TECHNISCHE UNIVERSITÄT MÜNCHEN
Lehrstuhl für Raumfahrttechnik

Autonomous Proximity Operations for Noncooperative Space Target

Jacopo Ventura

Vollständiger Abdruck der von der Fakultät für Maschinenwesen der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender: Prof. Dr.-Ing. habil. Boris Lohmann

Prüfer der Dissertation:

1. Prof. Dr. rer. nat. Dr. h. c. Ulrich Walter
2. Prof. Dr. Marcello Romano

Die Dissertation wurde am 15.09.2016 bei der Technischen Universität München eingereicht und durch die Fakultät für Maschinenwesen am 02.12.2016 angenommen.

Autonomous Proximity Operations for Noncooperative Space Target

Contents

Summary

Future space debris removal and On-Orbit Servicing missions to refuel, repair, or deorbit end-of-life and defunct satellites require proximity operations and, eventually, docking to noncooperative space vehicles. The main challenge in these operations is the noncooperative nature of the target, since it is not equipped with visual markers for attitude and distance (i.e. pose) estimation, it does not communicate with the chaser and it may rotate or tumble about the principal axes. Existing state-of-the-art robotic systems in to date test missions performed proximity operations and docking only to either cooperative or partially cooperative attitude-stabilized targets after overcoming contingencies through ground support. The capability to autonomously rendezvous and dock to noncooperative space targets was considered by NASA a high-priority cross-cutting technology required for the decade 2011-2021 and beyond. Autonomous systems would be able to service the majority of space debris and end-of-life satellites using the same algorithms and methods. Therefore, most servicing missions can be performed with the same kind of technology. This would ensure frequent On-Orbit Servicing missions.

This research was motivated by the hypothesis that it is possible to develop a complete Guidance, Navigation and Control system capable to perform autonomous proximity operations for noncooperative space targets. To investigate methods for safe and reliable noncooperative proximity operations, an innovative Guidance, Navigation and Control system is developed, implemented and verified step-by-step in this thesis.

The first part of the research develops the guidance algorithm for onboard generation of the approach trajectory. The dynamics model for proximity operations between a controlled chaser spacecraft and an uncontrolled target is first defined. This model includes the path constraints required for safe maneuvering. As a particular test case, the docking maneuver is considered. Based on this model and the requirements for the Guidance, Navigation and Control system, a methodology for rapid trajectory generation is chosen among state-of-the-art methods. The key feature of the selected algorithm is that the trajectory of the chaser spacecraft is parameterized by a set of polynomials. Using the dynamics model of the maneuver and a nonlinear programming solver, the coefficients of polynomials are optimized while respecting the path constraints. Due to the limited number of variables, the optimization of the 6 degrees-of-freedom trajectory can be executed in between 0.5 s and 9.0 s, allowing onboard execution. Optimality, maneuvering

precision and robustness to orbital perturbations and sensor noise are evaluated through extensive numerical experiments.

The second part of the research focuses on the navigation system. The algorithms, including the navigation filter, for model-based, noncooperative pose estimation using data from Lidar, stereo camera and mono camera are developed separately. These sensors are considered since they are state-of-the-art systems in terrestrial robotics. The performance of the proposed methods is evaluated through an experimental campaign using the RACOON laboratory, which provides real-time full degrees-of-freedom hardware simulation of proximity operations under realistic lighting conditions. Test results lead to the choice of the most suitable rendezvous sensor and related attitude estimation technique that complete the innovative Guidance, Navigation and Control system.

Zusammenfassung

Zukünftige Missionen zur Entfernung von Weltraummüll, oder Service-Missionen im Orbit zum Auftanken, Reparieren oder um veraltete oder fehlfunktionierende Satelliten aus dem Orbit zu entfernen, benötigen Nahbereichs-Operation und, abschließend, Andocken an nicht-kooperative Weltraumfahrzeuge. Die größte Herausforderung bei diesen Operationen ist die nicht-kooperative Eigenschaft des Zielsatelliten, da er nicht mit sichtbaren Markern für Lage und Abstand ausgerüstet ist, nicht mit dem Verfolger kommuniziert und eventuell rotiert oder um seine Hauptachsen taumelt. Bisherige robotische Weltraumsysteme haben Nahbereichsoperationen und Andocken entweder mit kooperativen oder teil-kooperativen und lagestabilisierten Zielsatelliten während Testmissionen durchgeführt, nachdem sie unvorhergesehene Vorfälle mit Bodenunterstützung überwinden konnten. Die Fähigkeit, autonom an nicht-kooperativen Weltraum-Fahrzeugen heranzufliegen und mit ihnen zu docken wurde von NASA als eine interdisziplinäre Hochprioritäts-Technologie für das Jahrzehnt 2011-2021 und darüber hinaus definiert. Autonome Systeme wären in der Lage, die Mehrheit des Weltraummülls oder ausgediente Satelliten mit Hilfe derselben Algorithmen und Methoden zu entsorgen. Daher können die meisten Service-Missionen mit der gleichen Satellitenart durchgeführt werden. Das würde häufige Servicemissionen ermöglichen.

Diese Forschungsarbeit basiert auf der Hypothese, dass es möglich ist, ein komplettes Steuerungs-, Navigations- und Kontrollsystem zu entwickeln, welches fähig ist, autonome Nahbereichsoperationen zu nicht-kooperativen Weltraumzielen durchzuführen. Um Methoden für sichere und verlässliche Nahbereichs-Operationen zu nicht-kooperativen Weltraumzielen zu untersuchen, wird in dieser Arbeit ein innovatives Steuerungs-, Navigations- und Kontrollsystem Schritt für Schritt entwickelt, implementiert und verifiziert.

Der erste Teil widmet sich der Entwicklung eines Steuerungsalgorithmus zur bordgestützten Erzeugung der Anflugs-Trajektorie. Zunächst wird das dynamische Modell für Nahberichtsoperationen zwischen dem verfolgenden Weltraumfahrzeug und dem unkontrollierten Zielfahrzeug definiert. Das Modell beinhaltet Pfadeinschränkungen für sicheres Manövrieren. Als spezieller Test wird das Andocken betrachtet. Basierend auf diesem Modell und den Anforderungen für das Steuerungs-, Navigations- und Kontrollsystem wird eine Methode für die schnelle Trajektoriengeneration aus den dem Stand der Technik entsprechenden Methoden ausgewählt. Das Hauptmerkmal des gewählten Algorithmus ist die durch Polynome parametrisierte Trajektorie des

Verfolgers. Die Polynomkoeffizienten werden unter Berücksichtigung der Wegbeschränkungen anhand des dynamischen Modells des Anflugmanövers und eines nichtlinearen Programmierlösers. Aufgrund der begrenzten Anzahl von Variablen kann die die Optimierung von 6 Freiheitsgraden zwischen 0.5 s und 9.0 s durchgeführt werden. Der Grad der Optimierung, Manöverpräzession und Robustheit zu Umlaufbahnstörungen und Sensorrauschen werden durch ausführliche numerische Experimente untersucht.

Der zweite Teil der Arbeit konzentriert sich auf das Navigationssystem. Die Algorithmen, inklusive des Navigationsfilters, für die modellbasierende Lageabschätzung werden unter Nutzung von Lidar, Stereo- und Monobildkameras separat entwickelt. Diese Sensoren werden berücksichtigt, da sie dem Stand der Technik terrestrischer robotischer Systeme entsprechen. Die Performance der vorgeschlagenen Methoden werden in einer experimentellen Versuchsreihe unter Nutzung des RACOON Labors durchgeführt, welches Echtzeitsimulation der Nahbereichsoperationen für alle Freiheitsgrade unter realen Beleuchtungssituationen in Hardware zur Verfügung stellt. Anhand der Testresultate kann der geeignetste Sensor und die damit zusammenhängende Lageabschätzungsmethode gewählt werden, was wiederum das Steuerungs-, Navigations- und Kontrollsystem vervollständigt.

List of symbols

Symbols

A	Exposed cross-section area for drag computation
b	Baseline of the stereo camera
$\boldsymbol{b}$	Boresight direction of the chaser rendezvous sensor
C	Principal body-fixed Cartesian coordinate system of the chaser spacecraft
CD	Auxiliary body-fixed Cartesian coordinate system on the chaser docking point
C_D	Aerodynamic drag coefficient
$\boldsymbol{D}$	Measured point cloud of the target spacecraft
d	Disparity
$\boldsymbol{F}$	Control thrust vector
H	Hill's Cartesian coordinate system
$\boldsymbol{I}$	Inertial Tensor
I_{xx}^{s}, I_{yy}^{s}, I_{zz}^{s}	Moments of inertia of the spacecraft s about the principal axes (s = C, T)
J	Performance index of the optimal control problem
J_i	Harmonic coefficient related to the non-homogeneous gravitational potential of the Earth
$\boldsymbol{l}^{s}$	Position of the spacecraft's docking point expressed in the system s (s = C, T)
m_s	Mass of the spacecraft s (s = C, T)
n	Polynomial order
$\boldsymbol{p}$	Position of the target center of mass relative to the chaser's rendezvous sensor
$\boldsymbol{q}$	Quaternion vector
$\boldsymbol{q}^{s}$	Quaternion vector of the attitude of the coordinate system s with respect to H (s = C, T)
$\boldsymbol{q}^{\text{T-S}}$	Quaternion vector of the attitude of T with respect to S
$\boldsymbol{R}_{\text{A}}^{\text{B}}$	Direction cosine matrix, from the coordinate system A to the system B
$\boldsymbol{r}$	Position vector of the spacecraft center of mass with respect to the ECI coordinate system
$\boldsymbol{r}_{\text{safe}}$	Safety radius for collision avoidance

$r_\oplus$	Average Earth equatorial radius ($r_\oplus$ = 6378.147 km)
S	Cartesian coordinate system of the chaser's rendezvous sensor
$\boldsymbol{s}$	Position vector of the chaser rendezvous sensors with respect to the chaser center of mass
T	Principal body-fixed Cartesian coordinate system of the target spacecraft
TD	Auxiliary body-fixed Cartesian coordinate system on the target docking point
$\boldsymbol{T}$	Control torque vector
t_f	Maneuver duration
t_GNC	Period of execution of the guidance loop
t_min	Sampling rate of the low level attitude and position controllers
$\boldsymbol{v}_\mathrm{con}$	Contact velocity between chaser and target docking points
$\boldsymbol{v}_\mathrm{rel}$	Velocity of the spacecraft relative to the atmosphere
$\boldsymbol{X}$	Point cloud of the target spacecraft geometric model
$\boldsymbol{x}$	State vector of the two-spacecraft rendezvous model
$[x,y,z]$	Position of the chaser center of mass in H system
α_FOV	Half-angle of the rendezvous sensor field-of-view cone
α_T	Angular distance from the rendezvous sensor boresight to the target spacecraft center of mass
$\beta_{i,n}()$	Bernstein polynomials of order n
$\boldsymbol{\sigma}$	Modified Rodrigues parameters vector
$\boldsymbol{\lambda}()$	Speed factor
$\boldsymbol{\mu}_\mathrm{D}$	Centroid of $\boldsymbol{D}$
$\boldsymbol{\omega}^{\,s}$	Angular velocity of the spacecraft s with respect to the inertial reference frame and referred to the frame s (s = C, T)
$\omega_\oplus$	Angular velocity of the Earth ($\omega_\oplus$ = 7.292115e-5 rad/s)
Ω	Angular velocity of the Hill's coordinate system
δa	Noise applied to the variable a
τ	Virtual domain
${}^\mathrm{H}\boldsymbol{\omega}^{\,s}$	Angular velocity of the spacecraft s with respect to the Hill's coordinate system (s = C, T)
$\Delta\boldsymbol{r}$	Relative distance between docking points at the docking
$\Delta\boldsymbol{v}$	Relative velocity between docking points at the docking
$\Delta\boldsymbol{\theta}$	Relative attitude (Euler angles 123) between auxiliary systems CD and TD at the docking instant
$\Delta\boldsymbol{\omega}$	Relative angular velocity between auxiliary systems CD and TD at the docking instant
ρ	Density of the atmosphere
μ	Earth's gravitational coefficient (μ = 398600.4416 km^3/s^2)

Subscripts

a_0	Value of a at the beginning of the maneuver
a_f	Value of a at the end of the maneuver
a_t	Value of a at the particular instant of time t

Superscripts

$\boldsymbol{a}^s$	Vector $\boldsymbol{a}$ whose components are expressed in the coordinate system s (s = H, C, CD, T, TD,S)
$\boldsymbol{X}^{cp}$	Closest points of $\boldsymbol{X}$ to the points of $\boldsymbol{D}$

Acronyms

CPU	Core Processing Unit
C-W	Clohessy-Wiltshire equations
DART	Demonstration of Autonomous Rendezvous Technology
DLR	Deutschen Zentrums für Luft- und Raumfahrt
DoF	Degrees of Freedom
ECI	Earth-Centered Inertia coordinate system
EKF	Extended Kalman Filter
FOV	Field of View
GNC	Guidance, Navigation and Control
GPOPS	General Purpose OPtimal Control Software
GPS	Global Positioning System
ICP	Iterative Closest Point
IDVD	Inverse Dynamics in the Virtual Domain
IMU	Inertial Measurement Unit
LEO	Low Earth Orbit
MILP	Mixed-Integer Linear Programming
MLI	Multi-Layer Insulation
MPC	Model-Predictive Control
MRP	Modified Rodrigues Parameters
NASA	National Aeronautics and Space Administration
OOS	On-Orbit Servicing
PWM	Pulse Width Modulation
RACOON	Robotic Actuation and On-Orbit Navigation Laboratory
RAM	Random Access Memory
RRT	Rapidly-exploring Random Tree
SAD	Sum of Absolute Differences
SDM	Sigma Delta Modulation
SNOPT	Sparse Nonlinear OPTimizer

Acknowledgments

This thesis was possible thanks to the collaboration and support of colleagues, friends and family.

First and foremost, I thank Prof. Walter for allowing me to write this thesis under his custody at his Institute and for to providing me the freedom to pursue my own ideas. My greatest thank goes to Dr. Hoehn, who supported me from the first day at the Institute when I was still a master student. I thank all my colleagues of RACOON laboratory for the constant and precious help to overcome the daily problems in the laboratory. I would like to express my gratitude to Andreas Fleischner for his passion and talent in building the RACOON laboratory, which played a key role in this research. I also have to thank Dr. Harder for his constant help.

I would like to thank Prof. Romano of the Naval Postgraduate School for his advices, especially in the toughest moment of the PhD being the first year, and for hosting me at his Spacecraft Robotics Laboratory in Monterey. I would like to express my gratitude to Dr. Ciarcia' (former Naval Postgraduate School, now Professor at South Dakota State University) not only for his ideas and visions, but also for being a source of inspiration. His suggestions on developing the control algorithms have greatly enhanced the quality, clarity and scope of this work. I greatly thank Prof. D'Amico of Stanford University for hosting me at his SLAB laboratory and for his precious suggestions during my stay. I really enjoyed to work together with the student of SLAB. Special thanks go to Sumant Sharma for sharing his talent with me on the development of the navigation software.

Above all, my gratitude is owed to my family, first and foremost my wife Lin, who had to live with my frustration, doubts and panic attacks for four long years and who never ceases to support me.

1 Thesis Scope

1.1 Motivation

Proximity operations and docking between spacecraft are key maneuvers in spaceflight. The capability of a spacecraft to approach and mechanically connect to another target vehicle enabled missions such as the Apollo program for Moon exploration, the servicing flights of the Space Shuttle and the assembly and supply of the space stations [1]. In these missions, proximity operations and docking procedures have been mainly performed by crewed space vehicles and involved cooperative target spacecraft, i.e. its attitude was controlled to help the maneuver execution and it was equipped with visual markers for distance estimation. As regard to the new generation of space robotics missions such as debris removal and On-Orbit Servicing (OOS) proximity operations and docking are key-enabling capabilities for either repair, refuel or deorbit end-of-life and nonfunctional satellites [2, 3]. The main challenge in these missions is to approach a noncooperative target vehicle, since it does not communicate with the ground station, it is not equipped with visual markers for relative attitude and distance estimation and it may rotate or tumble about the principal axes. Since ground estimates of the motion of the target are affected by large uncertainties, the approach trajectory must be determined during the mission using the flight data acquired by the rendezvous sensor suite. Several test OOS missions were performed in order to validate new technologies for autonomous or semi-autonomous rendezvous and docking. Successful maneuvers were executed with partially cooperative targets after overcoming mission contingencies through ground support. Therefore, proximity operations with noncooperative targets are currently at the edge of the capabilities of today's robotics systems.

This research therefore proposes a complete Guidance, Navigation and Control (GNC) system to enable autonomous proximity operations to noncooperative space targets. The desired outcome of performing autonomous missions is the increment of mission frequency, robustness and reliability with respect to the existing ground-in-the-loop guidance methods [4]. The capability to autonomously rendezvous and dock allows retrieving and servicing a variety of space targets being either end-of-life satellites, nonfunctional spacecraft or upper stage rockets using the same technology. Finally, this technology can be applied for performing deep space rendezvous and docking, as autonomous systems do not require real-time support from ground.

A rapid trajectory generation algorithm is developed in order to generate onboard sub-optimal approach trajectories. The key feature of the method is that the trajectory of the chaser spacecraft is parameterized through a set of polynomials. By using the known dynamics model of proximity operations and a nonlinear programming solver, the polynomial coefficients are optimized while enforcing the path constraints. The particular application to docking maneuvers is considered. Performances of the proposed algorithm such as maneuvering precision, robustness to system noise and optimality are verified through extensive numerical simulations. This software package defines the guidance and control subsystems.

To enable regular trajectory generation during the maneuver, the navigation subsystem of the proposed GNC is equipped with a dedicated model-based noncooperative pose estimation algorithm in order to calculate, in real-time, current attitude and distance of the target vehicle. An extended Kalman filter (EKF) estimates the velocities of the spacecraft while reducing sensor noise. This algorithm and the related rendezvous sensors for data acquisition are chosen after an experimental campaign that compared the performances of three different pose estimation systems. In particular, Lidar-, stereo camera- and mono camera-based systems were developed and tested.

The outcome of this research is an innovative GNC system composed by a dedicated rendezvous sensor suite and related software for onboard estimation and trajectory planning for proximity operations.

1.2 Why Autonomy

Proximity operations and, eventually, docking to noncooperative space targets are the last flight phases of OOS and space debris removal missions. The capability to approach and capture end-of-life or defunct space objects would allow either repairing, refueling or deorbiting the target. The main challenges in proximity operations is the noncooperative nature of the target vehicle, since it is not equipped with dedicated visual markers for attitude and position estimation, it does not communicate with the chaser and it may tumble about the principal axes. A concrete example of such vehicles is provided by Envisat, which is a defunct satellite in Low Earth Orbit (LEO) tumbling at about 3.2 deg/s with a constant loss of kinetic energy [5]. Furthermore, the precise trajectory for noncooperative proximity operations cannot be estimated before the launch of the mission since the available estimates of the target motion are affected by high uncertainties. For these reasons, the chaser spacecraft must be able to acquire, in proximity of the target, enough data for estimating relative distance, attitude and velocity with high precision. Knowing these parameters, the chaser system must calculate onboard safe trajectories for approaching and eventually docking to the target. Clearly, the chaser system must be flexible in order to cope with any rotational motion of the client and with uncertainties on state estimates

and target inertia. Moreover, due to the short distance between spacecraft, the chaser must ensure quick response to eventual mission contingencies.

In recent years, test OOS were performed in order to validate new GNC for proximity operations and docking maneuvers. These missions were operated either by ground control or in autonomous mode. In the first case, commands are generated on-ground after processing the flight data provided through the downlink. This control architecture introduces tight constraints due to ground station visibility and does not guarantee prompt response to mission contingencies. Furthermore, the usage of the onboard resources such as power or fuel may not be optimal. On the contrary, autonomous flight capabilities ensure more precise maneuvering, quick response to eventual contingencies and a more optimal usage of the spacecraft onboard resources. However, autonomous proximity operations and docking maneuvers were performed with cooperative and attitude stabilized target after overcoming mission contingencies through ground support. More recently, some studies have been conducted on the feasibility of the human-in-the-loop control for noncooperative proximity operations and docking [6, 7]. These researches were motivated by the fact that a human operator can potentially increment the situation awareness and quickly provide, through his intuition, spacecraft controls, state estimate from basic visual information and decision making to the robotic system. Despite potential advantages, feasibility studies showed important limitations of this method. First, the duration of the teleoperated phase is limited due to visibility from ground station. Second, the human operator has difficulties to command the 6 degrees-of-freedom (DoF) of the chaser spacecraft due to the unfamiliar orbital dynamics, communication delay and video quality. These experiences indicate the need of developing new control technologies to enable safe proximity operations and, eventually, docking to noncooperative space targets for OOS and debris removal missions.

This research work is motivated by the conviction that only autonomous control system would enable safe proximity operations for noncooperative space targets. In fact, autonomous systems would handle the complex 6 DoF orbital dynamics, optimize the usage of onboard resources and eliminate the constraints related to ground visibility. Moreover, the desired outcome of performing autonomous missions is the increment of mission frequency, robustness, and reliability with respect to the existing ground-in-the-loop guidance methods. The capability to autonomously rendezvous and dock to noncooperative clients would allow retrieving and servicing a variety of target objects such as space debris, end-of-life satellites and upper stage rockets. Furthermore, such technology for autonomous proximity operations for space debris can be applied for performing deep space rendezvous and docking procedures, as autonomous systems do not require real-time support from ground. These capabilities require innovative GNC techniques that must be developed and tested.

1.3 Roadmap of Future Space Robotics

In early 2011, the National Aeronautics and Space Administration's (NASA) Office of the Chief Technologist defined a set of roadmaps for innovative concepts and cross-cutting technologies required by NASA for the 2011-2021 decade and beyond. This report considered OOS and space debris removal missions as high priority space robotics challenges due to the increasing number of end-of-life and uncontrolled satellites in LEO. The proposed roadmaps indicate the need of innovative GNC technologies in order to enable autonomous rendezvous, proximity operations and docking to either cooperative targets or space debris. This capability would ensure high mission frequency, robustness, reliability and ability to service a large variety of clients.

The technical details for next generation GNC systems that enable autonomous proximity operations for noncooperative spacecraft have been consequently defined in Ref. [8]. In particular, these new systems must present the following key aspects:

1. *State estimation* of the *noncooperative* target object.
2. *Optimality*: For any feasible trajectory, optimal or near-optimal solution, which minimizes the fuel or energy consumption, is desired.
3. *Robustness* against orbital perturbations and sensor noise.
4. *Real-time implementable*: Algorithms must be executable onboard and in real-time.

As regard to the first requirement, the chaser spacecraft must be equipped with dedicated rendezvous sensors and estimation software capable to determine, in real-time, current distance, attitude and velocities of the noncooperative target. The reason for this requirement is that the offline estimated motion of the target vehicle is affected by high uncertainties and, therefore, the approach trajectory cannot be planned before the launch of the mission. The maneuver must be therefore planned onboard using the data from the rendezvous sensors. Furthermore, the chaser system must be able to provide quick response to mission contingencies. For this purpose, the precise knowledge of the current system state is required during the maneuver. Knowing the current state of target, the guidance software must generate onboard a safe and feasible approach trajectory. In order to minimize the usage of onboard resources, these trajectories should be optimized. Finally, the GNC system must ensure robustness to orbital perturbations, sensor noise and uncertainties on the model of the noncooperative target.

Following this roadmap, the conceptual architecture of a new generation GNC system for autonomous noncooperative proximity operations has been defined in Figure 1.1 by adapting state-of-the-art GNC schemes to the considered problem. A sensor suite composed by rendezvous sensors, a Global Positioning System (GPS) receiver, an Inertial Measurement Unit (IMU)

and star sensor acquires the flight data regularly during the maneuver. Measurements from the rendezvous sensors are processed in the noncooperative pose estimation block to estimate current attitude and distance of the target. The output of this block is fed into the navigation filter, which estimates target velocities and the remaining parameters of the system state while reducing the measurement noise. The EKF is generally used as filter in state-of-the-art system. The trajectory planning block generates in real-time the approach trajectory using the latest state estimates. The output of this block is used for control generation and might be employed for the pose estimation in the next loop execution. Since GPS, IMU, star sensor and control actuation are standard system in state-of-the-art satellites, the challenge is to develop the software for noncooperative pose estimation, navigation filter and trajectory planning.

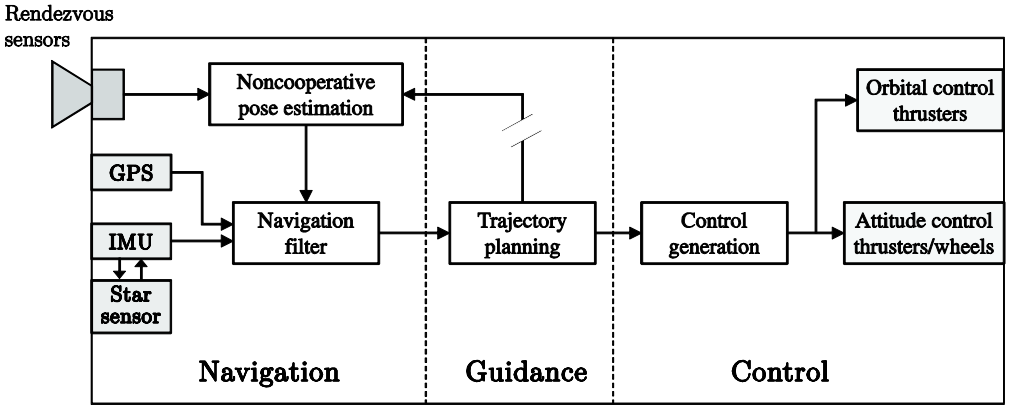

Figure 1.1: Schematic overview of a GNC system for autonomous noncooperative proximity operations. Grey blocks indicate hardware payloads, whereas white blocks represent software packages.

1.4 Research Hypothesis

As of now, proximity operations and docking procedures for OOS were performed in test missions with either cooperative or partially cooperative space targets after overcoming mission contingencies through ground support. Since the nature of the majority of the target objects for OOS and space debris removal missions is noncooperative, a lot of research has been done in recent years to enable proximity operations and docking procedures to such targets. Despite that, innovations in space GNC for OOS were related to guidance and navigation separately and, often, verification campaigns considered simplified test cases such as attitude stabilized target or 2D relative motion. Hence, the hypothesis of this work is:

Working Hypothesis

It is possible to develop a complete guidance, navigation and control system using the state-of-the-art technology to enable autonomous proximity operations for noncooperative space targets.

The desired outcome of performing autonomous missions is the increment of mission frequency, robustness and reliability with respect to the existing ground-in-the-loop guidance methods. Furthermore, the capability to autonomously rendezvous and dock allows retrieving and servicing a variety of space targets being either end-of-life satellites, nonfunctional spacecraft or upper stage rockets. Such technology can be applied also for executing deep space rendezvous and docking procedures, as autonomous systems do not require real-time support from the ground station.

Regarding the mission requirements, the chaser is a mid-size spacecraft designed to rendezvous, capture and deorbit the noncooperative target. The chaser of the future e.Deorbit mission to deorbit Envisat[1] is considered as reference spacecraft. The target satellite is orbiting in LEO, it is not equipped with dedicated visual markers for attitude and distance estimation, and it rotates about the principal axes. The technologies developed in this thesis must enable the autonomous approach to the target from an initial distance of about 10 m, being the beginning for standard proximity operations and docking maneuvers. The approach terminates at few centimeters to the target in order to enable docking or berthing. The GNC system for these proximity operations must ensure safe and successful approaches, meaning that the path constraints for collision avoidance and endpoint conditions must be enforced with high precision under measurement noise. The geometric model of the target spacecraft is assumed available either from the constructor or from shape reconstruction performed during the inspection phase.

The research hypothesis is to be investigated through the development and test of such innovative GNC system. The outcome of this research shall not only prove or refute the hypothesis, but also propose a GNC system for noncooperative proximity operations with the least human supervision.

[1] http://iaassconference2013.space-safety.org/wp-content/uploads/sites/19/2013/06/1200_Biesbroek_Innocenti.pdf

1.5 Research Approach

Following the research hypothesis and mission requirements, this thesis describes the step-by-step development, implementation and tests of an innovative GNC system as in Figure 1.1 to enable autonomous proximity operations with noncooperative space targets. Since IMU, GPS, star tracker and control thrusters are standard payloads in today's mission, this research focuses on the key-enabling software for this type of mission being noncooperative pose estimation, navigation filter and trajectory planning. The choice of the rendezvous sensor, which is strictly related to the pose estimation, is also addressed in this work.

The main research steps and the related outcomes are summarized in Figure 1.2. The first step is the definition of the dynamics model for proximity operations between a controlled chaser spacecraft and an uncontrolled target vehicle. As particular test case, the docking maneuver is considered. Based on this model and on the requirements for the GNC system, the methodology for rapid trajectory generation is chosen among state-of-the-art methods. The selected algorithm is first applied for attitude control only in order to choose the most suitable attitude parameterization. Subsequently, the algorithm for rapid 6 DoF suboptimal proximity operation trajectory generation is developed in details. The particular application to the docking maneuver to an uncontrolled target is considered. Optimality, maneuvering precision, real-time capabilities and robustness to orbital perturbations and sensor noise are evaluated through extensive numerical experiments. The second part of the research focuses on the navigation system. The algorithms, including the navigation filter, for noncooperative pose estimation using data from Lidar, stereo camera and mono camera are developed separately. The performances of the proposed methods are evaluated through an experimental campaign using the Robotic Actuation and On-Orbit Navigation Laboratory (RACOON) laboratory, which provides real-time full DoF hardware simulation of proximity operations under realistic lighting conditions. Test results lead to the choice of the most suitable rendezvous sensor and pose estimation technique that completes the innovative GNC system.

This thesis is organized as follows. Chapter 2 discusses the state of the art of OOS and space debris removal missions, including all the components of the related GNC system. Chapter 3 describes the dynamics model of proximity operations and docking between a controlled chaser spacecraft and an uncontrolled target vehicle. The most suitable trajectory planning algorithm for this maneuver is then chosen. Chapter 4 considers the attitude control problem in order to select the attitude representation that ensures the best trade-off between real-time capabilities and optimization. The detailed implementation of the 6 DoF trajectory planner for proximity operations and docking maneuvers is addressed in Section 5 along with the numerical experiments for performance evaluation. Section 6 describes in details the algorithms for noncooperative pose estimation using data from Lidar, stereo

camera and mono camera. The choice of the rendezvous sensor and pose estimation algorithm is made in Chapter 7, where the experimental campaign on the navigation system is discussed. Chapter 8 concludes the thesis by answering to the research hypothesis and addressing future work.

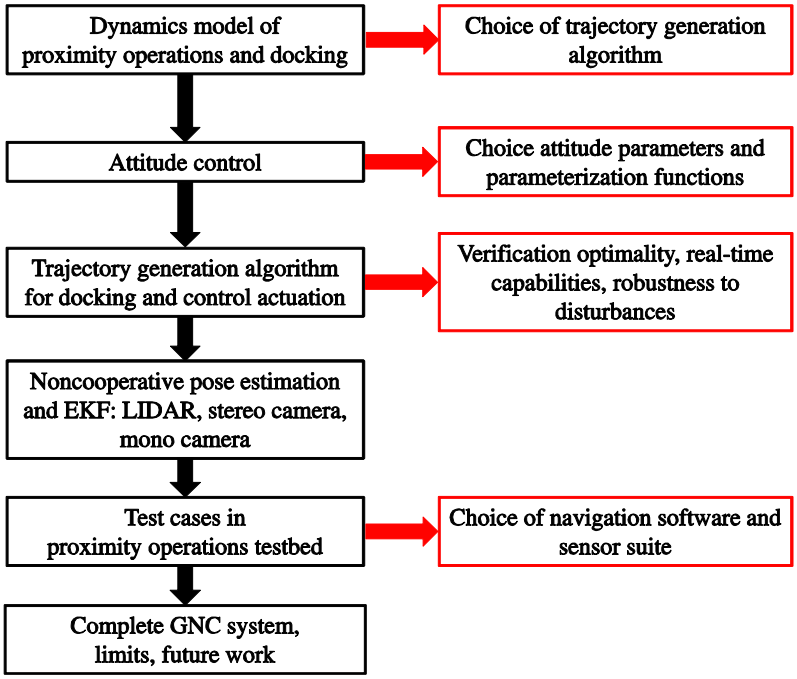

Figure 1.2: Conceptual representation of the research approach.

