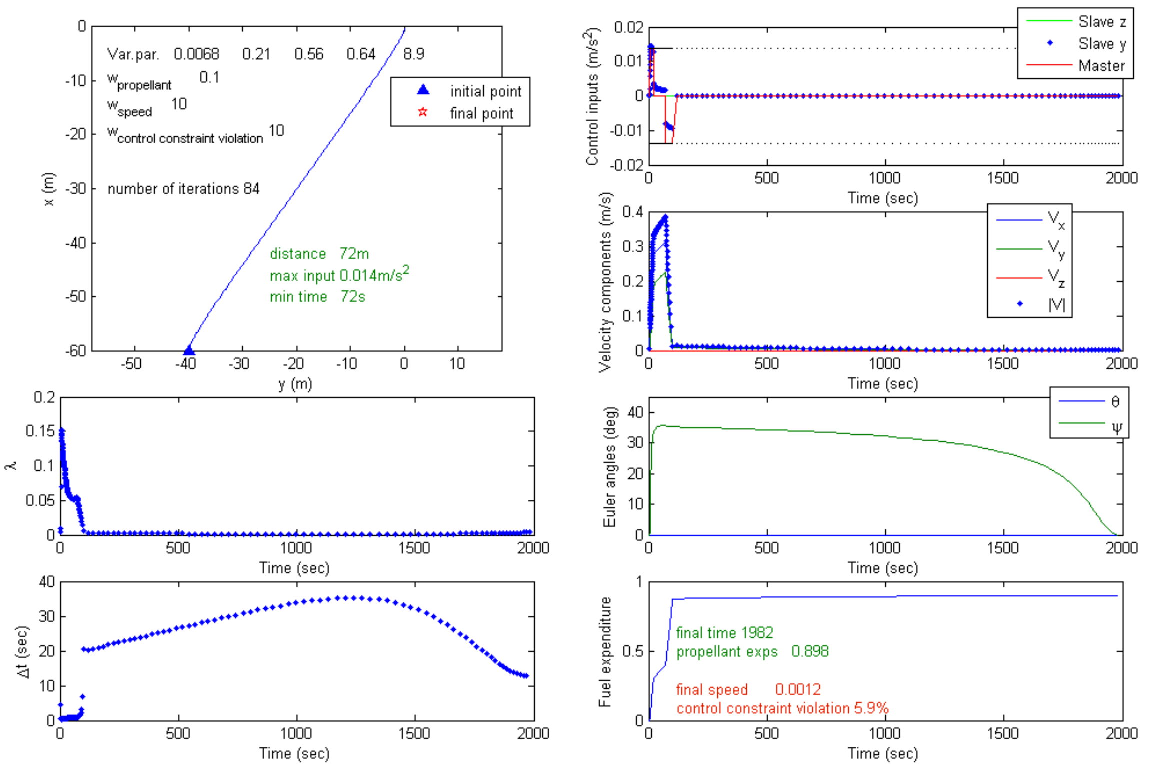

Figure 5.7: Results for simulation test case 2

The significant phase is zoomed for the controls in Figure 5.8. Again, the required iterations are less than 100. For this maneuver, more demanding than the simulation test case 1, we obtain a small control constraint violation of 5.9%, and the final velocity discrepancy is slightly higher than in the previous case.

Having required minimizing time, with a small consideration of propellant expenditure in this case, it results in the possibility of limit violations in the slaves' behavior. Note how the final time is $\sim$ 35% lower than in test case 1, as expected (1982 s vs. 3025 s). At the same time the propellant expenditure raised from 0.5 units to 0.898 units ($\sim$ 80% increase).

Relative CPU time is 4.2%4 on the AMD Athlon 2600 MHz and 6% on the Pentium III, 1200 MHz.

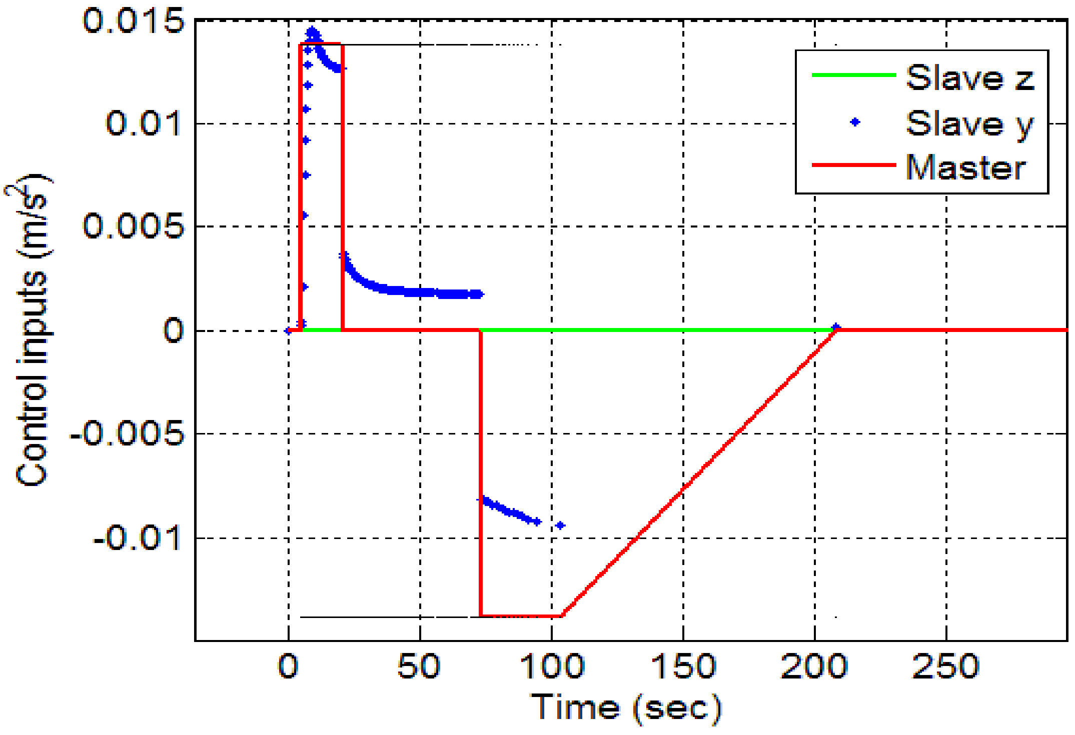

Figure 5.8: Zoom on the first 300 seconds for Test Case 2: Controls

5.7 Comments and Future Developments

The proposed direct method for trajectory optimization shows many advantages. First of all it guarantees the boundary conditions to be satisfied (always for position, numerically for velocity), no "wild" trajectories arise during optimization, an analytical (parametrical) representation of the reference trajectory is possible, and, the last but not the least, a small number of OPs have to be considered, requiring only a few iterations (< 100) to generate a solution. These two last features, together with a low relative CPU time for convergence, make possible to employ DMRP on board of a spacecraft for real-time prototyping of rendezvous and docking maneuvers. Resulting trajectory generation algorithms can be easily integrated with existing navigation/control algorithms.
As done in chapter 4 it is worth mentioning also the fact that non linear dynamics can be treated with the discussed technique.
The main limitation of the approach is that the *fminsearch* routine has a huge tendency to localize local minima and stop searching better solutions (the routine strongly depends on the initial guesses; see sections 5.6.1, 5.6.2). Figure 5.7 and Figure 5.8 show how the algorithm is not always capable to avoid the slaves to violate the constraints and how the final velocity is not much close to the desired one.
Second drawback are the relatively high values for the Δts (up to almost 1 *min*) obtained separating path from velocity (Figure 5.5 and Figure 5.7). *fminsearch* does not have good performances

not being able to correctly weight the penalties. Also weighting the maximum Δt obtained in a run, it does not improve significantly the result.
High Δts imply long intervals of interpolation once the obtained vector of controls is to be used for actually driving the chaser, in other words the direct integration of eq. 3.13 could be affected by errors due to poor knowledge of the controls for long time intervals, and could result in not matching the final desired condition.
For all these reasons one of the next mandatory developments will be the implementation of a better minimization routine. A multi-criterion multi-variable optimization routine based on Hooke-Jeeves pattern search algorithm was developed, not yet validated for the direct method illustrated in this chapter, but it showed good performances for the technique presented in the following one.
Once the routine will be updated to deal with constraints among the optimization variables (in the present approach we want $\tau_1 < \tau_2 < ... < \tau_f$) the first simulations could be performed.
It is important, anyway, to keep in mind that this kind of approach needs an ad-hoc tuning of the weights introduced for the cost function and the discrepancy. Some could call it an art, more than a mathematical choice, being often a trial and error procedure.
Further development of the present study will be also the implementation of the algorithm with the C language and the hardware-in-the-loop test, i.e. the translation into Embedded Matlab to proceed as done in chapter 4.

Chapter 6

Hybrid Low-High Thrust Fuel Optimal Control

6.1 Introduction

After the considerations on available space qualified thrusters made in appendix F, in this chapter an hybrid method is introduced for minimum-propellant proximity maneuvers based on the assumption of having two classes of real thrusters on board. The variational results (chapter 3) are here used again, as done in chapter 5, but in a more extensive way. The case of impulsive thrust ([34]) for the very last phases of rendezvous and continuous multi-level (discretized) low thrust for far away manoeuvering is addressed. The problem statement follows: given the initial and final desired states, determine the far away thrust switching history, having a finite set of admissible values for it, while, for the proximity stage, where more accuracy is required, determine number, magnitude and location in time of the pulses, minimizing the overall ΔV. The present technique aims considering real engines features, i.e. the limitations in terms of thrust adjustability. Several researchers ([28, 29, 50, 30, 57, 36]) have studied the case of rendezvous and docking maneuvers of spacecraft with continuous thrust. Furthermore, a vast literature exists on orbital change maneuvers with impulsive thrust ([39, 66]).
The case of rendezvous and docking is particularly critical because of its applicability to current-technology spacecraft with either chemical or cold gas, electrical on-off continuous thrusters ([67, 68]). The problem of determining an optimal impulsive thrust sequence is here faced according to a first order algorithm similar to that of [38, 37], previously applied to orbital transfers optimization. A very extensive previous work can be found, for instance, in [35, 33], where the problem is faced through the solution of a set of nonlinear equations.
Neutstadt ([46]) demonstrated that for linear systems the number of impulses is upper-limited by the number of the final conditions. Furthermore, Lawden ([34]) defined the primer vector as the adjoint velocity and established a set of four necessary conditions of optimality, being also sufficient for linear dynamics. The time evolution of the primer vector is analytically calculated in eq. 3.22.
The basic principles of the approach follows: for the first stage a multi-criterion multi-variable optimization routine based on Hooke-Jeeves pattern search algorithm ([69], see appendix C)

has been implemented in order to find the adjoint initial conditions and to adjust the maneuver time that bring the chaser vehicle in the vicinity of a desired intermediate condition. For the second stage the primer history is first analyzed for the analytically determined two-impulses maneuver. Then, if the Lawden's condition of optimality are not satisfied, additional impulses are added, one at the time ([38, 37]), by optimizing, with a gradient-search technique, their time location in order to satisfy the conditions. The maximum number of impulses is set by the previously mentioned Neutstadt theorem. Jezewski already studied the application of Lawden's primer conditions to the linear equations of motion for satellites in relative flight ([45]).
The approach in literature that most recalls the technique here proposed is that of Lembeck and Prussing ([41]). In their work a fast intercept impulsive manoeuvre from the origin of the LVLH frame to a pre-chosen position is performed, then a low-thrust return to the initial state. Only one impulse is used to reach the intercept point, not optimizing any sequence of more impulses, not imposing a desired velocity at the intercept. The impulsive segment is anyway optimal in a sense that the time for the maneuver is adjusted so that the Lawden's conditions are satisfied with only one impulse. The low-thrust return phase is optimized and thrust unbounded, assuming modulation capability for the thrusters.

6.2 Problem Definition and Optimality Conditions

For docking maneuvers, the chaser spacecraft starts from a generic initial condition with respect to the target centered LVLH frame and has to reach the vicinity of an intermediate state using low thrust, eventually zero relative velocity and position by using a sequence of impulsive thrusts, the overall operations within an initially fixed time interval $t_f - t_0$. For the more general case of either rendezvous or proximity operation maneuvers the final conditions are a state vector with all non-zero elements. The normalized form [29] (anomaly of the target substituting time: $\theta = \omega_{LVLH} t$) of the HCW equations are used to represent the relative state vector evolution (eq. 3.12, the modified HCW equations). For the low thrust segment let us express the cost as in [29]:

$$J = \frac{1}{2} \int_{t_0}^{t_{int}} \Gamma \hat{u} \cdot \Gamma \hat{u} \, dt \tag{6.1}$$

from which (see section 3.3.1):

$$\Gamma \hat{u} = -\vec{\lambda}_V \tag{6.2}$$

$\vec{\lambda}_V, \vec{\lambda}_r$ time evolution is governed by the dynamics of equation 3.22. In order to consider a finite set of levels of thrust, as it would be with a real cluster of different performances low thrust actuators, each one operating either at the maximum or at zero thrust (on-off), the optimal acceleration is constrained to assume only a set of values between zero and a maximum saturation. This makes the method applicable in principle to current electric thruster technology. In particular, let us consider a cluster of five electrical thrusters, each able to give a fixed level of acceleration: $10^{-3}\ ms^{-2}$. The maximum resulting acceleration with all electrical thrusters on is then $5 \cdot 10^{-3}\ ms^{-2}$. Let us call the different levels of thrust l_i, being, in this case,

$l_0 = 0,\ l_1 = \pm 10^{-3},\ l_2 = \pm 2 \cdot 10^{-3},\ l_3 = \pm 3 \cdot 10^{-3},\ l_4 = \pm 4 \cdot 10^{-3},\ l_5 = \pm 5 \cdot 10^{-3}$, units are obviously ms^{-2}. Discretization (i.e. limiting the admissible values to that of the multi-level set) and saturation in 6.2 will introduce an error in exact matching the intermediate condition, then the more accurate impulsive approach will exactly drive the vehicle to the target.

For the impulsive case the control vector is $\Gamma\hat{u} = \sum_{j=1}^{N} \Delta\vec{V}\delta\left(t - t_j\right)$. The Neutstadt ([46]) theorem limits the number of impulses to be at most six, when both position and velocity are imposed at final time. We need to find the number N, the time instants t_j, space collocation $\vec{r}_j$, and values $\Delta\vec{V}_j$ of the impulses to be applied for the dynamics in 3.12, in order to minimize the cost function:

$$J = \int_{t_{int}}^{t_f} \Gamma dt = \sum_{i=1}^{N} \Delta V_i,\ N = 2..6 \tag{6.3}$$

6.3 The Optimization Algorithms

The first stage control history is determined iterating on the unknown initial conditions for the adjoint vector, also allowing little changes in the required time for the maneuver. The objective function to be minimized is the norm of the final discrepancy on the desired state vector:

$$f\left(\Lambda_X(t_0),\ t_{int}\right) = \sqrt{\begin{array}{c} \left(x(t_{int}) - x_{des}\right)^2 + \left(y(t_{int}) - y_{des}\right)^2 + \left(z(t_{int}) - z_{des}\right)^2 + \\ + \left(\dot{x}(t_{int}) - \dot{x}_{des}\right)^2 + \left(\dot{y}(t_{int}) - \dot{y}_{des}\right)^2 + \left(\dot{z}(t_{int}) - \dot{z}_{des}\right)^2 \end{array}} \tag{6.4}$$

The minimization routine has been implemented using a first order, highly reliable, algorithm: the Hooke-Jeeves method. Starting from the initial guess for unbounded continuous control (eq. 3.25), 3.12 is integrated considering discretization in the control (i.e. in eq. 3.23, which uses the costate time evolution 3.22), the final error on the state being the minimization function. Multi-level control practically means changing the value for the adjoint velocity into the closest level of admissible thrust l_i. In [29] upper bounded continuous thrust is considered and the problem is solved as a non linear system, i.e. the discrepancy on each coordinate is considered, through the Matlab® *fsolve* routine. *fsolve* did not show the same capabilities of solving the present problem, never converging to a solution. Having only a set of levels for the thrust adds complexity and higher residual errors on final state and the minimization approach definitely showed a satisfactory behavior.

In the impulsive phase the second, third and fourth conditions in 3.3.2 are analytically studied (remember the primer evolves according to eq. 3.22, being the velocity costate). Starting from a two impulses maneuver, which is known in closed form ([39]):

$$\left.\begin{array}{l} \Delta\vec{V}_0 = \Phi_{12}\left(t_f - t_0\right)^{-1}\left[\vec{r}_f - \Phi_{11}\left(t_f - t_0\right)\vec{r}_0\right] - \vec{V}_0, \\ \Delta\vec{V}_f = -\Phi_{21}\left(t_f - t_0\right)\vec{r}_0 - \Phi_{22}\left(t_f - t_0\right)\left\{\Phi_{12}\left(t_f - t_0\right)^{-1}\left[\vec{r}_f - \Phi_{11}\left(t_f - t_0\right)\vec{r}_0\right]\right\} \end{array}\right\} \tag{6.5}$$

having partitioned the *CW* transition matrix (appendix A):

$$\Phi = \begin{pmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{21} & \Phi_{22} \end{pmatrix} \tag{6.6}$$

the primer's maxima during the maneuver are calculated. If any of these values exceeds one, a new impulse is introduced. The first condition is imposed numerically, by adjusting the time and space locations of the newly added impulse with a multidimensional nonlinear conjugate gradient method ([70], see also section 6.5 for a brief description) based on Flecther-Reeves (or Polak-Ribiere), following the procedure of Jezewsky and Rozendaal ([38]). Further impulses are iteratively added till the satisfaction of the first three conditions (3.3.2), with total upper limit of six as determined by the Neutstadt's theorem.
In particular at a generic step of the algorithm, the trajectory can be divided into sub-arcs which merge at the impulses times, with known boundary conditions for primer vector. In case a maximum of the magnitude of the primer vector $\vec{p}_m$ exists over one, an additional impulse is added at the corresponding time t_m. This reduces the cost with respect to the original reference trajectory ([38]), maximizing the scalar product between the primer vector and the thrust direction:

$$dJ = \Delta V_m \left(1 - \vec{p}_m \cdot \hat{u}\right) \tag{6.7}$$

While the direction of the added impulse is given by the fourth condition of optimality, its magnitude is determined according to the following relation ([38]):

$$\Delta V_m = \\ = \frac{\vec{b}\cdot\Delta\vec{V}_j \Big/ \Delta V_j - \vec{a}\cdot\Delta\vec{V}_{j-1} \Big/ \Delta V_{j-1} - 1}{\left[\vec{a}\cdot\vec{a} - \left(\vec{a}\cdot\Delta\vec{V}_{j-1}\right)^2 \Big/ \Delta V_{j-1}^2\right] \Big/ \Delta V_{j-1}^2 - \left[\vec{b}\cdot\vec{b} - \left(\vec{b}\cdot\Delta\vec{V}_j\right)^2 \Big/ \Delta V_j^2\right] \Big/ \Delta V_j^2} \tag{6.8}$$

where:

$$\vec{a} = \Phi_{12}^{-1}\left(t_m - t_{i-1}\right) A^{-1}\frac{\vec{p}_m}{p_m},\ \vec{b} = \Phi_{12}^{-1}\left(t_m - t_i\right) A^{-1}\frac{\vec{p}_m}{p_m} \tag{6.9}$$

and:

$$A = \Phi_{22}\left(t_m - t_i\right)\Phi_{12}^{-1}\left(t_m - t_i\right) - \Phi_{22}\left(t_m - t_{i-1}\right)\Phi_{12}^{-1}\left(t_m - t_{i-1}\right) \tag{6.10}$$

Relation 6.8 guarantees the respect of the boundary conditions every time an impulse is added. Its location in space, and the variations of the immediately previous and subsequent impulses are calculated from:

$$\Delta\vec{x}_m = \Delta V_m A^{-1}\frac{\vec{p}_m}{p_m},\ \delta\Delta\vec{V}_{i-1} = \Phi_{12}^{-1}\left(t_m - t_{i-1}\right)\Delta\vec{x}_m,$$

$$\delta\Delta\vec{V}_i = \Phi_{12}^{-1}\left(t_m - t_i\right)\Delta\vec{x}_m \tag{6.11}$$

adding $\delta\Delta\vec{V}_{i-1}$ to the old impulse immediately before the new one, resting $\delta\Delta\vec{V}_i$ from the old subsequent one.
If the new trajectory does not result in a C^1 primer vector evolution, then the conjugate gradient method is implemented to improve the cost function, moving the burn times and collocation. The gradient expression for the cost with respect to these last variables can be found to be ([38]):

$$\overrightarrow{\nabla J} = \left\{ \begin{array}{c} \dot{\vec{p}}_m^{+} - \dot{\vec{p}}_m^{-} \\ \dot{\vec{p}}_m^{+}\cdot\dot{\vec{r}}_m^{+} - \dot{\vec{p}}_m^{-}\cdot\dot{\vec{r}}_m^{-} \end{array} \right\} \tag{6.12}$$

representing the partial derivatives of cost with respect to the instants and collocations of the ΔVs. Clearly $+$ refers to values right after the impulse, $-$ right before. As regards the fifth and sixth conditions, we limit the scope of the present research to sample maneuvers which a priori satisfy them.
Indeed only initial coasting would make sense for this problem being the sixth condition always satisfied for the docking case, as demonstrated in the following.
The possibility of a final coast means to find a non trivial solution of the following 7 unknowns, 6 equations system:

$$\Phi\left(t_f - t^*\right)\left\{\begin{array}{c}\vec{r}(t^*)\\ \vec{V}(t^*)\end{array}\right\} =$$

$$= \left(\begin{array}{cc}\Phi_{11} & \Phi_{12}\\ \Phi_{21} & \Phi_{22}\end{array}\right)\left\{\begin{array}{c}\vec{r}(t^*)\\ \vec{V}(t^*)\end{array}\right\} =$$

$$= \left\{\begin{array}{c}\vec{r}(t_f)\\ \vec{V}(t_f)\end{array}\right\} = \left\{\begin{array}{c}\vec{0}\\ \vec{0}\end{array}\right\},$$

$$\Phi_{11} = \left(\begin{array}{ccc}1 & -6sin\left(t_f - t^*\right) + 6\left(t_f - t^*\right) & \\ 0 & 4 - 3cos\left(t_f - t^*\right) & 0\\ 0 & 0 & cos\left(t_f - t^*\right)\end{array}\right),$$

$$\Phi_{12} = \left(\begin{array}{ccc}-3t + 4sin\left(t_f - t^*\right) & -2cos\left(t_f - t^*\right) + 2 & 0\\ -2 + 2cos\left(t_f - t^*\right) & sin\left(t_f - t^*\right) & 0\\ 0 & 0 & sin\left(t_f - t^*\right)\end{array}\right),$$

$$\Phi_{21} = \left(\begin{array}{ccc}0 & -6sin\left(t_f - t^*\right) + 6 & 0\\ 0 & 3sin\left(t_f - t^*\right) & 0\\ 0 & 0 & -sin\left(t_f - t^*\right)\end{array}\right),$$

$$\Phi_{22} = \left(\begin{array}{ccc}-3 + 4cos\left(t_f - t^*\right) & 2sin\left(t_f - t^*\right) & 0\\ -2sin\left(t_f - t^*\right) & cos\left(t_f - t^*\right) & 0\\ 0 & 0 & cos\left(t_f - t^*\right)\end{array}\right) \tag{6.13}$$

To coast we should determine a time instant t^*, and its relative state vector, to achieve zero position and velocity at final time. The determinant of the transition matrix in 6.13 is:

$$det\left(\Phi\right) = cos^4\left(t_f - t^*\right) + 2sin^2\left(t_f - t^*\right)cos^2\left(t_f - t^*\right) + sin^4\left(t_f - t^*\right) \tag{6.14}$$

which never nullifies. At $t^* = t_f$ we obtain an identity matrix, that is a final $\Delta\vec{V}$ is always required in order to match a desired zero docking velocity.