2 State of the Art

2.1 OOS Test Missions

At the moment, scientific and commercial satellites were repaired, upgraded or rescued only by manned OOS missions aboard the Space Shuttle [9, 10, 11, 12]. Examples of such missions are the repair of the Solar Max satellite (STS-41-C in 1984) and the upgrades of the Hubble telescope (STS-61, STS-82, STS-103, STS-109, STS-125 between 1993 and 2009). To achieve the flexibility required to approach nonfunctional spacecraft, the Space Shuttle was manually commanded by the crew. The capture of space targets was performed either by astronauts through extravehicular activities or by the remote manipulator system Canadarm, which was controlled by a member of the crew from the Space Shuttle's aft flight deck. Finally, repairs and upgrades of satellites have been always performed by astronauts through extravehicular activities. Due to the high cost of such servicing missions, high risks for astronauts, and today's unavailability of Space Shuttle-like spacecraft, new technologies are required to perform safe and cost-friendly OOS again in the near future. Moreover, spacecraft capable to capture and deorbit noncooperative space targets without crew involvement are needed for space debris removal missions. Therefore, in the last two decades, research has been focusing on technology development to enable unmanned OOS and debris removal missions.

Due to the advantages of full autonomous systems, spacecraft rendezvous technologies for autonomous inspection and docking have been recently tested in several demonstration missions. The first experiment was the Engineering Test Satellite No. 7 (ETS-VII), which was launched by the Japanese space agency in 1997 [13]. Two attempts of autonomous proximity operations and docking to a cooperative and attitude-stabilized target were performed. In the first maneuver, the chaser successfully docked to the target from a relative distance of 2 m at a relative speed of 1 cm/s after a short formation flight. The second maneuver, the onboard rendezvous software aborted the approach due to a thrust failure. After ground investigation and software upgrade, the docking was successfully performed. The Experimental Satellite System 10 (XSS-10) and XSS-11, commissioned by the U.S. Air Force Research Laboratory, demonstrated the capability of a microsatellite to safely and autonomously rendezvous and inspect multiple noncooperative space objects [14]. In particular, XSS-10 was launched in 2003 and it demonstrated semi-autonomous inspection from a distance of 35 m to the target. Two years later, XSS-11 performed proximity operations with minimal

ground support. In 2005, NASA's Demonstration of Autonomous Rendezvous Test missions (DART) was launched to perform autonomous close proximity operations up to 5 m to the target vehicle [15, 16]. This mission was intended to validate the newly developed navigation system and ground simulation facilities using flight data. During the first approach to the target, an anomaly in the navigation software occurred. The mission was therefore aborted. Mishap investigations demonstrated that the estimation of the potential effects of nonfunctional subsystems on the system functionality was not sufficient. This lack of knowledge was mainly due to the limited mathematical models available to perform pre-launch valuation tests. The Orbital Express program, developed by the U.S.'s Defense Advanced Research Projects Agency (DARPA), successfully performed for the first time several fully autonomous approaches and captures of a cooperative target in 2007 [4, 17]. Furthermore, autonomous refueling and OOS interfaces were also tested. Notably, in this mission the target spacecraft attitude was completely stabilized. The latest demonstration mission was PRISMA, which was launched by the Swedish National Space Board together with the German Aerospace Center (DLR) in 2010. Here the servicer successfully performed semi-autonomous rendezvous maneuver from 30 km to the final hold point at 3 km of relative distance to the noncooperative target [18]. Also in this mission, the target satellite was attitude-stabilized.

2.2 Onboard Trajectory Planning

The role of the guidance system is to compute onboard a feasible trajectory from the actual configuration of the chaser spacecraft to the desired final configuration of the same vehicle. The computation of the trajectory is executed in real-time by the path planner algorithm. For the considered case of the 6 DoF docking maneuver, both translational and rotational trajectories of the chaser must be provided by the guidance. Moreover, the capability to generate a feasible docking trajectory onboard is crucial for capturing noncooperative targets such as space debris. As already discussed, the kinematic characteristics of a noncooperative client might be known with low precision before the launch of the mission [5] and, therefore, the docking trajectory cannot be determined offline. Instead, the chaser spacecraft must estimate a feasible docking trajectory onboard using the data from the navigation system, which provides the precise estimation of the target motion. Furthermore, the path planning algorithm must be able to enforce path constraints such as sensor pointing, collision avoidance and docking enabling conditions. Since the fuel is a crucial factor for the lifetime of a satellite, fuel-optimal trajectories are desired.

This section of the thesis discusses the existing optimal guidance strategies that have been proposed for autonomous rendezvous and docking to an uncontrolled target in the last decade.

2.2.1 Glideslope Algorithm

The glideslope algorithm is a hybrid method between a path planner and a velocity control algorithm [19, 20]. It simply calculates, with a certain frequency, the velocity corrections required to follow a predefined linear velocity profile. The glideslope algorithm was employed in the guidance systems of both Apollo and Space Shuttle for straight docking maneuvers along the docking axis of the target vehicle. The method is fast and robust, since no optimization problems are solved. Therefore, no iterative solvers are employed. For this reason, optimality of the trajectory is not guaranteed and the method cannot handle path constraints such as collision avoidance and sensor pointing. Recently, Nolet [21] applied the glideslope method to perform autonomous docking to a cooperative tumbling target using the SPHERES system.

2.2.2 Mixed-Integer Linear Programming

The Mixed-Integer Linear Programming (MILP)-based guidance algorithm was developed at the Aerospace Control Laboratory of the Massachusetts Institute of Technology [22, 23]. The linear dynamics model of the system is discretized and the future state variables are expressed as linear function of the controls, which are assumed constant in each time step. The fuel expenditure is minimized by optimizing the values of the controls in each interval. The MILP-based path planner has two important strengths: first, it provides fuel-optimal trajectories and second, path constraints can be enforced, including plume impingement and passive safety under thruster failures toward the end of the trajectory [24]. However, as demonstrated in previous applications, this technique cannot be applied to nonlinear dynamics models and it is not suitable for onboard docking trajectory generation due to the high computational time required to optimize the controls.

2.2.3 Model Predictive Control

The Model Predictive Control (MPC) is a receding horizon control strategy capable to solve constrained trajectory optimization problems [25]. The MPC employs the dynamics model of the system to sample the state vector for a relatively short time horizon in the future starting from the current instant. The discretized controls and states are subsequently used to formulate a constrained optimization problem, which is solved using numerical techniques. The MPC must be executed in closed loop to optimize the trajectory of the entire maneuver. The advantage of this approach is that the optimization problem formulated using MPC can be solved rapidly since only the solution over a finite time horizon is calculated. In case of linear systems, the optimization problem is convex. Therefore, most applications employ a linearized dynamics model of the system to ensure fast trajectory optimization. In the case of nonlinear dynamics, the optimization problem

is not convex anymore and therefore more advance iterative solvers are required.

The MPC technique has been applied by several authors to generate feasible docking trajectories [26, 27, 28]. Notably, Di Cairano [28] applied a Linear-Quadratic MPC to generate online suboptimal planar trajectories to approach a tumbling object. The method showed robustness even in presence of obstacles along the path. Virgili-Llop et al. tested a Linear-Quadratic MPC on the air-bearing-based testbed of the Naval Postgraduate School [29]. Successful docking maneuvers with a cooperative target spacecraft were achieved. However, the dynamics model of the system was based on the classical mechanics defined in an inertial system rather than the Clohessy-Wiltshire (C-W) model. Recently, Wolf implemented a Linear-Quadratic MPC for the 6 DoF docking maneuver in C-W dynamics [30]. In this case, the method showed two important limitations. First, if the target is at a rate of about 3 deg/s, successful docking was achieved by employing long horizons and dense discretization. This caused large memory allocation for trajectory storage and long computational time for solving the optimization problem generated by MPC. Notably these aspects limit the applicability of the method in onboard CPUs. Second, the method's parameters such as discretization nodes and horizons required a specific tuning for each tested scenario. A set of parameters valid for several scenarios was not found.

2.2.4 Rapidly Exploring Random Tree

The Rapidly-exploring Random Tree (RRT) algorithm was introduced by LaValle and Kuffner in 2001 as path planner for terrestrial robots [31]. Starting from the current position of the vehicle, the algorithm randomly builds a space-filling tree of the 3D space. The vehicle trajectory is obtained using the consecutive nodes of the randomized data structure that minimize a certain performance index while respecting path constraints [32, 33]. Enforcement of further path constraints or boundary conditions such as endpoint velocity is possible. Although RRT showed robust applicability in terrestrial robotics applications, relatively little application to spacecraft control systems has been seen. Philipps applied this technique for the 6 DoF rendezvous maneuver [34]. Results showed that the generation of the random tree database was computationally expensive, but this database was used for the entire maneuver, allowing rapid trajectory update. However, RRT might suffer for the following disadvantages if applied to rendezvous and docking: first, large memory allocation is required for storing the space-filling trees of attitude and position; second, enforcement of docking enabling conditions on final position, attitude, contact velocity and angular velocity is not straightforward since the trees are randomized; third, the application of the method for attitude control hasn't been tested yet.

2.2.5 Inverse Dynamics in the Virtual Domain

The Inverse Dynamics in the Virtual Domain (IDVD) method was introduced by Yakimenko in 2001 as suboptimal guidance for aircrafts [35]. The key feature of this approach is that the components of the trajectory of the vehicle are imposed using a set of interpolating functions defined in an abstract (also called virtual) domain of time [35, 36]. The boundary conditions on the trajectory are enforced by linking some of the functions' polynomial coefficients to the known endpoint conditions such as position and velocity. By inverting the dynamics model of the system, state variable, controls, path constraints and performance index are expressed as nonlinear functions in the virtual domain of the parameterized trajectory and its derivatives. The performance index of the trajectory is then minimized by optimizing the free polynomial coefficients of the initial set of interpolating functions.

The first application of IDVD to the rendezvous and docking problem is due to Bevilaqua in 2009 [37]. Here the method was applied in open-loop to the relative translational motion in C-W dynamics using a predefined bang-bang shape of the controls. The switching time nodes of the controls, the signs of the controls and the polynomial coefficients were the varied parameters of the problem. The closed-loop implementation of IDVD in C-W dynamics is due to Ciarcia' [38, 39]. In this case, no predefined control shapes were imposed. Demonstrations in simulated C-W dynamics and in the air-bearing-based testbed of the Naval Postgraduate School were performed. Recently, Boyarko applied the IDVD method for rapid suboptimal 6 DoF docking trajectory generation [40]. Although this method showed robustness to numerous initial conditions and rotation rates of the target spacecraft, the high computational time required to generate the trajectory allows only open-loop trajectory optimization followed by closed-loop trajectory reshape using the current data from the navigation system and the optimal polynomial coefficients. Kobilarov and Pellegrino [41] formulated a trajectory planning algorithm for autonomous docking between self-assembling nanosatellites. A simplified version of the proposed method was validated on an air-bearing-based testbed with cooperative docking maneuvers. Similarly as Virgili-Lloop and Ciarcia', the dynamics model of the system was based on classical dynamics defined in an inertial system rather than the C-W model. Recently, Wilde et al. tested IDVD on noncooperative planar maneuvers on the air-bearing-based testbed of the Naval Postgraduate School [42]. The method provided robust performance over a range of target angular velocities from 1 deg/s to 4 deg/s. This research showed some strengths of IDVD. First, the trajectory can be easily stored onboard using the dynamics model of the system and a small set of parameters being the polynomial coefficients. Second, the boundary conditions on the trajectory are automatically imposed by linking some polynomial coefficients to endpoint position and velocity. This is particularly important for docking maneuvers, since error tolerances on the mechanical contact between spacecraft are narrow. Third,

the trajectory can be rapidly reshaped by using the polynomial coefficients previously optimized and the most current data from the navigation system.

2.3 Noncooperative Pose Estimation

In order to enable onboard trajectory generation, current attitude and position, also called pose, of the target spacecraft must be regularly estimated during proximity operations. This estimation process consists of two main steps: data acquisition using the rendezvous sensors and data processing to determine, in real-time, the current pose of the target. The state-of-the-art of rendezvous sensors and of pose estimation algorithms for proximity operations with noncooperative space targets is discussed in this Section.

2.3.1 Rendezvous Sensors

The goal of the rendezvous sensors is to acquire, at a certain frequency during the maneuver, measurements on the target spacecraft in order to enable onboard pose estimation of the same vehicle. For the particular application to proximity operations, rendezvous sensors must operate from several meters up to centimeters to the target surface. In general, sensors are classified into two different categories: active and passive.

The key feature of active sensors is that they emit high intensity signals that illuminate the scene. Subsequently, the echo of the signal received from the target object is captured by the sensor and process. In general, time-of-flight and phase shift of the income signal are used to generate the depth image of the scene. As regard to space proximity operations, Lidar systems have been considered in recent years as primary rendezvous sensor in several researches. This particular sensor employs laser beams in visible or invisible wavelengths to illuminate the target spacecraft. This technique has two main advantages: first, measurements are independent of ambient lighting conditions and second, measured depth is very accurate since Lidars employ high-accuracy clocks to analyze the incoming signal echo. However, these sensors are demanding in terms of power consumption and mass. In addition, they are expensive, have a rather small field-of-view (FOV) [43] and cannot provide depth measurement if located centimeters away to the target spacecraft's surface.

Passive sensors capture the emissions of the target satellite in order to generate the measurement. In particular, cameras acquire the visible or near-infrared natural light reflected by the target objects to create the image of the scene. This type of sensor has been broadly used in robotics applications since it combines low power consumption, limited mass and high operational rates. Regarding space applications, current navigation systems on the Soyuz spacecraft employs mono cameras to generate the image of the target

docking port and to track the visual markers for relative distance and attitude estimation. Moreover, cameras were employed in OOS demonstration missions such as DART, Orbital Express and PRISMA. Furthermore, when the image of the scene is acquired simultaneously from two different points of view, the depth image of the scene can be retrieved by comparing the two images. However, cameras are highly sensitive to illumination conditions. In particular, the presence of the Earth in the background of the image, the high reflectivity of target's surface materials and the absence of sunlight during the orbit pose significant problem in detecting the key feature of a noncooperative spacecraft, since it is not equipped with dedicated visual markers for pose estimation.

2.3.2 Pose Estimation Algorithms

Existing algorithms employed in terrestrial robotic navigation have been recently applied for noncooperative pose estimation of spacecraft. The main challenge to overcome in this application is the noncooperative nature of the target satellite, since it is not equipped with dedicated markers for relative state estimation, it does not communicate with the chaser and it may tumble about the principal axes. The capability to estimate, in real-time, current pose of a noncooperative target would enable OOS and space debris removal mission.

Regarding the application of stereo cameras during rendezvous and docking procedures, Oumer and Panin employed the Iterative Closest Point (ICP) algorithm to estimate the current pose of a nonrotating target spacecraft during close-range maneuvers [44]. In this research, the triangulation technique generated the point cloud of the scene from the stereo image. Accordingly, the ICP routine computed attitude and position of the target by matching the acquired point cloud with a reference 3D model of the same vehicle. An EKF algorithm provided velocity estimates. Tests were performed in the European Proximity Operations Simulator at DLR in Oberpfaffenhofen. Stereo vision-based navigation techniques for noncooperative pose estimation were tested using SPHERES aboard the ISS [45]. In this experiment, both target inertia and relative pose were estimated. However, the target was partially cooperative.

Several research has been also conducted for noncooperative pose estimation using monocular imagery. In particular, D'Amico proposed a pose initialization algorithm based on perceptual grouping of the edges detected in the image and least-squares optimization techniques [46]. Tests were on actual space images acquired during the PRISMA mission. A tracking procedure based on line correspondences was tested in hardware facility by Petit [47].

Most of the proposed algorithms employed the point cloud obtained with Lidar-based sensors, due to the robustness to Sun illumination. The effectiveness of this type of sensors for attitude and position estimation of a spacecraft was demonstrated in both laboratory experiments [48] and tests

during space missions using the ISS as target vehicle [49]. In the first case, the ICP algorithm was employed to determine, in real-time, current attitude and position of the noncooperative target. An EKF was designed to estimate velocities and the inertia tensor of the target vehicle. However, experiments were limited to tracking procedures from a constant distance to the target vehicle. In the second case, tests were performed with an attitude-stabilized target during mid-range rendezvous maneuvers. Also in this case, the ICP algorithm was employed for pose estimation. The pose initialization problem was not addressed.

2.4 Technological Gap

As discussed in the previous sections of this chapter, future robotic systems for OOS and space debris removal missions must perform proximity operations and, eventually, docking to potential noncooperative targets as key-enabling capability. The GNC systems that enable such procedures must be flexible and adaptable to a wide range of missions and for a multitude of different targets, due to the large number of space debris and end-of-life spacecraft that must be serviced in the near future. Moreover, such robotics systems should be able to react quickly to unforeseen events like the unexpected target rotation rates or chaser subsystem failures. This places demanding requirements on robotic systems for noncooperative OOS. As can be observed in Figure 2.1, state-of-the-art technologies tested in demonstration missions were limited to cooperative docking or to noncooperative far-range rendezvous. The capability of approaching and dock to noncooperative targets remains a challenge for space robotics systems. In particular, the technological roadmaps defined by NASA in 2011 indicate autonomous proximity operations and docking as cross-cutting technology to be developed.

In order to reduce this technological gap and enable autonomous proximity operations to noncooperative targets, innovative GNC technologies must be developed and validated through extensive test campaigns. As already discussed in Section 1.3, this new generation of robotic systems shall be able to estimate, in real-time during proximity operations, the motion of the target object without relying on visual markers and in the worst-case scenario of tumbling object. This required dedicated rendezvous sensors along with rapid algorithm for state estimation. Knowing these parameters, the robotic system must be able to compute a feasible and safe approach trajectory in order to execute proximity operations. The algorithm must be fast enough to allow trajectory update during the maneuver to improve the robustness to sensor noise and model uncertainties. The reason for these requirements is that the estimate of the motion of the target vehicle available offline is affected by high uncertainties and, therefore, the approach trajectory cannot be planned before the launch of the mission.

To enable autonomous proximity operations, new methods for rapid trajectory generation and for noncooperative pose estimation have been recently proposed. As discussed in this chapter, most of the trajectory planning methods were tested in simplified 2D scenarios or in air-bearing testbeds under terrestrial dynamics. Few methods were applied to the full DoF maneuver without addressing the problems of imposing either path constraints or precise final conditions. Nevertheless, the high computational time required by these methods may limit the onboard application. Regarding the noncooperative pose estimation, most of the research focused on the development of Lidar-based navigation systems. Test cases considered attitude stabilized targets without addressing the crucial problem of pose initialization, which is required to execute the entire GNC system including the trajectory planning. Stereo vision navigation systems were applied to partially cooperative targets. Clearly, these innovative algorithms must be improved to be onboard executable and to respect the requirements for enabling OOS and space debris removal missions.

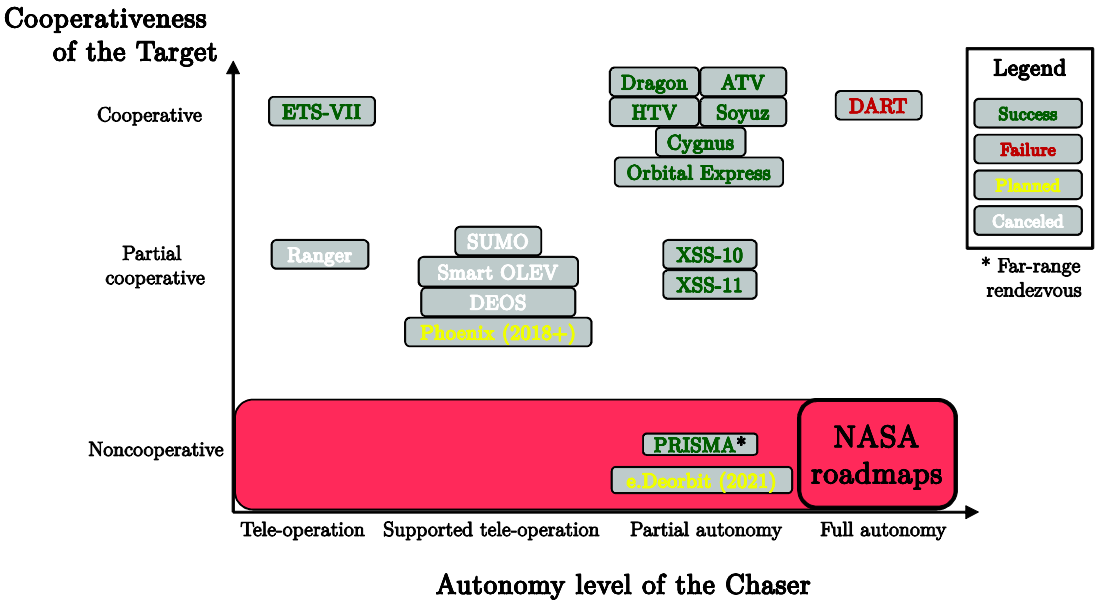

Figure 2.1: Overview of state-of-the-art space missions that execute proximity operation and docking procedures.

3 Dynamics of Proximity Operations

3.1 Dynamics Model

The main challenge for the trajectory generation algorithm for proximity operations for uncontrolled targets is to maneuver the chaser in full DoF using controls that minimize a certain performance index, i.e. time, energy expenditure or fuel consumption, while respecting eventual path constraints. State-of-the-art methods for solving numerically optimal control problems are demanding in terms of memory allocation and computational time and, therefore, not executable onboard. The purpose of this section is to describe in detail the dynamics of a docking maneuver between a controlled chaser spacecraft and an uncontrolled target vehicle according to the work of Ref. [50]. The major contribution with respect to the model proposed in Ref. [50] is the introduction of all the path constraints required to perform safe proximity operations and a more precise definition of the docking enabling conditions. The target is assumed in circular orbit. This hypothesis is motivated by the fact that the majority of space debris is orbiting in LEO and by the large number of end-of-life communication satellite situated in geostationary orbit.

Figure 3.1 shows a sketch of the geometry and coordinate systems of the target and the chaser spacecraft. For each vehicle, two body-fixed Cartesian coordinate systems are defined. The origin of the principal system is located at the center of mass of the spacecraft with the axis directed along the principal axes of inertia. The auxiliary coordinate system is centered at the vehicle's docking point and it is defined in such a way that the z-axis lies along the vehicle docking axis (outward direction for the chaser and inward direction for the target as in Figure 3.1) and the x-axis points to the top of the spacecraft. The y-axis completes the right-hand rule. Figure 3.1 also illustrates the orbiting Hill's coordinate system H: the origin placed at the center of mass of the target spacecraft, the x-axis points toward the zenith, the y-axis is directed along the velocity vector of the target spacecraft and the z-axis points the orbit normal.

The optimal control problem for this maneuver is then formulated.

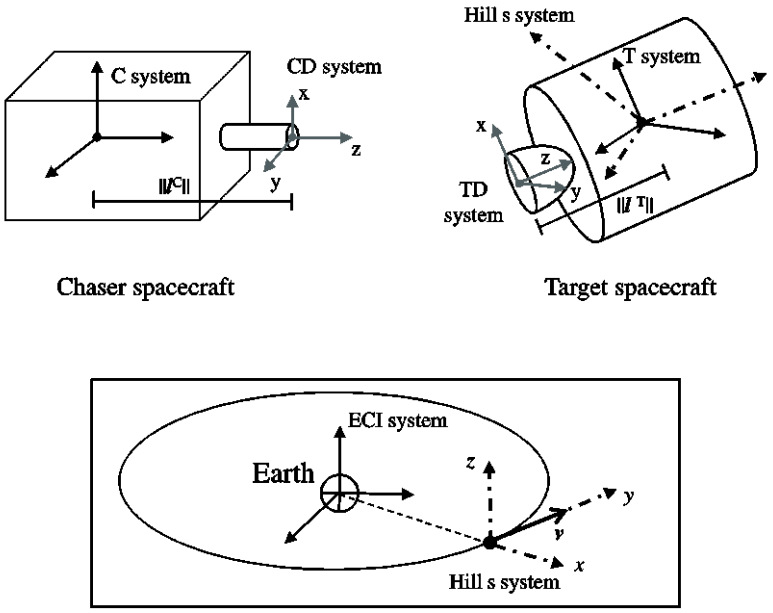

Figure 3.1: Spacecraft geometry and coordinate systems.

For close range proximity maneuvers between two spacecraft, the translational motion of the chaser spacecraft center of mass relative to the target center of mass is described in the Hill's coordinate system by the Clohessy-Wiltshire's (C-W) differential equations [51]:

$$\begin{aligned} \ddot{x} &= 2\Omega\dot{y} + 3\Omega^2 x + \frac{F_x^{\mathrm{H}}}{m_{\mathrm{C}}} \\ \ddot{y} &= -2\Omega\dot{x} + \frac{F_y^{\mathrm{H}}}{m_{\mathrm{C}}} \\ \ddot{z} &= -\Omega^2 z + \frac{F_z^{\mathrm{H}}}{m_{\mathrm{C}}} \end{aligned} \quad . \tag{3.1}$$

The dynamics of the rotational motion of the chaser spacecraft is governed by Euler's equation, which in scalar form and referenced to the chaser principal coordinate system C can be expressed as [52]

$$\begin{aligned} \dot{\omega}_x^{\mathrm{C}} &= \frac{(I_{yy}^{\mathrm{C}} - I_{zz}^{\mathrm{C}})\omega_z^{\mathrm{C}}\omega_y^{\mathrm{C}} + T_x^{\mathrm{C}}}{I_{xx}^{\mathrm{C}}} \\ \dot{\omega}_y^{\mathrm{C}} &= \frac{(I_{zz}^{\mathrm{C}} - I_{xx}^{\mathrm{C}})\omega_z^{\mathrm{C}}\omega_x^{\mathrm{C}} + T_y^{\mathrm{C}}}{I_{yy}^{\mathrm{C}}} \\ \dot{\omega}_z^{\mathrm{C}} &= \frac{(I_{xx}^{\mathrm{C}} - I_{yy}^{\mathrm{C}})\omega_y^{\mathrm{C}}\omega_x^{\mathrm{C}} + T_z^{\mathrm{C}}}{I_{zz}^{\mathrm{C}}} \end{aligned} \quad . \tag{3.2}$$

Similarly, the rotational dynamics of the target spacecraft (assumed in unforced motion and without disturbances) is given by

$$\begin{aligned}\dot{\omega}_x^{\mathrm{T}} &= \frac{(I_{yy}^{\mathrm{T}} - I_{zz}^{\mathrm{T}})\omega_z^{\mathrm{T}}\omega_y^{\mathrm{T}}}{I_{xx}^{\mathrm{T}}} \\ \dot{\omega}_y^{\mathrm{T}} &= \frac{(I_{zz}^{\mathrm{T}} - I_{xx}^{\mathrm{T}})\omega_z^{\mathrm{T}}\omega_x^{\mathrm{T}}}{I_{yy}^{\mathrm{T}}} \\ \dot{\omega}_z^{\mathrm{T}} &= \frac{(I_{xx}^{\mathrm{T}} - I_{yy}^{\mathrm{T}})\omega_y^{\mathrm{T}}\omega_x^{\mathrm{T}}}{I_{zz}^{\mathrm{T}}}\end{aligned} . \tag{3.3}$$

The attitude of a spacecraft can be represented by numerous parameters, for example Euler angles, quaternions, Rodrigues parameters, modified Rodrigues parameters (MRP), Cayley-Klein parameters and stereographic complex rotation variables [53, 54, 55]. However, quaternions are commonly used due to the absence of singularities. Therefore, the orientation of the chaser spacecraft (coordinate system C) with respect to the Hill's system can be express in terms of quaternion $\boldsymbol{q}^{\mathrm{C}} = [q_1{}^{\mathrm{C}}, q_2{}^{\mathrm{C}}, q_3{}^{\mathrm{C}}, q_4{}^{\mathrm{C}}]$ according to the following kinematic equation [53]:

$$\begin{bmatrix}\dot{q}_1^{\mathrm{C}} \\ \dot{q}_2^{\mathrm{C}} \\ \dot{q}_3^{\mathrm{C}} \\ \dot{q}_4^{\mathrm{C}}\end{bmatrix} = \frac{1}{2}\begin{bmatrix} 0 & {}^{\mathrm{H}}\omega_z^{\mathrm{C}} & -{}^{\mathrm{H}}\omega_y^{\mathrm{C}} & {}^{\mathrm{H}}\omega_x^{\mathrm{C}} \\ -{}^{\mathrm{H}}\omega_z^{\mathrm{C}} & 0 & {}^{\mathrm{H}}\omega_x^{\mathrm{C}} & {}^{\mathrm{H}}\omega_y^{\mathrm{C}} \\ {}^{\mathrm{H}}\omega_y^{\mathrm{C}} & -{}^{\mathrm{H}}\omega_x^{\mathrm{C}} & 0 & {}^{\mathrm{H}}\omega_z^{\mathrm{C}} \\ -{}^{\mathrm{H}}\omega_x^{\mathrm{C}} & -{}^{\mathrm{H}}\omega_y^{\mathrm{C}} & -{}^{\mathrm{H}}\omega_z^{\mathrm{C}} & 0 \end{bmatrix}\begin{bmatrix} q_1^{\mathrm{C}} \\ q_2^{\mathrm{C}} \\ q_3^{\mathrm{C}} \\ q_4^{\mathrm{C}}\end{bmatrix} . \tag{3.4}$$

In the previous equation, the components ${}^{\mathrm{H}}\omega_x{}^{\mathrm{C}}$, ${}^{\mathrm{H}}\omega_y{}^{\mathrm{C}}$, ${}^{\mathrm{H}}\omega_z{}^{\mathrm{C}}$ of the chaser spacecraft angular velocity with respect to the Hill's coordinate system and expressed in the chaser principal system C are obtained from $\boldsymbol{\omega}^{\mathrm{C}}$ with

$$\begin{bmatrix}{}^{\mathrm{H}}\omega_x^{\mathrm{C}} \\ {}^{\mathrm{H}}\omega_y^{\mathrm{C}} \\ {}^{\mathrm{H}}\omega_z^{\mathrm{C}}\end{bmatrix} = \begin{bmatrix}\omega_x^{\mathrm{C}} \\ \omega_y^{\mathrm{C}} \\ \omega_z^{\mathrm{C}}\end{bmatrix} - \boldsymbol{R}_{\mathrm{H}}^{\mathrm{C}}\begin{bmatrix}0 \\ 0 \\ \Omega\end{bmatrix}, \tag{3.5}$$

where

$$\boldsymbol{R}_{\mathrm{H}}^{\mathrm{C}} = \begin{bmatrix} q_4{}^{\mathrm{C}\,2} + q_1{}^{\mathrm{C}\,2} - q_2{}^{\mathrm{C}\,2} - q_3{}^{\mathrm{C}\,2} & 2\left(q_1^{\mathrm{C}}q_2^{\mathrm{C}} + q_3^{\mathrm{C}}q_4^{\mathrm{C}}\right) & 2\left(q_1^{\mathrm{C}}q_3^{\mathrm{C}} - q_2^{\mathrm{C}}q_4^{\mathrm{C}}\right) \\ 2\left(q_1^{\mathrm{C}}q_2^{\mathrm{C}} - q_3^{\mathrm{C}}q_4^{\mathrm{C}}\right) & q_4{}^{\mathrm{C}\,2} - q_1{}^{\mathrm{C}\,2} + q_2{}^{\mathrm{C}\,2} - q_3{}^{\mathrm{C}\,2} & 2\left(q_2^{\mathrm{C}}q_3^{\mathrm{C}} + q_1^{\mathrm{C}}q_4^{\mathrm{C}}\right) \\ 2\left(q_1^{\mathrm{C}}q_3^{\mathrm{C}} + q_2^{\mathrm{T}}q_4^{\mathrm{C}}\right) & 2\left(q_2^{\mathrm{C}}q_3^{\mathrm{C}} - q_1^{\mathrm{C}}q_4^{\mathrm{C}}\right) & q_4{}^{\mathrm{C}\,2} - q_1{}^{\mathrm{C}\,2} - q_2{}^{\mathrm{C}\,2} + q_3{}^{\mathrm{C}\,2} \end{bmatrix} \tag{3.6}$$

is the direction cosine matrix from the Hill's system H to the chaser principal system C. Similarly, the attitude of the target principal axes with respect to the Hill's coordinate system in terms of quaternion $\boldsymbol{q}^{\mathrm{T}} = [q_1{}^{\mathrm{T}}, q_2{}^{\mathrm{T}}, q_3{}^{\mathrm{T}}, q_4{}^{\mathrm{T}}]$ is given by

$$\begin{bmatrix} \dot{q}_1^{\mathrm{T}} \\ \dot{q}_2^{\mathrm{T}} \\ \dot{q}_3^{\mathrm{T}} \\ \dot{q}_4^{\mathrm{T}} \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 0 & {}^{\mathrm{H}}\omega_z^{\mathrm{T}} & -{}^{\mathrm{H}}\omega_y^{\mathrm{T}} & {}^{\mathrm{H}}\omega_x^{\mathrm{T}} \\ -{}^{\mathrm{H}}\omega_z^{\mathrm{T}} & 0 & {}^{\mathrm{H}}\omega_x^{\mathrm{T}} & {}^{\mathrm{H}}\omega_y^{\mathrm{T}} \\ {}^{\mathrm{H}}\omega_y^{\mathrm{T}} & -{}^{\mathrm{H}}\omega_x^{\mathrm{T}} & 0 & {}^{\mathrm{H}}\omega_z^{\mathrm{T}} \\ -{}^{\mathrm{H}}\omega_x^{\mathrm{T}} & -{}^{\mathrm{H}}\omega_y^{\mathrm{T}} & -{}^{\mathrm{H}}\omega_z^{\mathrm{T}} & 0 \end{bmatrix} \begin{bmatrix} q_1^{\mathrm{T}} \\ q_2^{\mathrm{T}} \\ q_3^{\mathrm{T}} \\ q_4^{\mathrm{T}} \end{bmatrix}, \tag{3.7}$$

where the components ${}^{\mathrm{H}}\omega_x{}^{\mathrm{T}}$, ${}^{\mathrm{H}}\omega_y{}^{\mathrm{T}}$, ${}^{\mathrm{H}}\omega_z{}^{\mathrm{T}}$ are obtained from Eqs. (3.5, 3.6) applied to the target spacecraft.

As can be observed from the dynamics model of the docking maneuver, the chaser must be controlled in both translation (C-W relative dynamics) and rotation at the same time. Furthermore, rotational and translational dynamics are decoupled in Eqs. (3.1–3.7) and, therefore, the docking procedure can be thought as a relative translational motion combined with a reorientation maneuver. However, attitude and relative distance to the target are coupled in the docking enabling conditions and path constraints as discussed in the following.

3.2 Enabling Conditions and Path Constraints

To achieve a smooth physical contact between target and chaser docking points, it is necessary to enforce end conditions on both relative position and velocity [56]. More precisely, the chaser docking point must grapple the target docking point with a certain contact velocity $\boldsymbol{v}_{\mathrm{con}}^{\mathrm{H}}$ at the end of the maneuver. Furthermore, the chaser's auxiliary coordinate system CD must match the target's auxiliary system TD orientation and angular velocity. This ensures alignment of the docking axes and synchronization between the rotational motions of the docking points. Since contact shocks can be avoided by enforcement of suitable end conditions on just relative position and relative velocity, no final conditions on relative acceleration are considered in this paper. Accordingly, the docking-enabling conditions are expressed in terms of state variables as:

$$\boldsymbol{R}_{\mathrm{T}}^{\mathrm{H}}\boldsymbol{l}^{\mathrm{T}} - \left(\boldsymbol{R}_{\mathrm{C}}^{\mathrm{H}}\boldsymbol{l}^{\mathrm{C}} + \begin{bmatrix} x_{\mathrm{f}} \\ y_{\mathrm{f}} \\ z_{\mathrm{f}} \end{bmatrix} \right) = \boldsymbol{0}\,, \tag{3.8}$$

$$\left(\boldsymbol{R}_{\mathrm{T}}^{\mathrm{H}}\boldsymbol{\omega}_{\mathrm{f}}^{\mathrm{T}} - \begin{bmatrix} 0 \\ 0 \\ \Omega \end{bmatrix} \right) \times \boldsymbol{R}_{\mathrm{T}}^{\mathrm{H}}\boldsymbol{l}^{\mathrm{T}} - \left\{ \left(\boldsymbol{R}_{\mathrm{C}}^{\mathrm{H}}\boldsymbol{\omega}_{\mathrm{f}}^{\mathrm{C}} - \begin{bmatrix} 0 \\ 0 \\ \Omega \end{bmatrix} \right) \times \boldsymbol{R}_{\mathrm{C}}^{\mathrm{H}}\boldsymbol{l}^{\mathrm{C}} + \begin{bmatrix} \dot{x}_{\mathrm{f}} \\ \dot{y}_{\mathrm{f}} \\ \dot{z}_{\mathrm{f}} \end{bmatrix} \right\} = \boldsymbol{v}_{\mathrm{con}}^{\mathrm{H}}, \tag{3.9}$$

$$\boldsymbol{R}_{\mathrm{TD}}^{\mathrm{H}} = \boldsymbol{R}_{\mathrm{CD}}^{\mathrm{H}}, \tag{3.10}$$

$$\boldsymbol{R}_{\mathrm{T}}^{\mathrm{TD\,H}}\boldsymbol{\omega}_{\mathrm{f}}^{\mathrm{T}} = \boldsymbol{R}_{\mathrm{C}}^{\mathrm{CD\,H}}\boldsymbol{\omega}_{\mathrm{f}}^{\mathrm{C}}. \tag{3.11}$$

In the previous expressions, the matrices $\boldsymbol{R}_{\mathrm{C}}^{\mathrm{H}}$ and $\boldsymbol{R}_{\mathrm{T}}^{\mathrm{H}}$ are calculated with Eq. (3.6) using the final chaser and target attitude, being $\boldsymbol{q}_{\mathrm{f}}^{\mathrm{C}}$ and $\boldsymbol{q}_{\mathrm{f}}^{\mathrm{T}}$ respectively.

During a docking maneuver to either a cooperative or a noncooperative target, the chaser spacecraft must follow safe and feasible trajectories until mechanical contact with the target spacecraft [8, 56]. For this reason, the following constraints are imposed to the chaser trajectory:

1) *Sensor field-of-view constraint*: The chaser spacecraft is equipped with rendezvous sensors to detect the relative distance and attitude of the target spacecraft. To allow the pose estimation at any instant of time during the docking procedure, the chaser must maneuver in such a way to keep the target spacecraft always inside the sensor FOV. As can be observed in Figure 3.2, the rendezvous sensor is mounted along the above the chaser's docking interface, whereas the sensor FOV is defined by the boresight direction $\boldsymbol{b}^{\mathrm{C}}$ and by the half-cone angle α_{FOV}. The sensor FOV constraint can be expressed mathematically as:

$$\left(-\boldsymbol{R}_{\mathrm{H}}^{\mathrm{C}}\begin{bmatrix}x\\y\\z\end{bmatrix}-\boldsymbol{s}^{\mathrm{C}}\right)\cdot\boldsymbol{b}^{\mathrm{C}} \geq \left\|-\boldsymbol{R}_{\mathrm{H}}^{\mathrm{C}}\begin{bmatrix}x\\y\\z\end{bmatrix}-\boldsymbol{s}^{\mathrm{C}}\right\|\cos\left(\alpha_{\mathrm{FOV}}\right), \tag{3.12}$$

where the vector $-\left(\boldsymbol{R}_{\mathrm{H}}^{\mathrm{C}}[x \quad y \quad z]^{\mathrm{T}}+\boldsymbol{s}^{\mathrm{C}}\right)$ is the position of the target center of mass with respect to the sensor expressed in the chaser principal coordinate system C. Therefore, Eq. (3.12) ensures that the angle defined by the sensor boresight direction and the vector $-\left(\boldsymbol{R}_{\mathrm{H}}^{\mathrm{C}}[x \quad y \quad z]^{\mathrm{T}}+\boldsymbol{s}^{\mathrm{C}}\right)$ remains lower than the maximum allowable angle α_{FOV}.

2) *Collision avoidance*: Any accidental collision between spacecraft must be avoided [8, 24]. Consequently, the distance between the chaser docking point and the target surface must be larger than a minimum distance r_{safe}, which represents the radius of the "keep-out" zone [8]. This constraint can be expressed as

$$\left\|\begin{bmatrix}x\\y\\z\end{bmatrix}+\boldsymbol{R}_{\mathrm{C}}^{\mathrm{H}}\boldsymbol{l}^{\mathrm{C}}\right\| \geq r_{\mathrm{safe}}. \tag{3.13}$$

Due to the constraint in Eq. (3.12), the chaser docking point is oriented toward the target spacecraft at any instant during the maneuver, moreover, it is the part of the chaser having the minimum distance from the target

surface (see also Figure 3.2). By imposing the collision avoidance at the chaser docking point, we enforce this constraint to any other parts of the chaser spacecraft.

3) *Handling controls upper and lower bounds:* Due to design constraints, thrusters and reaction wheels have finite upper (and lower) bounds on the available force and torque. These constraints are enforced in terms of control components as

$$\begin{aligned} \boldsymbol{T}_{min}^{\mathrm{C}} \le \boldsymbol{T}^{\mathrm{C}} \le \boldsymbol{T}_{max}^{\mathrm{C}} \\ \boldsymbol{F}_{min}^{\mathrm{C}} \le \boldsymbol{F}^{\mathrm{C}} \le \boldsymbol{F}_{max}^{\mathrm{C}} \end{aligned} . \tag{3.14}$$

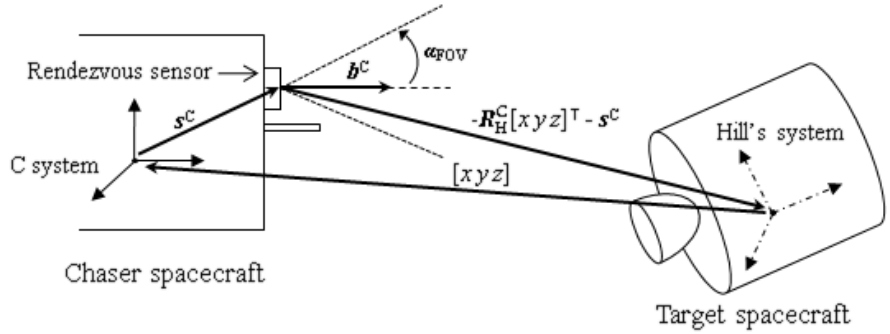

Figure 3.2: Schematic representation of the sensor field-of-view constraint.

3.3 Optimal Control Problem Formulation

In this work, minimum energy maneuvers are considered, due to the need to minimize propellant usage [8]; therefore, the performance index is defined as

$$J = \frac{1}{2}\int_0^{t_{\mathrm{f}}} \left(\left(\frac{F_x^{\mathrm{C}}}{m_{\mathrm{C}}}\right)^2 + \left(\frac{F_y^{\mathrm{C}}}{m_{\mathrm{C}}}\right)^2 + \left(\frac{F_z^{\mathrm{C}}}{m_{\mathrm{C}}}\right)^2 + \left(\frac{T_x^{\mathrm{C}}}{L_{\mathrm{eq}} m_{\mathrm{C}}}\right)^2 + \left(\frac{T_y^{\mathrm{C}}}{L_{\mathrm{eq}} m_{\mathrm{C}}}\right)^2 + \left(\frac{T_z^{\mathrm{C}}}{L_{\mathrm{eq}} m_{\mathrm{C}}}\right)^2 \right) \mathrm{d}t \,. \tag{3.15}$$

In the previous equation L_{eq} is a length parameter used to convert the control torque into an equivalent force. In this thesis, we consider L_{eq} a given parameter related to the size of the chaser spacecraft. However, one may also consider L_{eq} a weighting factor between force and torque, and, therefore, a varied parameter to be optimized. Notably in this thesis, the optimization of the maneuver energy is chosen over the propellant mass since the resulting performance index function, in Eq. (3.15), has the properties of being continuous, quadratic and differentiable. These aspects are indeed important for enabling rapid optimization using a nonlinear quadratic solver and, therefore, allow onboard execution of the trajectory optimization algorithm.

The optimal control problem for the docking maneuver can be stated as follows: find controls histories $\boldsymbol{T}^{\mathrm{C}}$ and $\boldsymbol{F}^{\mathrm{C}}$ that minimize J and, at the same time, transfer the system described by Eqs. (3.1–3.7) from an initial state

$$\boldsymbol{x}_0 = \begin{bmatrix} x_0 & y_0 & z_0 & \dot{x}_0 & \dot{y}_0 & \dot{z}_0 & \boldsymbol{\omega}_0^{\mathrm{C}} & \boldsymbol{q}_0^{\mathrm{C}} & \boldsymbol{\omega}_0^{\mathrm{T}} & \boldsymbol{q}_0^{\mathrm{T}} \end{bmatrix}$$

to a final state

$$\boldsymbol{x}_{\mathrm{f}} = \begin{bmatrix} x_{\mathrm{f}} & y_{\mathrm{f}} & z_{\mathrm{f}} & \dot{x}_{\mathrm{f}} & \dot{y}_{\mathrm{f}} & \dot{z}_{\mathrm{f}} & \boldsymbol{\omega}_{\mathrm{f}}^{\mathrm{C}} & \boldsymbol{q}_{\mathrm{f}}^{\mathrm{C}} & \boldsymbol{\omega}_{\mathrm{f}}^{\mathrm{T}} & \boldsymbol{q}_{\mathrm{f}}^{\mathrm{T}} \end{bmatrix}$$

that satisfies the docking enabling conditions in Eqs. (3.8–3.11), while respecting the path constraints in Eqs. (3.12–3.14).

In general, optimal control problems are nonlinear and, therefore, do not have close solutions. Two different approaches have been proposed in literature to solve numerically this type of problem: indirect methods and direct methods [57]. Indirect methods were introduced between 1950 and 1980 and employ the calculus of variations to define a set of necessary conditions for optimality, which are subsequently solved using proper numerical methods to attain the optimal solution. Direct methods were introduced from 1980. The key feature of this class of methods is that each state variable and control is discretized and approximated with piecewise parameterization functions. The original optimal control problem is converted into an equivalent nonlinear optimization problem whose varied parameters are the coefficients of the parameterization functions. Both approaches are computationally demanding. In fact, the boundary-value problem generated by indirect methods is often difficult to solve. On the other hand, the nonlinear optimization problem formulated by direct methods might have thousands of variable parameters. Clearly, the solution of the optimal control problem can be only obtained offline using either direct or indirect methods. Different approaches must be employed to obtain a sub-optimal docking trajectory in short amount of time that allows onboard execution.

3.4 Choice of the Trajectory Generation Algorithm

The choice of the trajectory generation algorithm highly depends on the dynamics model of the considered system. In fact, some methods are more suitable for linear system with fewer path constraints, whereas others are capable to handle nonlinear systems with path constraints. The path planner algorithms discussed in Section 2.2 are compared in the following in order to identify the most suitable approach for the 6 DoF docking maneuver to a tumbling target. The most important requirements to ensure successful docking to a tumbling target along with the requirements for next generation algorithms are considered [8]:

1. *Optimality*: fuel-optimal trajectories are required.
2. *Path constraints* along the trajectory such as collision avoidance, sensor pointing and control boundaries are crucial for ensuring safe and feasible trajectories. Furthermore, precise synchronization of chaser and target motions at the end of the maneuver is required to enable safe mechanical contact between spacecraft.
3. The 6 *DoF motion* of the chaser spacecraft must be controlled under *nonlinear* orbital dynamics.
4. Low *computational time* of the algorithm in order to ensure onboard trajectory generation. This aspect is particularly important in the case of docking missions to space debris, since the precise kinematics characteristics of the target vehicle can be estimated by the navigation system of the chaser spacecraft only. Therefore, the feasible docking trajectory must be calculated onboard using the most current data from the navigation system.
5. *Tumbling target:* since space debris might have tumble about the principal axes without control [5], the path planner must handle this particular case.

Table 3.1 summarizes the key features of each selected method. As previously discussed, glideslope, MILP and MPC do not meet all the requirements. In fact, the glideslope approach does not handle path constraints and the 6 DoF nonlinear orbital mechanics. Furthermore, this method requires predefined velocity profiles computed offline. The MILP algorithm is an offline path planner and MPC is not capable to compute a docking trajectory to a tumbling target in short time, since long time horizons are required. Moreover, the parameters of MPC must be changed for each scenario and therefore general applicability is not guaranteed. The RRT approach showed robustness for the translational motion only. The most suitable method is IDVD, since it ensures optimality, path constraints enforcement, capability to handle the case of 6 DoF docking maneuver to a tumbling target over a range of angular velocities. Furthermore, IDVD ensures precise enforcement of the docking conditions (in the ideal case of perfect knowledge of the final conditions) since the expected final conditions are direct linked to the polynomial coefficients. This is particularly important to achieve a safe mechanical contact between spacecraft.

Table 3.1: Comparison of the path planning algorithms.

Criteria	Glideslope	MILP	MPC	RRT	IDVD
Optimality	-	+	+	+/-	+
Path constraints	-	+	+	+	+
Nonlinear, 6DoF dynamics	-	-	+/-	-	+
Computational time	+	-	-	+/-	+/-
Tumbling target	+	-	-	-	+

In summary, the selected method for onboard trajectory generation is IDVD. The main disadvantage of this approach is the high computational

time for trajectory optimization. Since the computational performance of the algorithm highly depends on the trajectory parameterization, the application of the method to the 6 DoF docking to a tumbling target is revised in Chapters 3-5 in order to ensure closed-loop implementation and to meet the requirements of next generation guidance systems.

3.5 IDVD Approach

This section discusses the general application of IDVD to the problem of for rapid trajectory generation for proximity operations for uncontrolled space objects. The key feature of this approach is that the optimal control problem stated in Section 3.3 is converted into an equivalent nonlinear programming problem by describing the chaser trajectory components with a set of interpolating functions defined in the virtual domain τ [35, 36]. Some of the functions' polynomial coefficients will be linked to the boundary conditions on the trajectory, whereas other will be unconstrained and therefore become part of the varied parameters set of the resultant nonlinear programming problem. By inverting the dynamics model of the system, each state variable and each control component can be expressed as functions in the virtual domain τ of the parameterized trajectory and its derivatives. Notably for the considered two-spacecraft rendezvous problem defined by Eqs. (3.1–3.7), it results in the following expressions

$$\begin{aligned} \boldsymbol{\omega}^{\mathrm{C}}(\tau) &= \boldsymbol{f}_1\left(\boldsymbol{q}^{\mathrm{C}}(\tau), \boldsymbol{q}'^{\mathrm{C}}(\tau)\right), \qquad \boldsymbol{T}^{\mathrm{C}}(\tau) = \boldsymbol{f}_2\left(\boldsymbol{q}^{\mathrm{C}}(\tau), \boldsymbol{q}'^{\mathrm{C}}(\tau), \boldsymbol{q}''^{\mathrm{C}}(\tau)\right), \\ \boldsymbol{F}^{\mathrm{H}}(\tau) &= \boldsymbol{f}_3\left(x(\tau), y(\tau), z(\tau), x'(\tau), y'(\tau), z'(\tau), x''(\tau), y''(\tau), z''(\tau)\right). \end{aligned} \tag{3.16}$$

Virtual domain and time domain are related to each other by [35]

$$t(\tau_{\mathrm{f}}) = \int_0^{\tau_{\mathrm{f}}} \frac{\mathrm{d}\tau}{\lambda(\tau)}, \tag{3.17}$$

where

$$\lambda(\tau) = \frac{\mathrm{d}\tau}{\mathrm{d}t} \tag{3.18}$$

is the so-called speed factor, which is also expressed by a polynomial in the virtual domain. In turn, also the integrand of the performance index J in Eq. (3.15) can be expressed as function of the parameterized trajectory and the speed factor. Therefore, J can be minimized through the optimization of the varied parameters set composed by the unconstrained polynomial coefficients. The key advantage of describing the trajectory in the virtual domain

instead of time domain is the possibility to decouple space and time optimization and, consequently, a given trajectory can be followed using the different velocity profiles [35]. This allows the optimization of the velocity magnitude along the trajectory.

As already discussed, translation and rotation of the chaser spacecraft must be controlled simultaneously and, therefore, IDVD must be applied to the relative translational motion (C-W dynamics in Eq. (3.1)) and to the chaser rotational motion with respect to the Hill's coordinate system (Eq. (3.2) and Eq. (3.4)). The problem of applying IDVD to this problem has been addressed recently. Whereas the application of the method to the C-W dynamics is straightforward [38], the application to the rotation results more laborious due to the unit norm constraint of the quaternion vector [40]. Nevertheless, IDVD was applied to the full DoF docking maneuver using both quaternions and Euler angles by Boyarko [40]. In both cases, the computational time required by IDVD to generate a feasible trajectory resulted too high to ensure onboard execution, especially in closed loop. The high computational time required by the algorithm was caused by two main reasons:

1. laborious formulation for representing the quaternions due to the unit norm;
2. the parameterization of the virtual domain as in Eqs. (3.17, 3.18) increases the complexity of the nonlinear programming problem with respect to the basic representation of the trajectory in the time domain.

Since the motion of a noncooperative spacecraft can be estimated only with high uncertainty before the launch of the mission, the chaser spacecraft must determine onboard a feasible docking trajectory using the data from the relative navigation sensors [56]. For this reason, onboard trajectory generation capability is crucial. To ensure this capability to IDVD, the application of the method must be improved by choosing the most suitable attitude representation and the definition for the speed factor.

3.6 Summary

The dynamics model of proximity operations between a controlled chaser spacecraft and an uncontrolled target vehicle in circular orbit was presented in this Chapter. The particular case of docking to the target was analyzed in terms of required final conditions and path constraints. The IDVD algorithm was chosen among state-of-the-art methods for rapid sub-optimal trajectory planning due to its advantages with respect to the considered problem. In order to ensure real-time capabilities, the attitude parameters for attitude control and the domain definition of the polynomial functions must be carefully chosen.

4 Attitude Control

4.1 Introduction to Attitude Maneuvers

The study undertaken in this chapter is motivated by the dynamics of the two spacecraft rendezvous model and by the performance of IDVD applied to the docking problem evaluated in preliminary applications [40]. As matter of fact, attitude and position of the chaser spacecraft must be controlled simultaneously during the docking procedure and, therefore, the maneuver can be thought of as a translational motion relative to the target combined with an attitude control maneuver. Whereas the application of the IDVD method is straightforward for the relative translational dynamics (C-W equations) [38], several attitude representations can be employed for controlling the orientation with the same method. Moreover, due to the fact that IDVD might require a considerable amount of time to converge when applied to quaternions [58, 59], it is necessary to understand the impact of both virtual domain and attitude parameterization in the algorithm performance in order to ensure the best trade-off between optimality and onboard execution capabilities. For this reason, the IDVD method is first applied to attitude control maneuvers.

Spacecraft attitude control is a common task in most space missions. For instance, slew maneuvers are required for targeting imaging equipment and sensors, orienting antenna towards Earth or reorienting the spacecraft for solar power absorption. However, in particular applications the spacecraft is required to reorient while minimizing a certain mission parameter, such as maneuver duration or fuel expenditure. For example, time-optimal reorientation maneuvers are required by Earth imaging satellite in order to increase mission effectiveness. In fact, rapid retargeting capability increases the image collection capacity during a given observation window, which is a key aspect for commercial applications or climate and natural disaster monitoring [60]. Moreover, pointing the entire spacecraft rather than sweep the imaging system improves the resolution of the image [61]. Military space missions may also require agile reorientation capabilities. For instance, the TacSat-3 was developed to demonstrate responsive delivery of information to operational military users [62]. The key feature required by the satellite was the capability to change rapidly its attitude due to the limited operational window for mission acquisition, tasking and information delivery. Finally, agile attitude maneuvers are also required by tracking satellites for pointing moving ground targets [61]. Current onboard control systems exe-

cute the maneuver by steering about a spacecraft eigenaxis since it represents the shortest angular path between two orientations. However, Karpenko et al. demonstrated with the TRACE spacecraft that this procedure is not time-optimal, and onboard control systems that optimize the maneuver are required [63].

The optimal reorientation of a spacecraft has been studied theoretically for both minimum time and minimum energy problems [64, 65, 66]. In particular, Bilimoria and Wie investigated the time optimal rest to rest reorientation of a symmetric spacecraft, showing that bang-bang control is optimal and the resulting motion has a significant nutational component [65].

Recently, the problem has been investigated numerically using two different direct optimization approaches: pseudospectral methods and inverse dynamics. Pseudospectral methods are based on the numerical integration of the differential equations of motion and provide accurate solutions, but may converge slowly [67,68]. On the contrary, in inverse dynamics the trajectory is approximated by analytical functions. In most cases, the solution is suboptimal, but computational speed is high due to the reduced number of variable parameters. Therefore, the inverse dynamics approach is suitable for onboard applications. Louembert et al. first applied the inverse dynamics method to the optimal spacecraft reorientation problem using B-Splines to represent the MRP [69]. Recently, Boyarko proposed a rapid attitude trajectory generation method based on the IDVD method [58]. Finally, Yakimenko applied IDVD to Euler angles [59]. However, a direct comparison in terms of performance (computational speed and trajectory optimality) among IDVD applied to quaternions, MRP and Euler angles hasn't been addressed yet. This is particularly important in order to choose the most suitable method to control the rotation during the docking maneuver, since each approach has its own strengths and weaknesses.

4.2 Problem Formulation

The dynamics of the rotational motion of a spacecraft is governed by Euler's equations, which in scalar form with the angular velocity $\boldsymbol{\omega} = [\omega_x, \omega_y, \omega_z]^T$ and the inertial tensor $\boldsymbol{I} = \mathrm{diag}([I_{xx}, I_{yy}, I_{zz}])$ referenced to the body-fixed principal axes can be expressed as [52]

$$\begin{aligned}
\dot{\omega}_x &= \frac{(I_{yy} - I_{zz})\omega_z\omega_y + T_x}{I_{xx}} \\
\dot{\omega}_y &= \frac{(I_{zz} - I_{xx})\omega_z\omega_x + T_y}{I_{yy}} \\
\dot{\omega}_z &= \frac{(I_{xx} - I_{yy})\omega_y\omega_x + T_z}{I_{zz}}
\end{aligned} . \tag{4.1}$$

In Eq. (4.1) $\boldsymbol{T} = [T_x, T_y, T_z]^T$ represents the component of the bounded external torque vector referenced to the body frame.

The kinematics of the rotational motion of a spacecraft can be described using different attitude representations [70]. In terms of quaternion $\boldsymbol{q} = [q_1, q_2, q_3, q_4]^T$, the kinematic differential equation is given by

$$\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 0 & \omega_z & -\omega_y & \omega_x \\ -\omega_z & 0 & \omega_x & \omega_y \\ \omega_y & -\omega_x & 0 & \omega_z \\ -\omega_x & -\omega_y & -\omega_z & 0 \end{bmatrix}\begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \frac{1}{2}\boldsymbol{q} \otimes \boldsymbol{\Omega}, \tag{4.2}$$

where $\boldsymbol{\Omega} = [\omega_x, \omega_y, \omega_z, 0]^T$ and the symbol $\otimes$ denotes the (Hamiltonian) product between quaternions [70]. In terms of Euler angles $\boldsymbol{\theta} = [\theta_1, \theta_2, \theta_3]^T$ of the rotational sequence 1-2-3, the kinematic differential equation is

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix} = \frac{1}{\cos\theta_2}\begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 \\ \sin\theta_3\cos\theta_2 & \cos\theta_3\cos\theta_2 & 0 \\ -\cos\theta_3\sin\theta_2 & \sin\theta_3\sin\theta_2 & 1 \end{bmatrix}\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}. \tag{4.3}$$

Finally, using the MRP $\boldsymbol{\sigma} = [\sigma_1, \sigma_2, \sigma_3]^T$, the kinematics of the rotational motion can be described by the following equation

$$\begin{bmatrix} \dot{\sigma}_1 \\ \dot{\sigma}_2 \\ \dot{\sigma}_3 \end{bmatrix} = \frac{1}{4}\begin{bmatrix} 1+\sigma_1^2-\sigma_2^2-\sigma_3^2 & 2(\sigma_1\sigma_2-\sigma_3) & 2(\sigma_1\sigma_3+\sigma_2) \\ 2(\sigma_1\sigma_2+\sigma_3) & 1-\sigma_1^2+\sigma_2^2-\sigma_3^2 & 2(\sigma_2\sigma_3-\sigma_1) \\ 2(\sigma_1\sigma_3-\sigma_2) & 2(\sigma_2\sigma_3+\sigma_1) & 1-\sigma_1^2-\sigma_2^2+\sigma_3^2 \end{bmatrix}\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \frac{1}{4}\boldsymbol{B}(\boldsymbol{\sigma})\boldsymbol{\omega}. \tag{4.4}$$

The optimal spacecraft reorientation problem consists of minimizing a (given) cost function J by finding the optimal control vector $\boldsymbol{T}$, subjected to the control constraints $\boldsymbol{T}_{\min} \le \boldsymbol{T} \le \boldsymbol{T}_{\max}$, that brings the system described by Eq. (4.1) and Eqs. (4.2–4.4) from an initial state of angular velocity $\boldsymbol{\omega}_0$ and attitude $\boldsymbol{q}_0$, $\boldsymbol{\theta}_0$ or $\boldsymbol{\sigma}_0$ to a final state of angular velocity ω_f and attitude $\boldsymbol{q}_f$, $\boldsymbol{\theta}_f$ or $\boldsymbol{\sigma}_f$. The cost function J is defined as

$$J = \int_{t_0}^{t_f} \mathrm{d}t \tag{4.5}$$

for minimum time maneuvers and

$$J = \frac{1}{2}\int_{t_0}^{t_f} \left(T_x^2 + T_y^2 + T_z^2\right)\mathrm{d}t \tag{4.6}$$

for minimum quadratic control (or energy) expenditure.

4.3 IDVD and Inverse Dynamics Approaches

As discussed in Section 3.5, the IDVD algorithm allows rapid computation of suboptimal and feasible paths by describing the trajectory components with a set of polynomial functions defined in the virtual domain. Some of the functions' polynomial coefficients are linked to the boundary conditions on the trajectory, whereas the others remain free and therefore become part of the varied parameters set of the resultant nonlinear programming problem. By using the dynamics model of the system, each state variable and each control component can be expressed as function of the parameterized trajectory and its derivatives defined in the virtual domain. In particular, for the considered reorientation maneuver, it results in

$$\boldsymbol{\omega}(\tau) = f_{1\mathrm{q}}\left(\boldsymbol{q}(\tau),\boldsymbol{q}'(\tau)\right), \quad \boldsymbol{T}(\tau) = f_{2\mathrm{q}}\left(\boldsymbol{q}(\tau),\boldsymbol{q}'(\tau)\right) \tag{4.7}$$

for the quaternion-based model,

$$\boldsymbol{\omega}(\tau) = f_{1\theta}\left(\boldsymbol{\theta}(\tau),\boldsymbol{\theta}'(\tau)\right), \quad \boldsymbol{T}(\tau) = f_{2\theta}\left(\boldsymbol{\theta}(\tau),\boldsymbol{\theta}'(\tau)\right) \tag{4.8}$$

for the Euler angles-based model and

$$\boldsymbol{\omega}(\tau) = f_{1\sigma}\left(\boldsymbol{\sigma}(\tau),\boldsymbol{\sigma}'(\tau)\right), \quad \boldsymbol{T}(\tau) = f_{2\sigma}\left(\boldsymbol{\sigma}(\tau),\boldsymbol{\sigma}'(\tau)\right) \tag{4.9}$$

for the MRP-based model. Also the integrand of the performance index J in Eq. (4.5) or in Eq. (4.6) can be expressed as function of the parameterized trajectory and the speed factor. Finally, J is minimized through the optimization of the unconstrained polynomial coefficients.

The impact of the virtual domain is evaluated by applying the IDVD in the virtual domain and in the time domain. In the latter approach, called Inverse Dynamics method in this thesis, the trajectory components are defined in the domain of time. This can be considered a particular case of IDVD. In fact, when of $\lambda(\tau) = 1$, the time variable t is identical to the abstract argument τ and the problem is defined in the time domain. In the following sections both IDVD and Inverse Dynamics methods are described in details for each attitude parameterization considered.

4.4 Euler angles

This section describes the IDVD and inverse dynamics methods applied to the Euler angles for the rotational sequence 1-2-3.