6.4 Simulations and Results

The following boundary conditions have been set for a significant sample rendezvous manoeuvre:

Table 6.1: Boundary Conditions for Rendezvous

Variable Name	Measurement Units	Value
$\vec{r}(t_0)$	km	$\begin{bmatrix} 15 & 0 & 2 \end{bmatrix}^T$
$\dot{\vec{r}}(t_0)$	m/s	$\begin{bmatrix} -10 & 0 & -2 \end{bmatrix}^T$
$\vec{r}(t_{int})$	km	$\begin{bmatrix} -3 & 0 & 0 \end{bmatrix}^T$
$\dot{\vec{r}}(t_{int})$	m/s	$\begin{bmatrix} 2.5 & 0 & 0 \end{bmatrix}^T$
$\vec{r}(t_f)$	km	$\begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$
$\dot{\vec{r}}(t_f)$	m/s	$\begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$
t_0	s	0
t_{int}	s	8500
t_f	s	13000

The target orbits at $h = 480$ km above the Earth's surface. The resulting cost is $2.774 \cdot 10^{-8}$ $\frac{km^2}{s}$ for the first stage, if calculated as ΔV, i.e. as $\int |u| dt$ it results $19.771 \frac{m}{s}$. Optimal trajectory for the low-thrust segment is shown in Figure 6.1 together with the initial state and the reached intermediate one.

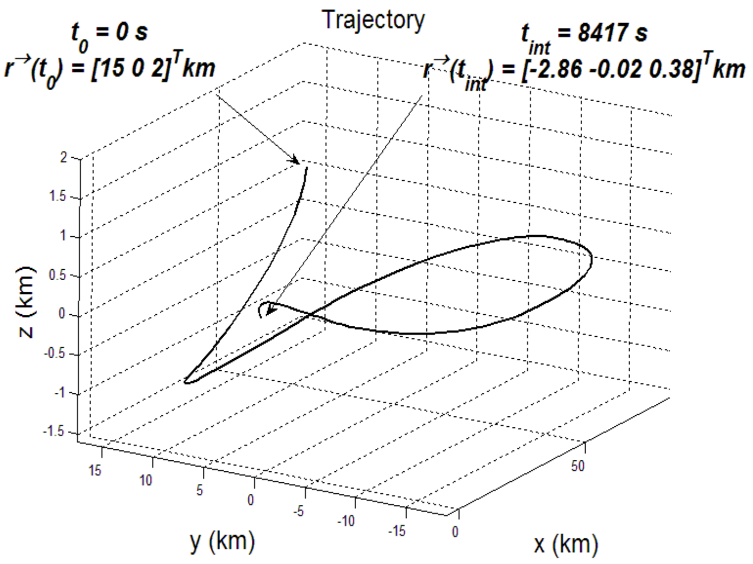

Figure 6.1: Multi-level Control resulting Trajectory

Figure 6.2 illustrates the obtained switching structure for the controls along the three axis:

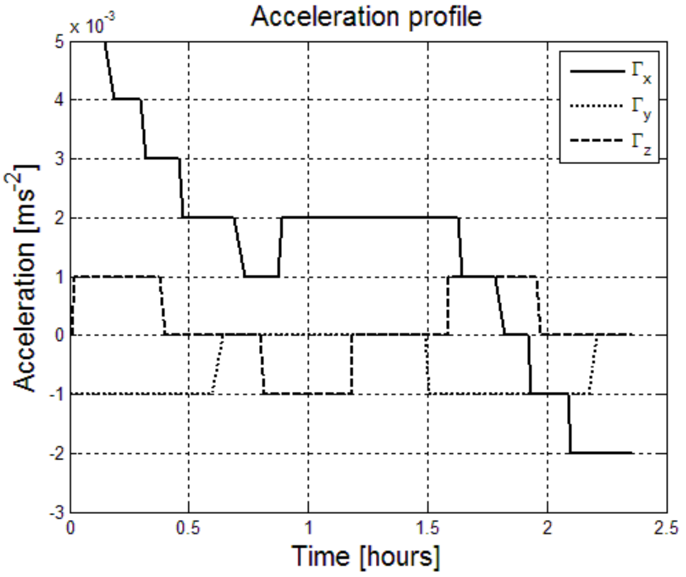

Figure 6.2: Multi-level Controls

The residual error in reaching the intermediate state is:

$$\begin{bmatrix} -0.14 & 0.02 & -0.38 \end{bmatrix}^T km$$

$$\begin{bmatrix} -4.22 & 2.11 & 5.45 \end{bmatrix}^T \cdot 10^{-4} km\, s^{-1}$$

Intermediate time is slightly adjusted by the algorithm from $8500s$ to $8417s$.

The following plot gives a visual representation of the impulses insertion, up to four, for the second stage and the "smoothening" of the primer vector norm through the gradient numerical method. It also shows the actual impulses and their firing instants:

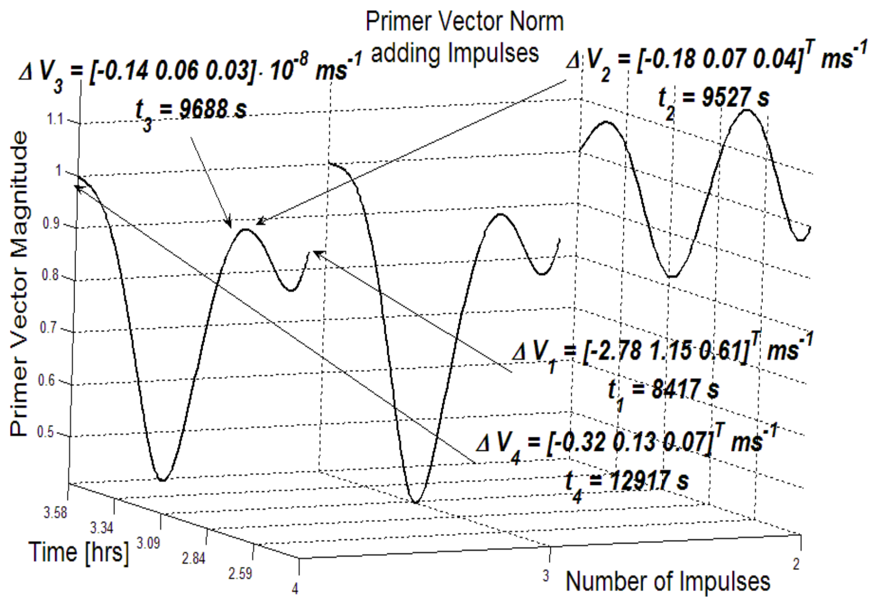

Figure 6.3: Primer Vector Magnitude Improving

The resulting, last phase, docking trajectory is:

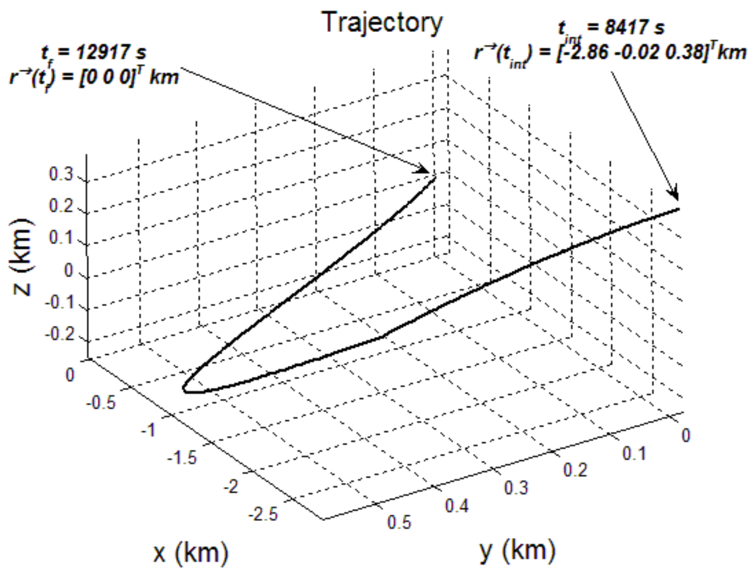

Figure 6.4: Impulsive Control Resulting Trajectory

The total cost for the manoeuvre, in terms of overall velocity variation, results to be $23.390\frac{m}{s}$.

6.4.1 Comparing with Low-Thrust Continuous Manoeuvre

In this section the same manoeuvre is optimized via the approach followed by Guelman [29], that is, the error in reaching the final docking condition $[0,\ 0,\ 0,\ 0,\ 0,\ 0]$ is minimized by using the Matlab® *fsolve* routine (solution of a non linear system). The upper limit on control acceleration is the same but no discretization into multi-levels is applied, i.e. the magnitude is supposed to be adjustable along time. A fundamental difference is also the fact that the bound refers to the overall acceleration module, not on a single channel. Time required for the manoeuvre is imposed to be 12917 s, as obtained in previous section by the Hooke-Jeeves algorithm.
By reproducing this simulation we want to show how the combination of low- and high-thrust brings to a same order cost manoeuvre with the advantage of being directly applicable to real space qualified thrusters.
The following are the optimal trajectory and controls:

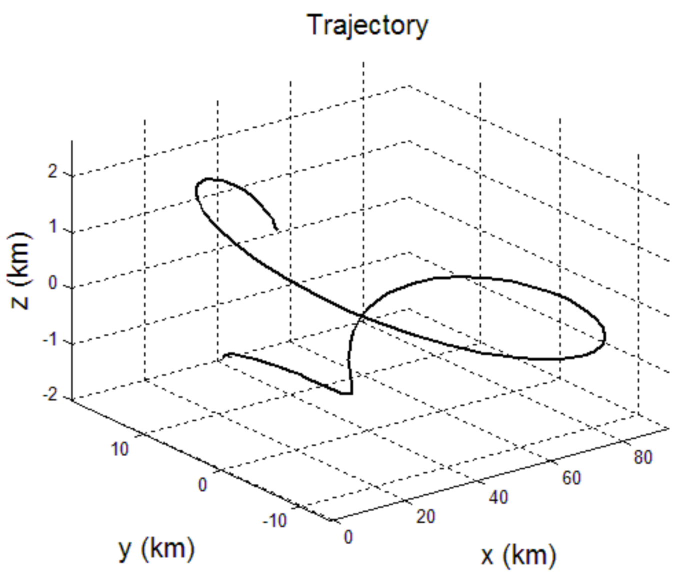

Figure 6.5: Continuous Control Resulting Trajectory (Guelman Approach)

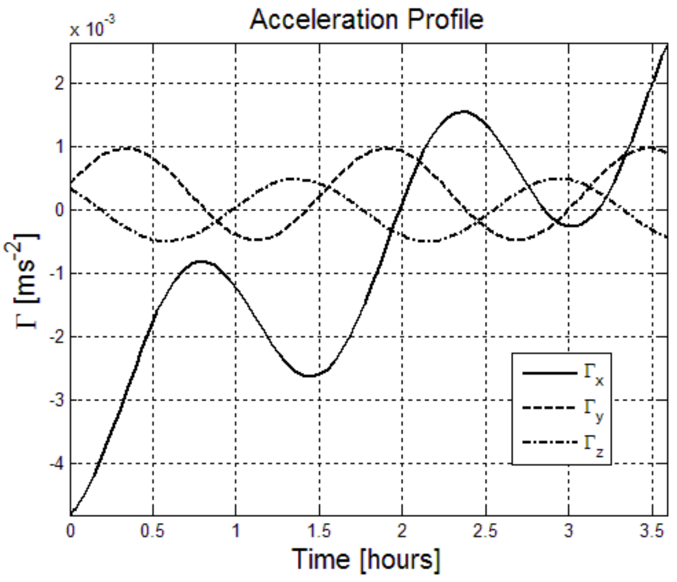

Figure 6.6: Continuous Control Resulting Trajectory (Guelman Approach)

Resulting cost, again in terms of ΔV, is $22.176\frac{m}{s}$.
This brief comparison highlighted how the use of impulses compensates the error in reaching the final desired state vector for RDD manoeuvres when the multi-level assumption is made in the optimal control generation. An intermediate state is fixed and the Hooke-Jeeves routine calculates the way to reach the vicinity of it, then pulses precisely drive the chaser to the final destination. The price to be paid is a little increase in the fuel expenditure (from $22.176\frac{m}{s}$ with upper bounded low-thrust to $23.390\frac{m}{s}$ with the hybrid technique) but still remaining acceptable if compared.

6.4.2 Are Thrust Values Acceptable?

Let us briefly make some considerations on the imposed values of continuous thrust and on the obtained pulses magnitude. As for the $5 \cdot 10^{-3}\frac{m}{s^2}$ maximum continuous acceleration, consider the case of the ATV (see Figure 9.4). After launch its mass approximates 20 *tons*, that means, considering the four main navigation engines on, a maximum acceleration of $9.8 \cdot 10^{-2}\frac{m}{s^2}$. With only one main engine on it would reduce to $2.5 \cdot 10^{-2}\frac{m}{s^2}$. Assuming to have a certain capability of orienting the smaller thrusters (220 N each) and considering to have 1 to 6 for each direction in space (x, y, z of LVLH frame) we derive an equally spaced range of accelerations from $1.1 \cdot 10^{-2}$ to $6.6 \cdot 10^{-2}\frac{m}{s^2}$. Then, the $5 \cdot 10^{-3}\frac{m}{s^2}$ assumed for the previous simulations is far lower than practical obtainable thrust on board the ATV.
As for the impulses, looking at Figure 6.3 we recognize the highest ΔV to be $2.78\frac{m}{s}$. With the maximum value of $9.8 \cdot 10^{-2}\frac{m}{s^2}$ it would require a firing for a $\Delta t = 28.4\ s$ (refer to the PWM described in chapter 3, eq. 3.29). If considering the lowest value of acceleration ($1.1 \cdot 10^{-2}$) it would be $\Delta t = 252.7\ s$. In the worst case, i.e. longest firing time interval, we are talking about 5% of the impulsive phase required time (4500 s). In the best case it reduces to 0.6% of total time.
This demonstrates the actual possibility of using the trajectories obtained with the hybrid approach on future, real space vehicles.
The following reports a simulation intended to optimize the ATV Rendezvous manoeuvre. As known "after three days of orbit adjustments, the ATV will come in sight of the ISS and will start relative navigation from about 30 *km* behind and 5 *km* below the Station". The chosen values are partly a guess, not completely responding to the so far unspecified ATV mission plan, but still will show how the hybrid method can be applied to real problems.
With a maximum thrust of $9.8 \cdot 10^{-2}\frac{m}{s^2}$, equally spaced (discretized) into 6 (multi-)levels, an initial guess on required time for the first stage of 1800 s, the same boundary conditions of table 6.1, a part from the initial point which is now $\vec{r}_0 = [30,\ -5,\ 0]^T\ km$, $\dot{\vec{r}}_0$ unchanged, the results for the low-thrust segment are shown in Figure 6.7 and Figure 6.8.

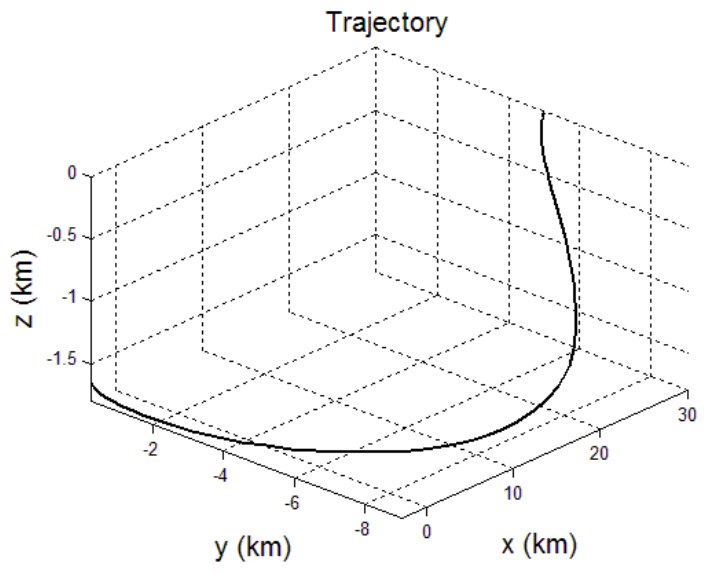

Figure 6.7: Multi-level Control resulting Trajectory for ATV

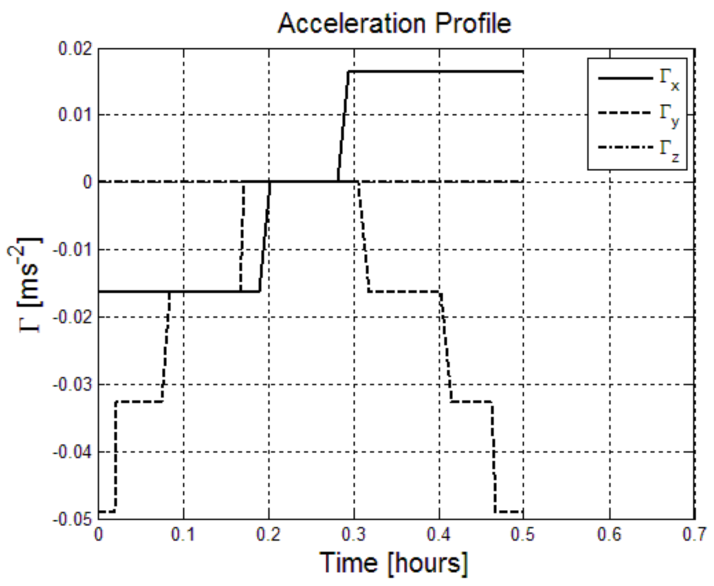

Figure 6.8: Multi-level Controls for ATV

For the present manoeuvre we also report the evolution of the error minimized by the Hooke-Jeeves routine (eq. 6.4) and of the step size as functions of the iterations (Figure 6.9).

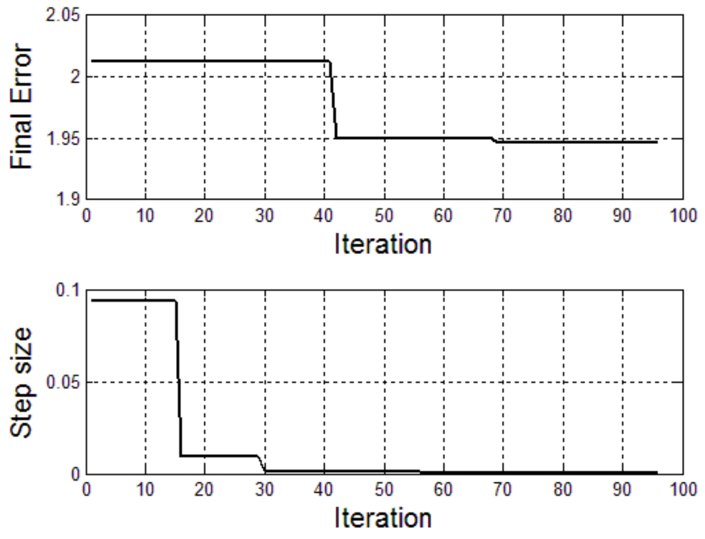

Figure 6.9: Hooke-Jeeves Routine running

The residual error in reaching the intermediate state is:

$$\begin{bmatrix} 0.226 & -0.24 & -1.631 \end{bmatrix}^T km$$

$$\begin{bmatrix} -0.0661 & -0.0335 & 0.0843 \end{bmatrix}^T \cdot 10^{-5} km\, s^{-1}$$

Time is adjusted to 1845 s.
The impulsive phase gives an optimal sequence of 6 impulses. Figure 6.11 shows the impulses insertion while in Figure 6.10 the first order discontinuity due to the pulses can be easily recognized for the last phase trajectory.

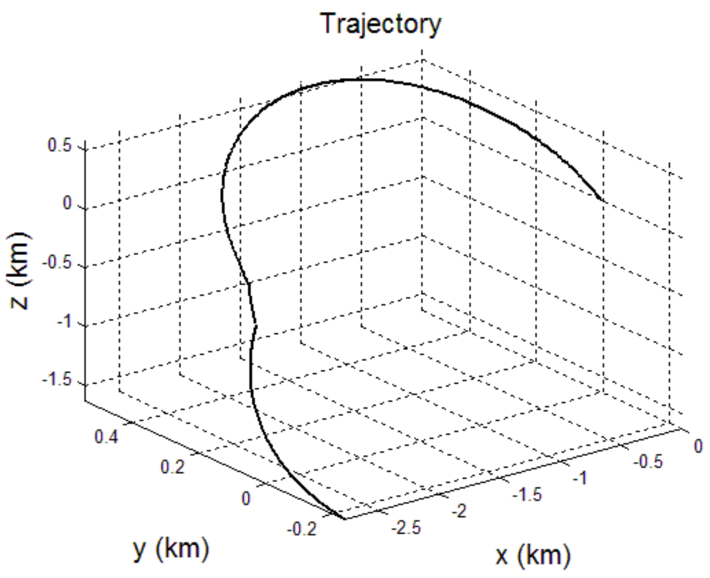

Figure 6.10: Impulsive Control Resulting Trajectory for ATV

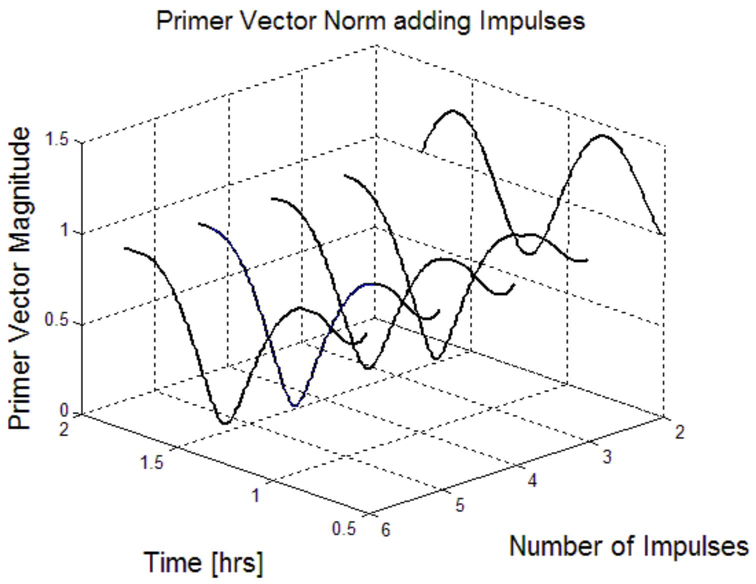

Figure 6.11: Primer Vector Magnitude Improving for ATV

Figure 6.12 shows how the overall ΔV for the impulsive phase reduces adding impulses up to six.

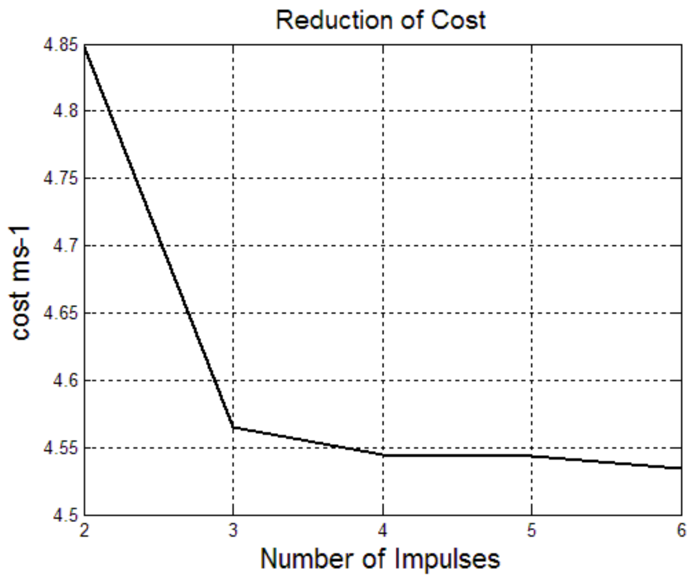

Figure 6.12: Cost Behavior as Function of the Number of Impulses

The corresponding impulses are as follows:

$$\begin{array}{rl}
\Delta V_1 = [-2.0218;\ 1.0834;\ -0.3989]\frac{m}{s} & t_1 = 1845\ s \\
\Delta V_2 = [-0.2904;\ 0.2448;\ -0.7214]\frac{m}{s} & t_2 = 2729.2\ s \\
\Delta V_3 = [-0.1414;\ 0.1451;\ -0.4949]\frac{m}{s} & t_3 = 2948.9\ s \\
\Delta V_4 = [-0.0267;\ 0.0298;\ -0.1167]\frac{m}{s} & t_4 = 3112.1\ s \\
\Delta V_5 = [0.0066;\ 0.0781;\ 0.1933]\frac{m}{s} & t_5 = 6132.3\ s \\
\Delta V_6 = [0.1011;\ 0.2345;\ 0.4582]\frac{m}{s} & t_6 = 6345\ s
\end{array}$$

Overall cost is $44.571 + 4.535 = 49.106\frac{m}{s}$.
To conclude this section we can definitely recognize that the proposed technique sees an immediate application for real space vehicles and real levels of thrust.

6.4.3 Tuning the Parameters

This section reports a parametric analysis of the multi-level control segment for the first manoeuvre of section 6.4. It is clear how the main parameters affecting the final result are:

- maximum value of thrust
- time required
- number of discretization levels for thrust (number of engines on board)

There is no reason to increase the refinement of the thrust when the final time and the maximum acceleration are not related. In fact, as well known, in fuel minimum problems the maximum control and the time required for the manoeuvre are strongly related. In [29] a numerically solvable expression is derived in order to give the minimum required time to have not saturated flight, i.e. the engines are never at their maximum performance. Here the approach is different: having imposed on-off behavior for the thrusters we are always saturating, at least a subset of the whole cluster. From this point of view it makes less sense to look for a minimum time for unsaturated flight, much more reasonable is to find the required time inferior limit under which there is no way to reach the final position, even exerting the maximum acceleration (all engines on).
Figure 6.13 shows the error on reaching the final state as function of maximum acceleration and refinement of it, final time is left 8500 s. Maximum acceleration is varied between $5 \cdot 10^{-3}$ and $24 \cdot 10^{-3}$, discretization from 5 to 30 thrust levels:

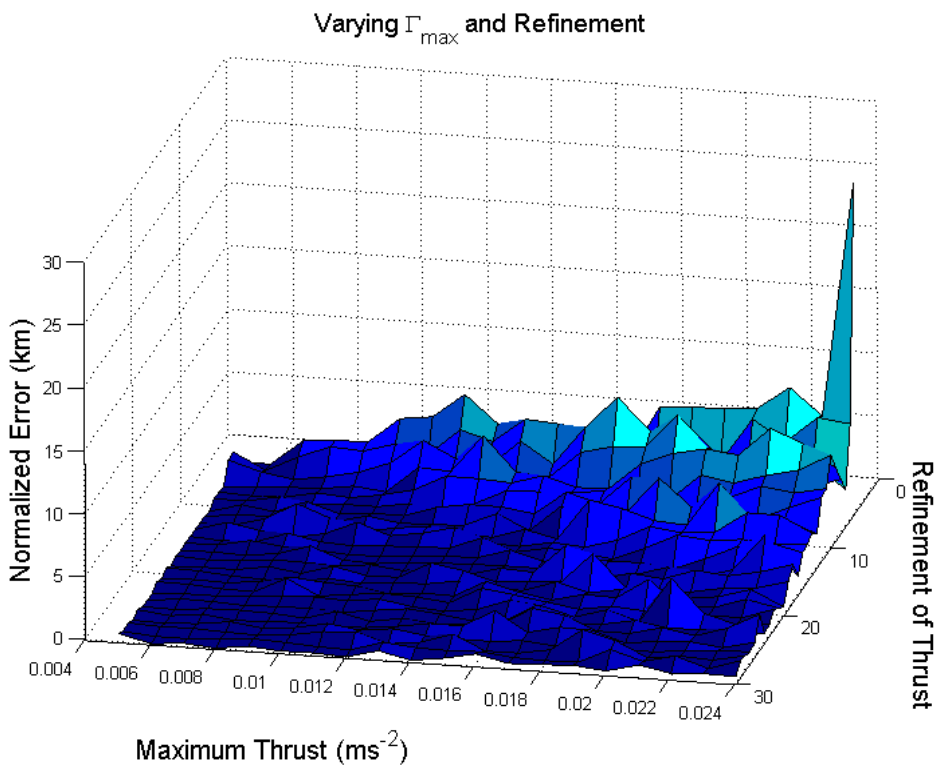

Figure 6.13: Analysis on Error Dependency w.r.t. Maximum Thrust and Refinement

The best combination is in fact for the lowest value of maximum thrust, 11 thrust levels. In this case the final condition is very close to the desired one:

$$\vec{r}(t_f) = [-2.9597,\ -0.0142,\ -0.0463]^T\ km$$

$$\dot{\vec{r}}(t_f) = [2.4825,\ 0.0281,\ 0.0079]^T\ \frac{m}{s}$$

Figure 6.14 catches the relation above mentioned between required time and maximum thrust, for a given number of 5 thrust levels. Time varies from 5000 s to 30000 s, maximum thrust in the range $5-8\cdot 10^{-3}\frac{m}{s^2}$:

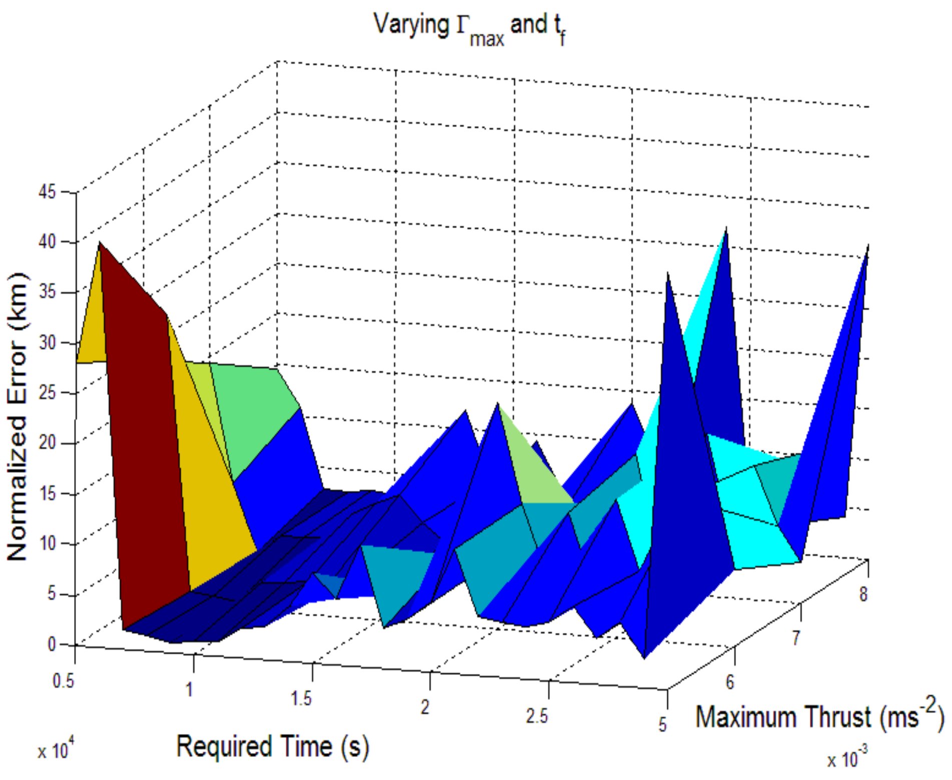

Figure 6.14: Analysis on Error Dependency w.r.t. Maximum Thrust and Required Time

The best combination arises for $6\cdot 10^{-3}\frac{m}{s^2}$ and 9000 s, with a normalized error of 0.7613 km. When a manoeuvre has to be performed a previous tuning of the parameters is required. Having related required time to the upper bound of thrust it is then straightforward to improve the accuracy in reaching the final state by increasing the thrust refinement. There is no analytical relation between time and thrust, one can solve the upper bound problem as done by Guelman in [29] adjusting these two values (better say adjusting required time when the upper bound is given). After this first tuning a good guess is given for the hybrid technique, in particular for the first segment of RDD. This is how we have proceeded for the previous simulations.

6.5 Considerations on Software

The Hooke-Jeeves routine, that has been completely developed from scratch, needed a very reduced number of iterations for this problem ($<$ 100 for each run). Another advantage of this kind of first order algorithms is their reliability (as the Nelder-Mead downhill simplex algorithm of chapter 5), in the worst case restituting a local minima, but still they always give an answer when the maximum number of iterations is reached. The first segment (the low-thrust one) solution for the ATV example required around 17 s of calculation on a Centrino Duo Intel® T2250 @$1.73Ghz$ machine. This elapsed time refers to a double use of the routine, the first time it generates a very good guess on Λ_0, with fixed final time, then it optimizes Λ_0 allowing t_f to change, i. e. minimizes function 6.4. Figure 6.9 shows the second set of iterations.
As for the impulsive phase, the conjugate gradient routine ([71]) it recalls the Numerical Recipes instructions ([64]). Adapted to the problem under study it took less than 25 s to calculate the optimal sequence of 4 impulses showed in Figure 6.11.
The basic idea of a conjugate gradient technique is to speed up what the steepest descent method does. Steepest descent often finds itself taking steps in the same direction as earlier steps. A conjugate gradient algorithm aims doing the correct move for each direction once, then working on the remaining and so on. The detailed description of such algorithms goes beyond the scopes of this dissertation as they have been a powerful already known mean to solve the impulses problem. For a wide explanation refer to [70].

6.6 Comments and Future Developments

In order to consider real thrusters for rendezvous and docking an hybrid technique has been presented to fuel optimize the maneuvers. The relative motion between the agents involved in the PROXOP is represented by the Clohessy and Wiltshire linear model. A finite set of low thrusts is considered for far away maneuvers, impulses for the very last phase of flight to the target. This way to face the problem sees its application when a set of different sized thrusters is thought to be mounted on the chaser agent. Electrical engines can be used for the far away segment, chemical ones for the more accurate final stage. The approach takes into account real limitations of the current space-qualified rockets, among all the difficulties in varying the thrust magnitude continuously in time (see appendix F).
The low thrust segment is solved setting up a minimization problem, using a first order algorithm. The second phase is tackled by respecting Lawden's conditions, via a conjugate gradient technique. In this way a very reliable procedure is generated to determine the switching structure of the low thrust part and the number, magnitudes, directions, times and space collocation of the firings related to the second span of trajectory.
It has also been shown how the trade-off between the continuous adjustable low-thrust approach and the one here developed is very convenient: the errors arising from discretizing the acceleration profile can be corrected via impulsive manoeuvres, without increasing too much the resulting fuel consumption. The approach has been verified on some data taken from the ATV vehicle, again showing good performances. Tuning is needed for final time (when maximum

thrust is fixed) and levels of discretization. A way to face the tuning process has been also reported.
The set up programs converge to the final solution in a total time less than 50 s, making the approach a very good candidate for real-time implementation. Major modifications on the Hooke-Jeeves and on the conjugate gradient routines will be the constraining of the optimization variables (see also section 5.7), after that the translation into Embedded Matlab will provide the final mean for real-time testing on the chosen target machine (section 4.7).

Chapter 7

Periodic Relative Motion

7.1 Introduction

The possibility to obtain natural periodic motion of formation flying Earth satellites is investigated through the use of a genetic global optimizer.
This aspect of relative motion sees its immediate usefulness in fuel saving when performing formation keeping operations. Possible applications of the results are presented in section 7.6 with a STS monitoring example. A wide range of possible uses of such results can be thought, both for formation keeping (telecom satellites and so on) and monitoring of big orbiting structures.
The introduced algorithm has been initially tested and tuned on the known unperturbed case (perfectly spherical uniform mass Earth, no other forces), where the period matching between the two agents is necessary and sufficient condition to have invariant relative trajectories. In the J_2 perturbed case, the conditions to obtain an invariant relative motion are known only in approximated closed forms, which, in some cases, guarantee the minimum drift, but not the relative motion periodicity. Using the genetic algorithm, periodic relative motion for satellites on particular J_2 perturbed orbits are found. Respecting the obtained initial conditions and being the reference orbit in the vicinity of four inclinations ($63.4°$ and $116.6°$ for all eccentricities (already known Molniya orbits) and $49°$ and $131°$ for nearly circular orbits) periodicity or quasi-periodicity can be achieved. For other perturbations the method is able to supply initial conditions for minimum drift, or, in the case of drag subjected orbits, for formations which get close after a predetermined time interval.
In order to keep the satellites of the formation in the designed configuration, and therefore to achieve the mission's goals, control actions are needed. The cost of this orbital control in terms of ΔV limits both the mission duration and the expected performances. Advantageous dynamics could significantly reduce the cost of these operations, in particular periodic or quasi-periodic motion would be a big saving, almost canceling the residual drift. Different approaches enrich the recent literature on this topic. Inalhan, Tillerson and How ([72]) found the analytical expression for the initial conditions giving periodic motion based on the classical Tschauner-Hempel equations ([12]); Kasdin and Koleman ([73]) used the epicyclic orbital elements theory to derive bounded, periodic orbits in presence of various perturbations; Vaddi, Vadali, and Alfriend

([74]) studied an Hill-Clohessy-Wiltshire ([7]) modified system to include second order terms; finally, Schaub and Alfriend ([9]) formulate the conditions for invariant J_2 relative motion basing on relations between the mean orbital elements of the two satellites.
In the previous works the analytical approach leads two kinds of results: initial conditions which ensure exact periodicity in approximated dynamical models or initial conditions resulting in bounded (i.e. with minimum drift, but not periodic) relative motion of more detailed dynamical models.
A numerical approach, though not providing a physical insight of the problem, definitely guarantees an answer about the possibility to have periodic trajectories for satellites in a fully non linear, perturbed environment. While some results are easily predictable, like the disruptive effect of non conservative forces as atmospheric drag, some others are quite surprising and interesting. In particular, the possibility to have periodic motion is negated, as shown later on, also for a conservative, symmetric perturbation like the J_2 effect with four remarkable exceptions: when the formation reference elliptical orbit shows an inclination in the vicinity of the critical values of $63.4°$ and $116.6°$ and, for nearly circular orbits, of $49°$ and $131°$, the relative motion can be considered really periodic. While the physical reasons of this behavior are still under study, a simple conclusion can be drawn: if two satellites have to remain in close formation, the proper choice of the parameters of the reference orbit is of fundamental importance, and it results in control cost saving for formation keeping.

7.2 Problem Statement

Consider the following generic optimization problem: given a model (in the present case the relative dynamics of satellites flying in formation) depending on a set of parameters κ, a functional relation $f(\kappa)$ returns a measure of the quality for the corresponding model, this will be called the fitness function. The optimization task consists in finding the point κ^* defining the model parameters that maximize the quality measure $f(\kappa)$. The software main architecture developed for this work is similar to that of [75] and its name is PIKAIA.
The problem statement follows. A chief satellite initial conditions are given, in terms of absolute position and velocity. A deputy is introduced by relative position and velocity with respect to the master. Both orbits are propagated, i.e. equation 3.1 is integrated for each satellite. Relative motion is deduce subtracting the obtained state vectors. For doing this double precision variables have been used.
The variables on which we want the GA to work on will be the relative state vector at initial time. For more details on Genetic Algorithms refer to the Appendix G.
An individual of the gene pool is characterized by a single chromosome, composed by seven genes, described by the unknowns of the problem (it simplifies reality: each individual has a DNA composed by a single chromosome):

$$\kappa = \begin{bmatrix} x_i \\ y_i \\ z_i \\ \dot{x}_i \\ \dot{y}_i \\ \dot{z}_i \\ t_{end} \end{bmatrix} \tag{7.1}$$

The chosen variables are the initial relative position (x_i) and velocities ($\dot{x}_i$) determining the formation motion. Also the epoch t_{end} of the fitness function evaluation is considered allowing the optimizer to find the motion's periodicity. In other words, after t_{end} seconds the relative vector between the spacecrafts is evaluated and compared with the one at t_0, through the use of a fitness function (eq. 7.2) which measures how close the state vectors are.

Note that it is actually not possible to use directly initial conditions and propagation time as genes. In fact a GA uses these chromosomes after encoding/decoding processes (see Appendix G).

Because the mathematical parameters of the GA are defined as $\bar{\kappa}_k \in [0,\ 1]$, we have to define a simple transformations between the original κ and the bar one. For the initial relative distances:

$$[x_i,\ y_i,\ z_i] = (-1 + 2[\bar{\kappa}_1,\ \bar{\kappa}_2,\ \bar{\kappa}_3])K$$

This limits the range of variation for the initial relative position to $[-K,\ K]\ km$, where K represents the maximum dimension for the formations we are interested in. Then, to relate the initial relative velocities to the GA parameters:

$$[\dot{x}_i,\ \dot{y}_i,\ \dot{z}_i] = \left(-10^{-2} + 2[\bar{\kappa}_4,\ \bar{\kappa}_5,\ \bar{\kappa}_6]2 \cdot 10^{-2}\right)K$$

In this way initial velocities are in the range $[-10^{-2},\ 10^{-2}]K$, in fact they are usually about two orders of magnitude lower than the initial distances (actually, order of the orbital angular velocity).

As for the propagation time, it will be chosen as $t_{end} = t_{kepler} \pm \bar{\kappa}_7 k$,where k is a constant properly chosen (some hundred of seconds) and t_{kepler} is the well known orbital period $2\pi\sqrt{\frac{a^3}{\mu}}$. This last variable t_{end} is a crucial one, because at this time the final relative coordinates are compared to the initial relative coordinates. These two compared sets of relatives states (initial and final) determine the quality of the individual. A good individual shows the difference between the two relative states (initial and final) close to zero. In this way its position in the individual ranking will be high, and so it has the chance to mate and to generate "good" sons. Its genes will survive in the next generation, and if they will be placed first, in the individual ranking at the last generation, they represent the set of initial conditions that generate a closed relative orbit, if it exists. Otherwise, they represent the set that more than other generates quasi-invariant orbits. How good is an individual, and consequently how its genes can be considered representative of the solution that generate closed relative orbits, is established by the value of the fitness function.

In this work, after an exhaustive set of tests on different functions, an individual's quality is measured by the following fitness:

$$f(\kappa) = \frac{1}{\sqrt{\left(\frac{x_f - x_i}{x_i}\right)^2 + \left(\frac{y_f - y_i}{y_i}\right)^2 + \left(\frac{z_f - z_i}{z_i}\right)^2 + \left(\frac{\dot{x}_f - \dot{x}_i}{\dot{x}_i}\right)^2 + \left(\frac{\dot{y}_f - \dot{y}_i}{\dot{y}_i}\right)^2 + \left(\frac{\dot{z}_f - \dot{z}_i}{\dot{z}_i}\right)^2}} \tag{7.2}$$

A high fitness indicates that after t_{end} from the initial time the spacecraft show a relative state vector very close to the initial one. For $f(\kappa) \to \infty$ the mechanical deterministic principle ensures that the relative motion will repeat itself, thus being invariant.
The N_{ind} individuals of the population are randomly initialized. After mating, the chromosomes of the offspring differ from the chromosomes of the parents, because of crossover and mutation processes. The best individuals have a greater probability to mate, and so their chromosomes have a greater probability to pass their good characteristics to the offspring. After N_{gen} generations the best individual's chromosomes represent the solution provided by the GA.
In this kind of heuristic methods, the algorithm tuning is an essential and very time-demanding part of the work. The number N_{ind} of individuals in the population, the number N_{gen} of generations, the minimum and maximum mutation rate, the crossover probability and many other parameters affect the GA results.