In computer graphics and animation smooth curves are represented by parametric functions such as Splines or Bezier curves for their capability to

approximate complex shapes through curve fitting and computational efficiency [71, 72, 73]. Given a set of control points p_0, p_1, p_2, …, p_n in $\mathfrak{R}$, the Bezier curve $p(\tau)$ which interpolates these points is defined as [71]

$$p(\tau)=\sum_{i=0}^{n} p_i \beta_{i,n}(\tau), \tag{4.10}$$

where $\beta_{i,n}(\tau)$ is the n-th order Bernstein polynomials

$$\beta_{i,n}(\tau)=\binom{n}{i}(1-\tau)^{n-i}\tau^{i} \tag{4.11}$$

acting as basis form of $p(\tau)$ and $\tau \in [0; 1]$ the virtual argument. This formulation yields to the following properties at the endpoints, which are employed by IDVD to define the trajectory with the Bezier curve [58, 72]:

1. $p(0) = p_0$ and $p(1) = p_n$;
2. $\mathrm{d}p(0)/\mathrm{d}\tau = n(p_1\text{-}p_0)$ and $\mathrm{d}p(1)/\mathrm{d}\tau = n(p_n\text{-}p_{n\text{-}1})$.

Following this approach, each Euler angle is represented in the virtual domain $\tau \in [0; 1]$ by a Bezier curve as follows

$$\theta_j(\tau)=\sum_{i=0}^{n} a_{j,i}\beta_{i,n}(\tau), \qquad j=1,2,3 \tag{4.12}$$

where $a_{j,i}$ are the polynomial coefficients acting as control points and $\beta_{i,n}(\tau)$ the Bernstein base as in Eq. (4.11). The map between the time domain and the virtual domain is defined by the speed factor $\lambda(\tau)$, which is expressed through a polynomial of order m [58]

$$\lambda(\tau)=\sum_{i=0}^{m} b_i \tau^{i}. \tag{4.13}$$

The corresponding points in the time domain are calculated with Eq. (3.17). As a result of this mapping, the time derivatives of the angles, expressed as function of τ, are obtained from the virtual derivatives according to

$$\dot{\theta}_j(\tau)=\frac{d\theta_j(\tau)}{\mathrm{d}\tau}\frac{\mathrm{d}\tau}{\mathrm{d}t}=\frac{\mathrm{d}\theta_j(\tau)}{\mathrm{d}\tau}\lambda, \qquad j=1,2,3 \tag{4.14}$$

$$\ddot{\theta}_j(\tau)=\frac{\mathrm{d}\dot{\theta}_j(\tau)}{\mathrm{d}\tau}\frac{\mathrm{d}\tau}{\mathrm{d}t}=\frac{\mathrm{d}^2\theta_j(\tau)}{\mathrm{d}\tau^2}\lambda^2+\frac{\mathrm{d}\theta_j(\tau)}{\mathrm{d}\tau}\frac{\mathrm{d}\lambda}{\mathrm{d}t}\lambda. \qquad j=1,2,3 \tag{4.15}$$

Note that the attitude trajectory components can be also represented with high-order polynomials, leaving the virtual argument unbounded and its final value as variable of the problem. However, preliminary experiments showed that this approach is computationally more expensive that the proposed one and requires a good first guess on the final value of the virtual argument, which is not intuitive.

By imposing attitude, angular velocity and acceleration at the endpoints, the polynomial coefficients $a_{j,i}$ can be defined in such a way that these boundary conditions are automatically satisfied. By exploiting the properties of the Bezier curve at the endpoints, it results

$$\begin{cases} a_{j,0} = \theta_{0j} \\ a_{j,1} = \dfrac{\dot{\theta}_{0j}}{n\lambda(0)} + a_{j,0} \\ a_{j,2} = \dfrac{1}{2}\binom{n}{2}^{-1} \dfrac{\ddot{\theta}_{0j} - \dot{\theta}_{0j}\lambda'(0)}{\lambda^2(0)} + 2a_{j,1} - a_{j,0} \end{cases} \qquad j = 1,2,3 \tag{4.16}$$

for the initial conditions and

$$\begin{cases} a_{j,n} = \theta_{\mathrm{f}j} \\ a_{j,n-1} = -\dfrac{\dot{\theta}_{\mathrm{f}j}}{n\lambda(1)} + a_{j,n} \\ a_{j,n-2} = \dfrac{1}{2}\binom{n}{n-2}^{-1} \dfrac{\ddot{\theta}_{\mathrm{f}j} - \dot{\theta}_{\mathrm{f}j}\lambda'(1)}{\lambda^2(1)} + 2a_{j,n-1} - a_{j,n} \end{cases} \qquad j = 1,2,3 \tag{4.17}$$

for the final conditions. In the previous expressions λ' is the virtual derivative of the speed factor, whereas θ_{0j} and $\theta_{\mathrm{f}j}$ denote the j-th Euler angle evaluated at the initial and final time respectively. The time derivatives of the angles are obtained with Eq. (4.3) and its time derivative, calculated at the endpoints. As a consequence, the degree n of the Bezier curves is related to the total number N_{B} of boundary conditions according to

$$n \geq N_{\mathrm{B}} + 1. \tag{4.18}$$

In this way, at least one polynomial coefficient for each angle is left as variable parameter for the optimization, allowing flexibility of the shape of the trajectory.

At this point it is important to note that the singularity in Eq. (4.3) limits the application of the proposed algorithm: the trajectories having $\theta_2 = \pm 90$

deg as final attitude cannot be represented by the parametric curve since the coefficients $a_{j,n-1}$ and $a_{j,n-2}$ defined in Eq. (4.17) cannot be determined.

The expressions of angular velocity and controls can be obtained from the angles by inverting the dynamics. Inversion of Eq. (4.3) provides the angular velocity $\boldsymbol{\omega}$

$$\begin{aligned} \omega_x &= \cos\theta_2\cos\theta_3\dot{\theta}_1 + \sin\theta_3\dot{\theta}_2 \\ \omega_y &= -\cos\theta_2\sin\theta_3\dot{\theta}_1 + \cos\theta_3\dot{\theta}_2 \\ \omega_z &= \sin\theta_2\dot{\theta}_1 + \dot{\theta}_3 \end{aligned}, \tag{4.19}$$

where the time derivatives of the angles are calculated using Eq. (4.14). The angular accelerations are obtained from the time derivative of Eq. (4.19) along with Eq. (4.15). Finally, the inversion of Eq. (4.1) provides the control torques

$$\begin{aligned} T_x &= I_{xx}\dot{\omega}_x + (I_{zz} - I_{yy})\omega_z\omega_y \\ T_y &= I_{yy}\dot{\omega}_y + (I_{xx} - I_{zz})\omega_z\omega_x \\ T_z &= I_{zz}\dot{\omega}_z + (I_{yy} - I_{xx})\omega_x\omega_y \end{aligned}. \tag{4.20}$$

Note that the obtained expressions of angular velocities and torques are functions of the virtual argument τ and the polynomial coefficients.

The proposed procedure allows to transform the cost J of the optimization problem into a (nonlinear) function $J(a_{j,i},b_i)$ of the polynomial coefficients: Eq. (4.5) is integrated numerically after substitution of Eq. (3.18) and Eq. (4.13) for the minimum time problem, whereas Eq. (4.6) is integrated after substitution of the expression of the controls for the minimum energy problem. The optimal control problem is consequently converted into the following equivalent nonlinear programming problem: Minimize the function $J(a_{j,i},b_i)$ subjected to the constraints

$$\begin{cases} \boldsymbol{T}_{\min} \le \boldsymbol{T}(\tau_i) \le \boldsymbol{T}_{\max} \\ t(\tau_i) \ge 0 \\ t(1) \le t_{\mathrm{fmax}} \end{cases}, \tag{4.21}$$

and the variables of the problem being the polynomial coefficients $a_{j,i}$ and b_i. The constraints are evaluated at several nodes τ_i of the virtual domain. The higher this number, the more accurate the enforcement of the bounds, but at the cost of more computational time. The second constraint in Eq. (4.21) imposes Eq. (3.17) to map the points of the virtual domain into positive points of the time domain. The third condition is applied to Eq. (3.17) and limits the maximum maneuver duration to t_{fmax}.

In the particular case of $\lambda(\tau) = 1$, the time variable t is identical to the abstract argument τ and the problem is defined in the time domain. Thus, IDVD is reduced to the basic inverse dynamics method. The Euler angles

can be represented in the time domain by Eq. (4.12) after imposing $\tau = t/t_f$, with t_f final maneuver time and variable of the problem. However, preliminary analysis showed that high-order polynomials are computationally more efficient than modified Eq. (4.12). Therefore, in this study the Euler angles are expressed in the time domain $t \in [0; t_f]$ by n-th order polynomials

$$\theta_j(t) = \sum_{i=0}^{n} a_{j,i} t^i . \qquad j = 1,2,3 \tag{4.22}$$

The time derivatives of the angles are given by the successive polynomial derivatives

$$\dot{\theta}_j(t) = \sum_{i=1}^{n} i a_{j,i} t^{i-1} , \quad \ddot{\theta}_j(t) = \sum_{i=2}^{n} i(i-1) a_{j,i} t^{i-2} . \qquad j = 1,2,3 \tag{4.23}$$

The boundary conditions are respected using the following expressions for the coefficients

$$\begin{cases} a_{j,0} = \theta_{0j} \\ a_{j,1} = \dot{\theta}_{0j} , \qquad j = 1,2,3 \\ a_{j,0} = \dfrac{\ddot{\theta}_{0j}}{2} \end{cases} \tag{4.24}$$

$$\begin{cases} a_{j,n} = \dfrac{\theta_{fj} - \sum_{i=0}^{n-1} a_{j,i} t_f^i}{t_f^n} \\ a_{j,n-1} = \dfrac{\dot{\theta}_{fj} - \sum_{i=0}^{n-2} i a_{j,i} t_f^{i-1} - n a_{j,n} t_f^{i-1}}{(n-1) t_f^{n-2}} \qquad . \qquad j = 1,2,3 \\ a_{j,n-2} = \dfrac{\ddot{\theta}_{fj} - \sum_{i=0}^{n-3} i(i-1) a_{j,i} t_f^{i-2} - \sum_{i=n-1}^{n} i(i-1) a_{j,i} t_f^{i-2}}{(n-2)(n-3) t_f^{n-4}} \end{cases} \tag{4.25}$$

The previous relationships are obtained by imposing the boundary conditions $\boldsymbol{\theta}_0$, $\dot{\boldsymbol{\theta}}_0$, $\ddot{\boldsymbol{\theta}}_0$ and $\boldsymbol{\theta}_f$, $\dot{\boldsymbol{\theta}}_f$, $\ddot{\boldsymbol{\theta}}_f$ to Eq. (4.22) and Eq. (4.23). Therefore, the order n of the polynomial in Eq. (4.22) must satisfy

$$n \geq N_B . \tag{4.26}$$

The expressions of angular velocity and controls are obtained from the angles and their derivatives with Eq. (4.19) and Eq. (4.20) respectively. Using a numerical integration method, the cost function J is transformed into a (nonlinear) function $J(a_{j,i},b_i,t_f)$ of the polynomials coefficients and final maneuver time.

The optimal control problem is therefore converted into the following equivalent nonlinear programming problem: Minimize the function $J(a_{j,i},b_i,t_f)$ subject to the constraints

$$\begin{cases} \boldsymbol{T}_{\min} \le \boldsymbol{T}(t_i) \le \boldsymbol{T}_{\max} \\ 0 \le t_f \le t_{f\,\max} \end{cases}, \tag{4.27}$$

with the final time t_f and the polynomial coefficients $a_{j,i}$ and b_i being the variables of the problem.

4.5 Quaternions

Following the parameterization approach employed for the Euler angles, the attitude trajectory in terms of quaternions can be represented by a set of control points and basis functions. However, the parameterization must ensure the nonlinear unit norm condition along the trajectory and therefore Eq. (4.12) cannot be employed directly. For this reason, the parametric curve proposed by Kim et al. and successively applied by Boyarko was chosen [58, 71].

Given two unit quaternions $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$, the quaternion curve $\gamma(\tau)$ which connects $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ can be defined as

$$\gamma(\tau) = \boldsymbol{q}_1 \otimes \exp(\boldsymbol{\omega}^* \beta(\tau)), \tag{4.28}$$

where $\beta(\tau)$ is the basis function of the curve, $\tau \in [0; 1]$ the virtual argument and

$$\boldsymbol{\omega}^* = \log(\boldsymbol{q}_1^{-1} \otimes \boldsymbol{q}_2) \tag{4.29}$$

a vector defined in $\Re^3$ whose norm is the Euler's rotation angle defined by the quaternion $\boldsymbol{q}_1^{-1} \otimes \boldsymbol{q}_2$ and direction the Euler's axis. The exponential and natural logarithmic maps are applied according to the following definition

$$\boldsymbol{q} = \exp(\boldsymbol{\omega}^*) = \begin{cases} \left[\sin\left(\boldsymbol{\omega}^*\right) \dfrac{\boldsymbol{\omega}^*}{\left\|\boldsymbol{\omega}^*\right\|} \quad \cos\left(\left\|\boldsymbol{\omega}^*\right\|\right) \right]^T & \left\|\boldsymbol{\omega}^*\right\| \neq 0 \\ \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T & \left\|\boldsymbol{\omega}^*\right\| = 0 \end{cases}, \tag{4.30}$$

$$\boldsymbol{\omega}^* = \log(\boldsymbol{q}) = \begin{cases} \left(\dfrac{\arccos(q_4)}{\sqrt{q_1^2 + q_2^2 + q_3^2}} \right) \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix}^{\mathrm{T}} & \sqrt{q_1^2 + q_2^2 + q_3^2} \neq 0 \\ \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}} & \sqrt{q_1^2 + q_2^2 + q_3^2} = 0 \end{cases}. \tag{4.31}$$

Note that $||\exp(\boldsymbol{\omega}^*)||=1 \; \forall \; \boldsymbol{\omega}^*$ and therefore Eq. (4.28) automatically satisfies the unit norm constraint.

Therefore, the quaternion trajectory curve $\boldsymbol{q}(\tau)$ can be represented by a set of parametric curves connecting a sequence of control points $\tilde{\boldsymbol{q}}_i$ according to [71]

$$\boldsymbol{q}(\tau) = \boldsymbol{q}_0 \otimes \prod_{i=1}^{n} \exp(\boldsymbol{\omega}_i^* \tilde{\beta}_{i,n}(\tau)), \tag{4.32}$$

where

$$\tilde{\beta}_{i,n}(\tau) = \sum_{j=i}^{n} \binom{n}{j} (1-\tau)^{n-j} \tau^j \tag{4.33}$$

and

$$\boldsymbol{\omega}_i^* = \log(\tilde{\boldsymbol{q}}_{i-1}^{-1} \otimes \tilde{\boldsymbol{q}}_i). \tag{4.34}$$

In Eq. (4.32) the symbol Π denotes the sequential quaternion product and the control points satisfy the following conditions at endpoints:

$$\tilde{\boldsymbol{q}}_0 = \boldsymbol{q}_0, \quad \tilde{\boldsymbol{q}}_n = \boldsymbol{q}_{\mathrm{f}} \; .$$

The speed factor $\lambda(\tau)$ is defined with a polynomial as in Eq. (4.13). The virtual derivatives of Eq. (4.32) are calculated using the chain rule and their expressions can be found in Ref. [58]. The time derivatives of the quaternion vector are calculated from its virtual derivatives using Eq. (4.14) and Eq. (4.15) after substituting the j-th angle θ_j (and its derivatives) with the quaternion vector $\boldsymbol{q}$.

To allow the correct optimization using this approach, all the control points $\tilde{\boldsymbol{q}}_i$ and the vectors $\boldsymbol{\omega}^*$ must be defined using the boundary conditions of the problem [58]. Consequently, the parameters to be optimized consist of the additional boundary conditions introduced to define all the $\tilde{\boldsymbol{q}}_i$ and $\boldsymbol{\omega}_i^*$ along with the polynomial coefficients of $\lambda(\tau)$. The total number N_{B} of boundary conditions required to define all the parameters is therefore

$$N_{\mathrm{B}} = n + 1. \tag{4.35}$$

In this study, the quaternion history is represented with a fifth-order Bezier function (and therefore a product of five exponential functions). Consequently, six boundary conditions are required to define all the control points and vectors $\boldsymbol{\omega}_i^*$. These conditions are initial and final attitude, angular velocity and angular acceleration.

By exploiting the properties of the Bezier polynomial and imposing the boundary conditions, the vectorial terms $\boldsymbol{\omega}_i^*$ are defined through $\boldsymbol{\Omega}_i^* = [\ \omega_{ix}^*\ \ \omega_{ix}^*\ \ \omega_{ix}^*\ \ 0]^T$ with the following expressions

$$\begin{cases} \boldsymbol{\Omega}_1^* = \dfrac{\boldsymbol{q}_0^{-1} \otimes \dot{\boldsymbol{q}}_0}{n} \\ \boldsymbol{\Omega}_2^* = \dfrac{\boldsymbol{q}_0^{-1} \otimes \ddot{\boldsymbol{q}}_0 - 2\left[\binom{n}{2} - n(n-1)\right]\boldsymbol{\Omega}_1^* - n^2 \boldsymbol{\Omega}_1^* \otimes \boldsymbol{\Omega}_1^*}{2\binom{n}{2}} \end{cases}, \tag{4.36}$$

$$\begin{cases} \boldsymbol{\Omega}_n^* = \dfrac{\boldsymbol{q}_\mathrm{f}^{-1} \otimes \dot{\boldsymbol{q}}_\mathrm{f}}{n} \\ \boldsymbol{\Omega}_{n-1}^* = \dfrac{\ddot{\boldsymbol{q}}_{n-1}^{-1} \otimes \left[\ddot{\boldsymbol{q}}_\mathrm{f} - n^2 \boldsymbol{q}_\mathrm{f} \otimes \boldsymbol{\Omega}_n^* \otimes \boldsymbol{\Omega}_n^* - 2\binom{n}{n-2}\boldsymbol{q}_\mathrm{f} \otimes \boldsymbol{\Omega}_n^*\right] \otimes \boldsymbol{q}_\mathrm{f}^{-1} \otimes \tilde{\boldsymbol{q}}_{n-1}}{2\left[\binom{n}{n-2} - n(n-1)\right]} \end{cases}, \tag{4.37}$$

where n is the order of the Bezier polynomial, which corresponds to 5 in the considered case. Therefore, Eq. (4.37) provides the expressions for Ω_5^* and Ω_4^*. All the control points $\tilde{\boldsymbol{q}}_i$ and the coefficient Ω_3^* are then calculated using Eq. (4.34). The expressions of angular velocity and accelerations are obtained with the inversion of Eq. (4.2) and its derivative

$$\boldsymbol{\Omega} = 2\boldsymbol{q}^{-1} \otimes \dot{\boldsymbol{q}}, \qquad \dot{\boldsymbol{\Omega}} = 2\boldsymbol{q}^{-1} \otimes \ddot{\boldsymbol{q}} + \frac{1}{2}\boldsymbol{\Omega} \otimes \boldsymbol{\Omega}. \tag{4.38}$$

The expressions of the torques are obtained with Eq. (4.20). By numerically integrating the cost function J, the optimal control problem is converted into an equivalent nonlinear programming problem, whose variables are the additional boundary conditions introduced to define all the parameter of Eq. (4.32) along with the polynomial coefficients of Eq. (4.13).

By imposing $\lambda(\tau) = 1$, the problem is formulated in the time domain $t \in [0; t_\mathrm{f}]$, where t_f is set as variable parameter of the problem. Consequently, Eq. (4.32) is modified as

$$\boldsymbol{q}(t) = \boldsymbol{q}_0 \otimes \prod_{i=1}^{n} \exp(\boldsymbol{\omega}_i^* \tilde{\beta}_{i,n}(t)), \tag{4.39}$$

with

$$\tilde{\beta}_{i,n}(t) = \sum_{j=i}^{n} \binom{n}{j} \left(1 - \frac{t}{t_\mathrm{f}}\right)^{n-j} \left(\frac{t}{t_\mathrm{f}}\right)^{j}. \tag{4.40}$$

Note that the unit norm constraint and the property of the Bezier polynomial at endpoints are still preserved. The transformation of the problem into the equivalent nonlinear programming problem is analogous to IDVD with virtual domain. Due to Eq. (4.39) and Eq. (4.40), the general expressions of the coefficients $\boldsymbol{\omega}_i$* for a Bezier curve of order n are given by

$$\begin{cases} \boldsymbol{\Omega}_1^* = \dfrac{t_\mathrm{f} \boldsymbol{q}_0^{-1} \otimes \dot{\boldsymbol{q}}_0}{n} \\ \boldsymbol{\Omega}_2^* = \dfrac{\boldsymbol{q}_0^{-1} \otimes \ddot{\boldsymbol{q}}_0 - \dfrac{2}{t_\mathrm{f}^2}\left[\binom{n}{2} - n(n-1)\right]\boldsymbol{\Omega}_1^* - \left(\dfrac{2}{t_\mathrm{f}}\right)^2 \boldsymbol{\Omega}_1^* \otimes \boldsymbol{\Omega}_1^*}{\dfrac{2}{t_\mathrm{f}^2}\binom{n}{2}} \end{cases} \tag{4.41}$$

$$\begin{cases} \boldsymbol{\Omega}_n^* = \dfrac{t_\mathrm{f} \boldsymbol{q}_\mathrm{f}^{-1} \otimes \dot{\boldsymbol{q}}_\mathrm{f}}{n} \\ \boldsymbol{\Omega}_{n-1}^* = \dfrac{\tilde{\boldsymbol{q}}_{n-1}^{-1} \otimes \left[\ddot{\boldsymbol{q}}_\mathrm{f} - \left(\dfrac{2}{t_\mathrm{f}}\right)^2 \boldsymbol{q}_\mathrm{f} \otimes \boldsymbol{\Omega}_n^* \otimes \boldsymbol{\Omega}_n^* - \dfrac{2}{t_\mathrm{f}^2}\binom{n}{n-2} \boldsymbol{q}_\mathrm{f} \otimes \boldsymbol{\Omega}_n^*\right] \otimes \boldsymbol{q}_\mathrm{f}^{-1} \otimes \tilde{\boldsymbol{q}}_{n-1}}{\dfrac{2}{t_\mathrm{f}^2}\left[\binom{n}{n-2} - n(n-1)\right]}. \end{cases} \tag{4.42}$$

For the considered case of fifth-order Bezier curve (n=5), Eq. (4.42) provides the expressions for Ω_5* and Ω_4*.

4.6 MRP

The IDVD method is applied to the MRP similarly to the Euler angles. Each MRP is represented in the virtual domain $\tau \in [0; 1]$ with an n-th order Bezier curve:

$$\sigma_j(\tau) = \sum_{i=0}^{n} a_{j,i}\beta_{i,n}(\tau). \qquad j = 1,2,3 \tag{4.43}$$

The speed factor $\lambda(\tau)$ is defined with Eq. (4.13). The boundary conditions are imposed with Eq. (4.16) and Eq. (4.17) after substituting the j-th angle θ_j (and its derivatives) with the j-th MRP σ_j. The required initial and final time derivatives of the MRP are obtained with Eq. (4.4) and its derivative calculated at the endpoints. By inverting the kinematic Eq. (4.4) and differentiating the obtained expression, results in the following compact expression of the angular velocity and acceleration [70]:

$$\boldsymbol{\omega} = 4\frac{1}{\left(1+|\boldsymbol{\sigma}|^2\right)^2}\boldsymbol{B}(\boldsymbol{\sigma})^{\mathsf{T}}\dot{\boldsymbol{\sigma}}, \tag{4.44}$$

$$\dot{\boldsymbol{\omega}} = \frac{1}{\left(1+|\boldsymbol{\sigma}|^2\right)^2}\boldsymbol{B}(\boldsymbol{\sigma})^{\mathsf{T}}\left[4\ddot{\boldsymbol{\sigma}} - \dot{\boldsymbol{B}}(\boldsymbol{\sigma})\boldsymbol{\omega}\right]. \tag{4.45}$$

The time derivatives of the MRP are obtained from the virtual derivatives with Eq. (4.14) and Eq. (4.15) by substituting the angle θ_j (and its derivatives) with σ_j. The control torques are obtained with Eq. (4.20).

Similarly to IDVD applied to Euler angles, the optimal control problem is consequently converted into an equivalent nonlinear programming problem, with constraints given by Eq. (4.21). The application of the method in the time domain is analogous to the case with the Euler angles.

4.7 Performance for Minimum Time Maneuvers

The numerical results of comparison among the IDVD algorithms applied to minimum time attitude control are presented in this section. Several test scenarios have been considered in nondimensional form as in Wie [65]. The duration of each maneuver is limited in the interval [2 10] s. All the scenarios addressed in this analysis are rest to rest, but IDVD can be extended to any angular velocity at endpoints [58].

The proposed algorithms have been implemented in Matlab on a Windows 7 laptop computer with an Intel 2.27 GHz i5 M430 processor and 4 Gb of RAM. The nonlinear programming problem generated by IDVD is solved using the Sequential Quadratic Solver method of Matlab *fmincon* [74]. A total number of 50 nodes is considered for imposing the constraints of the optimization problem. The IDVD applied to virtual domain employs the Gauss-Legendre Quadrature method with 4 points to integrate Eq. (3.17).

Reference solutions are computed with the Gauss Pseudospectral Optimization Software GPOPS [75]. The obtained controls are propagated with Matlab ODE 45 using Eq. (4.1) and Eq. (4.2) and a time step of 0.01s. The magnitude of the final angular velocity and attitude errors after propagation of the controls are denoted with $\Delta\omega$ and $\Delta\theta$ respectively.

The performance of the IDVD method are evaluated in terms of effects of the virtual domain, attitude representation and order of polynomials on solution quality and computational speed required by the algorithm to converge to an optimal solution. In particular, effects of the virtual domain are evaluated by solving a single optimization problem in the virtual domain and in the time domain separately, using an analogous polynomial to represent the attitude trajectory. To compare the results of the considered algorithms in terms of solution optimality, the difference in percentage ΔJ is defined as follows:

$$\Delta J = 100 \frac{| J_{\mathrm{IDVD}} - J_{\mathrm{GPOPS}} |}{J_{\mathrm{GPOPS}}}, \tag{4.46}$$

where J_{IDVD} is the optimal cost function computed by IDVD and J_{GPOPS} the optimal cost computed with GPOPS taken as reference solution.

The overall performance of IDVD is determined in terms of computational time $CPUt$ required by the algorithm to converge and solution quality ΔJ according to the following performance index

$$MI = K_{\mathrm{J}} \Delta J + K_{\mathrm{CPU}} CPUt, \tag{4.47}$$

where K_{J} and K_{CPU} are weighting coefficients for solution quality and computational time respectively. Since ΔJ and $CPUt$ have the same order of magnitude, the weights are defined in the range [0; 1]. Different combinations of weighting values are considered to analyze the performance of IDVD. Note that the best possible performance of the algorithm is obtained when ΔJ and $CPUt$ are both close to zero. Therefore, the lower the performance index, the better the overall performance.

The maneuver addressed in this section is a minimum time 180 deg reorientation about the z-axis of a symmetric spacecraft with the following conditions [65]

$$\begin{gathered} \boldsymbol{I} = \mathrm{diag}([1,1,1]) \\ -1 \le \boldsymbol{T} \le 1 \\ \boldsymbol{\omega}_0 = [0,0,0] \quad \boldsymbol{\theta}_0 = [0,0,0]\mathrm{deg} \quad \boldsymbol{q}_0 = [0,0,0,1] \quad \boldsymbol{\sigma}_0 = [0,0,0] \\ \boldsymbol{\omega}_{\mathrm{f}} = [0,0,0] \quad \boldsymbol{\theta}_{\mathrm{f}} = [0,0,180]\mathrm{deg} \quad \boldsymbol{q}_{\mathrm{f}} = [0,0,1,0] \quad \boldsymbol{\sigma}_{\mathrm{f}} = [0,0,1] \end{gathered}. \tag{4.48}$$

Initial guesses for the controls at endpoints are taken to be equal to one, which is the maximum torque. Accordingly, the guesses on the polynomial

coefficients related to angular accelerations are obtained from Eqs. (4.16, 4.17, 4.24, 4.25, 4.36, 4.37, 4.41, 4.42); initial guess on the optimal maneuver time is 2 s, which is the minimum allowable maneuver time. The first guess for b_1 is the inverse of the guess of the maneuver time, while initial guesses for all the other polynomial coefficients are zero as in Ref. [58].

Table 4.1 summarizes the results of the optimizations. The GPOPS method computes the optimal solution found by Bilimoria and Wie with all the considered attitude representations. The optimal controls have a bang-bang nature (see Figure 4.1) and the consequent rotation has a significant nutational component, evident from the large values of the angle θ_1 and θ_2 represented in Figure 4.2.

The solutions obtained with IDVD are sub-optimal, for every attitude parameter, polynomial order and domain considered, and the resulting optimal controls have always a smooth time history instead of bang-bang, as showed in Table 4.1. This is due to the nature of the method: the controls are calculated as combination of the trajectory components and their derivatives through the inversion of the dynamics (see Eq. (4.19)-(4.20)-(4.38)-(4.44)-(4.45)). Since the trajectory is represented with polynomials, the controls result nonlinear continuous function defined in the virtual argument or time. Notably, the controls obtained with this approach are always continuous since the trajectory components must be represented at least with class C^2 functions (see the conditions in Eq. (4.18) and Eq. (4.26)). Therefore, a bang-bang shape of the controls cannot be achieved and only sub-optimal solutions can be obtained. Better solutions can be computed by increasing the order of the polynomials for trajectory representation, but at the cost of higher computational time.

Table 4.1: Optimization of a minimum time rest to rest 180 deg reorientation of a symmetric spacecraft using IDVD and GPOPS. Controls and trajectories are obtained with the marked methods are represented in Figure 4.1 and Figure 4.2 respectively.

Attitude	Method	J (s)	ΔJ (%)	CPU time (s)	$\Delta\omega$ (deg/s)	$\Delta\theta$ (deg)
MRP	GPOPS	3.2431	-	68.7	1.8e-1	6.9e-3
	IDVD 5th - time	3.8153	17.6	0.3	2.7e-1	3.7e-3
	IDVD 5th - virtual	3.4351	5.9	2.2	2.5e-1	1.7e-3
	IDVD 7th – time	3.4853	7.5	0.7	1.6e-1	2.0e-3
	IDVD 7th – virtual	3.3179	2.3	7.7	1.9e-1	2.5e-2
Euler Angles	GPOPS*	3.2431	-	93.2	1.8e-1	6.9e-3
	IDVD 5th - time	3.6871	13.7	0.3	1.6e-1	3.4e-3
	IDVD 5th - virtual	3.4443	6.2	1.6	5.6e-1	1.3e-1
	IDVD 7th – time*	3.5277	8.8	1.1	1.2e-1	2.8e-3
	IDVD 7th – virtual*	3.3302	2.7	7.1	2.8e-1	6.1e-2
Quaternion	GPOPS	3.2431	-	100.3	1.8e-1	6.9e-3
	IDVD 5th – time	3.5418	9.2	2.0	9.7e-2	2.3e-3
	IDVD 5th - virtual	3.3971	4.7	7.8	2.4e-2	1.8e-3

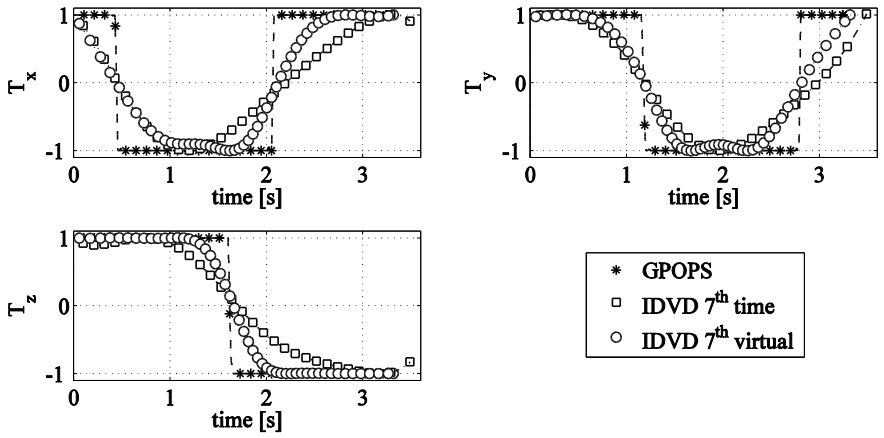

Figure 4.1: Optimal control time history generated with GPOPS and seventh-order IDVD applied to the Euler angles in virtual domain and time domain.