7.3 Orbital Propagator and Atmospheric Model

The numerical orbital propagator used for this work was previously developed at the University of Rome, La Sapienza, Department of Aerospace Engineering, by one of the authors of [76] and set up for dealing with two spacecraft orbits. It is capable of up to 23 zonal harmonics and 8 tesseral, including SRP and moon-sun combined attraction. The various effects, a part form the very basic keplerian force $-\frac{\mu}{r_{sat}^3}\vec{r_{sat}}$, can be switched on or off at will in the propagator.
As for the atmosphere (section 7.6) a modeling of the air density is very difficult to be faithful to reality. The atmosphere conditions change with altitude, time, and many other variables. In general the effects of drag are expected to be very low for heights superior to 500 *km*, while are very important for heights lower than 300 *km*. At $100 - 150$ *km* a satellite is usually considered lost because the air drag will cause a very fast orbit decay. At 80 *km* above the Earth's surface we have the so-called upper atmosphere.
The drag perturbation force per unit mass (acceleration) can be represented as:

$$\vec{f}_D = -\frac{1}{2}\frac{C_D A}{m}\rho V^2 \hat{V} \tag{7.3}$$

where ρ is the air density, $\vec{V}$ is the relative speed of the satellite with respect to the Earth's atmosphere, $\hat{V}$ the corresponding unit vector, A and m are the frontal area and the mass of the satellite. C_D is the air drag (or resistance) coefficient, typically ranging between 2.0 and 2.6 for low orbit satellites.
ρ is the most difficult quantity to evaluate. It can be approximated with various models. The one here used is a JACCHIA ([77]) model for high altitudes (100 *km* $< h <$ 600 *km*), which considers a great number of variables, mainly height, day of the year and time of the day.

The atmosphere can roughly be characterized as the region from sea level to about 1000 *km* altitude around the globe, where neutral gases can be detected. Below 50 *km* the atmosphere can be assumed to be homogeneously mixed and can be treated as a perfect gas. Above 80 *km* the hydrostatic equilibrium gradually breaks down as diffusion and vertical transport become important.
The major species in the upper atmosphere are N_2, O, O_2, H, He. Temperature-oriented nomenclature differentiates the strata of the atmosphere as follows: the troposphere, from sea level up to about 10 *km*, where the temperature decreases; the stratosphere, from 10 *km* up to about 45 *km*, where the temperature increases; the mesosphere, from 45 *km* up to about 95 *km*, where the temperature decreases again; the thermosphere, from 95 *km* to about 400 *km*, where the temperature increases again; and the exosphere, above about 400 *km*, where the temperature is constant.
The first global models of the upper atmosphere were developed by L. G. Jacchia in the early sixties based on theoretical considerations and satellite drag data. Since the launch of Sputnik 1 in 1957, orbit decay of artificial satellites has been used to derive atmospheric data. Several national and international organizations have established committees for the development of atmospheric reference models, e.g., the International Civil Aviation Organization (ICAO), the Committee on Space Research (COSPAR), and the Committee on Extension to the Standard Atmosphere (COESA). Probably the most widely used and well established model is the COSPAR International Reference Atmosphere (CIRA), an effort that started in 1961 with the publication of CIRA-61. CIRA-72, the third generation of this model, CIRA-86, includes Jacchia's 1971 model.
The Jacchia 1977 thermospheric model is combined with the U.S. Standard Atmosphere 1976 model. Temperature and component number densities are calculated from the ground to the maximum altitude requested. The only variable input parameter is the exospheric temperature.
When a perturbation like air drag is considered, other satellite's physical properties, and not only its orbital parameters, have to be determined. The following are the values here adopted:

- $m = 1850\ Kg$;
- Equivalent diameter $D = 2.52\ m$;
- Frontal area $A = \pi\frac{D^2}{4} = 5\ m^2$;
- Drag coefficient $C_D = 2.2$

7.4 Unperturbed Case

This paragraph analyzes the case of perfectly spherical uniform mass Earth, no other forces are considered. In this section classical approaches found in literature are used and compared against the numerical GA technique here developed.
The orbital period matching condition is the constraint to be met in order to have periodic motion. The results obtained in [74] and [73] are here used to set up the GA parameters and establish its performances.

The very first step is the use of the GA on the Hill-Clohessy-Wiltshire (HCW) linear equations, valid for circular unperturbed reference orbits:

$$\begin{cases} \ddot{x} - 2\omega_{LVLH}\dot{y} - 3\omega_{LVLH}^2 x = 0 \\ \ddot{y} + 2\omega_{LVLH}\dot{x} = 0 \\ \ddot{z} + \omega_{LVLH}^2 z = 0 \end{cases} \tag{7.4}$$

The periodicity condition being:

$$\dot{y}_0 = -2\omega_{LVLH} x_0 \tag{7.5}$$

In [52] the GA was already proved to be capable of reproducing condition in eq. 7.5. In the present work the following are the set up parameters and characteristics for the GA (see Appendix G):

Table 7.1: Set up Parameters for the Genetic Optimizer

N_{ind}	20
N_{gen}	5000
Number of significant digit (number of genes):	9
Crossover probability:	0.85
Mutation mode:	one-point, adjustable rate based on fitness
Initial mutation rate:	0.005
Minimum mutation rate:	0.0005
Maximum mutation rate:	1
Reproduction plan:	Steady-state-replace-worst

As the formation dimensions grow the non linearities make condition of eq. 7.5 no longer valid. Trajectories in Figure 7.1 are obtained by propagating the initial conditions of eq. 7.5 with a nonlinear model for 11 periods.
In Figure 7.2 the field of validity of the linear approximation is reported after defining the initial formation dimension as K. The initial condition will then be: $K[1, 0, 0.5, 0, -2\omega_{LVLH}, 0]\,km$. The obtained drift is the dependent variable, measured as the difference between the spacecraft relative distance at initial time and after a period of propagation with a nonlinear model.
Vaddi, Vadali and Alfriend ([74]) developed a model that takes into account the effects of non linearities, both for circular and for elliptical orbits.
A perturbative approach with respect to the parameter $\varepsilon = \frac{3\mu}{a^4}$ in the HCW equations, leads to the following model:

$$\begin{cases} \ddot{x} - 2\omega_{LVLH}\dot{y} - 3\omega_{LVLH}^2 x = \varepsilon\left[\frac{y^2}{2} + \frac{z^2}{2} - x^2\right] \\ \ddot{y} + 2\omega_{LVLH}\dot{x} = \varepsilon xy \\ \ddot{z} + \omega_{LVLH}^2 z = \varepsilon xz \end{cases} \tag{7.6}$$

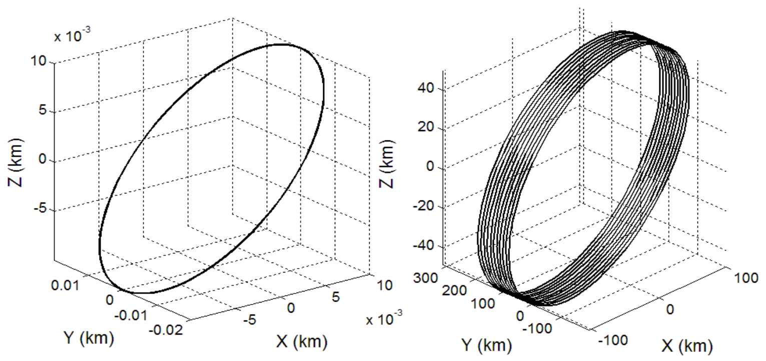

Figure 7.1: Relative trajectories descending from HCW condition for a small formation (left) and a large formation (right)

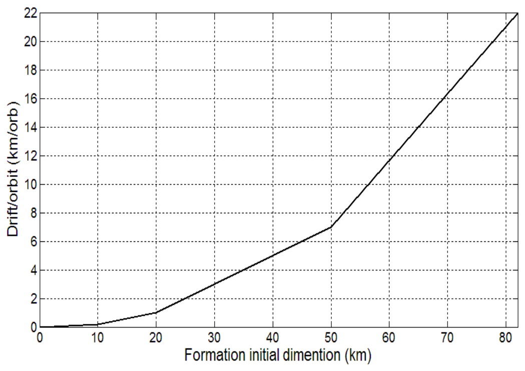

Figure 7.2: Field of validity of the HCW condition

A condition for periodic relative orbits is then reached:

$$\begin{aligned}
[x_0, y_0, z_0] &= \left[\frac{\rho}{2} sin(\omega_{LVLH} t + \alpha_0), \rho cos(\omega_{LVLH} t + \alpha_0), \rho sin(\omega_{LVLH} t + \alpha_0)\right] \\
[\dot{x}_0, \dot{y}_0, \dot{z}_0] &= \left[\frac{\omega_{LVLH}\rho}{2} cos(\omega_{LVLH} t + \alpha_0), \dot{y}, \omega_{LVLH}\rho cos(\omega_{LVLH} t + \alpha_0)\right]
\end{aligned} \tag{7.7}$$

where ρ is the relative distance and α_0 the initial relative phase angle. The only variable influencing the shape of the relative orbit is $\dot{y}$, which can be written as:

$$\dot{y}(0) = \dot{y}_h(0) + \varepsilon \dot{y}_{cn}(0) \tag{7.8}$$

where $\dot{y}_h$ is the initial condition from HCW (eq. 7.5) and $\dot{y}_{cn}$ is the correction for the non linearity:

$$\dot{y}_{cn}(0) = -\left(\frac{\rho^2}{48\omega_{LVLH}}\right)(12 + 6cos2\alpha_0) \tag{7.9}$$

A different analytical approach is found in [73]. Here Kasdin and Koleman use an Hamiltonian approach to derive the equations of motions for an object relative to a circular or slightly elliptical reference orbit. By solving the Hamilton-Jacobi equation in terms of the epicyclic elements they are able to provide analytical approximations of the invariance condition. By means of this formalism, they derive bounded, periodic orbits in the presence of various perturbations, among them the non linearities. Here we only report the conditions found for the circular reference orbit case. Two expressions are given to compute a normalized $\bar{y}_0$ (in eq. 7.13 the bars stand for the distances being normalized by the reference orbit semi-major axis a, the time by the angular velocity ω_{LVLH}, giving a-dimensional quantities): one considers second order terms in the series expansion for the initial conditions, the other also third order terms:

$$a_3(0) = -\frac{5}{2}a_1^2(0) - \frac{1}{2}\left(a_2^2(0) - b_1^2(0) + b_2^2(0)\right) - 3a_1(0)b_3(0) - b_3^2(0) \tag{7.10}$$

$$\begin{aligned}
a_3(0) = &-\frac{5}{2}a_1^2(0) - \frac{1}{2}\left(a_2^2(0) - b_1^2(0) + b_2^2(0)\right) - 3a_1(0)b_3(0) + \\
&-b_3^2(0) - \frac{3}{2}\left(a_1^2(0)b_1(0) + \right. \\
&\left. + a_2^2(0)b_1(0)\right) + \frac{1}{2}b_1^3(0)
\end{aligned} \tag{7.11}$$

In both cases:

$$\begin{aligned}
a_1 &= \sqrt{2\alpha_1}cos\beta_1 \\
b_1 &= \sqrt{2\alpha_1}sin\beta_1 \\
a_2 &= \sqrt{2\alpha_2}cos\beta_2 \\
b_2 &= \sqrt{2\alpha_2}sin\beta_2 \\
a_3 &= \alpha_3 \\
b_3 &= \beta_3
\end{aligned} \tag{7.12}$$

α_i, β_i are the initial canonical momenta and coordinates, which can be written as functions of the initial conditions (for brevity no 0 is reported to indicate initial time in eq. 7.13):

$$\begin{aligned}
\alpha_1 &= \frac{1}{2}\left(\dot{\bar{x}}^2 + (2\bar{y}+3\bar{x})^2\right)\\
\alpha_2 &= \frac{1}{2}\left(\dot{\bar{z}}^2 + \bar{z}^2\right)\\
\alpha_3 &= \dot{\bar{y}} + 2\bar{x}\\
\beta_1 &= -tan^{-1}\left(\frac{3\bar{x}+2\dot{\bar{y}}}{\dot{\bar{x}}}\right)\\
\beta_2 &= -tan^{-1}\left(\frac{\bar{z}}{\dot{\bar{z}}}\right)\\
\beta_3 &= -2\dot{\bar{x}} + \bar{y}
\end{aligned} \tag{7.13}$$

Substituting eq. 7.13 in eq. 7.12 and imposing the conditions in eq. 7.10 or in eq. 7.11 (according to the order of the chosen approximation), and solving for $\dot{y}$, gives the initial $\dot{y}$ for bounded orbits. The difference between the semi-major axes of the spacecraft in the formation is a good index of how near the approximation of the proposed analytical conditions is to the physical one (i.e. period matching); in fact a measure of the drift per orbit can be given ([78]) as:

$$\frac{drift}{orbit} = -3\pi\Delta a \tag{7.14}$$

The difference Δa resulting by using condition 7.8, 7.10 or 7.11 can be plotted for various formation dimensions; as shown by Figure 7.3, the third-order epicyclic conditions are a very good approximation of the period matching conditions, and indeed the use of a numerical approach such as GA seems not really necessary in this case:

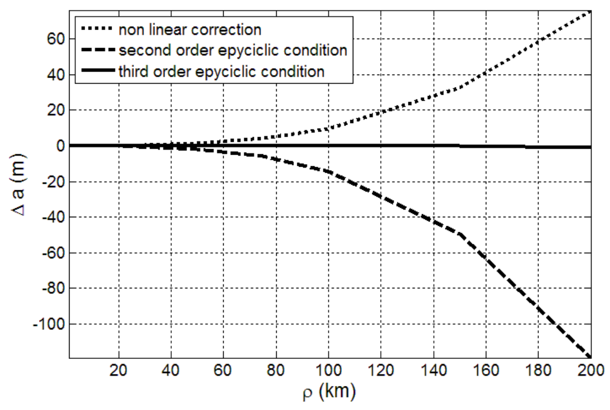

Figure 7.3: Difference of the semi major axis vs. initial dimensions

The comparison between the best analytical (third-order epicyclic) and numerical (GA) solutions is showed in Figure 7.4:

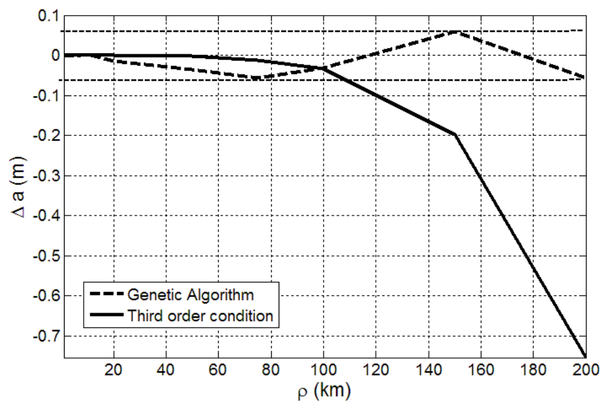

Figure 7.4: Comparison between GA and third-order epicyclic condition

Figure 7.4 shows the main difference between analytical and numerical approach: Δa due to the genetic algorithm conditions oscillates because of the stochastic nature of the optimizer, while Δa due to third-order conditions grows with the formation dimensions. However, also for very large formations, the results of the analytical condition are satisfactory and the use of GA is not really necessary. Then, at this stage, the set up GA showed its ability to find a known solution with great accuracy. With no variation, the same results can be obtained for elliptical unperturbed reference orbits.

7.5 J_2 Perturbed Case

In this paragraph the J_2 perturbation is considered and the aim is no more to test the GA performances with respect to a well known solution, but rather searching if such a solution exists. A numerical approach can supply precious information to be later explained.
For LEO and MEO the J_2 effect and air drag are by far the most important perturbations.
The results here reported are the most interesting the GA approach has returned. In fact, while in the Keplerian case of 7.4 the solution was very well known, and in the drag perturbed case 7.6 the solution is easily predictable not to exist, in the J_2 perturbed case the question is open.
An analytic method presented by Schaub and Alfriend ([9]) establishes J_2 invariant relative orbits. Working with mean orbital elements, the secular drift of the longitude of the ascending node and the sum of the argument of perigee and mean anomaly are set to be equal between two neighboring orbits. By having both orbits drifting at equal angular rates on the average, they will not separate over time due to the J_2 influence. Two first order conditions are established among the differences in momenta elements (semi-major axis, eccentricity and inclination angle):

$$\begin{aligned} \delta a &= 2 D a_0 \delta\eta \\ \delta e &= \frac{\left(1-e^2\right) tan(i)}{4e} \delta i \end{aligned} \tag{7.15}$$

where:

$$\begin{aligned} \delta\eta &= -\frac{\eta_0}{4} tan(i_0)\delta i \\ \eta &= \sqrt{1-e^2} \end{aligned} \tag{7.16}$$

and D is a parameter depending on i, a, η. The combination of eq. (7.15) and eq. (7.16) provides the two necessary conditions on the mean orbital element differences yielding a J_2-invariant relative orbit. When designing a relative orbit using the mean orbital element differences, δi or δe or δa is chosen, the remaining two elements differences are then prescribed through the two constraints. The remaining elements differences $\delta\Omega, \delta\omega$ and δM can be chosen at will no affecting the J_2-invariant conditions. Note that these two conditions are only valid for a first order approximation, the relative orbit still exhibiting a relative drift, as Figure 7.5 shows for an almost circular 35° inclined reference orbit. Propagation is again performed via a nonlinear model including this time second order terms.

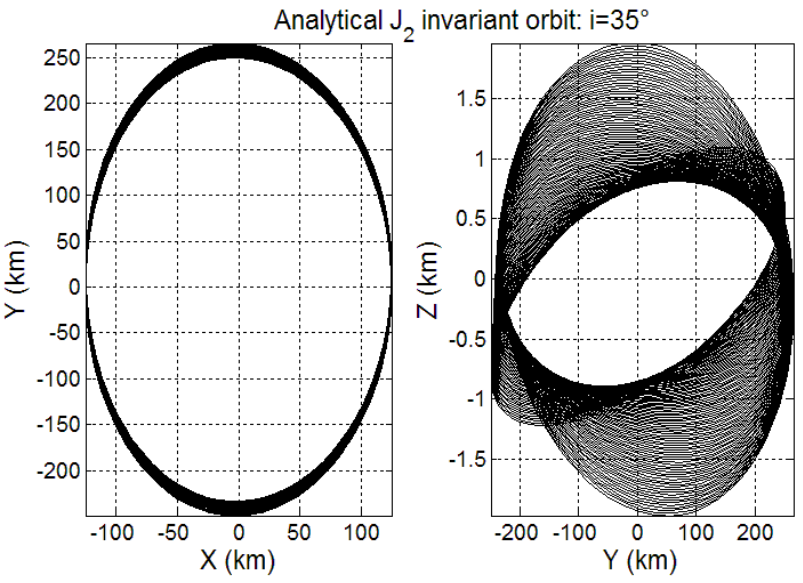

Figure 7.5: XY and YZ projections of relative orbits generated with analytical J_2 invariant conditions ($i = 35\,\mathrm{deg}$)

The conditions (7.15) and (7.16) supply two powerful means to find relative orbits not properly periodic but bounded, with minimum drift per orbit. GA can be then used to verify if there is an actual physical limit for the existence of really periodic orbits, or if the residual drift is only due to the introduced approximations. In general a run of GA does not improve in a remarkable way the analytical results: the drift is minimum but the periodicity is not reached for a generic inclination. Instead, if the simulation is repeated for the entire range of inclinations, the results vary sensibly disclosing a previously overlooked feature of the invariant relative motion. Figure 7.6 reports the best individuals fitness function values for inclinations from 0 to 180 degrees (very low eccentricity):

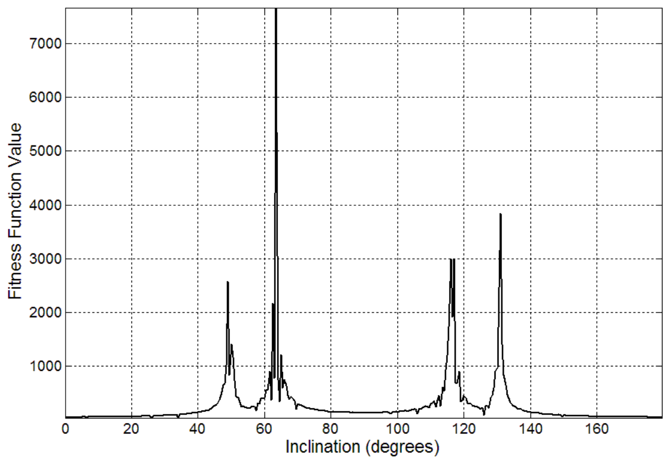

Figure 7.6: Fitness function values for the whole range of inclinations (nearly circular orbit)

Drifting trajectories as the one in Figure 7.5 are obtained, with two remarkable exceptions: 49° and 63.4°, and their symmetric counterparts (with respect to 90°), i.e. 131° and 116.6°.
The stochastic component proper of genetic algorithms is definitely not responsible for these peaks, being clearly particular cases where the orbits are periodic (Figure 7.7, Figure 7.8).
For 63.4° and 116.6°, the reason for this behavior has to be searched in the cancelation of the mean secular drift of the argument of perigee, according to eq. (7.17):

$$\Delta\bar{\omega} = \frac{3}{2}\pi J_2 \frac{R_\oplus^2}{p^2}\left(5\cos^2 i - 1\right) \tag{7.17}$$

Analyzing Gauss' equations:

$$\begin{aligned}
\frac{da}{dt} &= \sqrt{\frac{p}{\mu}}\frac{2p}{(1-e^2)^2}\left[esin\theta\frac{f_r}{m} + (1+ecos\theta)\frac{f_\theta}{m}\right]\\
\frac{de}{dt} &= \sqrt{\frac{p}{\mu}}\left[sin\theta\frac{f_r}{m} + \frac{e+2ecos\theta+ecos^2\theta}{1+ecos\theta}\frac{f_\theta}{m}\right]\\
\frac{d\Omega}{dt} &= \sqrt{\frac{p}{\mu}}\frac{1}{sin(i)}\frac{sin(\theta+\omega)}{1+ecos\theta}\frac{f_z}{m}\\
\frac{d\omega}{dt} &= \sqrt{\frac{p}{\mu}}\left[\frac{-cos\theta}{e}\frac{f_r}{m}\frac{(2+ecos\theta)sin\theta}{e(1+ecos\theta)}\frac{f_\theta}{m} - \frac{1}{tan(i)}\frac{sin(\theta+\omega)}{1+ecos\theta}\frac{f_z}{m}\right]\\
\frac{di}{dt} &= \sqrt{\frac{p}{\mu}}\frac{cos(\theta+\omega)}{1+ecos\theta}\frac{f_z}{m}
\end{aligned} \tag{7.18}$$

it is evident that the argument of perigee affects, through the inclination i, the whole set of equations. A time increasing ω forces the orbital elements to have different periods: this means that the absolute motion of the single satellite is not periodic and neither can be the relative motion between the spacecraft of the formation. Instead, at critical inclinations, a periodic absolute motion is possible, and so is a relative trajectory.

Figure 7.7, referring to a 100 orbits propagation, shows that at this inclination the orbit is not simply bounded, but really periodic.

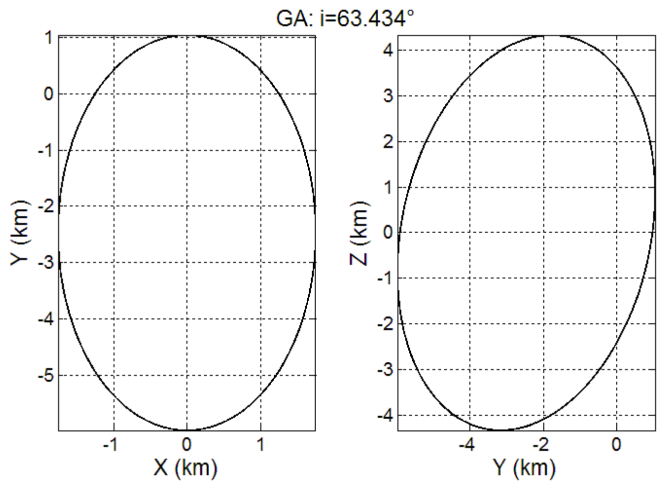

Figure 7.7: 100 relative orbits for a J_2 perturbed case at 63.4° inclination

The corresponding optimal relative state vector and time are:

$$\kappa^* = \begin{bmatrix} 0.602\ km \\ 0.848\ km \\ -6 \cdot 10^{-2}\ km \\ -5.320 \cdot 10^{-3} \frac{km}{s} \\ -1.402 \cdot 10^{-3} \frac{km}{s} \\ -9.339 \cdot 10^{-3} \frac{km}{s} \\ 5427.514 + 25.136\ s \end{bmatrix} \tag{7.19}$$

Though the result is very similar to the unperturbed case, here the condition is no more of period matching as a difference in all the six orbital elements is kept, as reported in table 7.2.

Table 7.2: Comparing Orbital Elements for J_2 Case

	chief	deputy	Difference
a	6678 km	6677.7091 km	-0.291 km
e	0.00118	0.01573	0.01455
i	63.435°	63.391°	−0.044°
ω	90°	50.126°	−39.87°
Ω	270°	−89.123°	0.877°
θ	0°	40.333°	40.33°
$\omega+\theta$	90°	90.46°	0.46°

Table 7.2 refers to the case of a very large formation (explaining the evident differences).

The second set of critical inclinations (49° and 131°, the remaining parameters for the chief are the same of table 7.2) represents the new result and yet needs to be explained. Figure 7.8 illustrates the quasi-periodicity obtained at this orbital plane angle.

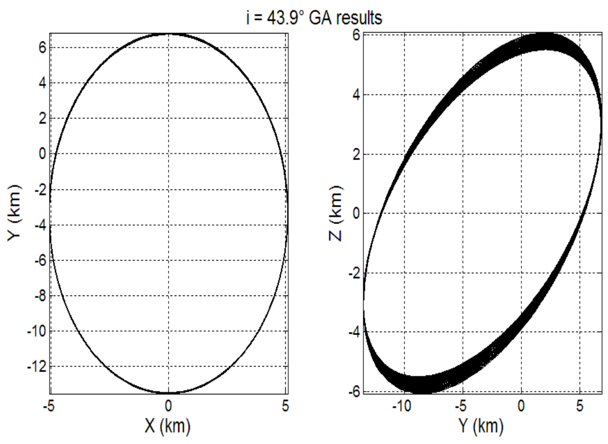

Figure 7.8: 100 relative orbits for a J_2 perturbed case at 49.3° inclination

The corresponding optimal relative state vector and time are:

$$\kappa^* = \begin{bmatrix} -4.176\ km \\ 6.001\ km \\ -9.248\ km \\ 8.848 \cdot 10^{-3} \frac{km}{s} \\ 9.706 \cdot 10^{-3} \frac{km}{s} \\ 2.708 \cdot 10^{-3} \frac{km}{s} \\ 5427.514 - 12.567\ s \end{bmatrix} \tag{7.20}$$

while, the previous example at 35° (again the remaining parameters for the chief are the same of table 7.2), shows how the best individual for a generic inclination presents a higher drift (Figure 7.9).

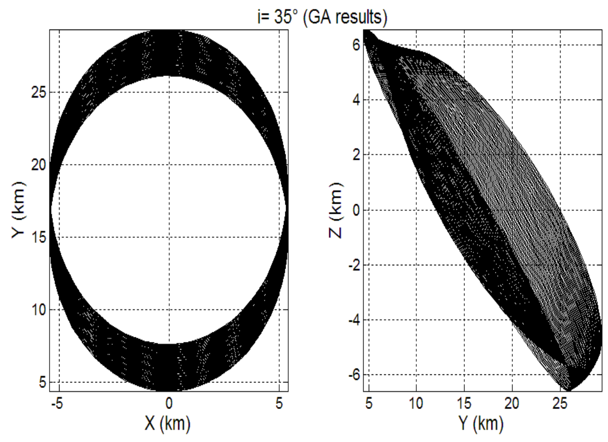

Figure 7.9: 100 relative orbits for a J_2 perturbed case at 35° inclination

being:

$$\kappa^* = \begin{bmatrix} -4.193\ km \\ 8.267\ km \\ 5.252\ km \\ -4.034 \cdot 10^{-3} \frac{km}{s} \\ 9.675 \cdot 10^{-3} \frac{km}{s} \\ -4.6 \cdot 10^{-3} \frac{km}{s} \\ 5427.514 + 100.571\ s \end{bmatrix} \tag{7.21}$$

A more detailed study ([76]) deals with the phenomenon. At this stage it is however clear that these inclinations are not universally valid as the critical inclinations are. In fact their validity is limited to the case of circular and nearly circular orbits, as confirmed by Figure 7.10, while the critical case is valid at all eccentricities. Moreover, inclinations of 49° and 131° are of interest for small and middle sized formations, while they behave quite like all the other inclinations for very large ones (see Figure 7.11).

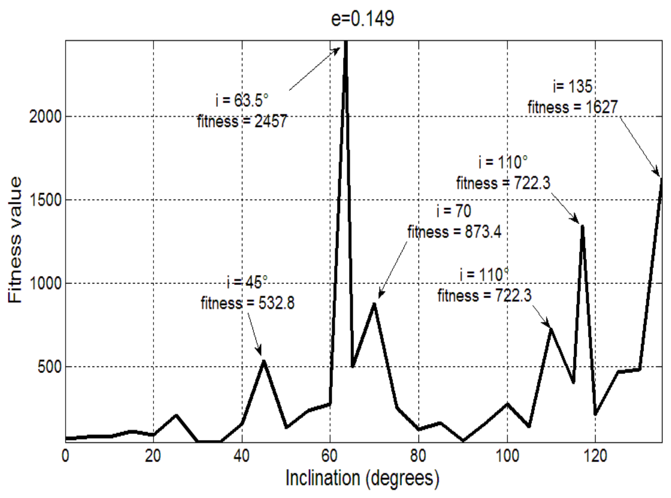

Figure 7.10: Fitness function values for the whole range of inclinations (elliptic reference orbit)

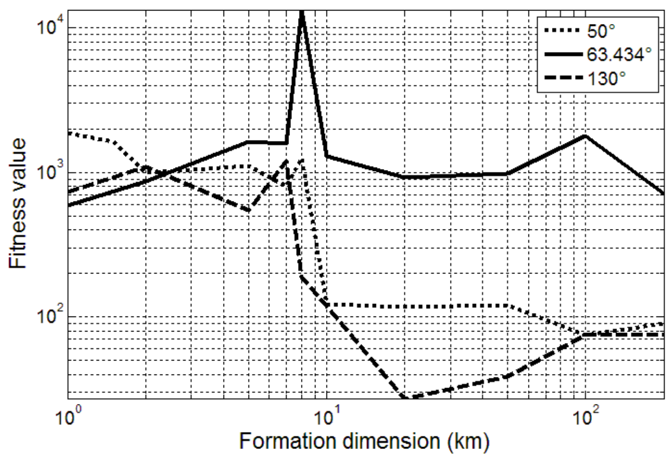

Figure 7.11: Fitness value as a function of the formation dimensions

Notwithstanding the limitations the importance of these result are clear and need more investigation.

7.6 Application of GA on J_2 and Drag Perturbed Case

This section takes also air drag into account. All the performed simulations have shown (Figure 7.12) what was obvious from the beginning: a dissipative perturbation such as air drag cannot allow periodic motion, even if the physical properties of the satellites (mass, area, C_D) are exactly the same, minimizing in that way the differential drag.
A different approach can be implemented in this case. Excluding the possibility of periodic motion, one can set the GA in order to have a close formation after a predetermined number of orbits, not necessarily just one: in this way the satellites behavior between the initial and final instants is of no interest, and the fitness function is evaluated at final time, when the motion is required to repeat itself .
Let us introduce an a-dimensional relative distance as a measure of how bounded the formation is after a chosen time interval t:

$$\frac{distance(t)}{max(distance\ 1^{st}\ orbit)} \tag{7.22}$$

If the relative motion is periodic, the a-dimensional distance oscillates between a minimum value and 1. Performing the optimization with the fitness function evaluated after 1 (case A), 50 (case B), or 100 (case C) orbits, it is possible to obtain the initial conditions which, once propagated for 100 orbits, result in different dynamics. Respectively a formation which is much bounded at the beginning, but then diverges (case A), formations which break apart during the first orbits, but then recompose at the desired time (case B and C: see Figure 7.13).

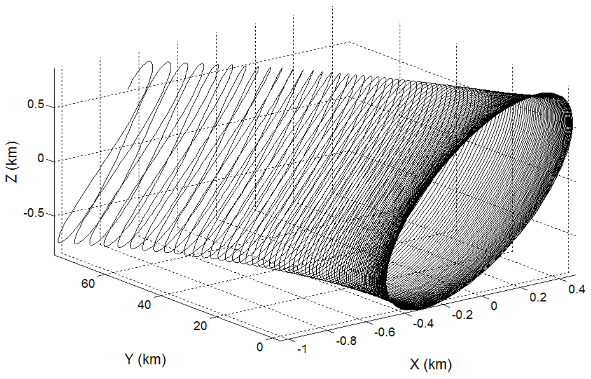

Figure 7.12: Best "periodic" trajectory as found by GA for a J_2 and drag perturbed orbit

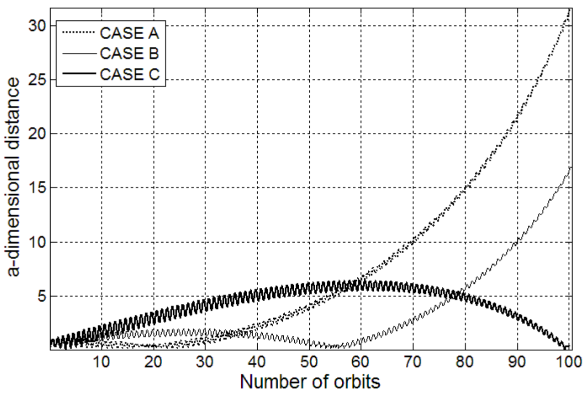

Figure 7.13: Comparison among a-dimensional relative distance for Case A ($N = 1$), Case B ($N = 50$) and Case C ($N = 100$)

As an example, Figure 7.14 shows the behavior in case C: as already mentioned, the formation breaks apart, but then it recomposes around the 100^{th} orbit:

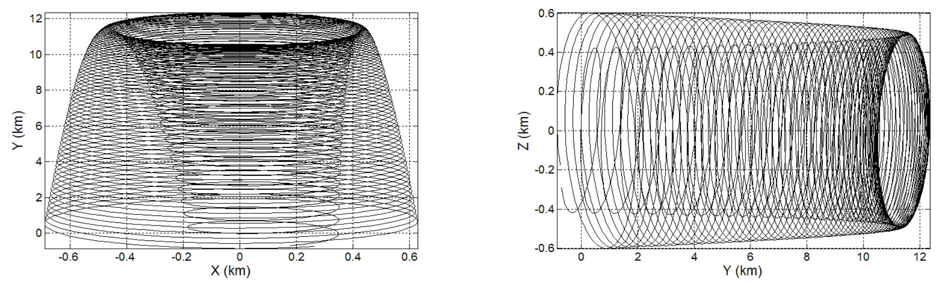

Figure 7.14: Projection on XY and YZ plane of the proposed strategy for the atmospheric drag effect

Possible application of relative orbits such as the one in Figure 7.14 could be close monitoring of a big structure. Imagine to have the STS or the ISS orbiting under J_2 and drag effect (as it is, due to their low operational altitude). A small agent moving back and forth in the vicinity of the origin of LVLH, where the structure would be located, could have a long time (up to hundreds of orbits in the performed simulations) to monitor (video cameras and similar) the external conditions of the “patient”.
More perturbations (moon-sun attraction, solar radiation pressure) have also been analyzed, but their effect is overtop by J_2 and drag effects at low altitude, while, also for high orbits like GEO, the time scale of their action is too long to be taken into account by the presented method.

7.7 Comments and Future Developments

The possibility to obtain natural periodic motion of formation flying satellites has been investigated through the use of a numerical global optimization technique such as Genetic Algorithms. After validating the approach on the well known unperturbed test case, the attention has been focused on perturbations. For a J_2 perturbed reference orbit the genetic optimizer has shown the existence of periodic relative orbits for satellites flying in two particular inclinations: $63.4°$ and $116.6°$ for all eccentricities (already known critical inclinations) and $49°$ and $131°$ for nearly circular orbits. The first couple of inclinations, proper of Molniya orbits, find an explanation through Gauss’ equations. The very interesting and new result represented by the second couple of inclinations need a more deep future investigation. Taking into account also air drag the GA supplies the initial conditions to obtain a close formation after a predetermined time span (with no warranty for the trajectory evolution before or after this imposed time instant).

Chapter 8

Conclusions

This dissertation deals with two aspects of satellites relative flight: fuel optimization of maneuvers for rendezvous and docking and a search for particularly good-natured dynamics for obtaining periodic or bounded relative orbits.
Control is numerically optimized, for the case of rendezvous and docking, by developing three different techniques. In all of the approaches a passive target in circular orbit is imagined at the origin of the Local Vertical Local Horizontal frame. An actively controlled chaser has to reach the target, starting its maneuvers from a generic initial condition, minimizing propellant consumption. Time required for the maneuvers can be combined with fuel consumption into the cost function, in order to have the rendezvous execution in a finite time. Attitude dynamics and control is not considered. Hill-Clohessy-Wiltshire equations describe the relative motion between the two spacecraft. A solution for the theoretical two boundary conditions, control unbounded, unconstrained trajectory problem is known in closed form, thanks to the linearity of the equations.
The current literature lacks in real-time testing of optimization algorithms for spacecraft approaching maneuvers. Likewise a contribution in considering real hardware onboard the spacecraft, in particular real space qualified thrusters, is strongly needed.
Inspired by the experimental test bed described in the appendix E, and by the just mentioned research open points, two of the developed algorithms (chapters 4 and 5) have been implemented paying particular attention to minimize computational burden, i.e. to their possible real time implementation.
In chapter 4 a direct method for real time sub-optimal control for spacecraft rendezvous and docking on pre-determined path is presented. The approach parameterizes the dynamics equations for spacecraft relative flight through the curvilinear abscissa along the trajectory, using cubic B-splines. Via dynamic programming the sub-optimal acceleration sequence on the path is determined. The developed algorithm is implemented under the form of Embedded Matlab® functions, allowing translation into C compilable code. Executable file was tested on a target PC, showing real time capabilities with a sample time of up to 0.2 seconds.
Thanks to the rapidness of the algorithm the designer (or an automatic system), by simply moving the splines control points, can modify the shape of the path while manoeuvring.
Future work on this methodology will be making the control bounded through the analysis of

the generated path before using it into the optimizer. The objective would be to maintain the curvature of the path limited.
The second technique, illustrated in chapter 5, is also a direct method. It is based on high order polynomials for trajectory parameterization, the introduction of a virtual arc, to be used instead of time, thus accelerating the algorithm, and a very few number of optimization parameters.
This method was already used for aircraft trajectory optimization. Among its advantages it enumerates the satisfaction of the boundary conditions (always for position, numerically for velocity), no "wild" trajectories arise during optimization, an analytical (parametrical) representation of the reference trajectory is possible, and, fundamental for real time future implementation, it requires only a few iterations (< 100) to generate a solution. A first order algorithm is used for solving the otained NLP problem, guarantying always a solution, even if a local minima, thus making the approach highly reliable. It also needs a low relative CPU time for convergence, making possible to employ it on board of a spacecraft for real-time prototyping of rendezvous and docking maneuvers.
Principal drawback of all direct methods is that they generate sub-optimal trajectories. Main current problem within this particular kind of direct optimization is the use of the *fminsearch* routine for solving the NLP problem. The limitation of this routine of restituting local minima implies some disadvantages like strong dependency on the initial guess solution, not precise matching of the final desired velocity and high values for the time intervals separating available data for the controls. A multi-criterion multi-variable optimization routine based on Hooke-Jeeves pattern search algorithm is then under development for overcoming the issue related to the *fminsearch*. Not yet validated for the direct method, the routine showed good performances for the hybrid approach implemented in chapter 6.
It is worth mentioning that non linear dynamics can be used with both the above cited techniques. Virtually any mathematical, more accurate, model of relative motion can be used, if needed.
Analyzing the results obtained with the previous described algorithms, and also looking at the most recent literature on optimal rendezvous and docking, the attention falls on the real possibility of exerting a time varying thrust profile. Both the direct methods here developed and the available research in literature, present control profiles which are very difficult to think as actually generated by space qualified rocket engines.
The investigation reported in appendix F shows the current technological difficulties of realizing space engines with a really continuous thrust modulation. Current engines should be still considered on-off capable.
In order to take into account these limitations for optimal rendezvous and docking, a hybrid technique is introduced in chapter 6. A set of low thrust engines is imagined onboard the chaser for far away maneuvers, high thrust impulsive thrusters for the very last phase of flight to the target. This way to face the problem sees its application when a set of different sized thrusters is thought to be mounted on the chaser agent. Electrical engines can be used for the far away segment, chemical ones for the more accurate final stage.
The far away control profile is determined setting up a minimization problem, using the first order Hooke-Jeeves algorithm to find the adjoint velocity initial conditions and modifying the final time for the low thrust segment. In order to consider finite values of admissible thrust the

optimal unbounded control profile is discretized.
The second phase is tackled by considering the Lawden's conditions, via a conjugate gradient technique. In this way a very reliable procedure is generated to determine the switching structure of the low thrust segment and the number, magnitudes, directions, times and space collocation of the firings related to the second span of trajectory.
The trade-off between the continuous adjustable low-thrust approach and the one here developed is very convenient: the errors arising from discretizing the acceleration profile can be corrected via impulsive manoeuvres, without significantly increasing the resulting fuel consumption. The approach has been verified on some data taken from the ATV vehicle showing satisfying performances.
The hybrid method looks like a good candidate for real time implementation: the convergence to the final solution, using simple Matlab® scripts, requires a total time less than 50 s.
Then major future developments will be modifications on the Hooke-Jeeves and on the conjugate gradient routines, in order to be able of constraining the optimization variables.
A very interesting work will be focused on optimizing the whole cost (fuel for low-thrust segment plus fuel for impulsive manoeuvre) having the junction point as optimization parameter, i.e. the state vector used as subdivision between the low- and high-thrust spans. In this way the optimal combination of the different families of engines could be found.
The last chapter of this dissertation presents the use of a genetic algorithm optimizer for searching particular conditions under which the relative motion between two spacecraft in close orbits can be periodic or at least bounded. Measure of the periodicity is the introduced fitness function, which basically indicates how close the relative orbit is to repeat itself after a certain amount of time.
Using the initial relative state vector and the fitness function evaluation time as genetic variables, the numerical genetic optimizer restitutes new interesting results.
With a nonlinear J_2 perturbed model for the two spacecraft orbit propagation, the genetic optimizer shows the existence of periodic relative orbits for satellites flying in two particular inclinations: $63.4°$ and $116.6°$ for all eccentricities (already known critical inclinations) and $49°$ and $131°$ for nearly circular orbits. The first couple of inclinations, proper of Molniya orbits, find an explanation through Gauss' equations. The very interesting and new result represented by the second couple of inclinations need a more deep future investigation. Taking into account also air drag the genetic algorithm supplies the initial conditions to obtain a close formation after a predetermined time span.
Application of such trajectories can be found in close monitoring of orbiting structures in low orbits, where the J_2 and drag effects are predominant.