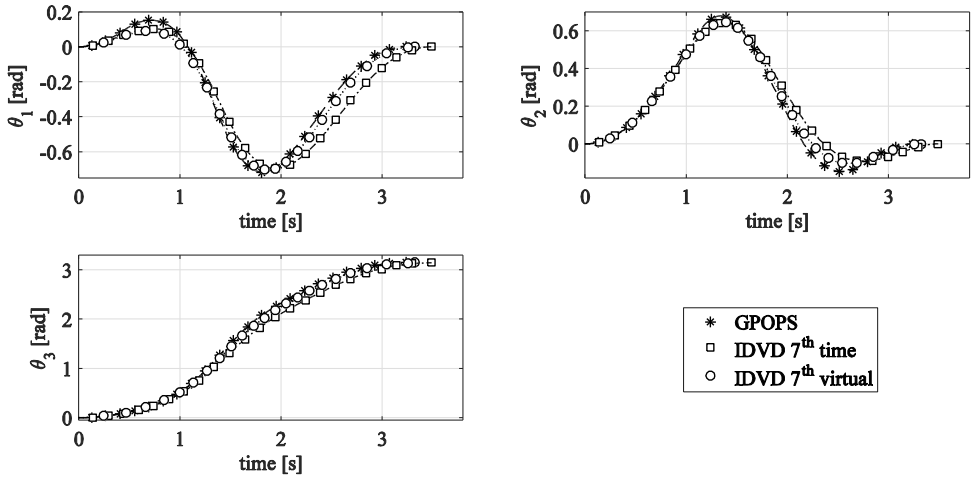

Figure 4.2: Optimal attitude trajectory in terms of Euler angles, generated with GPOPS, seventh-order IDVD in virtual domain and seventh-order IDVD in time domain, applied to the Euler angles.

The definition of the trajectory in the virtual domain improves the sub-optimal solution with respect to the same polynomial formulation in the time domain, for every attitude representation and polynomial order considered. In fact, the speed factor introduces additional variables to the problem, which allow optimization of the velocity profile along the trajectory (see also Eq. (4.14) and Eq. (4.15)). Figure 4.1 graphically represents the effect of the virtual domain on the control time histories. The introduction of the speed factor modifies the shape of the controls, which result closer to the bang-bang optimal shape. The other effect of the virtual domain is the increment of computational time required by the solver due to the additional variable parameters, additional constraints (see Eq. (4.21)) and integration of Eq.

(3.17) at every iteration of Matlab *fmincon*. The effect of the parameterization in the virtual domain can be observed also in Figure 4.2: the attitude trajectory computed with IDVD in virtual domain is closer to the solution of GPOPS.

The attitude parameter affects the computational time and solution quality. The IDVD applied to quaternions requires more time to converge than the analogous IDVD applied to MRP and Euler angles due to the usage of the exponential and logarithmic. Moreover, the solutions obtained with quaternions are closer to the reference optimal solution provided by GPOPS than MRP and Euler angles represented by fifth-order polynomials. The overall best optimal solution is obtained with the seventh-order IDVD applied to the MRP in the virtual domain. The polynomial order influences the solution quality and computational time required by the solver. Higher polynomial orders provide better optimal solution, but at the cost of more computational time due the higher number of variable parameters.

Figure 4.3 summarizes the performance of IDVD based on the performance index *MI*, for different combinations of weighting coefficients. Notably, in Figure 4.3 a) the performance bars are ordered according to the weights $K_{\mathrm{J}} = 0$ and $K_{\mathrm{CPU}} = 1$. In this case, the performance criteria is the computational speed required by IDVD to converge to a sub-optimal solution. The figure shows that higher computational speeds are achieved when IDVD is applied to fifth-order polynomials (lowest order considered) in the time domain. The reason of this behavior is that by increasing the order of polynomials representing the trajectory, more variables are introduced in the optimization problem and the Sequential Quadratic Solver of MatLab *fmincon* requires more time to converge. Furthermore, the definition of the parametric curves in the virtual domain introduces additional variables, being the polynomial coefficients for the speed factor (see Eq. (4.13)), which further increment the computational time. The lowest values of *MI* are obtained with seventh-order IDVD applied to the virtual domain. Figure 4.3 b) shows the performance of IDVD ordered according to the weights $K_{\mathrm{J}} = 0.5$ and $K_{\mathrm{CPU}} = 0.5$. In this case, the performance criteria is the computational speed and the optimality of the solution equally balanced. The best performance are obtained with fifth-order IDVD applied to MRP and Euler angles defined in the virtual domain. The low polynomial order ensures high computational speed, while the usage of the virtual domain improves the optimality of the solution. Lowest performance are obtained with fifth-order IDVD applied to Euler angles and MRP in the time domain, due to the low optimality of the solution.

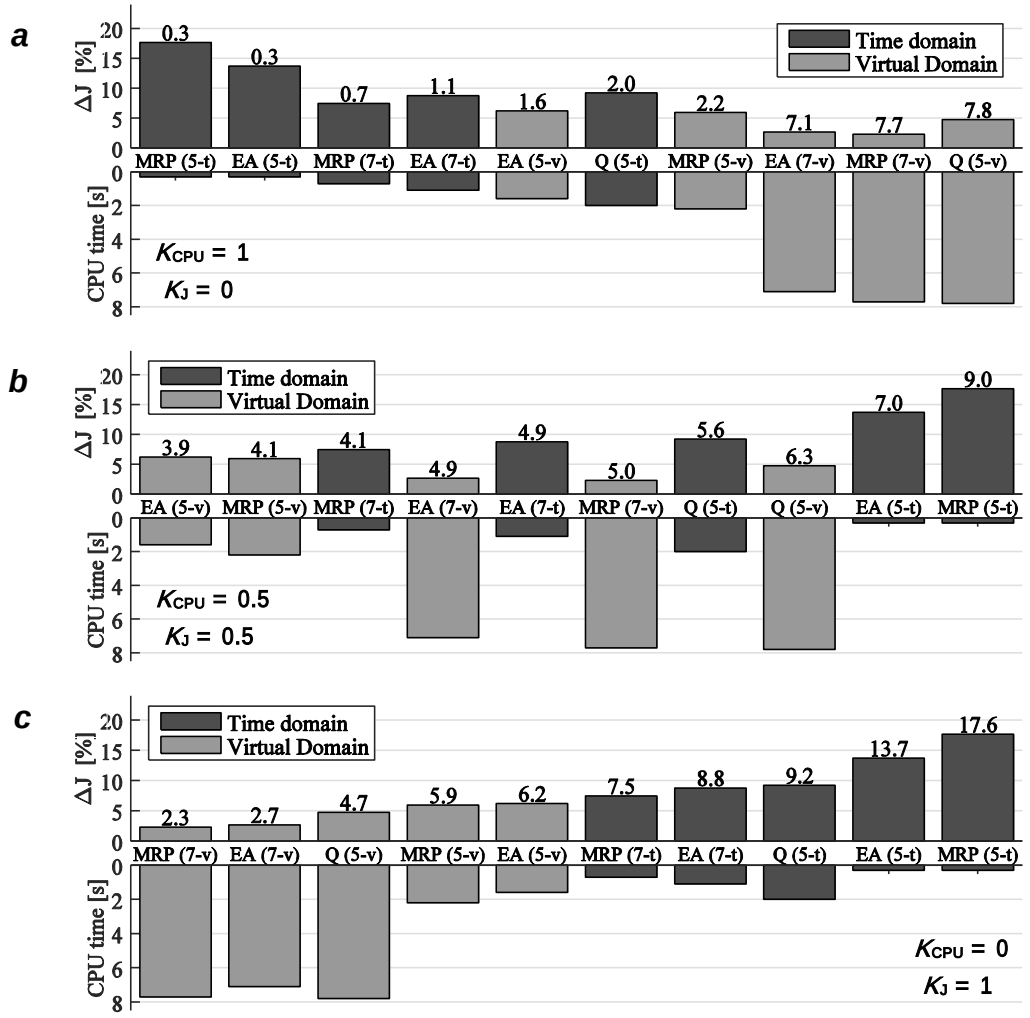

Figure 4.3: Performance of IDVD for the minimum time 180 deg reorientation. The symbol Q denotes the quaternion and EA indicates the Euler angles. The order of IDVD and domain are indicated in parenthesis. The graphs are ordered according to the performance index (see Eq. (4.47)), for several weighting coefficients. The performance index is indicated above each bar.

Figure 4.3 c) shows the performance of IDVD ordered according to the weights $K_J = 1$ and $K_{CPU} = 0$. In this case, the performance criteria is the optimality of the solution. The best performance are obtained with high-order polynomials defined in the virtual domain, due to the high number of variables to be optimized.

Additional test scenarios have been generated by varying inertia, control bounds and reorientation angles using the Monte Carlo method. The symbol $x_{A\text{-}B}$ denotes a random value of the parameter x in the range $[A, B]$. The generated scenarios have been analyzed using IDVD as in the previous section.

The first guesses on the controls at endpoints are taken as the endpoints controls computed by GPOPS. The first guesses for the maneuver time of Scenario 1, Scenario 2 and Scenario 3 are 6 s, 1 s and 1 s respectively.

Table 4.2: Reference scenarios generated with Monte Carlo analysis.

Scenario	Inertia	$\boldsymbol{T}_{\max}$	$\boldsymbol{\theta}_f$ (deg)	Number of cases
1	diag([$I_{1\text{-}10}$, $I_{1\text{-}10}$, $I_{1\text{-}10}$])	[1,1,1]	[0,0,180]	500
2	diag([1,1,1])	[$T_{1\text{-}10}$, $T_{1\text{-}10}$, $T_{1\text{-}10}$]	[0,0,180]	500
3	diag([1,1,1])	[1,1,1]	[$\theta_{\text{-}180\text{-}180}$, $\theta_{\text{-}180\text{-}180}$, $\theta_{\text{-}180\text{-}180}$]	1000

Table 4.3 summarizes the performance of IDVD. The symbol σ_x denotes the standard deviation of the data x, whereas Conv. indicates the converge percentage. Similarly to the analysis of the minimum time 180 deg reorientation of a symmetric spacecraft of the previous section, the virtual domain improves the solution quality, but at the cost of more computational time, for any polynomial order and attitude parameter considered. Moreover, all the solutions obtained with IDVD are sub-optimal, since the optimal bang-bang nature of the controls cannot be achieved due to the nature of the method.

The virtual domain also affects the convergence percentage. In the case of IDVD applied to time domain, all the algorithms converge to a sub-optimal solution (100% of convergence percentage). In the case of IDVD applied to virtual domain, the convergence percentage is always below 100%. The reason is that the polynomial coefficients of $\lambda(\tau)$ are not related to any state variable or physical behavior of the spacecraft and therefore the guesses of these parameters may not be accurate enough to ensure convergence of the optimization solver. Lowest convergence rates are observed with seventh-order IDVD applied to virtual domain, due to the higher number of variable parameters, which require an accurate first guess to ensure convergence of the solver. The attitude representation affects the solution quality and computational time. As in the ideal 180 deg reorientation scenario, the average optimal cost closest to the one computed from GPOPS is obtained with a seventh-order IDVD applied to MRP in the virtual domain.

All the tested methods present large standard deviation for ΔJ. In fact, *fmincon* is a local optimization solver and therefore an inaccurate first guess on the variable of the problem may cause convergence to a solution not even close to the one obtained with GPOPS. The first guess effects also the computational time to obtain a solution, since a bad first guess increases the number of iterations required by the optimization solver to converge.

Figure 4.4 shows the performance of IDVD based on the performance index MI, for different combinations of weighting coefficients. Similarly to the minimum time 180 deg reorientation, IDVD applied to time domain presents the best performance when the criteria is the computational speed (Figure

4.4 a)), K_J = 0 and K_{CPU} = 1). This behavior is due to the lower number of variables to be optimized than IDVD applied to virtual domain, which results in fewer iterations of the solver to converge. Figure 4.4 b) shows the performance of IDVD ordered when the performance criteria is the computational speed and the optimality of the solution equally balanced (K_J = 0.5 and K_{CPU} = 0.5). The best performance is obtained with seventh-order IDVD applied to MRP in time domain, due to the solution quality and the high computational speed. Furthermore, IDVD applied to virtual domain does not present favorable performance as in Figure 4.3 b) due to the high computational time required to converge in some scenarios. Figure 4.4 c) shows the performance of IDVD ordered according to the weights K_J = 1 and K_{CPU} = 0. In this case, the performance criteria is the optimality of the solution. The best performance are obtained with seventh-order and fifth-order IDVD applied MRP in virtual domain. However, IDVD applied to virtual domain does not always offer better performance than IDVD in time domain due to the sensitivity to the first guess (compare Figure 4.4 c) with Figure 4.3 b)). As already observed, the first guess on the polynomial coefficients of $\lambda(\tau)$ is not straightforward and an inaccurate guess may cause convergence to a non-optimal solution.

Table 4.3: Performance of IDVD in case of pseudorandom reorientation test scenarios. The standard deviation is denoted with σ.

Attitude	Method	Conv. (%)	ΔJ (%)	$\sigma_{\Delta J}$ (%)	$CPUt$ (s)	σ_{CPUt} (s)
MRP	IDVD 5th - time	100	19.0	21.8	0.4	0.2
	IDVD 5th - virtual	99.9	8.7	15.1	3.1	1.5
	IDVD 7th – time	100	10.0	16.2	1.7	1.1
	IDVD 7th – virtual	97.4	6.4	13.8	15.0	5.5
Euler angles	IDVD 5th - time	100	27.2	31.2	0.4	0.7
	IDVD 5th - virtual	98.1	18.7	26.7	4.0	15.6
	IDVD 7th – time	100	20.7	27.8	1.9	1.8
	IDVD 7th – virtual	95.7	14.9	22.6	15.6	21.2
Quaternion	IDVD 5th - time	100	14.7	17.2	2.8	1.5
	IDVD 5th - virtual	99.7	12.0	20.1	13.3	8.2

Table 4.4: Final errors on angular velocity and attitude after propagation of the controls obtained from IDVD in case of pseudorandom reorientation test scenarios.

Attitude	Method	$\Delta\omega$ (deg/s)	$\sigma_{\Delta\omega}$ (deg/s)	$\Delta\theta$ (deg)	$\sigma_{\Delta\theta}$ (deg)
MRP	IDVD 5th – time	1.4e-2	1.7e-2	2.5e-1	3.6e-1
	IDVD 5th – virtual	9.9e-2	2.7e-2	1.2e-1	2.2e-1
	IDVD 7th – time	9.4e-3	1.1e-2	2.5e-1	3.6e-1
	IDVD 7th – virtual	1.5e-1	2.9e-1	2.3e-1	4.9e-1
Euler angles	IDVD 5th – time	1.0e-2	1.3e-2	2.2e-1	3.5e-1
	IDVD 5th - virtual	3.7e-1	7.2e-1	1.9e-1	3.7e-1
	IDVD 7th – time	1.0e-2	1.1e-2	2.4e-1	3.5e-1
	IDVD 7th – virtual	8.2e-1	1.6	3.7e-1	7.4e-1
Quaternion	IDVD 5th – time	1.1e-2	1.1e-2	2.2e-1	3.4e-1
	IDVD 5th - virtual	3.2e-1	3.2e-1	9.0e-2	2.7e-1

4.8 Performance for Minimum Energy Maneuvers

The ideal 180 deg reorientation about the z-axis of a symmetric spacecraft is analyzed for the minimum energy problem. The endpoint conditions are defined in Eq. (4.48).

The first guesses for the dimensionless initial and final controls are [0 0 0.25] and [0 0 -0.25] respectively. Accordingly, the guesses on the polynomial coefficients related to angular accelerations are calculated with Eqs. (4.16, 4.17, 4.24, 4.25, 4.36, 4.37, 4.41, 4.42). The guess on the maneuver duration is 10 s. The trapezoidal rule is employed to integrate the cost function in Eq. (4.6) using 100 equally spaced nodes in the virtual and time domains. To ensure convergence of the optimization algorithm, the speed factor is represented by a third-order polynomial.

As summarized in Table 4.5, the IDVD method closely matches the optimal solution computed by GPOPS in both time and virtual domains and for any considered attitude representation, with a maximum difference percentile of ΔJ = 0.22 % obtained with MRP parameterized in time domain. In fact, the optimal controls that minimize the maneuver energy do not saturate and their continuous shapes can be represented by polynomial functions. This behavior is showed in Figure 4.5. Here the controls obtained with GPOPS are compared to the ones obtained from IDVD applied to the Euler angles in time and virtual domains. As a consequence, the usage of the virtual domain is not necessary since it increases the computational time without improving the solution quality. However, the virtual domain slightly improves the solution quality in the case of IDVD applied to MRP.

Figure 4.5 also provides significant information about the maneuver: only the torque about the z-axis of the spacecraft is actuated and the consequent energy-optimal maneuver is an eigenaxis rotation about the same axis.

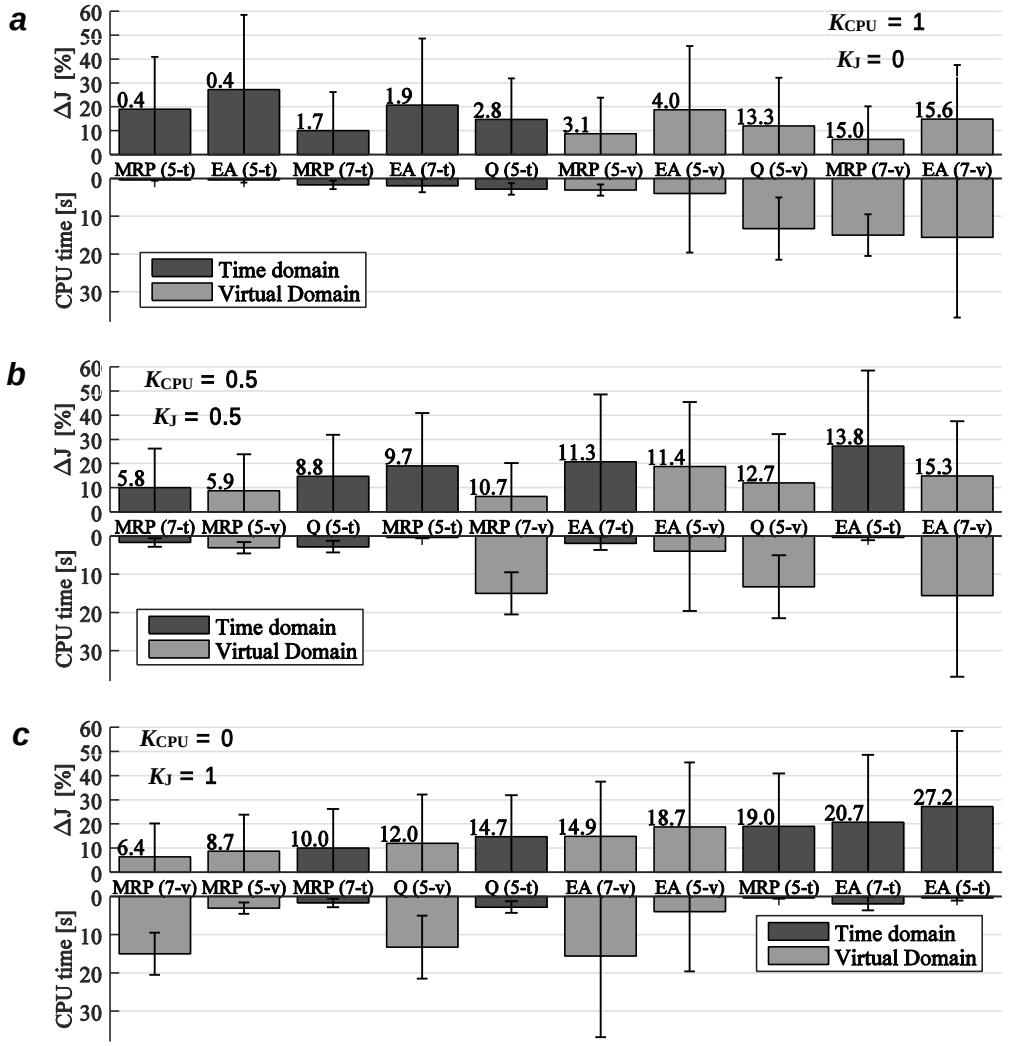

Figure 4.4: Performance of IDVD for several pseudorandom spacecraft geometries and maneuvers. The symbol Q denotes the quaternion and EA indicates the Euler angles. Order of IDVD and domain are indicated in parenthesis. The graphs are ordered according to the performance index (see Eq. (4.47)), for several weighting coefficients. The performance index is indicated above each bar. The vertical lines indicate the standard deviations.

Figure 4.6 summarizes the performance of IDVD based on the performance index *MI*, for different combinations of weighting coefficients. Independently of the choice of the weights, the IDVD method applied to the time domain presents the best performance, since the optimal solution can be achieved without the virtual domain. The presence of the virtual domain increases the computational time only and therefore the performances are lower.

Table 4.5: Optimization of a minimum energy rest to rest 180 deg reorientation of a symmetric spacecraft using IDVD and GPOPS. Controls obtained with the marked methods are represented in Figure 4.5.

Attitude	Method	J	$CPUt$ (s)	$\Delta\omega$ (deg/s)	$\Delta\theta$ (deg)	ΔJ (%)
Quaternion	GPOPS*	5.922e-2	198.8	2.8e-4	2.9e-4	0
MRP	IDVD 5th – time	5.935e-2	1.0	5.3e-6	8.0e-5	0.22
	IDVD 5th – virtual	5.923e-2	5.8	3.3e-4	2.9e-4	0.02
Euler angles	IDVD 5th – time*	5.923e-2	0.7	4.9e-11	1.8e-4	0.02
	IDVD 5th – virtual*	5.923e-2	1.7	4.4e-6	3.8e-4	0.02
Quaternion	IDVD 5th - time	5.923e-2	1.1	3.3e-12	1.8e-4	0.02
	IDVD 5th - virtual	5.923e-2	16.8	1.9e-5	3.3e-4	0.02

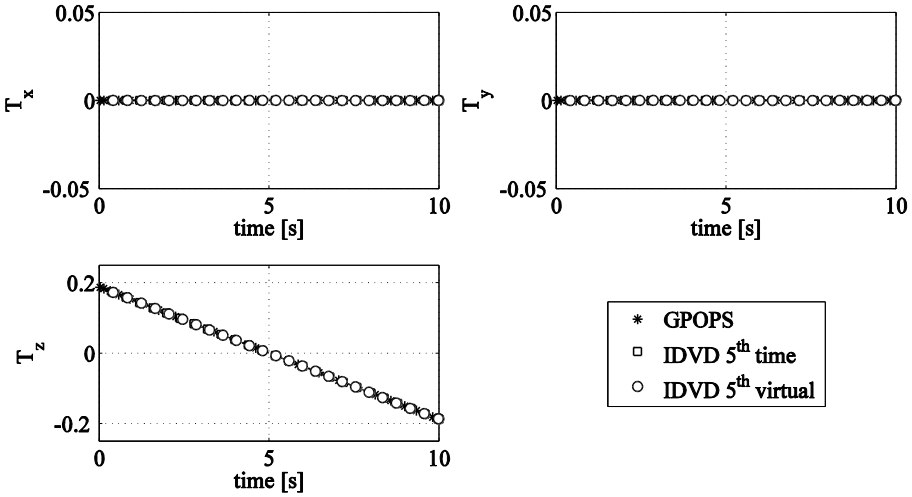

Figure 4.5: Optimal control time history generated with GPOPS and fifth-order IDVD applied to the Euler angles in virtual domain and time domain.

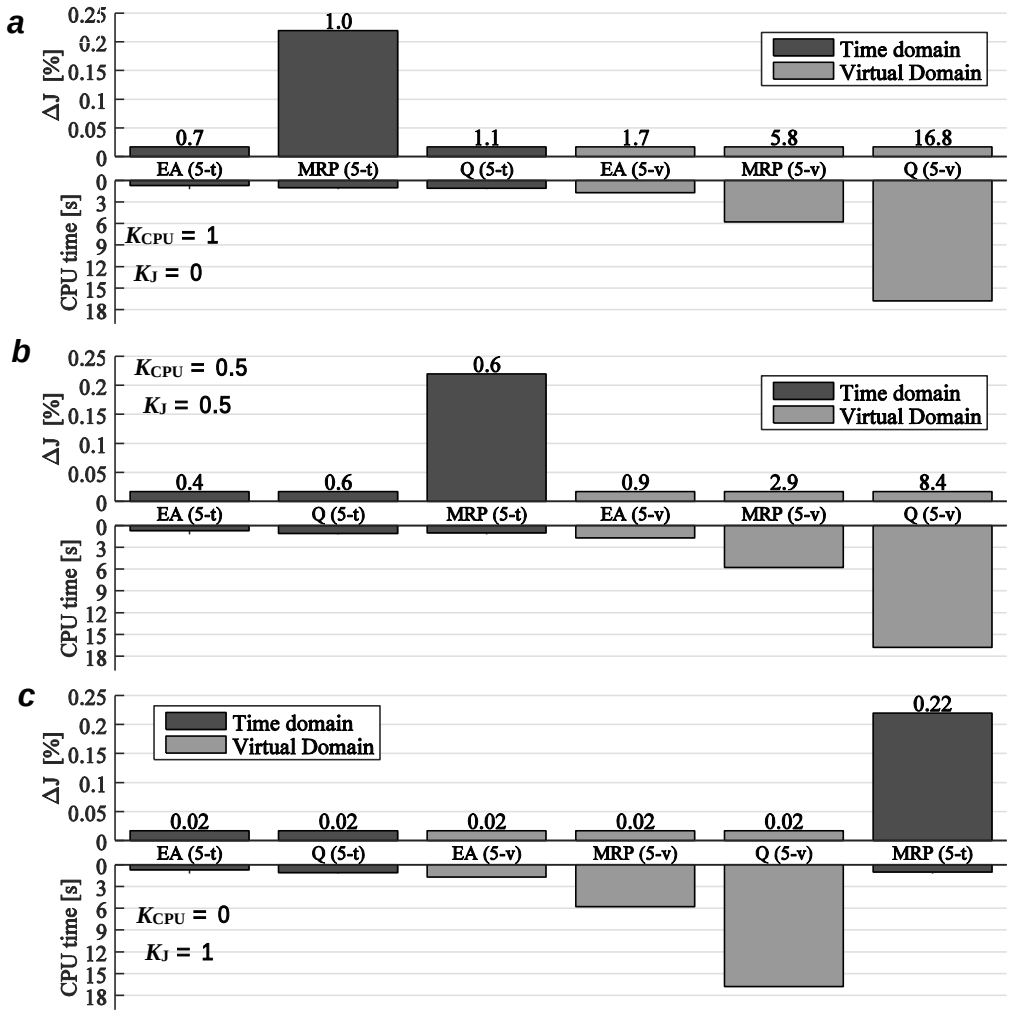

Figure 4.6: Performance of IDVD for the minimum energy 180 deg reorientation. The symbol Q denotes the quaternion and EA indicates the Euler angles. Order of IDVD and domain are indicated in parenthesis. The order of IDVD and domain are indicated in parenthesis. The graphs are ordered according to the performance index (see Eq. (4.46)), for several weighting coefficients. The performance index is indicated above each bar.

4.9 Choice of Attitude Parameters and Domain

The application of the inverse dynamics approach to rapid optimal attitude trajectory generation for spacecraft reorientation maneuvers has been investigated. The method has been applied to MPR, Euler angles (sequence 1-2-3) and quaternions with different polynomial orders defined in the time domain and in the virtual domain.

In the case of minimum time maneuvers, the optimal bang-bang shape of the controls cannot be achieved with IDVD owing to the nature of the

method. As matter of fact, IDVD imposes the trajectory components to follow polynomial shapes (C^2 function) and, therefore, only smooth controls can be achieved. The definition of the attitude trajectory in the virtual domain improves the solution quality with respect to the analogous formulation in the time domain since the speed factor increases the flexibility of the velocity along the trajectory. The speed factor, in fact, allows optimization of the magnitude of the velocity along the trajectory. However, the computational time increases due to the higher number of variables and constraints, since the speed factor is defined using a low-order polynomial. The attitude representation also influences the computational time and solution quality: quaternions require high computational time due to the logarithmic and exponential maps; MRP and Euler angles are faster since no constraints on the attitude history must be respected.

The choice of the trajectory parameterization for minimum time maneuvers depends on the requested performance. Looking for a quick computation of a sub-optimal solution, IDVD applied to time domain represents the most reasonable choice, since the limited number of variable parameters ensure rapid convergence to a solution, but at the cost of low optimality. Looking for the optimality of the solution, high-order polynomials defined in the virtual domain provide solutions closer to the ones obtained with a Gauss Pseudospectral method, but require accurate first guess to ensure 100% convergence. If the performance criteria takes into account both convergence speed and optimality of the solution, then high order IDVD used in the time domain or low-order IDVD used in the virtual domain should be considered and applied to the MRP.

In the case of the considered minimum energy maneuver, IDVD applied to the time domain computes the optimal solution found by the pseudospectral method GPOPS, since the optimal controls do not saturate and can be represented with continuous functions of the time.

The inverse dynamics approach is applicable as sub-optimal guidance for spacecraft reorientation maneuvers. For the considered minimum energy docking maneuvers, MRP represent the most suitable choice due to the absence of singularities in typical reorientation ranges and the favorable compromise between computational speed and solution quality. Notably, low order polynomials in time domain should be considered when the optimal controls do not saturate.

5 The Guidance Algorithm

5.1 Overview

This chapter of the thesis introduces an innovative fast and near-optimal trajectory planner for full DoF proximity operation and docking maneuvers. Following the results of the previous chapter, the attitude of the chaser spacecraft is expressed in terms of MRP. In order to drastically reduce the computational speed of the path planner and, therefore, allow onboard execution, the Inverse Dynamics method is applied in the time domain. This choice is also motivated by the fact that the optimal controls for the minimum energy problem do not saturate and can be represented by polynomials with high precision.

The main contribution of this work is a rapid trajectory generation algorithm for the full 6 DoF docking maneuver to an uncontrolled tumbling target achieved by combining the inverse dynamics approach and a sequential optimization algorithm; the methodology includes a procedure for the estimation of a feasible docking maneuver duration before the trajectory optimization. The proposed algorithm is implemented in closed-loop fashion to attain near-optimality of the resulting trajectory. Robustness of the guidance strategy to orbital perturbations, sensor noise and model uncertainty is tested through a Monte Carlo analysis since these issues are critical for realistic applications. Through this study, the applicability of the proposed guidance to noncooperative docking maneuvers is evaluated.

5.2 Open Loop Trajectory Optimization

The guidance algorithm proposed in the following employs the inverse dynamics method for rapid suboptimal trajectory generation. As discussed in Chapters 3 and 4, the optimal control problem is converted into an equivalent nonlinear programming problem by describing the chaser trajectory components with a set of interpolating functions defined in the time domain [36]. Once again, the choice of defining the polynomials in the time domain instead of the virtual domain is motivated by the fact that inverse dynamics is capable to compute minimum energy trajectories close to the optimal ones in a smaller amount of time if compared to IDVD, allowing onboard execution. By inverting the dynamics model of the system, each state variable and each control component can be expressed as function in the time domain

of the parameterized trajectory and its derivatives. In turn, also the integrand of performance index J of the optimal control problem can be expressed as function of the parameterized trajectory and, therefore, it can be minimized through the optimization of the unconstrained polynomial coefficients.