Chapter 9

Appendices

9.1 Appendix A: State Transition and Convolution Matrices for HCW

This appendix reports the normalized transition matrices for state and costate associated to the HCW equations in the two forms used in this dissertation (see 3.2). It also shows the convolution matrix obtained with optimal unbounded control. The matrices are calculated on the normalized form of HCW, leaving a lighter notation without the LVLH angular rate in it. The only variable change to keep in mind is from the time t to the anomaly on circular orbit θ: $\theta = \omega t$. In this way the time derivatives are calculated with respect to θ (the subscript θ stands for it) giving a normalized version, for example, of eq. 3.8:

$$\begin{cases} x_{\theta\theta} - 2y_{\theta} - 3x = \frac{u_x}{\omega^2} \\ y_{\theta\theta} + 2x_{\theta} = \frac{u_y}{\omega^2} \\ z_{\theta\theta} + z = \frac{u_z}{\omega^2} \end{cases} \tag{9.1}$$

By doing this the position and velocity will be both expressed in *km* or *m*, having eliminated time as independent variable. The state transition matrix called Φ in chapter 3, eq. 3.11 is calculated as the matrix exponential $e^{\mathbf{A}\theta}$. Via the Matlab® Symbolic Toolbox we get:

$$\Phi(\theta) = \begin{pmatrix} 4-3\cos(\theta) & 0 & 0 & \sin(\theta) & 2-2\cos(\theta) & 0 \\ 6\sin(\theta)-6\theta & 1 & 0 & 2\cos(\theta)-2 & -3\theta+4\sin(\theta) & 0 \\ 0 & 0 & \cos(\theta) & 0 & 0 & \sin(\theta) \\ 3\sin(\theta) & 0 & 0 & \cos(\theta) & 2\sin(\theta) & 0 \\ 6\cos(\theta)-6 & 0 & 0 & -2\sin(\theta) & -3+4\cos(\theta) & 0 \\ 0 & 0 & -\sin(\theta) & 0 & 0 & \cos(\theta) \end{pmatrix} \tag{9.2}$$

The adjoint vector transition matrix of eq. 3.22 is:

$$\Phi_\lambda(\theta) = \begin{pmatrix} 4-3\cos(\theta) & -6\sin(\theta)+6\theta & 0 & -3\sin(\theta) & 6\cos(\theta)-6 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \cos(\theta) & 0 & 0 & \sin(\theta) \\ -\sin(\theta) & 2\cos(\theta)-2 & 0 & \cos(\theta) & 2\sin(\theta) & 0 \\ 2-2\cos(\theta) & 3\theta-4\sin(\theta) & 0 & -2\sin(\theta) & -3+4\cos(\theta) & 0 \\ 0 & 0 & -\sin(\theta) & 0 & 0 & \cos(\theta) \end{pmatrix} \tag{9.3}$$

The convolution matrix of eq. 3.24 is:

$$\Psi(\theta) = \begin{pmatrix} \Psi_{11}(\theta) & \Psi_{12}(\theta) \\ \Psi_{21}(\theta) & \Psi_{22}(\theta) \end{pmatrix} \tag{9.4}$$

where:

$$\Psi_{11}(\theta) =$$

$$= \begin{pmatrix} -2.5\cos(\theta)\theta+6.5\sin(\theta)-4\theta & 16(1-\cos(\theta))-5\sin(\theta)\theta-3\theta^2 & 0 \\ 16(\cos(\theta)-1)+5\sin(\theta)\theta+3\theta^2 & -10\cos(\theta)\theta+38\sin(\theta)-28\theta+1.5\theta^3 & 0 \\ 0 & 0 & -0.5(\cos(\theta)\theta+\sin(\theta)) \end{pmatrix}$$

$$\Psi_{12}(\theta) = \begin{pmatrix} -4\cos(\theta)-2.5\sin(\theta)\theta+4 & 5\cos(\theta)\theta-11\sin(\theta)+6\theta & 0 \\ -5\cos(\theta)\theta+11\sin(\theta)-6\theta & -28\cos(\theta)+28-10\sin(\theta)\theta-4.5\theta^2 & 0 \\ 0 & 0 & -0.5\sin(\theta)\theta \end{pmatrix}$$

$$\Psi_{21}(\theta) = \begin{pmatrix} 4\cos(\theta)-4+2.5\sin(\theta)\theta & -5\cos(\theta)\theta+11\sin(\theta)-6\theta & 0 \\ 5\cos(\theta)\theta-11\sin(\theta)+6\theta & -28+28\cos(\theta)+10\sin(\theta)\theta+4.5\theta^2 & 0 \\ 0 & 0 & 0.5\sin(\theta)\theta \end{pmatrix}$$

$$\Psi_{22}(\theta) = \begin{pmatrix} -2.5\cos(\theta)\theta+1.5\sin(\theta) & -6\cos(\theta)+6-5\sin(\theta)\theta & 0 \\ 6\cos(\theta)+5\sin(\theta)\theta-6 & -10\cos(\theta)\theta+18\sin(\theta)-9\theta & 0 \\ 0 & 0 & -0.5\cos(\theta)\theta-0.5\sin(\theta) \end{pmatrix}$$

For the modified set of HCW equations we easily obtain:

$$\Phi(\theta) = \begin{pmatrix} 1 & 6(\theta-\sin\theta) & 0 & 4\sin\theta-3\theta & 2(1-\cos\theta) & 0 \\ 0 & 4-3\cos\theta & 0 & -2(1-\cos\theta) & \sin\theta & 0 \\ 0 & 0 & \cos\theta & 0 & 0 & \sin\theta \\ 0 & 6(1-\cos\theta) & 0 & 4\cos\theta-3 & 2\sin\theta & 0 \\ 0 & 3\sin\theta & 0 & -2\sin\theta & \cos\theta & 0 \\ 0 & 0 & -\sin\theta & 0 & 0 & \cos\theta \end{pmatrix} \tag{9.5}$$

and:

$$\Phi_\lambda(t) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 6\sin(\theta)-6\theta & 4-3\cos(\theta) & 0 & -6\cos(\theta)+6 & -3\sin(\theta) & 0 \\ 0 & 0 & \cos(\theta) & 0 & 0 & \sin(\theta) \\ 3\theta-4\sin(\theta) & -2+2\cos(\theta) & 0 & -3+4\cos(\theta) & 2\sin(\theta) & 0 \\ -2\cos(\theta)+2 & -\sin(\theta) & 0 & -2\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & -\sin(\theta) & 0 & 0 & \cos(\theta) \end{pmatrix} \tag{9.6}$$

finally:

$$\Psi(\theta) = \begin{pmatrix} \Psi_{11}(\theta) & \Psi_{12}(\theta) \\ \Psi_{21}(\theta) & \Psi_{22}(\theta) \end{pmatrix} \tag{9.7}$$

where:

$$\Psi_{11}(\theta) = \begin{pmatrix} 38\sin\theta - 10\theta\cos\theta + 1.5\theta^3 - 28\theta & 16\,(1-\cos\theta) - 5\theta\sin\theta - 3\theta^2 & 0 \\ 16\,(\cos\theta - 1) + 5\theta\sin\theta + 3\theta^2 & 6.5\sin\theta - 2.5\theta\cos\theta - 4\theta & 0 \\ 0 & 0 & 0.5\,(\sin\theta - \theta\cos\theta) \end{pmatrix}$$

$$\Psi_{12}(\theta) = \begin{pmatrix} 28\,(1-\cos\theta) - 10\theta\sin\theta - 4.5\theta^2 & 5\theta\cos\theta - 11\sin\theta + 6\theta & 0 \\ 11\sin\theta - 5\theta\cos\theta - 6\theta & 4\,(1-\cos\theta) - 2.5\theta\sin\theta & 0 \\ 0 & 0 & -0.5\theta\sin\theta \end{pmatrix}$$

$$\Psi_{21}(\theta) = \begin{pmatrix} 10\theta\sin\theta + 28\,(\cos\theta - 1) + 4.5\theta^2 & 11\sin\theta - 5\theta\cos\theta - 6\theta & 0 \\ -11\sin\theta + 5\theta\cos\theta + 6\theta & 4\,(\cos\theta - 1) + 2.5\theta\sin\theta & 0 \\ 0 & 0 & 0.5\theta\sin\theta \end{pmatrix}$$

$$\Psi_{22}(\theta) = \begin{pmatrix} 18\sin\theta - 10\theta\cos\theta - 9\theta & 6 - 5\theta\sin\theta - 6\cos\theta & 0 \\ 6\,(\cos\theta - 1) + 5\theta\sin\theta & 1.5\sin\theta - 2.5\theta\cos\theta & 0 \\ 0 & 0 & -0.5\,(\theta\cos\theta + \sin\theta) \end{pmatrix}$$

9.2 Appendix B: Cubic B-Splines

This appendix briefly shows how two additional control points ($\vec{r}_0'$ and $\vec{r}_{end}'$) for the cubic B-splines used in section 4.3 allow the curve to pass for the initial and final desired positions ($\vec{r}_0$ and $\vec{r}_{end}$).
The additional points are inserted one at the beginning of the curve, one at the end, to satisfy:

$$\vec{r}_0 = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \frac{1}{6} \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{pmatrix} \begin{bmatrix} \vec{r}_0' \\ \vec{r}_0 \\ \vec{r}_1 \\ \vec{r}_2 \end{bmatrix}$$

and

$$\vec{r}_{end} = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} \frac{1}{6} \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{pmatrix} \begin{bmatrix} \vec{r}_{end-2} \\ \vec{r}_{end-1} \\ \vec{r}_{end} \\ \vec{r}_{end}' \end{bmatrix}$$

From these relations we can deduce the additional points:

$$\vec{r}_0' = 2\vec{r}_0 - \vec{r}_1$$

and

$$\vec{r}_{end}' = 2\vec{r}_{end} - \vec{r}_{end-1}$$

9.3 Appendix C: Hooke-Jeeves Algorithm

The algorithm here presented aims finding the best point $X^* = [x_1^*,\ x_2^*,\ ...,\ x_n^*]$ minimizing a scalar function $f = f\left([x_1^*,\ x_2^*,\ ...,\ x_n^*]\right)$. Being a numerical routine it obviously approximates the solution. The direct search on which the algorithm is based implies the existence of a space Σ of possible solutions X. An extremal point is such that $f(X^*) < f(X),\ \forall X \in\ \Sigma$.
An initial base point X_0 and an initial step k_0 are needed to start running the routine. The procedure is based on the concept of exploratory (local) and pattern (global) moves. The exploratory moves can be simply represented by a change equal to the step size for each coordinate x_i independently. The pattern moves change the whole vector X at once. A brief description of the Hooke-Jeeves algorithm follows (see [69] for more details):

1. Start evaluating $f(X_0)$;

2. From the current base point make exploratory moves with step size k_0, if the function decreases with respect to the one at current base point go to 3, otherwise go to 4;

3. Set a new base point, make a pattern move, make exploratory moves. If the function decreases with respect to the one at current base point go to 3, otherwise go to 2;

4. If the step size is lower than a specified tolerance STOP, otherwise decrease step size and go to 2.

If the function decreases with a certain exploratory move the new value for that coordinate is kept, otherwise the old one is retained.
Starting from the initial base point X_0 the exploratory moves give the direction for the first pattern move. The first pattern move is performed in the direction given by the exploratory moves from a base point which is equal to the old one but modified by the exploratory moves. Then "the presumption that whatever constituted a successful set of moves in the past is likely again to prove successful" ([69]) makes the algorithm perform pattern moves in the same previous direction, unless the move is a failure (function increases instead of reducing).
Any pattern move is followed by exploratory moves to locally improve the solution (and obviously it updates the base point). The amount of the move duplicates the combined moves from the previous base point, i.e. all coordinates are changed by an amount equal to the difference between the present base point and the previous base point. A set of exploratory moves is performed also when a pattern move is a failure, starting from the old base point, in order to determine a new better pattern direction.

9.4 Appendix D: Developed Software

A part from the FORTRAN routines (PIKAIA free-ware code, orbital propagator, etc.) used for the genetic search in chapter 7, the whole set of simulations presented in this dissertation has been run using Matlab® code, Embedded Matlab® code, and Simulink® models.
In chapter 4 the routines calculating the cubic B-splines have been taken from a previous work ([59]) and modified for the specific purpose. The dynamic programming tree generation and exploration has been implemented following the guidelines of [61]. After the software validation via numerical simulation, real-time tests have been run. Before downloading the executable file on the target machine there was the need of converting the whole software into C compilable code. The Embedded Matlab® functions help in automatic generation of C code. Embedded Matlab® functions are claimed to be able of converting Matlab® routines directly into C programs. The reality is that a very limited subset of the Matlab® functions are recognized and converted by the tool. Furthermore no dynamic memory is allowed, that is obvious when real-time programs needs to be run.
All these limitations meant an annoying work in adapting the software to make it much closer to a C structure. Some functions, such as the *fzero*, were re-written from scratch to make them recognizable by the tool.
Once the set of Embedded Matlab® functions is ready they appear as Simulink® blocks (see Figure 4.10). At this stage the model can be compiled by XPCtarget and downloaded as executable file on the target machine.
In chapter 5 a Matlab® script calls, at each iteration, a Simulink® model, which calculates states, slaves and performances.
The software in chapter 6 is completely written as Matlab® code. It is basically ready to be translated into Embedded Matlab®, the only routine which needs to be re-written is the *ODE45* integration one. This will be a minor work for the future implementation as compilable code.

9.5 Appendix E: Autonomous Docking Test Bed: Brief Description

The autonomous docking test bed (see [5] for a detailed description and experiments) set up at the Spacecraft Robotics Laboratory of the Naval Postgraduate School is composed of three main systems (see Figure 9.1 and Figure 9.2):

1. A floating epoxy surface 4.9 *m* by 4.3 *m* installed by Rock Art LTD;
2. a chaser spacecraft simulator;
3. a target spacecraft simulator.

Furthermore a separate desktop computer is used to upload the software on the chaser, initiate the experimental tests and receive data.
Air pads on the spacecraft simulators reduce the friction to negligible levels.
The chaser spacecraft simulator is composed of five modular decks mounted one on top of the other (see table 9.1). Modularity eases maintenance and modifications. The lowest module houses the floatation and thrust subsystems: a carbon fiber tank that feeds compressed air through two independent pressure regulators to eight thrusters and four air pads. A surge chamber is inserted along the thrust air supply line between the regulator and the solenoid valves in order to limit pressure oscillations during the thrusters' operation to about $\pm 2\%$ of the nominal value. A single thruster consists of a convergent nozzle and is activated by a normally closed on-off solenoid valve. The second deck of the chaser hosts the active portion of a docking interface mechanism. This system, designed for soft-docking applications by Starsys Research Corporation, is the prototype capture system for DARPA's Orbital Express Mission. This part consists of a motor driven lead screw that actuates three individual four-bar linkages. At docking the linkages engage the passive portion of the capture system that is mounted on the target and, by retracting, seat the passive portion into a three point kinematic mount, supported by preloaded springs, thus establishing a rigid interface. The third and fourth decks house a reaction wheel and the electric and electronic subsystems respectively. The reaction wheel, made by Ball Aerospace, controls the attitude of the vehicle. Two lithium ion batteries providing 28 VDC bus tension feed the subsystems. The tension is transformed, when needed, by an array of four DC-DC converters. Two PC104 computer form the central part of the electronics subsystem. One computer works on the vision sensor, the other runs the real time state estimation and control code, receiving data from the sensors and commanding the actuators. A Micro-Electro-Mechanical-System (MEMS) based Inertial Measurement Unit (IMU) is in the center of the fourth module. The fifth module hosts a monochrome 1.3 million pixel Complementary Metal Oxide Semiconductor (CMOS) camera and a wireless Ethernet router. Through the router the two on-board computers can communicate with each other and with the separate desktop computer. An infrared filter allows the camera to only recognize the LEDs in its field of vision.
The Target Spacecraft Simulator hosts the passive portion of the docking interface mechanism. Three infrared LEDs are used as reference for the vision sensor and are mounted on its top deck. The target is not currently actively maneuvered and pre-existed the more recent chaser. More

Table 9.1: Main Characteristics of the Chaser Spacecraft Simulator

Size	Length and Width	0.4 [m]
	Height	0.85 [m]
	Mass	63 [Kg]
	Moment of Inertia about Y_{ch}	2.3 [Kg m^2]
Propulsion	Propellant	Air
	Equivalent Storage Capacity	0.72 [m^3] @ 0.35 [Mpa]
	Operating Pressure, Thrust	0.35 [Mpa] (50 [PSI])
	Operating Pressure, Floating	0.24 [Mpa] (35 [PSI])
	Continuous Operation	20 – 40 [min]
	Thrust of each thrusters	0.45 [N]
	Reaction Wheel Max Torque	0.16 [Nm]
	Reaction Wheel Max Angular Momentum	20.3 [Nms]
Subsystems	Battery Type	Lithium-Ion
	Storage Capacity	12 [Ah] @ 28[V]
	Continuous Operation	$\sim$ 6 [h]
	Computers	2 PC104 Pentium III
Sensors	IMU	Crossbow 400CC
	Vision Sensor	custom developed
	CMOS Camera	Pixelink PL-A471
	Camera Field of View	40 [deg]
	Vision Sensor Range	0-10 [m]
Docking I/F	Max Axial Misalignment	+/- 7.62 [cm]
	Max Lateral Misalignment	+/- 5.08 [cm]
	Max Angular Misalignment (Pitch, Yaw, Roll)	+/- 5 [deg]

details on the target can be found in [79].
As just mentioned, vision is used to measure the relative state vector of the target with respect to the chaser. The relative position vector $\overrightarrow{r}$, and the relative attitude angle θ, compose the relative state vector.
Each spacecraft simulator has its own body reference frame, centered on its center of mass.
The three infrared LEDs mounted on the target are distributed as follows: two LEDs are positioned along the one body axis, the third one is offset and positioned along the perpendicular axis.
The transmitted data consists of the relative position and angle measurements plus a check signal that is nominally 0 and becomes 1 when a new measurement is available.

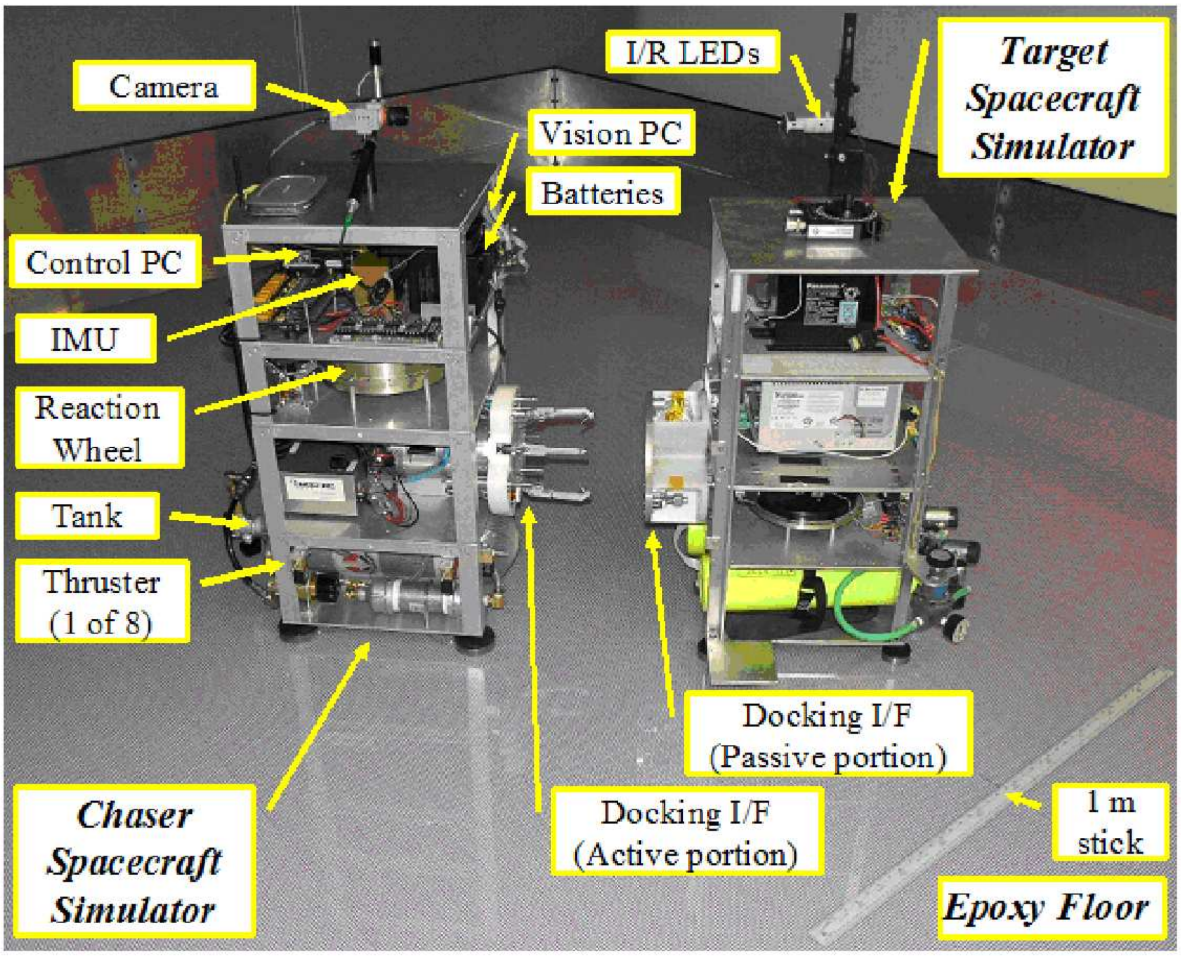
Camera
I/R LEDs
Target Spacecraft Simulator
Vision PC
Batteries
Control PC
IMU
Reaction Wheel
Tank
Thruster (1 of 8)
Docking I/F (Passive portion)
1 m stick
Chaser Spacecraft Simulator
Docking I/F (Active portion)
Epoxy Floor

Figure 9.1: Autonomous Docking Test Bed at the Spacecraft Robotics Laboratory, Naval Postgraduate School

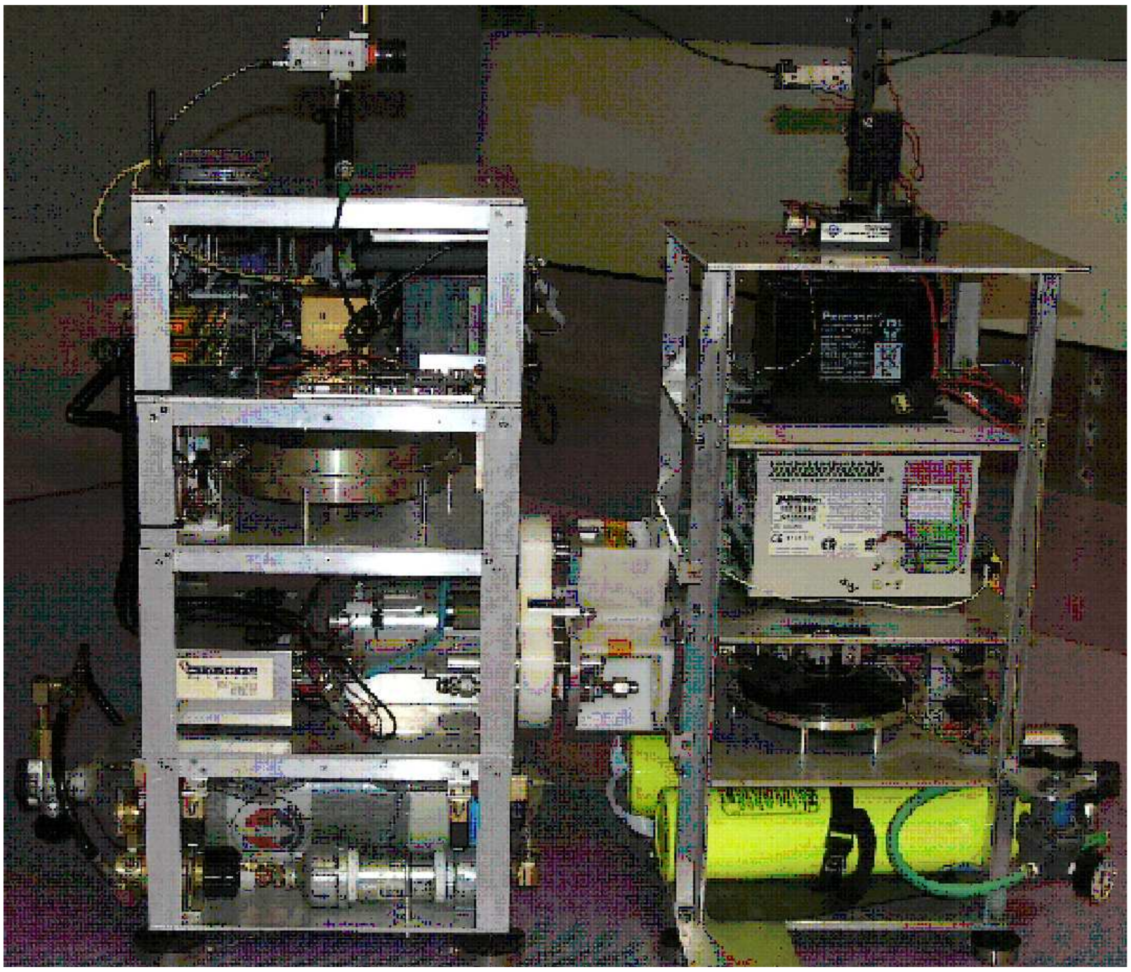

Figure 9.2: Chaser and Target in Docked Configuration

Finally, Figure 9.3 reports the block diagram of the test bed, showing the links and data exchange among the systems.

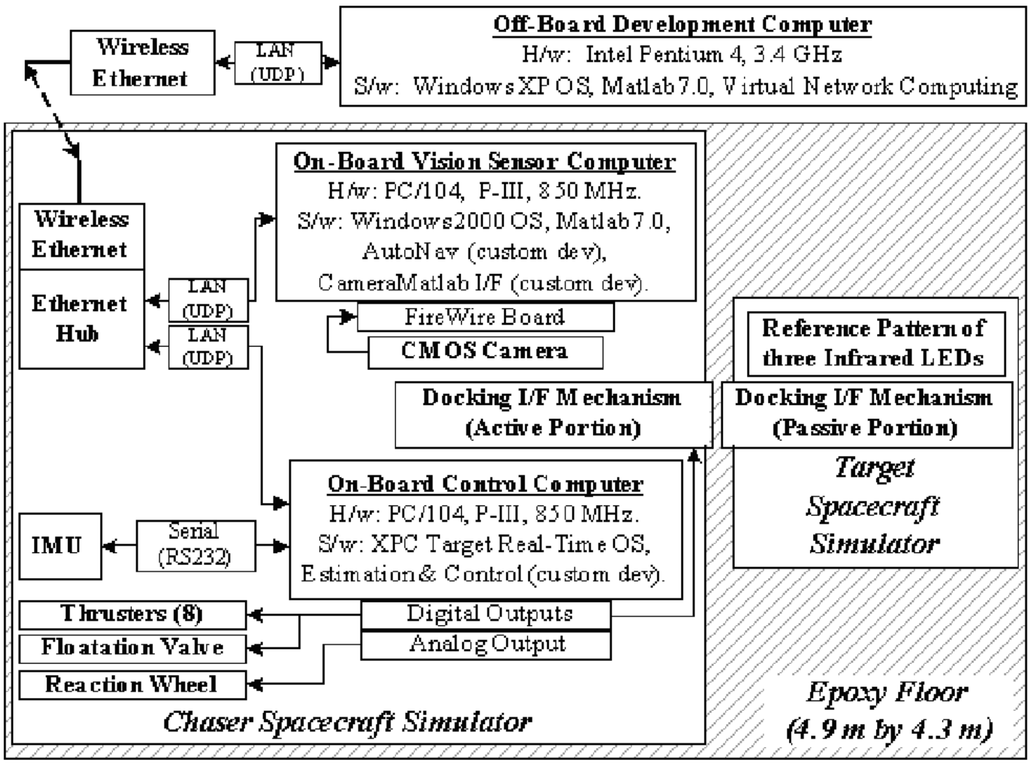
Wireless Ethernet
LAN (UDP)
Off-Board Development Computer
H/w: Intel Pentium 4, 3.4 GHz
S/w: Windows XP OS, Matlab 7.0, Virtual Network Computing
On-Board Vision Sensor Computer
H/w: PC/104, P-III, 850 MHz.
S/w: Windows2000 OS, Matlab 7.0, AutoNav (custom dev), CameraMatlab I/F (custom dev).
Wireless Ethernet
Ethernet Hub
LAN (UDP)
LAN (UDP)
FireWire Board
CMOS Camera
Reference Pattern of three Infrared LEDs
Docking I/F Mechanism (Active Portion)
Docking I/F Mechanism (Passive Portion)
Target Spacecraft Simulator
On-Board Control Computer
H/w: PC/104, P-III, 850 MHz.
S/w: XPC Target Real-Time OS, Estimation & Control (custom dev).
IMU
Serial (RS232)
Thrusters (8)
Floatation Valve
Reaction Wheel
Digital Outputs
Analog Output
Chaser Spacecraft Simulator
Epoxy Floor (4.9 m by 4.3 m)

Figure 9.3: Block Diagram of the Autonomous Docking Test Bed

The test bed allows for 2-D limited simulations, and obviously cannot reproduce the real HCW dynamics showed by two satellites flying in very close orbit. Anyway, it still permits to test control algorithms with hardware-in-the-loop simulations, with a very reduced cost.
Furthermore, for distances such as a few meters, the HCW accelerations can be considered a disturbance with respect to the simple double integrator dynamics which can be reproduced in this test bed.

9.6 Appendix F: Survey of Current Technology Space Thrusters

Two different ways of facing the problem are investigated, depending on the assumptions made about the thrusters on board the chaser satellite. Following [41] we can differentiate the thrust systems in high- and low-thrust engines.
High-thrust systems (tenths of N and more) usually are on for a very short time with respect to the mission duration, from which their definition as impulsive systems. They are usually chemical engines with a medium specific impulse and relatively high consumption rate.
The low-thrust thrusters (mN to some N), on the other hand, are characterized by low fuel consumption (high specific impulse) which allows them to be on for very long time intervals.
The specific impulse I_{sp} measures the capability of a rocket to accelerate the particles which constitute the propellant. Its definition is in fact related to the gas exhaust velocity c: $I_{sp} = \frac{c}{g_0}$, being g_0 the gravity acceleration at sea level.
Being the rocket thrust proportional to the gas exhaust velocity and the mass flow rate, it is straightforward to define the specific consumption as the inverse of the I_{sp}. Given a thrust level, an high I_{sp} engine needs less mass flow rate to generate the required push, i.e. less fuel consumption.
For both the systems above introduced a further categorization can be made based on the possibility to vary the gases exhaust velocity (CSI = Constant Specific Impulse, VSI = Variable Specific Impulse).
A very delicate issue is that of modulating the thrust magnitude generated by space rockets. In theory in CSI systems the thrust level can be regulated by varying the mass flow rate. They can be either high- or low-thrust devices. For VSI engines, mainly low thrust, the thrust can be controlled by varying the exhaust velocity. Low-thrust CSI engines are also called thrust-limited, as the mass flow rate is limited, limiting the thrust as well. VSI systems are limited by the power available from a separate energy source required to run the engine and are called power-limited.
Section 3.3.1 deals with the optimal control problem for RDD when continuous controls are available, i.e. medium-low-thrust devices. In general these kind of engines are electrical engines, which gives them the capability of long duration operativity.
In order to have a more precise idea of the state of the art for space qualified or in-qualification engines, some data and information have been taken from the EADS ([80]), the Centrospazio ([81]) and the Loral ([82]) web pages. The EADS is a European thruster producer providing engines for a number of important vehicles, among them the ESA ATV (see Figure 9.4). Centrospazio cooperates with EADS and ESA since many years for the development and test of electric thrusters. Loral is an American space systems company. Also some information have been founded through the web encyclopedia Wikipedia. Searching for technical data regarding the available space qualified engines, especially looking at low thrust, long lasting motors, one realizes that not much information is given regarding the actual possibilities of modulating the thrust magnitude.
Focusing the interest on low-thrust systems, assuming the high thrust ones operating at constant impulsive thrust, let us report a brief collection of some representative EADS engines:

- ION: ion propulsion is based on the electrostatic acceleration of electrically charged gas atoms (ions). The most interesting engine is the space qualified RITA-10. It flew on the

European Retrievable Carrier (EURECA) in 1992 and on the Advanced Relay Technology Mission (ARTEMIS) in 2002. The most interesting feature is the thrust variation

Table 9.2: RITA-10 technical data

propellant	Xenon
ionisation principle:	radio frequency-discharge ($v = 0.7 - 1MHz$)
Thrust level (nominal and demonstrated)	15 mN, $0.3 - 41mN$
Specific Impulse (nominal and demonstrated)	3300 $sec.$, $2500 - 3700$ $sec.$
Design Life	$>$ 20000 hrs
Mass	1.8 kg

from 15% to 135% of its nominal thrust. The problem is: how the thrust can be adjusted? How long does it take to pass from a stable level to a new one? Is it a real modulation or just different states not easy to interchange?
No data is available, for example, on the bandwidth for the thrust adjustability, making us to think that no possibility of real modulation yet exists. In other words, given a thrust profile function of time it is really hard to follow it without great deviations due to rising and decaying delays, instabilities, inaccuracy at intermediate thrust levels, etc..
Of the same class the two following engines:

Table 9.3: RIT-22 technical data

propellant	Xenon
ionisation principle:	radio frequency excitation
Thrust level (nominal and demonstrated)	$120 - 200$ mN, $80 - 250mN$
Specific Impulse (nominal and demonstrated)	$3000 - 5500$ $sec.$, $2500 - 6400$ $sec.$
Design Life	$>$ 10000 hrs
Mass	7 kg

Table 9.4: RIT-XT technical data

propellant	Xenon
ionisation principle:	radio frequency-discharge ($v = 0.7 - 1MHz$)
Thrust level (nominal and demonstrated)	$50 - 150\ mN,\ 210mN$
Specific Impulse (nominal and demonstrated)	$4200 - 4500\ sec.,\ 2500 - 5500\ sec.$
Design Life	$> 15000\ hrs$
Mass	$7\ kg$

- <u>HYDRAZINE</u>: Thrust is produced by the decomposition of hydrazine as it passes through a catalyst bed. Thrusters are designed for operation in both steady state and pulse mode operation. These thruster allow for almost no variability of the thrust. Again, in the case of variability it is not a real modulation-ability.
  Hydrazine can run both high- and low-thrust systems, it does not give high specific impulses as electric propulsion, but it is here mentioned as also belonging to the family of low-thrust engines.
  The following table summarizes the characteristics of some of the low-thrust EADS hydrazine motors:

Table 9.5: EADS Hydrazine Low-Thrust Engines

Model / Data	CHT 0.5	CHT 1	CHT 2	S10-01
Thrust (N)	0.5	0.32-1.1	0.6-2	10
Specific Impulse (s)	227.3	200-223	210-227	286
Required Power (kW)	0.55	Not declared	Not declared	14
Mass (kg)	0.195	0.29	0.2	0.35
Burn Time (hrs)	Not declared	46	22.5	Not declared

Centrospazio is currently involved in the research, development and testing of basically any class of electric engines. The FEEP (Field Emission Electric Propulsion) ones are probably the most interesting for this review as claimed to be capable of delivering very low thrust with very high accuracy and controllability. In FEEP emitters, unlike most ion engines, ions are directly extracted from the liquid phase. The thruster can accelerate a large number of different liquid metals, cesium is usually selected. A voltage difference accelerates the particles and a ion neutralizer is needed to keep neutrality of the whole system. In [83] the FEEP is recognized as the only existing thruster capable of accurate thrust modulation in the $1\ \mu N - 1\ mN$ thrust level, making it a very good candidate for drag-free scientific missions, small satellites attitude control, orbit maintenance and fine pointing of scientific spacecraft ([84]). In [85] Centrospazio

performed an interesting simulation of attitude and orbit control with FEEP thrusters. The modulation is not actually used, not even for attitude control. Drag and J_2 compensation and attitude control are obtained by firing at constant thrust with a certain frequency, basically bang-bang functions of time. The authors kept this conservative approach, not exploiting the modulation ability of the FEEPs. The same paper talks about high bandwidth for thrust variation but it does not report the typical values.
Then modulation ability can be achieved but only for very low thrust in a limited range (no more than 1 mN) and not much data is yet available.
For the sake of completeness other representative types of electric engines are following described. They do not fulfill any modulation request.

- Hall Effect Thrusters: a Hall effect thruster (HET) is a type of ion thruster in which the propellant is accelerated by an electric field in a plasma discharge with a radial magnetic field. Also known simply as plasma thrusters, HETs use the Hall effect to trap electrons and then use the electrons to ionize propellant, efficiently accelerate the ions to produce thrust, and neutralize the ions in the plume. Typical specific impulse ranges from 1200 to 1800 s. The amount of thrust is very small, on the order of 80 mN for a typical thruster. Current research is focused on:

  1. Scaling the typically 1 kW Hall thruster to higher powers (50 to 100kW) and lower powers (50 to 100 W).
  2. Resolving spacecraft integration issues regarding the large plume divergence.
  3. Enabling operation at higher specific impulse and variable specific impulse.
  4. Flight validating thrusters for use on western spacecraft.

  This technology was used on the European lunar mission SMART-1 and is used on a number of commercial geostationary satellites.

- Magnetoplasmadynamic thrusters: a (MPD) thruster (MPDT) is a form of electric propulsion (a subdivision of spacecraft propulsion) which uses the Lorentz force (a force resulting from the interaction between a magnetic field and an electric current) to generate thrust. In theory, MPD thrusters could produce extremely high specific impulses up to and beyond 11000 s, triple the value of current xenon-based ion thrusters, and about 20 times better than liquid rockets. Even more impressive is that MPD technology is capable of thrust levels of up to 200 N, the highest for any form of electric propulsion, and nearly as high as many interplanetary chemical rockets.
  MPD engines have received only academic interest by now due to the very high energy sources required to make them run (order of MegaWatts). A Japanese MPDT (EPEX (Electric Propulsion EXperiment)) was deployed during flight STS-72.

- Pulsed Plasma Thrusters: PPT thrusters use an arc of electric current adjacent to a solid propellant (teflon), to produce a quick and repeatable burst of impulse. PPTs are used for attitude control, and for main propulsion on particularly small spacecraft with a surplus of electricity (those in the hundred-kilogram or less category). However they are also one of the least efficient electric propulsion systems, with a thrust efficiency of less than 10%.

Let us now describe the main features of the ATV propulsion system, for its importance in the immediate future and as practical example of medium-high-thrust equipped space vehicle.
The main elements of the ATV propulsion are:

- 4 x 490 N main navigation engines;
- 28 x 220 N attitude control and braking thrusters;
- 8 titanium propellant tanks of 7 tonnes capacity;
- 2 high pressure carbon fibre-wound helium pressurant vessels.

The 220 N engines are highlighted in red in Figure 9.4. The bipropellant propulsion system is pressure fed with the propellant combination monomethyl hydrazine fuel and nitrogen tetroxide oxidizer.
Before closing this review on thrusters let us also mention the NASA Deep Space 1, the New Millennium Program first spacecraft. It is run by 0.09 N of thrust and has a specific impulse of 3300 s (over $32000\frac{m}{s}$ exhaust velocity), with a service life of $8000hrs$. The propellant gas in Deep Space 1 is xenon.

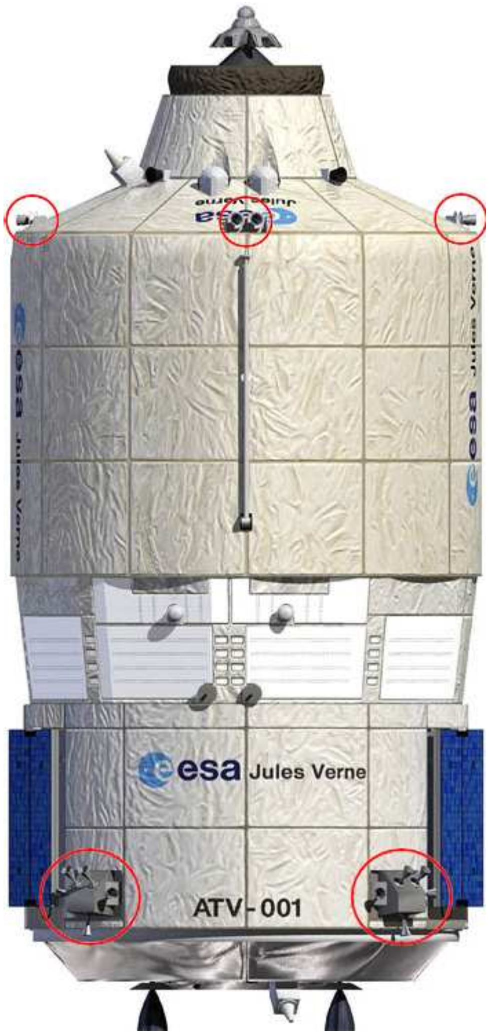

Figure 9.4: ESA Automated Transfer Vehicle

9.7 Appendix G: Notes on Genetic Algorithms

Genetic algorithms (GA) make use of a reduced version of the biological evolutionary process. Each cell of each individual of a given population (or phenotype) contains a complete set of instructions effectively defining its physical (and possibly behavioral) makeup. This information is encoded in the form of linear gene sequences stored on pairs of homologous chromosomes, which constitute the individual's genotype. Reproduction involves the combination of genetic material from both parents, one half of each chromosome pair coming from each parent. A fundamental aspect of this breeding process is that the relationship between phenotype and genotype is unidirectional: a given individual can be thought of as an external manifestation of its genotype (although there exist environmental influences in development and growth that are beyond genetic control), but the individual cannot influence its own genetic makeup. It can, however, influence the genetic makeup of subsequent generations through differential reproductive success, which is of course where natural selection plays its crucial role.
To a large extent, variability turns out to be maintained by the machinery of heredity itself. The production of reproductive cells often entails the recombination of genetic material across homologous chromosomes through the processes of crossover and inversion. Copying mistakes and/or true random events also occasionally introduce mutations in the genotype. The ensemble of all genes existing at a given time in the breeding population makes up the gene pool. For a given gene associated with a chromosomal locus, there exist in general more than one allowed "gene value" (or allele in biological terminology). Evolution can be thought of (and mathematically modeled) in terms of temporal changes in allele frequencies throughout the gene pool.
Genetic Algorithms (hereafter "GA") are a class of heuristic search techniques that incorporate these ideas in a setting that is computational rather than biological. Strictly speaking, genetic algorithms do not optimize, and neither does biological evolution. Evolution uses whatever material that is at its disposal to produce above average individuals. Evolution is blind. Evolution has no ultimate goal of "perfection". Even if it did, evolution must accommodate physical constraints associated with development and growth, so that not all paths are possible in genetic "parameter space". One could perhaps argue that evolution performs a form of highly constrained optimization, but even then it certainly does not optimize in the mathematical sense of the word. Nevertheless, genetic algorithms form the basis of a class of extremely robust optimization method known as GA-based optimizers.
The gene pool (and its associated phenotypic population) then evolves through the generations in response to three drivers: differential reproductive success in the population, genetic recombination (crossover) occurring at breeding and random mutations affecting a subset of breeding events. The differential reproductive success is given by the so-called fitness function, measuring the genetic quality of an individual: high fitness means high mating probability. The genetic recombination mixes the genes of two individuals at the moment of breeding, recombining genetic material, while the mutation adds a random feature to the process which allows a more deep exploration of the solution space. Figure 9.5 and Figure 9.6 give an idea of crossover and mutation. In analogy with biological systems, a phenotype is encoded in the form of a string (or chromosome) of digits. In contrast to biological systems, in basic genetic algorithms it is common to fully encode a phenotype on a single chromosome, as opposed to groups of homolo-

gous chromosome pairs with dominant/recessive character. A genotype then is made of a single chromosome, and both terms can be used interchangeably.

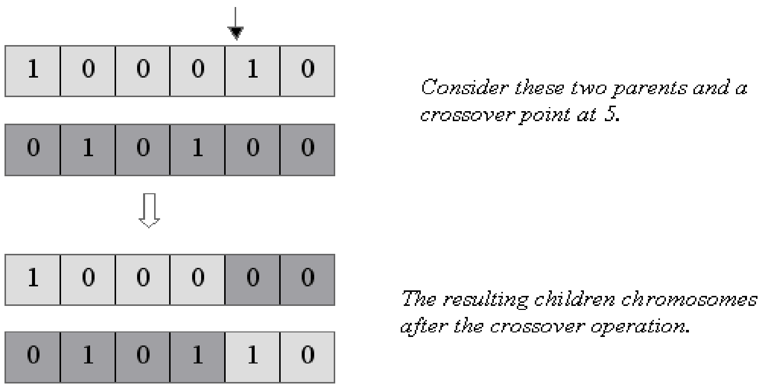
1 0 0 0 1 0
0 1 0 1 0 0
Consider these two parents and a crossover point at 5.
1 0 0 0 0 0
0 1 0 1 1 0
The resulting children chromosomes after the crossover operation.