Following the results of Chapter 4, the attitude of the chaser spacecraft with respect to the Hill's coordinate system is described in terms of MRP $\boldsymbol{\sigma}^{\mathrm{C}} = [\sigma_1^{\mathrm{C}}, \sigma_2^{\mathrm{C}}, \sigma_3^{\mathrm{C}}]$ according to the following kinematic equation [53]

$$\begin{bmatrix} \dot{\sigma}_1^{\mathrm{C}} \\ \dot{\sigma}_2^{\mathrm{C}} \\ \dot{\sigma}_3^{\mathrm{C}} \end{bmatrix} = \frac{1}{4}\boldsymbol{B}(\boldsymbol{\sigma}^{\mathrm{C}})\,{}^{\mathrm{H}}\boldsymbol{\omega}^{C}, \tag{5.1}$$

where

$$\boldsymbol{B}(\boldsymbol{\sigma}^{\mathrm{C}}) = \begin{bmatrix} 1+\sigma_1^{\mathrm{C}2}-\sigma_2^{\mathrm{C}2}-\sigma_3^{\mathrm{C}2} & 2(\sigma_1^{\mathrm{C}}\sigma_2^{\mathrm{C}}-\sigma_3^{\mathrm{C}}) & 2(\sigma_1^{\mathrm{C}}\sigma_3^{\mathrm{C}}+\sigma_2^{\mathrm{C}}) \\ 2(\sigma_1^{\mathrm{C}}\sigma_2^{\mathrm{C}}+\sigma_3^{\mathrm{C}}) & 1-\sigma_1^{\mathrm{C}2}+\sigma_2^{\mathrm{C}2}-\sigma_3^{\mathrm{C}2} & 2(\sigma_2^{\mathrm{C}}\sigma_3^{\mathrm{C}}-\sigma_1^{\mathrm{C}}) \\ 2(\sigma_1^{\mathrm{C}}\sigma_3^{\mathrm{C}}-\sigma_2^{\mathrm{C}}) & 2(\sigma_2^{\mathrm{C}}\sigma_3^{\mathrm{C}}+\sigma_1^{\mathrm{C}}) & 1-\sigma_1^{\mathrm{C}2}-\sigma_2^{\mathrm{C}2}+\sigma_3^{\mathrm{C}2} \end{bmatrix}.$$

The components ${}^{\mathrm{H}}\omega_x^{\mathrm{C}}$, ${}^{\mathrm{H}}\omega_y^{\mathrm{C}}$, ${}^{\mathrm{H}}\omega_z^{\mathrm{C}}$ of the chaser spacecraft angular velocity with respect to the Hill's coordinate system and expressed in the chaser principal system C are obtained from $\boldsymbol{\omega}^{\mathrm{C}}$ with

$$\begin{bmatrix} {}^{\mathrm{H}}\omega_x^{\mathrm{C}} \\ {}^{\mathrm{H}}\omega_y^{\mathrm{C}} \\ {}^{\mathrm{H}}\omega_z^{\mathrm{C}} \end{bmatrix} = \begin{bmatrix} \omega_x^{\mathrm{C}} \\ \omega_y^{\mathrm{C}} \\ \omega_z^{\mathrm{C}} \end{bmatrix} - \boldsymbol{R}_{\mathrm{H}}^{\mathrm{C}} \begin{bmatrix} 0 \\ 0 \\ \Omega \end{bmatrix}, \tag{5.2}$$

where

$$\boldsymbol{R}_{\mathrm{H}}^{\mathrm{C}} = \frac{1}{\left(1+\boldsymbol{\sigma}^{\mathrm{C}}\cdot\boldsymbol{\sigma}^{\mathrm{C}}\right)^2} \begin{bmatrix} 4\left(\sigma_1^{\mathrm{C}2}-\sigma_2^{\mathrm{C}2}-\sigma_3^{\mathrm{C}2}\right)+\Sigma^2 & 8\sigma_1^{\mathrm{C}}\sigma_2^{\mathrm{C}}+4\sigma_3^{\mathrm{C}}\Sigma & 8\sigma_1^{\mathrm{C}}\sigma_3^{\mathrm{C}}-4\sigma_2^{\mathrm{C}}\Sigma \\ 8\sigma_1^{\mathrm{C}}\sigma_2^{\mathrm{C}}-4\sigma_3^{\mathrm{C}}\Sigma & 4\left(-\sigma_1^{\mathrm{C}2}+\sigma_2^{\mathrm{C}2}-\sigma_3^{\mathrm{C}2}\right)+\Sigma^2 & 8\sigma_2^{\mathrm{C}}\sigma_3^{\mathrm{C}}+4\sigma_1^{\mathrm{C}}\Sigma \\ 8\sigma_1^{\mathrm{C}}\sigma_3^{\mathrm{C}}+4\sigma_2^{\mathrm{C}}\Sigma & 8\sigma_2^{\mathrm{C}}\sigma_3^{\mathrm{C}}-4\sigma_1^{\mathrm{C}}\Sigma & 4\left(-\sigma_1^{\mathrm{C}2}-\sigma_2^{\mathrm{C}2}+\sigma_3^{\mathrm{C}2}\right)+\Sigma^2 \end{bmatrix} \tag{5.3}$$

is the direction cosine matrix from the Hill's system H to the chaser principal system C in terms of MRP and

$$\Sigma = 1-\boldsymbol{\sigma}^{\mathrm{C}}\cdot\boldsymbol{\sigma}^{\mathrm{C}}. \tag{5.4}$$

The relative translational dynamics between chaser and target is given by the C-W model in Eq. (3.1). Furthermore, the rotational dynamics of the target vehicle is described by Eqs. (3.3, 3.7). This leads to the following state vector of the two spacecraft system:

$$\boldsymbol{x} = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} & \boldsymbol{\omega}^{\mathrm{C}} & \boldsymbol{\sigma}^{\mathrm{C}} & \boldsymbol{\omega}^{\mathrm{T}} & \boldsymbol{q}^{\mathrm{T}} \end{bmatrix}. \tag{5.5}$$

Path constraints and docking enabling conditions are defined in Section 3.2.

5.2.1 Translational Trajectory

Following the inverse dynamics approach, the translational trajectory of the chaser spacecraft is represented by a set of functions defined in the time domain. Among all the possible parameterization functions (such as B-splines, Hermite polynomials, Bezier curves), we choose polynomials since they ensure near-optimality of the translational trajectory (when the path planner is employed in closed loop) and fast computational speed [38, 76, 77]. Therefore, the trajectory of the chaser center of mass in the Hill's coordinate system is represented by a set of polynomials of order n defined in the time domain

$$x(t) = \sum_{i=0}^{n} b_{1,i} t^i, \quad y(t) = \sum_{i=0}^{n} b_{2,i} t^i, \quad z(t) = \sum_{i=0}^{n} b_{3,i} t^i, \tag{5.6}$$

which imply

$$\dot{x}(t) = \sum_{i=1}^{n} i b_{1,i} t^{i-1}, \quad \dot{y}(t) = \sum_{i=1}^{n} i b_{2,i} t^{i-1}, \quad \dot{z}(t) = \sum_{i=1}^{n} i b_{3,i} t^{i-1}, \tag{5.7}$$

$$\ddot{x}(t) = \sum_{i=2}^{n} i(i-1) b_{1,i} t^{i-2}, \quad \ddot{y}(t) = \sum_{i=2}^{n} i(i-1) b_{2,i} t^{i-2}, \quad \ddot{z}(t) = \sum_{i=2}^{n} i(i-1) b_{3,i} t^{i-2} \tag{5.8}$$

As mentioned before, certain polynomial coefficients such as $b_{1,i}$, $b_{2,i}$ and $b_{3,i}$ can be chosen in such a way that the boundary conditions on chaser position and velocity are systematically enforced. By imposing in Eqs. (5.6, 5.7) the initial values x_0, y_0, z_0 and $\dot{x}_0$, $\dot{y}_0$, $\dot{z}_0$ at $t = 0$ s, we obtain the following expressions for the coefficients $b_{1,0}$, $b_{2,0}$, $b_{3,0}$, $b_{1,1}$, $b_{2,1}$, and $b_{3,1}$:

$$\begin{aligned} b_{1,0} &= x_0 \quad b_{2,0} = y_0 \quad b_{3,0} = z_0 \\ b_{1,1} &= \dot{x}_0 \quad b_{2,1} = \dot{y}_0 \quad b_{3,1} = \dot{z}_0 \end{aligned}. \tag{5.9}$$

Similarly, the expressions of the coefficients $b_{1,n}$, $b_{2,n}$, $b_{3,n}$, $b_{1,n\text{-}1}$, $b_{2,n\text{-}1}$, and $b_{3,n\text{-}1}$ are obtained by imposing the final values x_f, y_f, z_f and $\dot{x}_\mathrm{f}$, $\dot{y}_\mathrm{f}$, $\dot{z}_\mathrm{f}$ in Eqs. (5.6, 5.7) at $t = t_\mathrm{f}$. For the coefficients $b_{1,n}$ and $b_{1,n\text{-}1}$ of the x-component of the trajectory it results in

$$b_{1,n} = \frac{x_\mathrm{f} - \sum_{i=0}^{n-1} b_{1,i} t_\mathrm{f}^i}{t_\mathrm{f}^n}, \quad b_{1,n-1} = \frac{\dot{x}_\mathrm{f} - \sum_{i=0}^{n-2} i b_{1,i} t_\mathrm{f}^{i-1} - n b_{1,n} t_\mathrm{f}^{i-1}}{(n-1) t_\mathrm{f}^{n-2}} \tag{5.10}$$

whereas analogous expressions are obtained for $b_{2,n}$, $b_{3,n}$ $b_{2,n\text{-}1}$ $b_{3,n\text{-}1}$. Note that the initial conditions x_0, y_0, z_0 and $\dot{x}_0, \dot{y}_0, \dot{z}_0$ are directly provided by the estimation of the current chaser state. The final conditions $x_\mathrm{f}, y_\mathrm{f}, z_\mathrm{f}$ and $\dot{x}_\mathrm{f}, \dot{y}_\mathrm{f}, \dot{z}_\mathrm{f}$ are calculated from the final target state using Eqs. (3.8–3.11), where the final state of the system is estimated by propagating the dynamics of the docking dynamics model. This aspect will be discussed with more details in Section 5.4. As a consequence of this procedure, the polynomial coefficients $b_{j,0}$, $b_{j,1}$, $b_{j,n}$ and $b_{j,n\text{-}1}$ (j = 1,2,3) are determined, whereas the rest of the coefficients are varied parameters to be optimized. In order to allow the optimization of the trajectory, the order of each polynomial must satisfy the equation

$$n \geq N_\mathrm{B}, \tag{5.11}$$

where N_B is the number of boundary conditions imposed to a single trajectory component (by imposing endpoint position and velocity, it results in N_B = 4). In this way, at least one polynomial coefficient for each trajectory component is a varied parameter to be optimized, allowing flexibility on the shape of the trajectory. The higher the order n, the higher the number of varied parameters. On the other hand, to avoid the Runge's phenomena (oscillations at the edges of the time interval) typical of high order polynomials, the polynomial order must be limited to $n \leq 8$ [78].

By inverting Eq. (3.1), it yields to the following expression of the control thrusts expressed in the Hill's coordinate system:

$$\begin{aligned} F_x^\mathrm{H} &= m_\mathrm{C}\left(\ddot{x} - 2\Omega\dot{y} - 3\Omega^2 x\right) \\ F_y^\mathrm{H} &= m_\mathrm{C}\left(\ddot{y} + 2\Omega\dot{x}\right) \\ F_z^\mathrm{H} &= m_\mathrm{C}\left(\ddot{z} + \Omega^2 z\right) \end{aligned} . \tag{5.12}$$

The procedure provides analytical expressions in the time domain of translational velocities, accelerations and control thrusts. In particular, the expressions of the controls are in the following form:

$$\begin{aligned} &F_j^\mathrm{H} = F_j^\mathrm{H}\left(x_0, \dot{x}_0, x_\mathrm{f}, \dot{x}_\mathrm{f} y_0, \dot{y}_0, y_\mathrm{f}, \dot{y}_\mathrm{f}, b_{1,2}, \ldots, b_{1,n-2}, b_{2,n-2}, m_\mathrm{C}, \Omega, t_\mathrm{f}, t\right), \\ &j = x,y \;\; F_z^\mathrm{H} = F_z^\mathrm{H}\left(z_0, \dot{z}_0, z_\mathrm{f}, \dot{z}_\mathrm{f}, b_{3,2}, \ldots, b_{3,n-2}, m_\mathrm{C}, \Omega, t_\mathrm{f}, t\right), \qquad t \in \left[0, t_\mathrm{f}\right]. \end{aligned} \tag{5.13}$$

These expressions involve a set of given parameters $x_0, \dot{x}_0, y_0, \dot{y}_0, z_0, \dot{z}_0, x_\mathrm{f}, \dot{x}_\mathrm{f}, y_\mathrm{f}, \dot{y}_\mathrm{f}, z_\mathrm{f}, \dot{z}_\mathrm{f}$, m_C, Ω, t_f and a set of varied parameters $b_{1,2}$, $b_{2,2}$, $b_{3,2}$,…, $b_{1,n\text{-}2}$, $b_{2,n\text{-}2}$, $b_{3,n\text{-}2}$ to be optimized.

5.2.2 Attitude Trajectory

Regarding the attitude trajectory of the chaser spacecraft, we proceed similarly to the procedure described for the translation; namely, the attitude trajectory of the chaser principal coordinate system C with respect to the Hill's system in terms of MRP is represented with a set of polynomials in the time domain:

$$\sigma_j^{\mathrm{C}}(t) = \sum_{i=0}^{n} a_{j,i} t^i \qquad j = 1,2,3. \tag{5.14}$$

Also for the parameterization of the translation, we choose low-order polynomials rather than more sophisticated functions such as B-splines since they ensure the best trade-off between optimality and computational speed [40,76]. Accordingly, the time derivatives of the trajectory components are given by

$$\dot{\sigma}_j^{\mathrm{C}}(t) = \sum_{i=1}^{n} i a_{j,i} t^{i-1} \qquad j = 1,2,3 \tag{5.15}$$

$$\ddot{\sigma}_j^{\mathrm{C}}(t) = \sum_{i=2}^{n} i(i-1) a_{j,i} t^{i-2} \qquad j = 1,2,3. \tag{5.16}$$

Also in this case the polynomial coefficients $a_{j,i}$ can be obtained by imposing the boundary conditions $\boldsymbol{\sigma}_0^{\mathrm{C}}$, $\dot{\boldsymbol{\sigma}}_0^{\mathrm{C}}$, $\boldsymbol{\sigma}_{\mathrm{f}}^{\mathrm{C}}$ and $\dot{\boldsymbol{\sigma}}_{\mathrm{f}}^{\mathrm{C}}$ to Eqs. (5.14, 5.15). The expressions for the coefficients $a_{j,0}$, $a_{j,1}$, $a_{j,n}$ and $a_{j,n-1}$ are given by

$$a_{j,0} = \sigma_{j0}^{\mathrm{C}}, \quad a_{j,1} = \dot{\sigma}_{j0}^{\mathrm{C}}, \qquad j = 1,2,3 \tag{5.17}$$

$$a_{j,n} = \frac{\sigma_{j_{\mathrm{f}}}^{\mathrm{C}} - \sum_{i=0}^{n-1} a_{j,i} t_{\mathrm{f}}^i}{t_{\mathrm{f}}^n}, \quad a_{j,n-1} = \frac{\dot{\sigma}_{j_{\mathrm{f}}}^{\mathrm{C}} - \sum_{i=0}^{n-2} i a_{j,i} t_{\mathrm{f}}^{i-1} - n a_{j,n} t_{\mathrm{f}}^{i-1}}{(n-1) t_{\mathrm{f}}^{n-2}} \qquad j = 1,2,3. \tag{5.18}$$

By inverting Eq. (4.4), we obtain the following expression of the angular velocity ${}^{\mathrm{H}}\boldsymbol{\omega}^{\mathrm{C}}$:

$${}^{\mathrm{H}}\boldsymbol{\omega}^{\mathrm{C}} = 4\boldsymbol{B}\left(\boldsymbol{\sigma}^{\mathrm{C}}\right)^{-1} \dot{\boldsymbol{\sigma}}^{\mathrm{C}}, \tag{5.19}$$

where the inverse of the matrix $\boldsymbol{B}(\boldsymbol{\sigma}^{\mathrm{C}})$ is given by [53]

$$\boldsymbol{B}\left(\boldsymbol{\sigma}^{\mathrm{C}}\right)^{-1}=\frac{1}{\left(1+\left\|\boldsymbol{\sigma}^{\mathrm{C}}\right\|^{2}\right)^{2}}\boldsymbol{B}\left(\boldsymbol{\sigma}^{\mathrm{C}}\right)^{\mathrm{T}}. \tag{5.20}$$

The time derivative of Eq. (5.19) provides the expressions of the angular acceleration with respect to the Hill's coordinate system [36, 53]:

$${}^{\mathrm{H}}\dot{\boldsymbol{\omega}}^{\mathrm{C}}=\frac{1}{\left(1+\left\|\boldsymbol{\sigma}^{\mathrm{C}}\right\|^{2}\right)^{2}}\boldsymbol{B}\left(\boldsymbol{\sigma}^{\mathrm{C}}\right)^{\mathrm{T}}\left[4\ddot{\boldsymbol{\sigma}}^{\mathrm{C}}-\dot{\boldsymbol{B}}\left(\boldsymbol{\sigma}^{\mathrm{C}}\right){}^{\mathrm{H}}\boldsymbol{\omega}^{\mathrm{C}}\right]. \tag{5.21}$$

The expressions of the control torques are finally obtained by inverting the dynamics expressed in Eq. (3.2):

$$\begin{aligned}
T_x^{\mathrm{C}}&=I_{xx}^{\mathrm{C}}\dot{\omega}_x^{\mathrm{C}}+\left(I_{zz}^{\mathrm{C}}-I_{yy}^{\mathrm{C}}\right)\omega_z^{\mathrm{C}}\omega_y^{\mathrm{C}}\\
T_y^{\mathrm{C}}&=I_{yy}^{\mathrm{C}}\dot{\omega}_y^{\mathrm{C}}+\left(I_{xx}^{\mathrm{C}}-I_{zz}^{\mathrm{C}}\right)\omega_z^{\mathrm{C}}\omega_x^{\mathrm{C}},\\
T_z^{\mathrm{C}}&=I_{zz}^{\mathrm{C}}\dot{\omega}_z^{\mathrm{C}}+\left(I_{yy}^{\mathrm{C}}-I_{xx}^{\mathrm{C}}\right)\omega_x^{\mathrm{C}}\omega_y^{\mathrm{C}}
\end{aligned} \tag{5.22}$$

where the components of $\boldsymbol{\omega}^{\mathrm{C}}$ are obtained from Eq. (5.2). Similarly, the components of $\dot{\boldsymbol{\omega}}^{\mathrm{C}}$, which represents the angular acceleration vector with respect to the inertial reference frame, are obtained from ${}^{\mathrm{H}}\dot{\boldsymbol{\omega}}^{\mathrm{C}}$ using the time derivative of Eq. (5.2):

$$\begin{bmatrix}\dot{\omega}_x^{\mathrm{C}}\\ \dot{\omega}_y^{\mathrm{C}}\\ \dot{\omega}_z^{\mathrm{C}}\end{bmatrix}=\begin{bmatrix}{}^{\mathrm{H}}\dot{\omega}_x^{\mathrm{C}}\\ {}^{\mathrm{H}}\dot{\omega}_y^{\mathrm{C}}\\ {}^{\mathrm{H}}\dot{\omega}_z^{\mathrm{C}}\end{bmatrix}+\dot{\boldsymbol{R}}_{\mathrm{H}}^{\mathrm{C}}\begin{bmatrix}0\\0\\\Omega\end{bmatrix}. \tag{5.23}$$

This procedure allows obtaining analytical expressions in the time domain of angular velocities, accelerations and control torques. In particular, the expressions of the control torques are in the following form:

$$\begin{aligned}
T_j^{\mathrm{C}}=T_j^{\mathrm{C}}\Big(&\boldsymbol{\sigma}_0^{\mathrm{C}},\dot{\boldsymbol{\sigma}}_0^{\mathrm{C}},\boldsymbol{\sigma}_{\mathrm{f}}^{\mathrm{C}},\dot{\boldsymbol{\sigma}}_{\mathrm{f}}^{\mathrm{C}},a_{1,2},a_{2,2},a_{3,2},\ldots,a_{1,n-2},a_{2,n-2},a_{3,n-2},\ldots\\
&\ldots I_{xx}^{\mathrm{C}},I_{yy}^{\mathrm{C}},I_{zz}^{\mathrm{C}},\Omega,t_{\mathrm{f}},t\Big)\quad t\in\left[0,t_{\mathrm{f}}\right]\quad j=x,y,z
\end{aligned}. \tag{5.24}$$

In these expressions, $\boldsymbol{\sigma}_0^{\mathrm{C}}$, $\dot{\boldsymbol{\sigma}}_0^{\mathrm{C}}$, $\boldsymbol{\sigma}_{\mathrm{f}}^{\mathrm{C}}$, $\dot{\boldsymbol{\sigma}}_{\mathrm{f}}^{\mathrm{C}}$, Ω, I_{xx}^{C}, I_{yy}^{C}, I_{zz}^{C}, t_{f} are known parameters, whereas $a_{1,2}$, $a_{2,2}$, $a_{3,2}$,…, $a_{1,n\text{-}2}$, $a_{2,n\text{-}2}$, $a_{3,n\text{-}2}$ are the varied parameters.

5.2.3 Trajectory Optimization

At this point, the time domain $[0;\ t_{\mathrm{f}}]$ is discretized into N intervals with N+1 equidistant temporal nodes and the performance index J of the optimal control problem in Eq. (3.15) is integrated numerically using the expressions

of the controls in Eqs. (5.12, 5.22). This procedure allows transforming J into a (nonlinear) function of the given parameters and the varied parameters:

$$J = J\left(\boldsymbol{\sigma}_0^{\mathrm{C}}, \dot{\boldsymbol{\sigma}}_0^{\mathrm{C}}, \boldsymbol{\sigma}_{\mathrm{f}}^{\mathrm{C}}, \dot{\boldsymbol{\sigma}}_{\mathrm{f}}^{\mathrm{C}}, x_0, \dot{x}_0, x_{\mathrm{f}}, \dot{x}_{\mathrm{f}} y_0, \dot{y}_0, y_{\mathrm{f}}, \dot{y}_{\mathrm{f}}, z_0, \dot{z}_0, z_{\mathrm{f}}, \dot{z}_{\mathrm{f}}, \right.$$
$$\left. a_{j,2}, b_{j,2}, \ldots, a_{j,n-2}, b_{j,n-2}, m_{\mathrm{C}}, I_{xx}^{\mathrm{C}}, I_{yy}^{\mathrm{C}}, I_{zz}^{\mathrm{C}}, \Omega, t_{\mathrm{f}} \right), \quad j = 1,2,3. \tag{5.25}$$

Consequently, the initial optimal control problem is converted into the following equivalent nonlinear programming problem: Given the set of known parameters $x_0, \dot{x}_0, y_0, \dot{y}_0, z_0, \dot{z}_0, x_{\mathrm{f}}, \dot{x}_{\mathrm{f}}, y_{\mathrm{f}}, \dot{y}_{\mathrm{f}}, z_{\mathrm{f}}, \dot{z}_{\mathrm{f}}$, m_{C}, $\boldsymbol{\sigma}_0^{\mathrm{C}}$, $\dot{\boldsymbol{\sigma}}_0^{\mathrm{C}}$, $\boldsymbol{\sigma}_{\mathrm{f}}^{\mathrm{C}}$, $\dot{\boldsymbol{\sigma}}_{\mathrm{f}}^{\mathrm{C}}$, Ω, I_{xx}^{C}, $I_{yy}^{\mathrm{C}}, I_{zz}^{\mathrm{C}}$, t_{f} minimize the function J in Eq. (5.25) with respect to the varied parameters $a_{1,2}$, $a_{2,2}$, $a_{3,2}$, $b_{1,2}$, $b_{2,2}$, $b_{3,2}$,…, $b_{2,n\text{-}2}$, $b_{3,n\text{-}2}$, subjected to the constraint Eqs. (3.12–3.14) evaluated at each temporal node.

It is important to note that the constraints are enforced at finite number of points, being the temporal nodes. In addition, the discrete approximation of the continuous function J in Eq. (3.15) depends on the choice of the integration technique along with the number of temporal nodes.

This constrained optimization problem is solved using SNOPT, which employs a sequential quadratic programming algorithm to optimize the varied parameters of the problem [79]. This solver has been selected because it is able to solve efficiently problems with a limited number of varied parameters and gradients of the constraints difficult to evaluate [80].

Before starting the optimization process, the rotational dynamics of the target, i.e. Eq. (3.3) and Eq. (3.7) applied to the target, are integrated and the constraints in Eqs. (3.8–3.11) are then applied to calculate the chaser's final state, required to impose the endpoint conditions of the trajectory. Note that the maneuver duration t_{f} is assumed to be a fixed parameter of the optimization problem and it is estimated before the first trajectory optimization loop using the algorithm described in Section 5.3. This choice is driven by the need to further decrease the computational speed required by the solver to converge to an optimal solution [36]. Conversely, leaving the maneuver duration as varied parameter requires the determination of the final state at every iteration of the solver, which reflects in lower computational speed.

5.3 Estimation of the Maneuver Time

This chapter of the thesis describes the pseudocode for the onboard estimation of a feasible maneuver duration, which is calculated in the guidance block during the first execution of the control loop, as discussed in Section 5.4. In the proposed iterative method, a linear acceleration profile is imposed along each axis of the Hill's coordinate system. Initially, the acceleration profile is defined on the axis corresponding to the maximum chaser-target

distance assuming the action of the maximum available thrust at the beginning and at the end of the maneuver. Subsequently, by propagating the translation dynamics we verify that the maximum thrust engaged in all components is lower than the maximum thrust available. If this condition is unverified, an acceleration profile, with lower magnitude is considered and the process is repeated. Once the constraint on the maximum thrust level is verified in all the three axes, we compute the first estimate of the maneuver duration using the obtained linear thrust profiles. However, we must ensure that the chaser docking point grapples the target docking point at the end of the maneuver. Subsequently, we propagate the rotation dynamics of the target spacecraft for a time larger than the first estimation of the maneuver duration until the target docking point points the chaser at the end of the maneuver. The constraint on the maximum torque level is verified in the same way as for the control thrust.

The case of a rest to rest R-bar approach is presented here, but the proposed procedure can be applied to other maneuvers by changing the endpoint conditions and the reference acceleration profile defined in *Step 1*.

Step 1: Impose the following reference linear acceleration profile along the x-axis of the Hill's coordinate system:

$$\ddot{x}(t) = a_x\left(1 - 2\frac{t}{t_\mathrm{f}}\right), \tag{5.26}$$

where

$$a_x = -\mathrm{sign}(x_0)K\frac{F_{x,\max}}{m_\mathrm{C}} \tag{5.27}$$

is the maximum acceleration amplitude and $K \in [0; 1]$ a weighting coefficient. Set a value for K.

Step 2: Impose the endpoint condition $x(t_\mathrm{f}) = 0$ m to the motion law $x(t)$ obtained from Eq. (5.26). Then compute the first estimate of the maneuver duration with

$$t_\mathrm{f}^* = \sqrt{-6\frac{x_0}{a_x}}\,. \tag{5.28}$$

Step 3: Impose a linear acceleration profile along the y and z axes of the Hill's coordinate system as in Eq. Eq. (5.26). The required initial accelerations to achieve $y(t^*_\mathrm{f}) = 0$ m and $z(t^*_\mathrm{f}) = 0$ m are

$$a_y = -6\frac{y_0}{t_\mathrm{f}^{*2}}, \qquad a_z = -6\frac{z_0}{t_\mathrm{f}^{*2}}. \tag{5.29}$$

Step 4: Evaluate the force components F_x^{H}, F_y^{H} and F_z^{H} at a given set of instants of time $t_i \in [0, t^*_{\mathrm{f}}]$ using Eq. (5.12) and the trajectory generated by the linear acceleration profiles. Verify the following conditions

$$\left\|F_x^{\mathrm{H}}\right\| \le F_{x,\max}, \qquad \left\|F_y^{\mathrm{H}}\right\| \le F_{y,\max}, \qquad \left\|F_z^{\mathrm{H}}\right\| \le F_{z,\max}. \tag{5.30}$$

If the conditions are verified for each considered instant of time, then go to *Step* 5, otherwise go back to *Step* 1 and decrease the weighting coefficient K.

Step 5: Set a value for the parameter $\Delta t_{\mathrm{f}} \ge 0$ and estimate the state of the target spacecraft at $t^* = t^*_{\mathrm{f}} + \Delta t_{\mathrm{f}}$ by propagating Eq. (3.3) and Eq. (3.7).

Step 6: Compare the final orientation of the coordinate system TD with the initial orientation of the system CD:

$$\boldsymbol{R}_{\mathrm{CD}}^{\mathrm{TD}} = \boldsymbol{R}_{\mathrm{CD}}^{\mathrm{C}} \boldsymbol{R}_{\mathrm{C}}^{\mathrm{H}}(0) \boldsymbol{R}_{\mathrm{H}}^{\mathrm{T}}(t^*) \boldsymbol{R}_{\mathrm{T}}^{\mathrm{TD}}. \tag{5.31}$$

If the relative orientation in Eq. (5.31) is lower than a maximum relative orientation defined by $\boldsymbol{R}_{\mathrm{CD}\max}^{\mathrm{TD}}$, then go to *Step* 7. Otherwise go to *Step* 5 and increase Δt_{f}. The matrix $\boldsymbol{R}_{\mathrm{CD}\max}^{\mathrm{TD}}$ was set using the Euler angles 1-2-3. In our experimental campaign, we limited the maximum relative orientation to ±30 deg for each angle. With this procedure, we ensure that 1) the target is in a favorable configuration for the docking at the end of the maneuver, i.e. the target's auxiliary coordinate system TD points the chaser 2) the final orientation of the system TD is similar to the initial orientation of the chaser, resulting in lower control expenditure.

Step 7: Impose the docking-enabling conditions in Eqs. (3.8–3.11) at $t = t^*$. Calculate the expected chaser angular velocity as

$$\begin{bmatrix} \omega_{x_{\mathrm{f}}}^{\mathrm{C}} \\ \omega_{y_{\mathrm{f}}}^{\mathrm{C}} \\ \omega_{z_{\mathrm{f}}}^{\mathrm{C}} \end{bmatrix} = \begin{bmatrix} {}^{\mathrm{H}}\omega_{x_{\mathrm{f}}}^{\mathrm{C}} \\ {}^{\mathrm{H}}\omega_{y_{\mathrm{f}}}^{\mathrm{C}} \\ {}^{\mathrm{H}}\omega_{z_{\mathrm{f}}}^{\mathrm{C}} \end{bmatrix} + \boldsymbol{R}_{\mathrm{H}}^{\mathrm{CD}} \boldsymbol{R}_{\mathrm{CD}}^{\mathrm{C}} \begin{bmatrix} 0 \\ 0 \\ \Omega \end{bmatrix}, \tag{5.32}$$

where $\boldsymbol{R}_{\mathrm{H}}^{\mathrm{CD}} = \boldsymbol{R}_{\mathrm{H}}^{\mathrm{TD}}(t^*)$ due to Eq. (3.10) and ${}^{\mathrm{H}}\boldsymbol{\omega}_{\mathrm{f}}{}^{\mathrm{C}}$ is computed from ${}^{\mathrm{H}}\boldsymbol{\omega}_{\mathrm{f}}{}^{\mathrm{TD}}(t^*)$ via Eq. (3.11). Compute the expected chaser angular acceleration as

$$\begin{bmatrix} \dot{\omega}_{x_{\mathrm{f}}}^{\mathrm{C}} \\ \dot{\omega}_{y_{\mathrm{f}}}^{\mathrm{C}} \\ \dot{\omega}_{z_{\mathrm{f}}}^{\mathrm{C}} \end{bmatrix} = \boldsymbol{R}_{\mathrm{T}}^{\mathrm{H}}(t^*) \boldsymbol{R}_{\mathrm{H}}^{\mathrm{CD}} \boldsymbol{R}_{\mathrm{CD}}^{\mathrm{C}} \begin{bmatrix} \dot{\omega}_{x_{\mathrm{f}}}^{\mathrm{T}} \\ \dot{\omega}_{y_{\mathrm{f}}}^{\mathrm{T}} \\ \dot{\omega}_{z_{\mathrm{f}}}^{\mathrm{T}} \end{bmatrix}. \tag{5.33}$$

Step 8: Assuming an initial rest configuration, impose the following linear angular acceleration profile along each axis of the chaser principal coordinate system C:

$$\begin{aligned}\dot{\omega}_j^{\mathrm{C}}(t) &= \frac{\dot{\omega}_{j_\mathrm{f}}^{\mathrm{C}} - \dot{\omega}_{j_0}^{\mathrm{C}}}{t^*} t + \dot{\omega}_{j_0}^{\mathrm{C}} \\ \omega_j^{\mathrm{C}}(t) &= \frac{\dot{\omega}_{j_\mathrm{f}}^{\mathrm{C}} - \dot{\omega}_{j_0}^{\mathrm{C}}}{2t^*} t^2 + \dot{\omega}_{j_0}^{\mathrm{C}} t\end{aligned} \quad , \quad j = x,y,z \tag{5.34}$$

The maneuver duration is t^*. Impose $\omega_j^{\mathrm{C}}(t^*) = \omega_{j\mathrm{f}}^{\mathrm{C}}$ in Eq. (5.34) and calculate the initial angular acceleration as

$$\dot{\omega}_{j_0}^{\mathrm{C}} = \frac{2\omega_{j_\mathrm{f}}^{\mathrm{C}}}{t^*} - \dot{\omega}_{j_\mathrm{f}}^{\mathrm{C}} \qquad j = x,y,z\,. \tag{5.35}$$

Step 9: Evaluate the torque components T_x^{C}, T_y^{C} and T_z^{C} at a given set of instants of time $t_i \in [0, t^*]$ using Eq. (5.22) and the imposed angular velocity and acceleration profiles as Eq. (5.34). At each instant of time, verify the conditions

$$\left\|T_x^{\mathrm{C}}\right\| \le T_{x,\max}^{\mathrm{C}}\,, \qquad \left\|T_y^{\mathrm{C}}\right\| \le T_{y,\max}^{\mathrm{C}}\,, \qquad \left\|T_z^{\mathrm{C}}\right\| \le T_{z,\max}^{\mathrm{C}}\,. \tag{5.36}$$

If the conditions are verified, then set $t_\mathrm{f} = t^*$. Otherwise go back to *Step* 5 and increase Δt_f.