Figure 9.5: Crossover Process at Mating

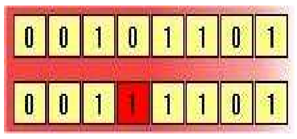

Figure 9.6: Mutation Process

A Genetic Algorithm always needs to encode the variables it works on in a chromosome like structure. The encoding process produces, for each selected parent, a structure that will subsequently be used for breeding through the action of the various genetic operators. The complementary process of decoding is the equivalent of development and growth in biology, i.e., the reconstruction of an individual from its defining genetic material. More pragmatically, the aim of the encoding process is to produce a “chromosome” from the n parameters (seven in this case) defining the function $f(\kappa)$ to be maximized. Given:

$$\kappa = [\kappa_1,\ ...,\ \kappa_n]$$

PIKAIA encodes these parameters using a decimal alphabet. Schematically, assuming to have each $\bar{\kappa}_k \in [0,\ 1]$, through a later on explained transformation on κ, we need to transform it into a $X_k = [X_1,\ ...,\ X_{nd}]_k$, where $X_j \in [0,\ 9]$ are positive integers. The encoding algorithm is:

$$X_j = mod\left(10^{nd-j+1}\bar{\kappa}_k,\ 10\right),\ j = 1,\ ...,\ nd$$

where the $mod(x,\ y)$ returns the remainder of the division of x by y. Each of the n defining parameters thus becomes a sequence of nd 1digit integers, so that the encoding of all n parameters to nd significant digits produces a 1D integer array (or “chromosome”) of length $n \times nd$. Each element of this array can be thought of as a “gene” having 10 possible alleles. For each encoded parameter, the complementary decoding process is:

$$\bar{\kappa}_k = \frac{1}{10^{nd}} \sum_{j=1}^{nd} X_j \times 10^j$$

Consider the task of maximizing a function $f(x,\ y)$ of two variables. In this case an individual (or "phenotype") is a point $(x,\ y)$ in 2D parameter space. The encoding process would produce $(x,\ y) = (0.34567890,\ 0.23456789) \rightarrow 3456789023456789$ for $nd = 8$. The chromosome 3456789023456789 is made up of 16 genes, and is the full genotype of the phenotype $(x,\ y)$. The number of digits retained in the encoding/decoding, *nd*, is an input quantity that remains fixed throughout the run. In order to increase the precision of the optimizer, some changes have been made to PIKAIA. In fact, all variables have been set to double precision, and the maximum number of digits for each gene has been brought from 6 to 9.
Referring to table 7.1 here follows a brief description of the remaining aspects of a GA.
There are three possible reproduction plans: Full generational replacement, Steady-state-replace-random and Steady state-replace-worst. The first case is perhaps the simplest reproduction plan. Throughout one iteration of the generation cycle, offspring are accumulated in temporary storage. Once a number of offspring equal to number of individuals have been so produced and stored, the entire parent population is wiped out and replaced by the offspring population , after which a new generational iteration begins. Under this reproduction plan, individuals have a fixed lifetime equal to a single generation. Steadystate reproduction plans insert individuals as they are being bred. Criteria must be specified to decide:

1. under which conditions newly bred offspring are to be inserted
2. how members of the parent population are to be deleted to make room for the new members
3. if any limit is to be imposed on an individual's lifetime

PIKAIA incorporates two steadystate plans. In both cases a newly bred offspring is inserted whenever its fitness exceeds that of the least fit member of the parent population, unless it is identical to an existing member of the population. Furthermore, PIKAIA imposes no limit on the generational lifetime of a population member; a very fit individual can survive through many iterations of the generational cycle. The two plans differ in how room is made to accommodate the offspring to be inserted. Under the steadystatedeleteworst plan, the least fit member of the parent population is eliminated and replaced by the offspring. Under the steadystatedelete-random plan, a member of the old population is chosen at random and deleted, independently of its fitness. In this work the steadystatedeleteworst is used.
The crossover operator is, in essence, what distinguishes genetic algorithms from other heuristic search techniques. PIKAIA incorporates a single crossover operator known as onepoint crossover. This operator acts on a pair of parentchromosomes to produce a pair of offspring-chromosomes. Consider again two prototypical "parents", for example:

$$(x,\ y)_1 = (0.34567890,\ 0.23456789)$$

$$(x,\ y)_2 = (0.87654321,\ 0.65432198)$$

Encoding to eight significant digits ($nd = 8$) would produce the corresponding parentchromosomes:

$$3456789023456789$$

$$8765432165432109$$

The crossover operation begins by randomly selecting a cutting point along the chromosomes, for example by generating a random integer K and cutting both parent chromosomes at the corresponding location.
The chromosomal fragments located right after the cutting point are then interchanged and concatenated to the fragments left of the cutting points:

$$345678902 - 5432109 \rightarrow 3456789025432109$$

$$876543216 - 3456789 \rightarrow 8765432163456789$$

The two strings resulting from this operation are the offspring chromosomes. These two chromosomes decode into the two offspring phenotypes:

$$(x_1,\ y_1) = (0.34567890,\ 0.25432198)$$

$$(x_2,\ y_2) = (0.87654321,\ 0.63456789)$$

The resulting offspring in general differ from either parent, although they do incorporate intact "chunks" of genetic material from each parent. In practice the crossover operator is applied only if a probabilistic test yields true. Define first a crossover rate $pcross \in [0,\ 1]$ and generate a random number $R \in [0,\ 1]$. The crossover operator is then applied only if $R \leq pcross$. If $R \geq pcross$, the two offspring remain exact copies of the two parents. The crossover rate (or probability) *pcross* is an input quantity, and remains constant throughout the evolution.
Regarding the mutation probability, PIKAIA incorporates a single mutation operator known as uniform onepoint mutation, but allows the mutation rate to vary dynamically in the course of the evolutionary run. The mutation operator runs as follows.
For each gene of an offspring chromosome, a random number $R \in [0,\ 1]$ is generated, and mutation hits the gene only if $R \leq pmut$, where $pmut \in [0,\ 1]$ is the mutation rate. The mutation itself consists in replacing the targeted gene by a random integer $K \in [0,\ 9]$. A maximum mutation probability of 1 is here chosen in order to ensure a greater variability in cases the fitness function rests for long time on a plateau (see [75] for more details on mutation dynamics in the algorithm).
Genetic algorithms have been used successfully to solve a number of difficult optimization problems arising in computer science, artificial intelligence, computer-aided engineering design, geosysmic modeling, and are attracting increasing attention in other branches of the physical sciences. Yet, generally speaking, they are not yet a standard component of the numerical modeler's toolboxes and many think that global optimization using partially stochastic algorithms is more art than science. The benefits of these techniques are, though, huge as they may approach many problems, otherwise unsolvable.

Bibliography

[1] Vadali S. R., Sengupta P., A Survey of Recent Work on Dynamics and Control of Formation Flying Satellites in Earth Orbits, 6th International Conference on Dynamics and Control of Structures and Systems in Space, Riomaggiore, Cinque Terre, Liguria, Italy, 2004.

[2] NASA, Bulletin *http://www.bulletinnews.com/nasa*, Restricted Access Website.

[3] *http://apod.nasa.gov/apod/ap060921.html*

[4] ESA, Ariadna *http://www.esa.int/gsp/ACT/ariadna/open_calls.htm*, Ariadna Calls 2004.

[5] Romano M., Friedman D. A., Shay T. J., Laboratory Experimentation of Autonomous Spacecraft Approach and Docking to a Collaborative Target, accepted for publication on AIAA Journal of Spacecraft and Rockets, 2006.

[6] Sabatini M., Bevilacqua R., Pantaleoni M., Izzo D., Periodic Relative Motion of Formation Flying Satellites, Paper AAS 06-206, AAS/AIAA Space Flight Mechanics Conference, Tampa, Florida, 2006.

[7] Clohessy W. H., Wiltshire R. S., Terminal Guidance System for Satellite Rendezvous, Journal of Aerospace Sciences, 653-658, 1960.

[8] Hill G. W., On differential equations with periodic integrals, Annals of Mathematics, 1887.

[9] H. Schaub, K. T. Alfriend, J2 Invariant Relative Orbits for Spacecraft Formations, Celestial Mechanics and Dynamical Astronomy, Volume 79, Number 2, February 2001, pp. 77-95(19).

[10] Sabatini M., Modellizzazione e Controllo della Dinamica Relativa di Satelliti in Formazione, M. Sc. Thesis, Università degli Studi di Roma, La Sapienza, A.A. 2001/2002.

[11] Ricelli R., Dinamica e Controllo di Satelliti in Formazione, M. Sc. Thesis, Università degli Studi di Roma, La Sapienza, A.A. 2001/2002.

[12] Tschauner J., Hempel P., Elliptic Orbit Rendezvous, AIAA Journal, Vol.5, No.6, 1967, pp. 1110-1113.

[13] Melton R. G., Time-Explicit Representation of Relative Motion Between Elliptical Orbits Journal of Guidance, Control and Dynamics, Vol. 23, No. 4, 2000, pp. 604-610.

[14] Schaub H., Vadali S. R., Junkins J. L., Alfriend K. T., Spacecraft Formation Flying Control using Mean Orbital Elements, Journal of the Astronautical Science, Vol. 48, No.1, Jan.-March 2000, 69-87.

[15] Brouwer, Dirk, Solution of the Problem of Artificial Satellite Theory Without Drag, The Astronautical Journal, vol. 68, oct 1963, pp. 555-558.

[16] Wiesel W. E., Relative Satellite Motion About an Oblate Planet, Journal of Guidance, Control and Dynamics, Vol. 25, pp. 776-785.

[17] Schweighart S. A., Sedwick R., A High Fidelity Linearized Model for Satellite Formation Flying, Journal of Guidance Control and Dynamics, Vol.25, No. 6, Nov-Dec. 2002, p.1073-80.

[18] Izzo D. , Sabatini M., Valente C., A New Linear Model Describing Formation Flying Dynamics Under J2 Effects, Proceedings of the XVII AIDAA congress held in Rome, Italy, 15-19 September 2003, Vol.1, pp.493-500.

[19] Battin R. H., An Introduction to the Mathematics and Methods of Astrodynamics, AIAA Education Series.

[20] Vadali S. R., An Analytical Solution for Relative Motion of Satellites, Proceedings of the 5th International Conference on Dynamics and Control of Structures and Systems in Space, Cranfield University Press, 2002, pp.309-316.

[21] Sangalli A., L'importanza di essere fuzzy, Bollati Boringhieri.

[22] Bauer F., Bristow J., Folta D., Hartman K., Quinn D., Satellite Formation Flying Using an Innovative Autonomous Control System (AUTOCON) Environment, AIAA paper, 1997.

[23] Dachwald B., Optimization of Very-Low-Thrust Trajectories Using Evolutionary Neurocontrol, 55th International Astronautical Congress, Vancouver, Canada, 2004.

[24] Ren W., Beard R. W., Quinn Decentralized Scheme for Spacecraft Formation Flying via the Virtual Structure Approach, Journal of Guidance, Control and Dynamics, vol. 27, No. 1, J.-F. 2004.

[25] Campbell M. E., Quinn D., Planning Algorithm for Multiple Satellite Clusters, Journal of Guidance, Control and Dynamics, vol. 26, No. 5, S.-O. 2003.

[26] Vaddi S. S., Vadali S. R., Linear and Non Linear Control Laws for Formation Flying, 13th AAS/AIAA Space Flight Mechanics Meeting, 2003.

[27] Underwood C. I., Richardson G., Savignol J., Quinn D., In-orbit Results from the SNAP-1 Nanosatellite and Its Future Potential, Philosophical Transactions of the Royal Society, London A., vol. 361, Dec. 2003, pp.199-203.

[28] Palmer P., Optimal Relocation of Satellites Flying in Near-Circular-Orbit Formations, Journal of Guidance, Control and Dynamics, Vol. 29, No. 3, May-June 2006, pp. 519-526.

[29] Guelman M., Aleshin M., Optimal Bounded Low-Thrust Rendezvous with Fixed Terminal-Approach Direction, Journal of Guidance, Control and Dynamics, Vol. 24, No. 2, March-April 2001, pp. 378-385.

[30] Carter T. E., Pardis C. J., Optimal Power-Limited Rendezvous with Upper and Lower Bounds on Thrust, Journal of Guidance, Control and Dynamics, Vol. 19, No. 5, Sep.-Oct. 1996, pp. 1124-1133.

[31] Carter T. E., Optimal Power-Limited Rendezvous for Linearized Equations of Motion, Journal of Guidance, Control and Dynamics Vol. 17, No. 5, Sept.-Oct. 1994, pp. 1082-1086.

[32] Pardis C. J., Carter T. E., Optimal Power-Limited Rendezvous with Thrust Saturation, Journal of Guidance, Control and Dynamics Vol. 18, No. 5, Sept.-Oct. 1995, pp. 1145-1150.

[33] Carter T. E., Alvarez S. A., Quadratic-Based Computation of Four-Impulse Optimal Rendezvous near Circular Orbit, Journal of Guidance, Control and Dynamics, Vol. 23, No. 1, Jan.-Feb. 2000, pp. 109-117.

[34] Lawden D. F., Optimal Trajectories for Space Navigation, London Butterworths, 1963, pp. 60-64.

[35] Carter T. E., Optimal Impulsive Space Trajectories Based on Linear Equations, Journal of Optimization Theory and Applications, Vol. 70, No. 2, August 1991, pp. 277-297.

[36] Carter T. E., Brient J., Fuel-Optimal Rendezvous for Linearized Equations of Motion, Journal of Guidance, Control and Dynamics, Vol. 15, No. 6, November-December 1992, pp. 1411-1416.

[37] Lion P. M., Hendelsman M., Primer Vector on Fixed-Time Impulsive Trajectories, AIAA Journal, Vol. 6, No. 1, January 1968, pp. 127-132.

[38] Jezewsky D. J., Rozendaal H. L., An Efficient Method for Calculating Optimal Free-Space N-Impulse Trajectories, AIAA Journal, Vol. 6, No. 11, November 1968, pp. 2160-2165.

[39] Prussing J. E., Optimal Two- and Three-Impulse Fixed-Time Rendezvous in the Vicinity of a Circular Orbit, AIAA Journal, Vol. 8, July 1970, pp. 1221-1228.

[40] Prussing J. E., Optimal Multiple-Impulse Time-Fixed Rendezvous Between Circular Orbits, Journal of Guidance, Control and Dynamics, vol. 9, No. 1, Jan.-Feb. 1986, pp. 17-22.

[41] Lembeck C. A., Prussing J. E., Optimal Impulsive Intercept with Low-Thrust Rendezvous Return, Journal of Guidance, Control and Dynamics, vol. 16, No. 3, May-June 1993, pp. 426-433.

[42] Edelbaum T. N., Minimum Impulse Transfers in the Near Vicinity of a Circular Orbit, The Journal of the Astronautical Sciences, Vol. XIV, No. 2, Mar.-Apr. 1967, pp. 66-73.

[43] Edelbaum T. N., How Many Impulses?, Astronautics and Aeronautics, Nov. 1967, pp. 64-69.

[44] Jeng-Hua Chiu, Optimal Multiple-Impulse Nonlinear Orbital Rendezvous, Ph. D. Thesis, Univerity of Illinois at Urbana-Champaign, 1984.

[45] Jezewsky D. J., Primer Vector Theory Applied to the Linear Relative-Motion Equations, Optimal Control Applications and Methods, Vol. 1, 1980, pp. 387-401.

[46] Neustadt L. W., Optimization, A Moment Problem and Nonlinear Programming, SIAM Journal on Control, Vol. 2, No. 1, 1964, pp. 33-53 .

[47] Prussing J. E., Optimal Impulsive Linear Systems: Sufficient Conditions and Maximum Number of Impulses, The Journal of the Astronautical Sciences, Vol. 43, No. 2, Apr.-June 1995, pp. 195-206.

[48] Leonard C.L., Bergmann E. V., A Survey of Rendezvous and Docking Issues and Developments, AAS 89-158, pp. 85-101.

[49] Bevilacqua R., Romano M., Real Time Quasi-Optimal Unbounded Control for Path Constrained Relative Spacecraft Manoeuvres, Proceedings of the 7^{th} Dynamics and Control of Systems and Structures in Space Conference, organized by Cranfield University, held in Greenwich, UK, July 2006, pp. 219-237.

[50] Bevilacqua R., Yakimenko O., Romano M., On-line Generation of Quasi-Optimal Docking Trajectories, Proceedings of the 7^{th} Dynamics and Control of Systems and Structures in Space Conference, organized by Cranfield University, held in Greenwich, UK, July 2006, pp.203-218.

[51] Bevilacqua R., Romano M., Optimal Guidance of Proximity Maneuvers for a Spacecraft with Hybrid On-Off Continuous and Impulsive Thrust, submitted for pubblication in Journal of Guidance, Control and Dynamics.

[52] Sabatini M., Bevilacqua R., Pantaleoni M., Izzo D., A Search for Invariant Relative Satellite Motion, accepted for publication in Journal of Non Linear Dynamics and Systems, also Proceedings of the 4^{th} Workshop on Satellite Constellations and Formation Flying, Sao Josè Dos Campos, Brazil, 14-16 February 2005, pp. 222-229.

[53] Vinh, N. X., Optimal Trajectories in Atmospheric Flight, Elsevier Scientific Publishing Company, 1981, pp. 10-19.

[54] Isidori A., Sistemi di Controllo, Siderea, Seconda Edizione, Volume Primo, 1992.

[55] Bolza O., Lectures on the Calculus of Variations, University of Chicago Press, Chicago, 1904.

[56] Pontryagin L.S., Boltjanskiy V.G., Gamkrelidze R.V., Mishenko E.F., The Mathematical Theory of Optimal Processes, Willey-Interscience, New York, NY, 1969.

[57] Carter T. E., Fuel-Optimal Power-Limited Maneuvers of a Spacecraft Relative to a Point in Circular Orbit, Journal of Guidance, Control and Dynamics, Vol. 7, No. 6, November-December 1984, pp. 710-716.

[58] Bellman R., Dynamic Programming, Princeton University Press, Princeton, New Jersey, 1957.

[59] Bernelli-Zazzera F., Romano M., Time-Optimal Motion for Robotic Manipulators with Obstacles Avoidance, Proceedings of 4^{th} International Conference on Dynamics and Control of Structure in Space, Cranfield University, Cranfield, UK, May 1999.

[60] Shiller Z., Dubowsky S., Robot Path Planning with Obstacles, Actuator, Gripper, and Payload Constraints, International Journal of Robotics Research, Vol. 8, No. 6, pp. 3-18, December 1989.

[61] Cruciani I., De Divitiis N., De Matteis G., Filippone E., Autonomous Guidance for a Sub-Orbital Re-Entry Vehicle, Paper IAC-03-A.7.06, 54^{th} International Astronautical Congress, Bremen, Germany, September 2003.

[62] Miles D. W., Real Time Dynamic Trajectory Optimization with Application to Free-Flying Space Robots, Stanford University Ph. D. Thesis, 1997.

[63] Mortensons M.E., Geometric Modeling, John Wiley & Sons: New York, 1985.

[64] *http://www.nr.com/*

[65] Yakimenko O. A., Direct Method for Rapid Prototyping of Near-Optimal Aircraft Trajectories, Journal of Guidance, Control and Dynamics, Vol. 23, Number 5, Pages 865-875.

[66] Prussing J. E., Optimal Two- and Three-Impulse Fixed-Time Rendezvous in the Vicinity of a Circular Orbit, Journal of Spacecraft and Rockets, Vol. 40, No. 6, Nov.-Dec. 2003, pp. 952-959.

[67] Ruth M., Tracy C., Orbital Sciences Corp. (USA), Video-Guidance Design for the DART Rendezvous Mission, Proceedings of SPIE, Vol. 5419, 14 April 2004, Orlando, Florida, USA, pp. 92-106.

[68] Shoemaker J., Wright M., DARPA (USA) and SRS Technologies (USA), Orbital Express Space Operations Architecture Program, Proceedings of SPIE, Vol. 5419, 14 April 2004, Orlando, Florida, USA, pp. 57-65.

[69] Hooke R., Jeeves T. A., Direct Search Solution of Numerical and Statistical Problems, Journal of the Association for Computing Machinery, vol. 8, No. 2, April 1961, pp. 212-229.

[70] Shewchuk J. R., An Introduction to the Conjugate Gradient Method Without the Agonizing Pain, Edition 1 $\frac{1}{4}$, School of Computer Science Carnegie Mellon University Pittsburgh, PA 15213, August 4, 1994.

[71] *http : //www.ictp.trieste.it/ mpking/cg.html*

[72] Inalhan G., Tillerson M., How J. P., Relative Dynamics and Control of Spacecraft Formations in Eccentric Orbits, Journal of Guidance, Control and Dynamics, Vol. 25, No. 1, 2002, pp. 48-59.

[73] Kasdin N. J., Koleman E., Bounded Periodic Relative Motion using Canonical Epicyclic Orbital Elements, Paper AAS 05-186, 15^{th} AAS/AIAA Space Flight Mechanics Meeting, Copper Mountain, Colorado, 2005.

[74] Vaddi S. S., Vadali S.R., Alfriend K.T., Formation Flying: Accommodating Nonlinearities and Eccentricity Perturbations, Journal of Guidance, Control and Dynamics, Vol. 26, No. 2, March-April 2003, pp. 214-223.

[75] Charbonneau P., Knapp B., A User's Guide to PIKAIA 1.0, NCAR Technical note 418+1A, Boulder: National Centre for Atmospheric Research, 1995.

[76] Sabatini M., Izzo D., Palmerini G.B., Analysis and Control of Convenient Orbital Configurations for Formation Flying Missions, AAS/AIAA Space Flight Mechanics Meeting Tampa, Florida, 2006.

[77] Jacchia L. G., Standard Jacchia Reference Atmosphere 1977, Smithsonian Astrophysical Observatory Cambridge, Massachusetts.

[78] Rimrott, Fred P.J., Introductory Orbit Dynamics, Braunschweig, Wiesbaden, Vieweg, 1989.

[79] Romano M., On-the-ground Experiments of Autonomous Spacecraft Proximity Navigation using Computer Vision and Jet Actuators, Proceedings of IEEE International Conference on Advanced Intelligent Mechatronics, 2005, pp.1011 - 1016.

[80] *http://cs.space.eads.net/sp/*.

[81] *http://www.centrospazio.cpr.it/*

[82] *http://www.ssloral.com/*

[83] Marcuccio S., Genovese A., Andrenucci M., FEEP Microthruster Technology Status and Potential Applications, technical paper from Centrospazio web page [81].

[84] Marcuccio S., Paita L., Saviozzi M., Andrenucci M., Flight Demonstration of FEEP on Get Away Special, technical paper from Centrospazio web page [81], also AIAA 98-3332, 33rd AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit, July 13-15, 1998, Cleveland, OH.

[85] Marcuccio S., Giannelli S., Andrenucci M., Attitude and Orbit Control of Small Satellites and Constellations with FEEP Thrusters, technical paper from Centrospazio web page [81].