5.4 Closed Loop Trajectory Optimization

The proposed guidance algorithm is based on the closed-loop implementation of the trajectory optimization algorithm described in the previous section. During the maneuver, a new docking trajectory is recomputed at a certain frequency (depending on the computational time required to solve the optimization problem) using the latest update of the system state vector. In this way, the docking trajectory is repeatedly adjusted based on the current state of both chaser and target. In addition, a control actuation system commands the controls generated by the guidance into the dynamics system at a sampling rate of $t_{\min}$ for both position and attitude.

Figure 5.1 represents the block diagram of such guidance algorithm. The current state vector $\boldsymbol{x}_t$, (see Eq. (5.5)) corresponding to the particular instant of time t of the maneuver and assumed to be available from the navigation system, is updated at a certain frequency and fed to the guidance block. In it, a new near-optimal trajectory is recomputed with the inverse dynamics approach. The output of the guidance block consists of the refreshed values of near-optimal controls $\boldsymbol{F}^{\mathrm{C}}$ and $\boldsymbol{T}^{\mathrm{C}}$, which have smooth histories owing to the nature of the inverse dynamics method [36]. In order to be actuated, these controls are fed to the actuation block as shown in Figure 5.1. The

low-level attitude controller samples the torque history $\boldsymbol{T}^{\mathrm{C}}$ at a rate of $t_{\min}$ and commands it using the reaction wheels. On the contrary, the continuous force history $\boldsymbol{F}^{\mathrm{C}}$ is first converted into equivalent sequences of on-off pulses and then sampled by the low-level position controller at a rate of $t_{\min}$. Control forces are finally actuated by cold gas thrusters. This entire process is repeated until maneuver completion. Notably, in the first execution of the loop, the maneuver duration t_{f} is computed in the guidance block before the trajectory optimization using the algorithm described in the Section 5.3.

The main constraint of this closed-loop implementation is that the execution frequency must be chosen such to ensure SNOPT to converge to an optimal solution, for every loop execution. In fact, the period of the loop execution must be higher than the maximum expected computational time required by SNOPT, which can be estimated by analyzing the performance of the trajectory optimization algorithm (in open-loop).

Due to the fact that SNOPT is an iterative solver, there is no guarantee that an optimal solution will be always found within a certain computational time. Therefore, it is necessary to ensure that the guidance block always provides the output controls for every loop execution. This is done by considering the following precautions:

Limiting the number of iterations: The number of maximum iterations is limited to ensure that SNOPT provides a sub-optimal solution within the available time t_{GNC} (note that a safety margin should be considered). The computational time required by SNOPT to execute a single iteration can be estimated by analyzing the performance of the trajectory optimization algorithm in open-loop. Then the maximum number of iterations within t_{GNC} is calculated.

Using previous solution of the nonlinear programming problem: One of the advantages of the inverse dynamics approach is that the controls are determined by the polynomial coefficients along with the chaser endpoint conditions. Therefore, if unexpected problems are encountered by SNOPT, the controls are determined using the polynomial coefficients from the previous trajectory optimization.

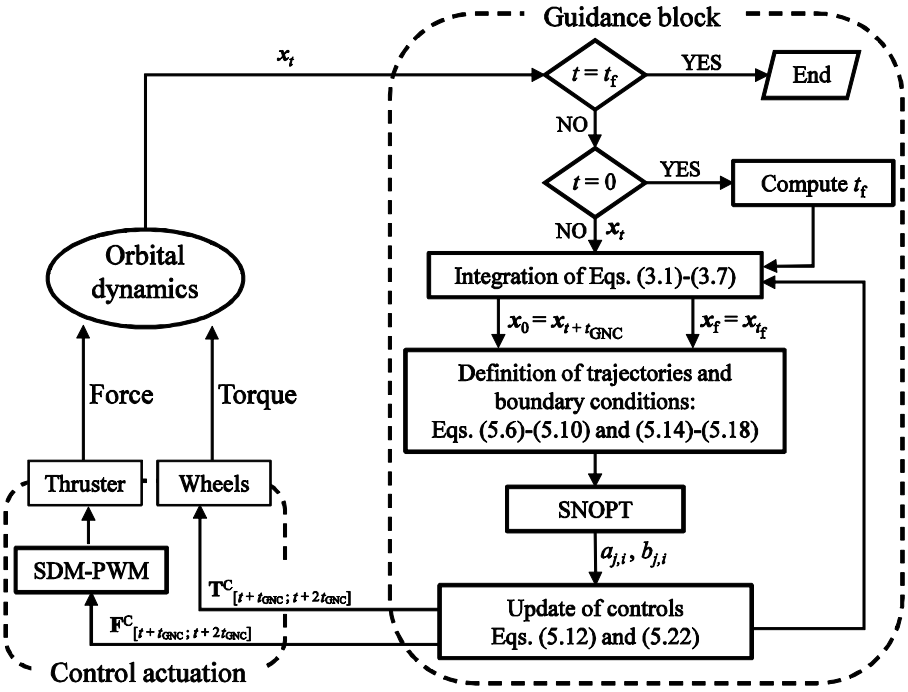

Figure 5.1: Block diagram of the guidance system.

As shown in Figure 5.1, the guidance block reformulates the optimal control problem at each loop execution. Starting from the most current update of the state vector $\boldsymbol{x}_t$ the rendezvous dynamics described by Eqs. (3.1–3.7) is propagated in time for the remaining maneuver duration t_f -(t + t_{GNC}) in order to determine the new initial state vector $\boldsymbol{x}_{t+tGNC}$ and the new final state vector $\boldsymbol{x}_{tf\,-\,(t\,+\,tGNC)}$ for the new trajectory generation. In particular, the initial state vector for the new trajectory optimization corresponds to the time t+t_{GNC} since the updated controls will be commanded at the end of the current loop iteration. Furthermore, although the maneuver duration t_f is a fixed parameter, the final state vector is updated in every loop execution based on the latest current system state. This procedure increase flexibility of the guidance algorithm and robustness to noise in the initial state vector.

Once the endpoint conditions are estimated, the trajectory is parameterized using Eqs. (5.6, 5.9, 5.10, 5.14, 5.17, 5.18). Finally, optimal polynomial coefficients computed by SNOPT are used to update the time histories of the controls, which will be commanded with the control actuation block and actuated through the chaser thrusters and reaction wheels.

In general, standard navigation systems provide the chaser attitude in terms of quaternion [56]. Therefore, before the definition of the trajectory's boundary conditions, the quaternion $\boldsymbol{q}$ must be converted into the MRP vector $\boldsymbol{\sigma}$. To avoid the singularity at $\boldsymbol{q}$ = [0 0 0 -1], the following definition is applied

$$\begin{cases} \sigma_i = \dfrac{q_i}{1+q_4} & q_4 \neq -1 \\ \sigma_i = \dfrac{-q_i}{1-q_4} & q_4 = -1 \end{cases} \qquad i = 1,2,3. \tag{5.37}$$

Notably this procedure exploits the property of quaternions that $\boldsymbol{q}$ and - $\boldsymbol{q}$ defines the same rotation. Clearly, Eq. (5.37) may cause jumps in the time history of MRP and, therefore, a change in the polynomial coefficients. This might affect the computational performance of SNOPT.

As mentioned before, the chaser attitude is controlled by a set of three reaction wheels, therefore the low-level attitude controller samples the time history of the torque $\boldsymbol{T}^{\mathrm{C}}{}_{[t+t_{GNC};t+2t_{GNC}]}$ and commands it at a rate of $t_{\min}$. Conversely, the chaser translation capability is assumed to be featured by six cold gas thrusters, three of them aligned along the positive semi-axis of the principal coordinate system C and the other three directed in the opposite directions. Each thruster is assumed to produce only on-off thrust states, i.e. either zero or maximum thrust (bang-bang control). In this configuration, the three components of the thrust control history $F^{\mathrm{C}}{}_{[t+t_{GNC};t+2t_{GNC}]}$ must be converted into six sequences of on-off pulses, components of the actuated thrust vector $\tilde{\boldsymbol{F}}^{\mathrm{C}}(lt_{\min}) = \left[F_1^C(lt_{\min}), F_2^C(lt_{\min}), \ldots, F_6^C(lt_{\min})\right]^{\mathrm{T}} \quad l \in [0,1,2,\ldots,L]$, with L = integer($t_{\mathrm{GNC}} / t_{\min}$) the number of intervals in the guidance period t_{GNC}.

This analog-to-digital conversion process is performed by the Sigma-Delta modulation combined with the Pulse-Width modulation (SDM-PWM technique) [81, 82] as shown in Figure 5.2. In particular, this technique is executed after the guidance block to convert the history $F^{\mathrm{C}}{}_{[t+t_{GNC};t+2t_{GNC}]}$ into equivalent sequences of max-thrust pulses, which is sampled by the low-level position controller and actuated by the thrusters.

The basic principle of the SDM-PWM consists in measuring the integral $\hat{\boldsymbol{F}}^{\mathrm{C}}(t) \in \boldsymbol{R}^{3x1}$ of the actuation error $\boldsymbol{e}_{\mathrm{F}}(t) \in \boldsymbol{R}^{3x1}$; namely, the difference between the input analog thrust $\boldsymbol{F}^{\mathrm{C}}$ and the actuated thrust $\boldsymbol{F}^{\mathrm{C}}_{\mathrm{act}}$. Such integral is then mapped into an equivalent vector $\overline{\boldsymbol{F}}^{\mathrm{C}}(t) \in \boldsymbol{R}^{6x1}$ characterized by six nonnegative continuous components. Each of these is then converted into sequences of on-off pulses using a PWM [83]. It is important to note that the components of $\hat{\boldsymbol{F}}^{\mathrm{C}}$ are three real numbers (which can attain negative values), whereas the six thrusters of the chaser spacecraft can actuate nonnegative values of thrust. This requires the following mapping process to convert these three components into six equivalent nonnegative components of the vector $\overline{\boldsymbol{F}}^{\mathrm{C}}$:

$$\overline{\boldsymbol{F}}^{\mathrm{C}} = \begin{bmatrix} \boldsymbol{\rho}\left(\hat{\boldsymbol{F}}^{\mathrm{C}}\right) \\ \boldsymbol{\rho}\left(-\hat{\boldsymbol{F}}^{\mathrm{C}}\right) \end{bmatrix} \tag{5.38}$$

with $\boldsymbol{\rho} : \boldsymbol{R}^3 \to \boldsymbol{R}^3$ the ramp function, defined as

$$\rho_i\left(\hat{F}_i^C\right)=\begin{cases}\hat{F}_i^C & \text{if } \hat{F}_i^C \geq 0\\ 0 & \text{if } \hat{F}_i^C < 0\end{cases} \qquad i=1,2,3 \tag{5.39}$$

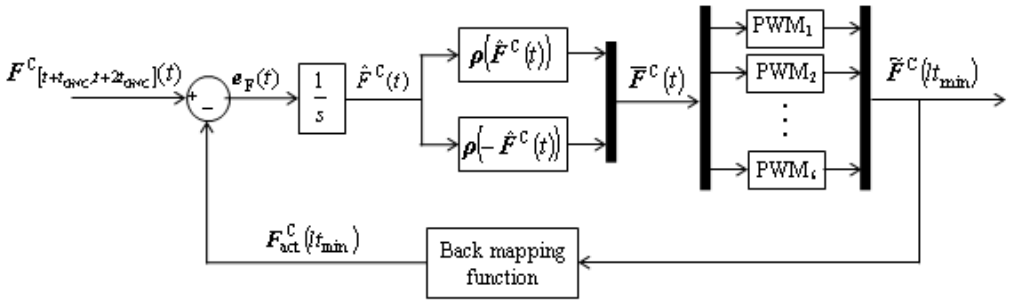

Figure 5.2: SDM-PWM actuation schematic.

Because of this analog to digital conversion process, the actuated thrust slightly differs from the requested thrust. In particular, from Eqs. (5.38, 5.39), we have the following back mapping function:

$$\boldsymbol{F}_{\text{act}}^{\text{C}}\left(lt_{\min}\right)=\begin{bmatrix}\tilde{F}_1\left(lt_{\min}\right)-\tilde{F}_4\left(lt_{\min}\right)\\ \tilde{F}_2\left(lt_{\min}\right)-\tilde{F}_5\left(lt_{\min}\right)\\ \tilde{F}_3\left(lt_{\min}\right)-\tilde{F}_6\left(lt_{\min}\right)\end{bmatrix}. \tag{5.40}$$

5.5 Performance of the Guidance Algorithm

This section investigates the performance of the guidance algorithm in terms of docking precision, optimality of the trajectories, enforcement of the path constraints and computational time required by SNOPT to solve the nonlinear programming problem of Section 5.2.3. In particular, this part focuses on evaluating the performance of the trajectory generation algorithm in the ideal case of perfect knowledge of current state vector and target inertia. The docking maneuvers between an actuated chaser spacecraft and a noncooperative tumbling target spacecraft are considered for the numerical experiments. Table 5.1 and Table 5.2 list the characteristics of the two spacecraft. Data of the chaser are taken from the servicer of the DEOS mission [84], whereas data of the target are taken from Envisat [85, 86]. The safety radius r_{safe} that defines the keep-out zone in Eq. (3.13) is set to 4.60 m according to the position of the target docking port (Table 5.2). The chaser docking point must enter into the target docking interface with a relative translational speed $\|\boldsymbol{v}_{\text{con}}\|$ of 0.01 m/s along the docking axis [58]. The shortest on-time of the chaser's cold gas thrusters is 0.01 s.

Table 5.1: Chaser spacecraft characteristics.

Parameter	Value	Unit of measure
I_{xx}^{C}, I_{yy}^{C}, I_{zz}^{C}	2 014, 1 897, 1 357	kg m^2
m_C	961	kg
Dimensions (body)	3.10 x 2.0 x 1.80	m
$\boldsymbol{l}^C$	[0 0 2.0]	m
$\boldsymbol{s}^C$	[0 0 2.0]	m
$\boldsymbol{R}^C_{CD}$	[0 -1 0; 1 0 0; 0 0 1]	-
$\boldsymbol{b}^C$	[0 0 1]	-
$F^C_{j,\max}$, $F^C_{j,\min}$	8, -8	N
$T^C_{j,\max}$, $T^C_{j,\min}$	10, -10	N m
$t_{\min}$	0.01	s
α_{FOV}	25	deg

Table 5.2: Target spacecraft characteristics.

Parameter	Value	Unit of measure
I_{xx}^{T}, I_{yy}^{T}, I_{zz}^{T}	17 023, 124 825, 129 112	kg m^2
mass	7828	kg
Dimensions (body)	9.20 x 2.6 x 2.6	m
Dimensions (solar array)	15.00 x 2.00 x 0.03	m
$\boldsymbol{l}^T$	[0 -4.60 0]	m
$\boldsymbol{R}^T_{TD}$	[1 0 0; 0 0 1; 0 -1 0]	-
Orbit altitude (eccentricity)	773 (0)	km
Orbit inclination	98.4	deg

The algorithm described in this paper has been implemented in Matlab Simulink environment [87] on a laptop with an Intel Core i7 U 2.8 GHz and 8.0 GB of RAM. The Simulink model for the orbital dynamics consists of the translational and rotational motions of each spacecraft with respect to the Earth-Centered Inertia (ECI) coordinate system. The two spacecraft are assumed rigid bodies. The translational motion of the center of mass is described by the two-body differential equation in Cowell's formulation [88]

$$\ddot{\boldsymbol{r}} = -\frac{\mu}{\|\boldsymbol{r}\|^3}\boldsymbol{r} + \boldsymbol{a}_{\text{ext}}, \tag{5.41}$$

where $\boldsymbol{a}_{\text{ext}}$ the total external force vector acting on the spacecraft and μ the standard gravitational parameter of the Earth. The dynamics of the rotational motion of each spacecraft is represented by the Euler's equation

$$\boldsymbol{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \boldsymbol{I}\boldsymbol{\omega} = \boldsymbol{T}_{\text{ext}}, \tag{5.42}$$

where $\boldsymbol{I}$ denotes the principal inertial tensor and $\boldsymbol{T}_{\text{ext}}$ the external torque vector. The attitude of the spacecraft principal axes with respect to the ECI system is provided by Eq. (3.4) applied to the inertial angular velocity vector

$\boldsymbol{\omega}$ obtained from Eq. (5.42). The orbital dynamics model is solved in Simulink using a fourth-order Runge-Kutta method applied with a fixed time step of 0.01 s [88]. The current state vector $\boldsymbol{x}$ of the two-spacecraft rendezvous model is computed from the solutions of Eqs. (5.41, 5.42).

The trajectory optimization algorithm of the control system employs fifth order polynomials to represent the MRP of the chaser spacecraft attitude trajectory. The same representation is used for each component of the translational trajectory. Since the boundary conditions on position and velocity (attitude and angular velocity for the rotation) are enforced through Eqs. (5.9, 5.10, 5.17, 5.18), two varied parameters are optimized for each trajectory component. The reference length L_{eq} in Eq. (3.15) is 1.5 m, which corresponds to the average distance from the chaser center of mass and the reaction wheels position. The number of intervals that discretize the remaining maneuver time is $N = 24$ in every guidance cycle (see Section 5.4). The constrained optimization problem generated with the inverse dynamics method is solved using SNOPT 7.2 [79] with an accuracy of 1e-7. Eqs. (3.1–3.7) are integrated in the guidance block (see Figure 5.1) with a fourth-order Runge-Kutta method and a fixed time step of 0.2 s.

For the first trajectory optimization (first execution of the guidance block), the initial guess on the polynomial coefficients to be optimized is calculated using Eqs. (5.8, 5.16) after imposing zero controls at the endpoints. Since two varied parameters are defined for each trajectory component, initial and final values of controls (thrust and torque) are enough to specify the initial guess. Successive optimizations use the previous optimal polynomial coefficients as new initial guess. In our experimental campaign, we have tested the trajectory optimization performed by SNOPT in open-loop, recording peaks of computational time of about 8.0 s. Consequently, in the numerical experiments, the guidance algorithm has been executed at a rate of 0.1 Hz ($t_{\text{GNC}} = 10$ s) in order to ensure a safety margin. Since SNOPT requires about 5ms to execute a single internal iteration, the maximum number of iterations is set to 1800 in order to ensure a result within 10 s.

Three reference scenarios summarized in Table 5.3 are considered. The maneuvers are R-bar approaches from identical initial conditions for both chaser and target, but differ in the rotational motion of the target spacecraft. The angular rates of the target are taken from the estimated measurements of Envisat [85]. For the numerical experiments presented in this section, the analog-to-digital conversion process of the thrust components described in Section 5.4 is bypassed and the continuous controls generated by the inverse dynamics are directly injected into the dynamics model in order to analyze the performance of the guidance algorithm alone. We also assume perfect knowledge of the target state and inertia. The optimality of the resulting maneuvers is evaluated by comparison with reference optimal trajectories obtained by solving the optimal control problem posted in Section 3.3 with GPOPS II v1.0 [89]. For each scenario, GPOPS is executed ten times using

different initial guesses, generated by randomly perturbing the initial guess on the controls used for the cold start of the guidance system and the final state achieved with the maneuver (first guess on the final condition). The chosen reference optimal trajectory corresponds to the one having the minimum performance index. In this way, we avoid possible local minima of the reference solution. The accuracy of GPOPS has been set to 1e-7 for this analysis. Each optimization with GPOPS required between 20 min and 60 min according to the first guess.

Table 5.3: Initial states of the target and chaser spacecraft for Scenarios 1-3.

Parameter	Value	Unit of measure
x_0, y_0, z_0	-50, -11, 7	m
$\dot{x}_0,\ \dot{y}_0,\ \dot{z}_0$	0, 0, 0	m/s
$\boldsymbol{\omega}_0^{\mathrm{C}}$	[0 0 0]	deg/s
$\boldsymbol{\sigma}_0^{\mathrm{C}}$	[0.34 0.41 0.37]	-
$\boldsymbol{\omega}_0^{\mathrm{T}}$ (Scenario 1)	[3.5 0.5 0.5]	deg/s
$\boldsymbol{\omega}_0^{\mathrm{T}}$ (Scenario 2)	[0.5 3.5 0.5]	deg/s
$\boldsymbol{\omega}_0^{\mathrm{T}}$ (Scenario 3)	[0.5 0.5 3.5]	deg/s
$\boldsymbol{q}_0^{\mathrm{T}}$	[-0.5 -0.5 -0.5 0.5]	-

Table 5.4 summarizes the maneuver durations calculated with the algorithm described in Section 5.3 and the relative states between the target spacecraft docking point and the chaser spacecraft docking point achieved at the end of the maneuvers. Notably the guidance algorithm allows reaching all the docking conditions in Eqs. (3.8–3.11) with high precision at the end of each maneuver. Furthermore, the algorithm for estimating the maneuver duration is capable to calculate feasible maneuver time durations.

Figure 5.3 illustrates the enforcement of the path constraints by the guidance during the maneuvers execution. The time history of the distance between the chaser spacecraft docking point and the keep-out zone is evaluated with Eq. (3.13) and it is represented in Figure 5.3a. The minimum distance is 0 m in all the maneuvers and occurs at the mechanical contact between docking points. The time history of the angular distance from the chaser spacecraft rendezvous sensor boresight to the target spacecraft center of mass is calculated with Eq. (3.12) and it is shown in Figure 5.3b. Also in this case, the guidance imposes accurately the constraint. The achieved maximum angular distance is 25.02 deg during Scenario 3.

Figure 5.4 represents the evolution of the computational time required by SNOPT to converge to a solution for each cycle (cycle duration t_{GNC} = 10 s). In all the considered scenarios, higher computational times arise during the initial trajectory optimizations (within 15 calls of SNOPT), whereas lower computational times, with an average of 0.4 s, are observed in the second part of the maneuvers. Such behaviour can be explained by considering that SNOPT enforces the path and control constraints in new

temporal nodes at each call and the resulting trajectories progressively converge to the sub-optimal solution of the constrained optimization problem. For this reason, SNOPT requires more iterations to converge in the initial calls, resulting in higher computational time. In the second part of the maneuver SNOPT only updates the computed feasible trajectory since the constraints are imposed accurately enough, resulting in a faster convergence speed.

Figure 5.5 shows the evolution of the polynomial coefficients computed by SNOPT for Scenario 1. As previously discussed, after the first dozen calls of SNOPT the coefficients $a_{j,3}$ and $b_{j,3}$ stabilize to quasi-steady values. In the second part of the maneuver, only the coefficients $a_{j,2}$ and $b_{j,2}$ keep changing since these parameters are related to the new initial acceleration components according the following

$$b_{1,2} = \frac{\ddot{x}_0}{2}, \quad b_{2,2} = \frac{\ddot{y}_0}{2}, \quad b_{3,2} = \frac{\ddot{z}_0}{2}, \quad a_{j,2} = \frac{\ddot{\sigma}^{\mathrm{C}}_{j0}}{2} \quad j = 1,2,3 \ . \tag{5.43}$$

Table 5.4: Achieved docking conditions for Scenarios 1-3. Relative states are calculated using Eqs. (3.8–3.11). Axial and radial directions are intended with respect to the target spacecraft docking axis.

	t_f, s	Δr , m		Δv , m/s		$\Delta\theta_{max}$, deg	$\Delta\omega_{max}$, deg/s
		axial	radial	axial	radial		
Scenario 1	410	2.4e-4	1.2e-4	1.0e-2	2.0e-5	5.7e-3	2.9e-4
Scenario 2	310	-5.5e-4	1.8e-4	1.0e-2	2.4e-5	6.9e-3	1.4e-3
Scenario 3	310	-2.9e-4	1.6e-4	1.0e-2	1.6e-5	8.0e-3	1.1e-3

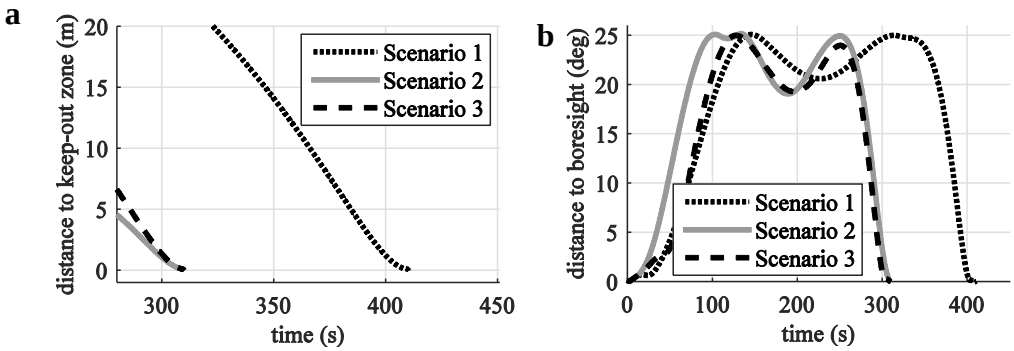

Figure 5.3: Path constraints during the maneuvers. Distance from the keep-out safety zone (a), and angular distance of the target spacecraft center of mass from the chaser sensor boresight (b).

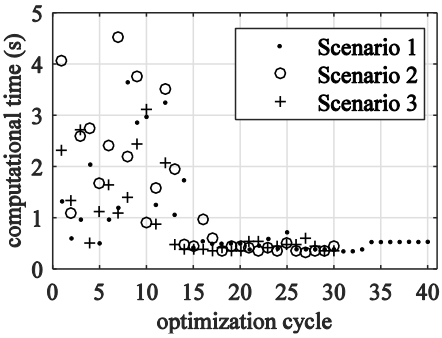

Figure 5.4: Computational time required by SNOPT to converge to a solution at each cycle.

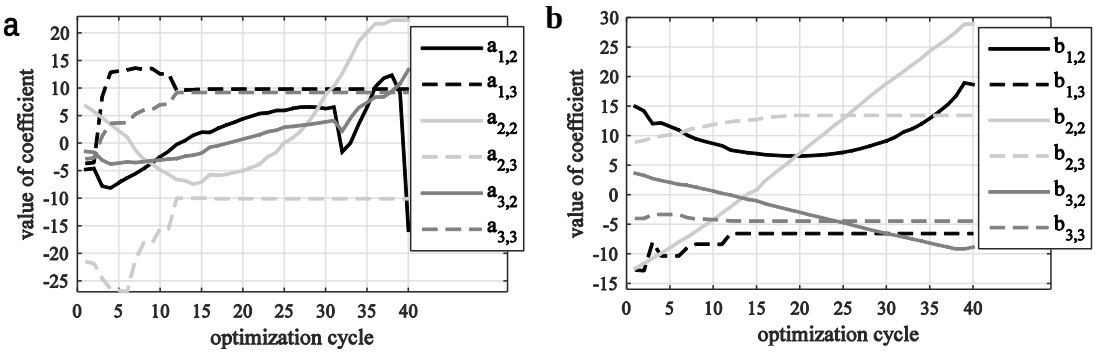

Figure 5.5: Evolution of the variable polynomial coefficients optimized by SNOPT during Scenario 1: attitude (a) and relative distance (b).

The performance index J related to the three test maneuvers are summarized in Table 5.5. The index J_{GNC} for the controlled trajectory is calculated with Eq. (3.15) using the controls generated by the guidance system. Similarly, J_{GPOPS} is calculated with Eq. (3.15) using the optimal controls. Comparison among optimal indexes shows that the trajectories generated by the guidance are near-optimal, with a maximum difference in percentage of 1.1 % in terms of optimal performance index occurring in Scenario 1. The reason of these discrepancies in optimality can be understood by comparing the trajectories and controls generated by the guidance system with the solutions of the correspondent optimal control problems. Let us now consider Scenario 1, which is subjected to the major difference in optimality. Figure 5.6 shows the translational and rotational trajectories performed by the chaser spacecraft. At any instant of time, the components of the translational trajectory generated by the guidance closely match the optimal ones (Figure 5.6a). In fact, the time histories of the optimal trajectory components present smooth shapes and can be represented precisely by high order

polynomials. On the contrary, as can be observed in Figure 5.6b, the rotational trajectories differ remarkably from the optimal ones. In this case, the time histories of the optimal MRP feature several inflection points, which are harder to track with the chosen polynomials.

Figure 5.7 graphically compares the controls generated by the guidance and the optimal controls obtained with GPOPS. The thrust components closely match the optimal ones (Figure 5.7a). Owing to the nature of the inverse dynamics, this is the consequence of the precise representation of the translational trajectory with the chosen function. However, significant differences are observed between the control torque curves due to the presence of first order discontinuities in the time histories of the torques generated by the inverse dynamics (Figure 5.7b). As previously discussed, fifth-order polynomials are unable to represent the rotational trajectory of the chaser spacecraft with high precision and, consequently, the controls generated by the inverse dynamics method differ from the optimal controls. At each guidance cycle, the polynomial coefficients are recomputed in order to minimize the difference between the controls generated by the guidance and the optimal ones, i.e. the performance index of the maneuver is minimum. As a result, first order discontinuities arise as the controls are updated. Analogous observations for Scenarios 2 and 3 hold.

Figure 5.7b also provides interesting information about the motion of the chaser spacecraft. In the first part of the maneuver, the torque action is moderate, meaning that the chaser approaches the target without large changes in attitude and angular velocity. Only in the last part of the maneuver (about 60s before the docking), the chaser performs intense angular accelerations in order to synchronize with the rotational motion of the target.

Table 5.5: Optimal performance index J achieved with GPOPS and with the controls generated by the guidance system for the selected reference scenarios.

	J_{GPOPS} , N^2 s	J_{GNC} , N^2 s	ΔJ , %
Scenario 1	617.1	624.0	1.1
Scenario 2	2490.6	2508.6	0.7
Scenario 3	1331.5	1337.5	0.4

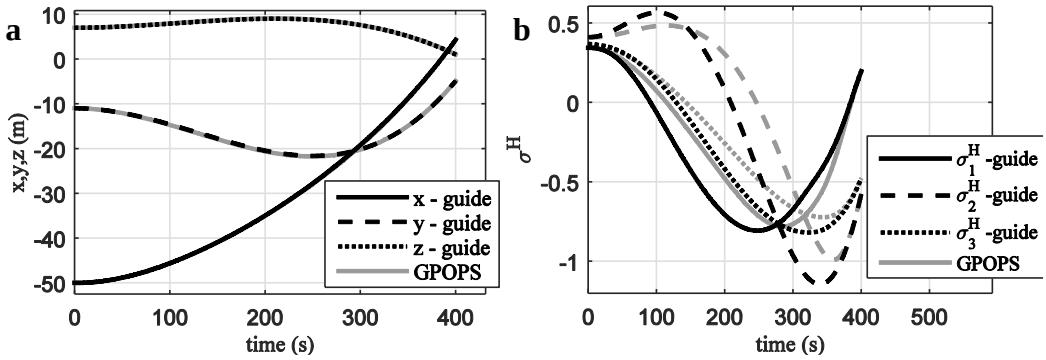

Figure 5.6: Scenario 1: time history of the translational and rotational components of the chaser trajectory obtained with the guidance system and with GPOPS.

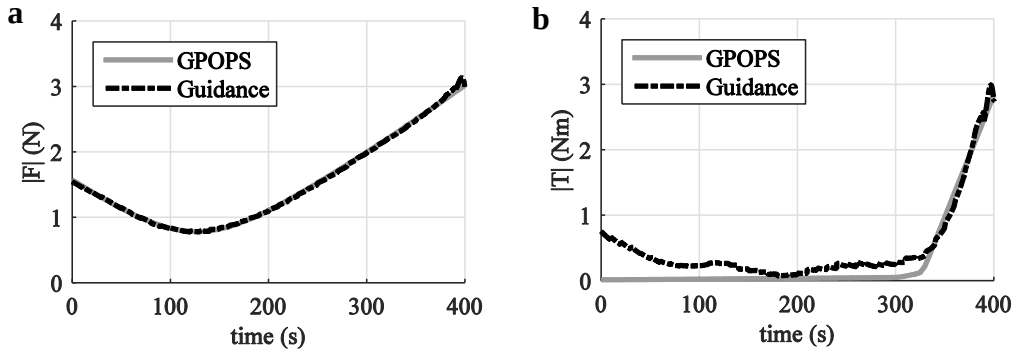

Figure 5.7: Scenario 1: time histories of the thrust and torque norms generated with the guidance algorithm and GPOPS.

5.6 Monte Carlo Simulations

The robustness of the guidance algorithm in Figure 5.1 to various maneuvering situations under realistic orbital perturbations is discussed in this section. The guidance system has been tested on 500 docking scenarios obtained by varying randomly both the initial relative distance between the spacecraft and the initial attitude and rotation rates of the target spacecraft. The orbit of the target spacecraft is circular with an altitude of 773 km as in Section 5.5. All the maneuvers start with zero relative translational speed with the chaser at rest (station keeping configuration) and its sensor boresight pointing the target spacecraft center of mass. The initial conditions are randomly generated within the intervals defined in Table 5.6, whereas the current state vector $\boldsymbol{x}_t$ and target inertia are the same as in the previous analysis. With the goal to test the algorithm under realistic control applications, the SDM-PWM technique described in Section 5.4 is used to convert

the continuous force components into equivalent sequences of thrust pulses. We also assume perfect knowledge of the target inertia and current state vector as in Section 5.5.

Table 5.6: Ranges of value for the random initial conditions. The initial attitude of the target principal system T is given in terms of Euler angles $\left[\theta_{1_0}^{\mathrm{T}} \quad \theta_{2_0}^{\mathrm{T}} \quad \theta_{3_0}^{\mathrm{T}}\right]$.

Parameter	Range of values	Unit of measure
x_0	[-100; -20]	m
y_0	[-100; 100]	m
z_0	[-20; 20]	m
ω_{x0}^{T}, ω_{y0}^{T}, ω_{z0}^{T}	[-4; 4]	deg/s
θ_{10}^{T}, θ_{20}^{T}, θ_{30}^{T}	[-180; 180]	deg

Table 5.7 summarizes the final docking conditions and the performance of the control system after numerical tests on the 500 random scenarios. The final conditions are calculated using Eqs. (3.8–3.11) and expressed in the CD coordinate system. Since the target's center of mass may not always reach the boundary of the chaser sensor FOV, the associated path constraint is considered active when α_{T} (peak) $\geq$ 24.5deg. The parameter α_{T} is calculated using Eq. (3.12). Remarkably, all the final docking conditions requirements are achieved with high precision, even using the DSM-PWM conversion for the control force. Furthermore, the guidance algorithm enforces the path constraints accurately. In all cases, the peaks of computational time required by SNOPT to converge are lower than 10 s. This result confirms the appropriateness of the choice to update the trajectory at a rate of 0.1 Hz.

Table 5.7: Docking conditions and performance of the control system obtained with the random scenarios. Axial and radial directions are intended with respect to the target spacecraft docking axis.

Parameter	Mean value [3σ]	Unit of measure
$\Delta r_{\mathrm{axial}}$	2.4e-3 ± 7.0e-3	m
$\Delta r_{\mathrm{radial}}$	3.7e-3 ± 10.1e-3	m
$\Delta v_{\mathrm{axial}}$	1.0e-2 ± 0.1e-2	m/s
$\Delta v_{\mathrm{radial}}$	6.7e-4 ± 9.0e-4	m/s
$\Delta\theta_{\max}$	2.6e-2 ± 0.51	deg
$\Delta\omega_{\max}$	3.4e-2 ± 9.3e-2	deg/s
α_{Tmax}	25.0 ± 0.2	deg
Minimum distance from "keep-out" zone	1.3e-3 ± 2.2e-3	m
CPU time (average)	0.99 ± 1.73	s
CPU time (peak)	5.12 ± 3.76	s

In order to verify the robustness to orbital perturbations, the most significant orbital disturbances in LEO have been introduced into the Simulink dynamics models (Eqs. (5.41, 5.42)) for both target and chaser spacecraft.

In particular, disturbances due to geopotential anomaly, atmospheric drag and gravitational gradient are considered.

The non-homogeneous gravitational potential of the Earth is considered up to the harmonic coefficient J_6. In particular, the translational acceleration due to the term J_2 is calculated with the following [88]

$$\boldsymbol{a}_{J2} = \frac{3J_2\mu\, r_\oplus^2}{2\|\boldsymbol{r}\|^5}\begin{bmatrix}\left(1-\dfrac{5r_z^2}{\|\boldsymbol{r}\|^5}\right)r_x\\ \left(1-\dfrac{5r_z^2}{\|\boldsymbol{r}\|^5}\right)r_y\\ \left(3-\dfrac{5r_z^2}{\|\boldsymbol{r}\|^5}\right)r_z\end{bmatrix}, \tag{5.44}$$

where $r_\oplus$ is the average Earth equatorial radius. The expressions for the other acceleration terms until J_6 are analogous to Eq. (5.44) and can be found in Ref. [88]. The translational acceleration due to the atmospheric drag is described as [88]

$$\boldsymbol{a}_{\text{drag}} = -\sum_i \frac{1}{2}\rho\frac{C_{\text{D}}A_i}{m_{\text{C}}}\|\boldsymbol{v}_{\text{rel}}\|\boldsymbol{v}_{\text{rel}} \tag{5.45}$$

where the density ρ of the atmosphere is estimated with the exponential model described in Ref. [88]. The drag coefficient is C_D = 2.2. The term A_i is the exposed cross-sectional area of the i-th surface of the spacecraft defined as the area that is normal to the vehicle velocity vector. The shape of each spacecraft is assumed rectangular parallelepiped. The velocity $\boldsymbol{v}_{\text{rel}}$ denotes the velocity of spacecraft center of mass relative to the rotating atmosphere and it is defined as

$$\boldsymbol{v}_{\text{rel}} = \dot{\boldsymbol{r}} - \begin{bmatrix}0\\0\\\omega_\oplus\end{bmatrix}\times\boldsymbol{r}\,, \tag{5.46}$$

where $\omega_\oplus$ is the rotation velocity of the Earth.

The external torque acting on each spacecraft due to the atmospheric drag is given by

$$\boldsymbol{T}_{\text{drag}} = -\sum_i \boldsymbol{L}_i \times\left(\frac{1}{2}\rho C_{\text{D}}A_i\|\boldsymbol{v}_{\text{rel}}\|\boldsymbol{v}_{\text{rel}}\right) \tag{5.47}$$

where the displacement vector $\boldsymbol{L}_i$ relative to the surface A_i corresponds to the distance between the spacecraft center of mass and the surface geometric center. The external torque due to the gravitational gradient is [90]

$$\boldsymbol{T}_{GG} = \frac{3\mu}{\|\boldsymbol{r}\|^5}\begin{bmatrix} \left(I_{yy}^{\mathrm{C}} - I_{zz}^{\mathrm{C}}\right) r_z r_y \\ \left(I_{zz}^{\mathrm{C}} - I_{xx}^{\mathrm{C}}\right) r_x r_z \\ \left(I_{xx}^{\mathrm{C}} - I_{yy}^{\mathrm{C}}\right) r_z r_y \end{bmatrix}. \tag{5.48}$$

Analogous expressions are applied for the target spacecraft.

Under these conditions, the guidance system has been tested on the same 500 random scenarios of Table 5.6. As can be observed in Table 5.8, the required final docking conditions are achieved with high precision also in presence of the orbital perturbations. In fact, by updating the trajectory every 10 s, the effects of the orbital perturbations within this time range are mitigated. However, the orbital perturbations cause an increment of the energy expenditure with respect to the ideal case.

Table 5.8: Docking conditions and performance of the control system obtained with the random scenarios under the orbital perturbations. Axial and radial directions are with respect to the target docking axis.

Parameter	Mean value [3σ]	Unit of measure
$\Delta r_{\mathrm{axial}}$	3.0e-3 ± 17.0e-3	m
$\Delta r_{\mathrm{radial}}$	4.2e-3 ± 15.0e-3	m
$\Delta v_{\mathrm{axial}}$	1.0e-2 ± 0.2e-2	m/s
$\Delta v_{\mathrm{radial}}$	8.0e-4 ± 38.0e-4	m/s
$\Delta\theta_{\max}$	3.4e-2 ± 0.79	deg
$\Delta\omega_{\max}$	3.9e-2 ± 12.2e-2	deg/s
α_{Tmax}	25.1 ± 0.3	deg
CPU time (average)	1.04 ± 1.99	s
CPU time (peak)	5.33 ± 4.21	s
$\mathcal{J}_{\mathrm{disturbances}}/\mathcal{J}_{\mathrm{ideal}}$	1.1 ± 0.5	%

5.7 Sensitivity Analysis

The proposed guidance system has been subjected to a sensitivity analysis in order to study its behavior in presence of model errors and noise in the chaser-target system state vector $\boldsymbol{x}_t$ (see Figure 5.1) and therefore its applicability to noncooperative scenarios. Errors in relative distance and translational velocity between spacecraft centers of mass along with errors in target spacecraft principal inertia, angular velocities and attitude are considered in our analysis. This choice is driven by the fact that these parameters are subjected to major uncertainties during the docking maneuver to a noncooperative target [56, 91]. Conversely, angular velocities and attitude

of the chaser spacecraft with respect to the ECI coordinate system are usually provided by the Inertial Measurement Unit with higher accuracy [92].

At this point, it is important to note that the proposed guidance system is a model-based algorithm and requires the knowledge of the target inertia. In the case of noncooperative maneuvers, this parameter can be retrieved from the target spacecraft constructor, as the case of Envisat. When this solution is not available, the target inertia can be estimated onboard the chaser during closed range observation maneuvers before the docking procedure [93, 94]. In both cases, however, the target inertia might be subjected to significant uncertainties.

The sensitivity analysis is performed on reference Scenario 1 (data in Table 5.3) with the following combinations of parameter variation and/or noises:

1. noise on relative distance and relative translational velocity
2. errors in target spacecraft principal inertia and noise in target spacecraft angular velocity
3. errors in target spacecraft principal inertia and noise in target spacecraft attitude
4. noise on target spacecraft angular velocity and attitude.

The choice of these particular combinations is motivated by the synergic correlation between these factors in the rendezvous dynamics model described in Section 3.1. In particular, the components x, y, z of the relative distance and the components $\dot{x}$, $\dot{y}$, $\dot{z}$ of the relative velocity are coupled in the C-W dynamics in Eq. (3.1). Moreover, the target spacecraft inertia, angular velocity, and attitude are coupled in the attitude dynamics Eq. (3.3) and Eq. (3.7). For each category of system disturbances (relative distance, relative velocity, target inertia, target attitude, and target angular velocity), we consider 5 levels of noise applied to every component of the vector (see Table 5.9). In this way, 5 x 5 = 25 different combinations of noise level for each of the four combinations of disturbances are obtained. For each particular noise combo, Scenario 1 is simulated 50 times. Errors in the inertia of the target spacecraft are applied with random signs to I_{xx}^{T}, I_{yy}^{T}, I_{zz}^{T} at the beginning of each simulation. The noise on the selected components of the state vector is modeled as white noise signal. The performance of the guidance system is evaluated in terms of final docking conditions achievement and energy expenditure as in Section V.B. Furthermore, to better evaluate the impact of the disturbances, we acknowled a successful docking when $||\Delta r_{\mathrm{radial}}|| \leq 0.05$ m, $||\Delta v_{\mathrm{radial}}|| \leq 0.01$ m/s, $||\Delta \theta_{\max}|| \leq 5$ deg and $||\Delta \boldsymbol{\omega}_{\max}|| \leq 0.5$ deg/s as in the Orbital Express mission [95].

Table 5.9: Summary of the noise levels (3σ, except for the inertia) for each category of system disturbance. In the case of vectors, the noise is applied to each component.

System disturbance	Noise level	Unit of measure
δx , δy , δz	0, 0.5, 1, 2, 4	cm
$\delta \dot{x}$, $\delta \dot{y}$, $\delta \dot{z}$	0, 0.1, 0.25, 0.5, 1	cm/s
$\delta I^{\mathrm{T}}_{xx}$, $\delta I^{\mathrm{T}}_{yy}$, $\delta I^{\mathrm{T}}_{zz}$	0, 1, 2, 3, 4	%
$\delta \boldsymbol{q}^{\mathrm{T}}(\boldsymbol{\theta}_{123})$	0, 0.25, 0.5, 1, 2	deg
$\delta \boldsymbol{\omega}^{\mathrm{T}}$	0, 0.1, 0.15, 0.2, 0.25	deg/s

The guidance loop is again executed at a rate of 0.1 Hz. Moreover, after the last optimization of the trajectory, i.e. 10 s before the docking, the guidance system in Figure 5.1 is executed at a rate of 2 Hz bypassing the call of SNOPT. Notably this rate still ensures enough time for noncooperative pose estimation [96]. The control time histories are consequently computed using the last optimal varied parameters and the current endpoint conditions evaluated with Eqs. (5.9, 5.10, 5.17, 5.18). This procedure can be implemented only after the last optimization of the trajectory since the computation time is mostly used by SNOPT during the nominal execution of the control loop. In this way, we increase the robustness to disturbances, since the docking conditions depend on the predicted final state of the target, which is influenced by the propagation of the noise.

Figure 5.8 and Figure 5.9 show the effects of the disturbances on final relative attitude and angular velocity between docking points. The bounded zones in the graphs depict where the docking precision requirements are met. As can be observed from the figures, the precision requirements are respected for every combination of disturbance levels. Moreover, relative attitude and angular velocity are not sensitive to errors in relative distance and velocity between spacecraft centers of mass (Figure 5.8a and Figure 5.9a). In fact, the guidance system imposes the synchronization of the rotational motion between docking points through the conditions in Eqs. (3.10, 3.11), which are independent from distance and velocity between spacecraft.

Figure 5.10 and Figure 5.11 illustrate the influence of the disturbances on the final radial distance and velocity between docking points. Once again, the bounded zones in the graphs depict where the docking precision requirements are met. In this case, the final relative velocity results sensitive to the noise in target angular velocity (Figure 5.11c and Figure 5.11d). In fact, as it can be observed in Eq. (3.9), an error $\delta \boldsymbol{\omega}^{\mathrm{T}}$ in the target angular velocity causes an error in translational velocity proportional to $\delta \boldsymbol{\omega}^{\mathrm{T}} \times \boldsymbol{l}^{\mathrm{T}}$. Clearly, large target spacecraft dimensions amplify the errors in the predicted translational velocity of the target docking port due to the noise on $\boldsymbol{\omega}^{\mathrm{T}}$. Furthermore, disturbances in target angular velocity generate errors in the target final attitude $\boldsymbol{R}^{\mathrm{H}}_{\mathrm{T}}$, which causes a further error in the final translational velocity related to the term $\boldsymbol{\omega}^{\mathrm{T}}_{\mathrm{F}} \times \boldsymbol{R}^{\mathrm{H}}_{\mathrm{T}} \boldsymbol{l}^{\mathrm{T}}$. Also in this case, large dimension and high rotation rates of the target spacecraft enhance the effect of $\delta \boldsymbol{\omega}^{\mathrm{T}}$. As can

be observed from Figure 5.11, the precision requirement on radial translational velocity are respected when errors in target angular velocity are limited to 0.1 deg/s 3σ and, at the same time, errors in target inertia and attitude are limited to 2% and 0.5 deg 3σ respectively. Regarding the final relative distance between docking points, the major offsets are caused by disturbances in relative translational velocity and target angular velocity. In fact, the large errors in radial velocity caused by these disturbances also generate large offsets in the radial direction.

Concerning the optimality of the maneuvers, the presence of system disturbances in the current state vector increases the energy expenditure. In fact, disturbances cause errors in the current estimated state and in the expected final state (obtained by propagation from the current state) and, consequently, the resulting trajectories diverge significantly from the optimal one. Noise in the target angular velocity increases the performance index J between 19 % 3σ (lowest noise level) and 64 % 3σ (highest noise level) with respect to the ideal case without noise, whereas the rest of disturbances case an increment between 5 % 3σ and 20 % 3σ.

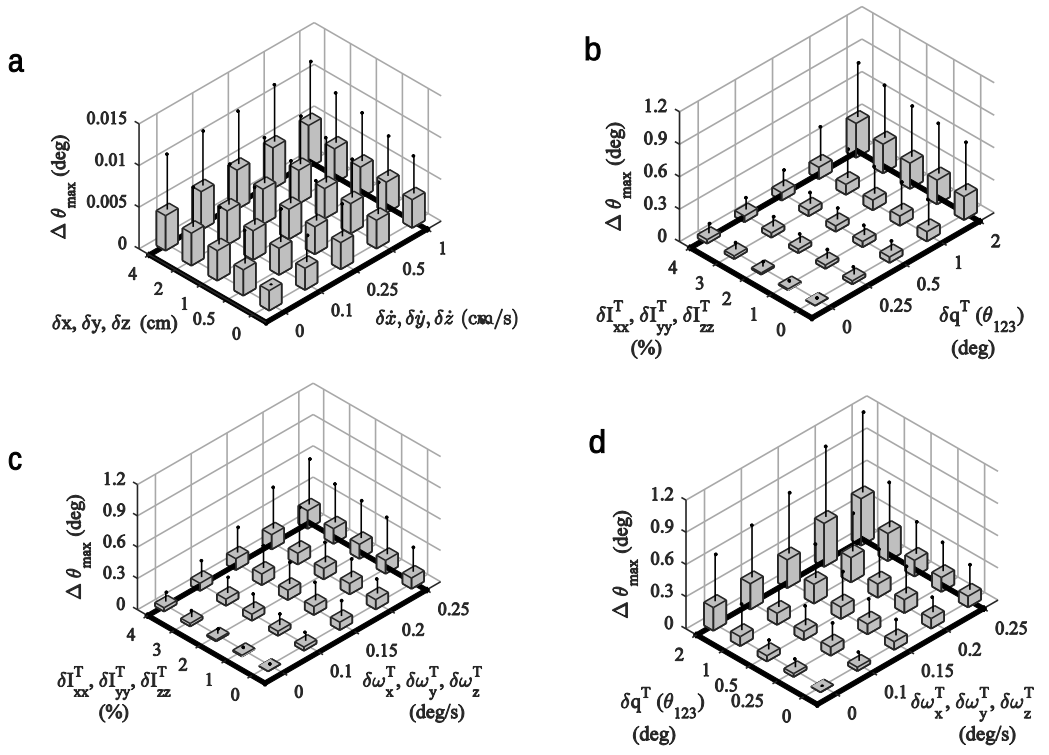

Figure 5.8: Final misalignment between docking points for different combinations of system disturbances. The vertical bars denote the mean value, whereas the vertical lines denote the 3σ value.

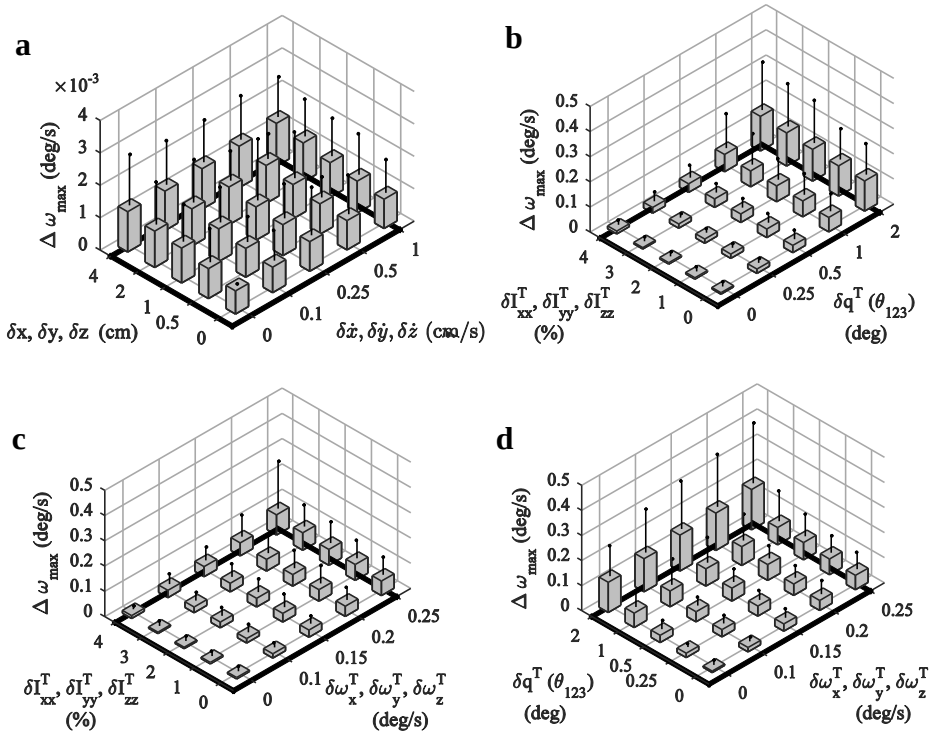

Figure 5.9: Final relative angular velocity between docking points for different combinations of system disturbances. The vertical bars denote the mean value, whereas the vertical lines denote the 3σ value.

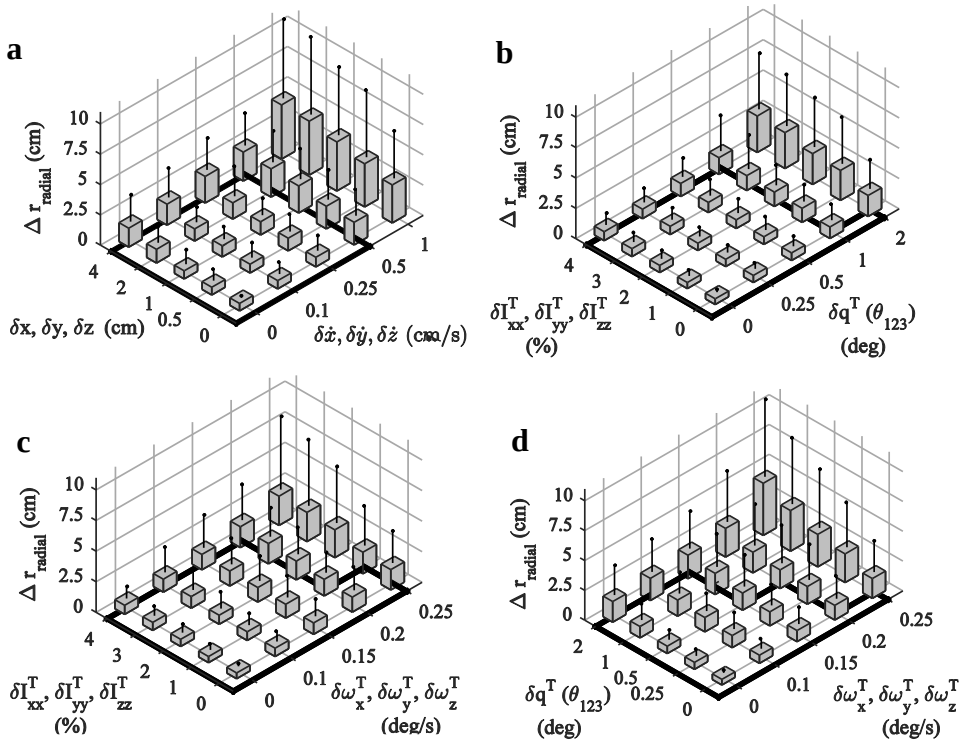

Figure 5.10: Final offset (radial direction from the target docking axis) between docking points for different system disturbances. Vertical bars represent the mean value, whereas vertical lines denote the 3σ value.

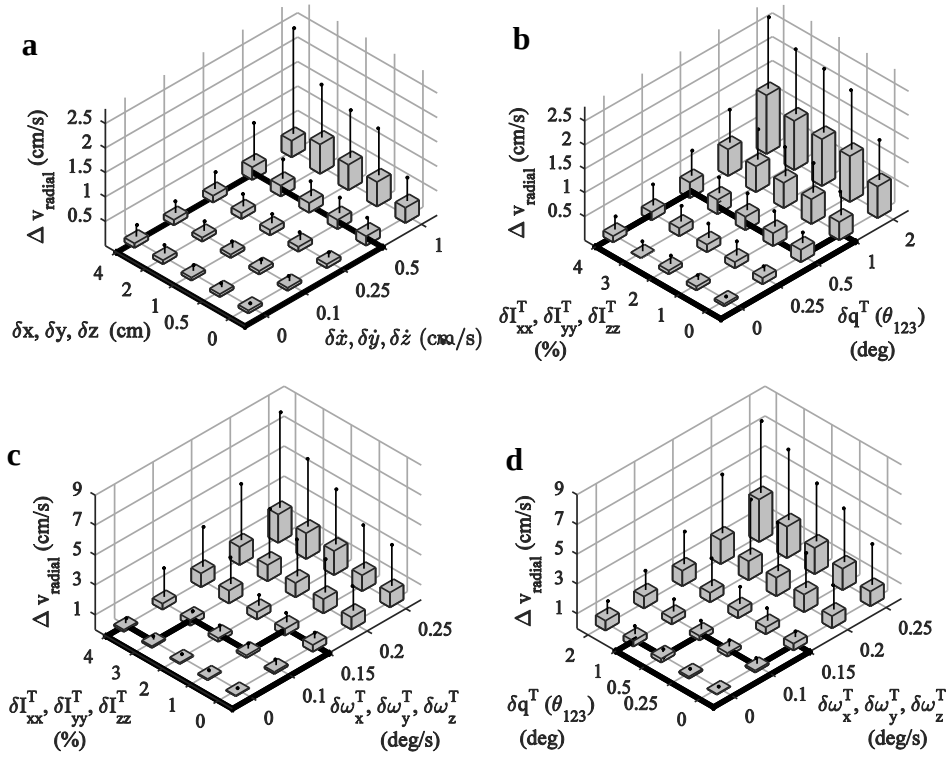

Figure 5.11: Final velocity (radial direction from the target docking axis) between docking points for different system disturbances. Vertical bars represent the mean value, whereas vertical lines denote the 3σ value.

5.8 Summary

This chapter presented a guidance system for minimum-energy 6 DoF docking maneuvers between a controlled chaser spacecraft and an uncontrolled target in circular orbit. A direct optimization method is formulated for rapid trajectory generation: polynomials are employed for trajectory representation, the inversion of the system dynamics allows to converts the optimal control problem into an equivalent nonlinear programming problem, a sequential quadratic programming solver optimizes the unconstrained polynomial coefficients. Such approach has three main advantages. First, the reduced number of variables to be optimized leads to fast computational speed for trajectory optimization. Second, the reference trajectory and controls are determined by the polynomial coefficients only, enabling easy onboard storage of the solution and rapid control update during the maneuver. Third, the docking-enabling conditions are automatically imposed by simple reconstruction of the polynomials. The closed-loop implementation of such technique includes a combined SDM-PWM to convert the continuous control forces generated by the inverse dynamics into sequences of max-

thrust pulses, and a low-level attitude and position controllers to sample and actuate the generated control time histories.

In the ideal case of perfect knowledge of the current state vector and target inertia, the proposed guidance system is able to perform near-optimal maneuvers while enforcing all the path constraints and achieving accurate docking conditions. The small difference in optimality is due to slight discrepancies in the chaser attitude trajectory, since piecewise polynomials can match the optimal trajectory up to a certain level. The Monte Carlo simulation demonstrates that the guidance scheme, in the same ideal case of perfect knowledge of the target conditions, performs accurate maneuvers in most maneuvering scenarios, also in presence of orbital perturbations in low Earth orbit.

The applicability of the guidance system to more realistic noncooperative scenarios is tested with a sensitivity analysis via Monte Carlo simulation. We have considered uncertainties on both target inertia and relative state, driven by the fact that these two parameters are the most critical in noncooperative maneuvers. The sensitivity study shows that the proposed guidance system is particularly sensitive to noise in target angular velocity. In fact, this type of disturbance causes significant errors in the final translational velocity between spacecraft docking points. As regards errors in the target inertia, which might be estimated during the observation phase and therefore subjected to significant uncertainties, the guidance is capable to dock successfully to the target whenever the error is less than 4% of the nominal value.

Even though the proposed trajectory optimization algorithm is a model-based method that requires the knowledge of the target inertia and the current relative state, the proposed closed-loop implementation is able to perform noncooperative maneuvers to uncontrolled tumbling targets under limited errors on target angular velocity and inertia. In fact, the key features that ensure noncooperative applications are: 1) onboard/online generation of the docking trajectory using the current state vector and maneuver duration only, 2) prediction of the docking conditions by onboard numerical propagation of the system state 3) rapid trajectory and control reshaping based on the latest state estimation.

6 Noncooperative Pose Estimation

6.1 Algorithm Architecture

During autonomous close-range proximity operations for OOS and space debris removal missions, the chaser spacecraft executes rendezvous and docking to the target satellite. Before the launch of the mission, the kinematic characteristics of the target can be only estimated by laser measurements from ground due to its noncooperative nature. The acquired data are consequently affected by significant uncertainties. Furthermore, the rotational motion of an uncontrolled space debris may change with the time due to the loss of kinetic energy. In the case of Envisat, the estimated spin rate is 3.2 deg/s with an increment of the spin period of 36.7 ms/day [5]. These values are given with 15% of accuracy. It is therefore difficult to plan the docking maneuver before the launch of the mission since the motion of the target spacecraft cannot be estimated with high precision. Safe docking trajectories that ensure accurate mechanical contact between spacecraft can only be generated onboard the chaser by the guidance using the data from the navigation system, whose goal is to determine current position, attitude and velocities of the target with high precision.

The guidance algorithm proposed in Chapter 5 requires the current state vector $\boldsymbol{x}_t$ of the dynamics model of the docking maneuver, which is defined by position and velocity of the chaser center of mass with respect to the Hill's coordinate system along with attitude and angular velocity of the two spacecraft with respect to the same system (see Eq. (5.5)). To determine the state vector $\boldsymbol{x}_t$, the chaser spacecraft is equipped with the navigation system showed in Figure 6.1. In particular, the sensor suite consists of the following payloads:

1. Close-range rendezvous sensors such as Lidar or cameras for estimating attitude and distance (pose) of the target spacecraft.
2. GPS providing absolute position $\boldsymbol{r}_{\text{chaser}}$ of the chaser spacecraft in ECI coordinate system.
3. IMU providing the orientation of the chaser spacecraft principal coordinate system C with respect to ECI and the inertial angular velocity $\boldsymbol{\omega}^{\text{C}}$. Star tracker or Sun sensor is required to initialize the attitude estimation.

The navigation software is composed by the following algorithms:

1. Noncooperative pose estimation algorithm for determining the current pose of the target spacecraft, which is not equipped with visual markers, using the data from the close range rendezvous sensors.
2. Software package for estimating inertial position, attitude and velocities from the GPS and IMU data.
3. EKF for determining position, attitude and velocities with respect to the Hill's system given the output data from software 1 and 2.

The main challenge of the navigation system for OOS and space debris removal missions is to determine, with high precision, the current pose of the noncooperative target spacecraft. In fact, this vehicle does not communicate with the chaser spacecraft, it is not equipped with dedicated visual markers for pose estimation, its motion may be uncontrolled and the surfaces may be covered by high-reflective material such as the Multi-Layer Insulation (MLI). On the contrary, GPS and IMU systems are commonly used in space missions, including OOS test missions such as PRISMA [18]. For this reason, the software package related to GPS and IMU is assumed available.

This chapter of the thesis describes the software for noncooperative pose estimation during proximity operations for a space target. Since the analysis in Section 2.3.1 showed that none of the considered close-range rendezvous sensors results superior to the others for this application, the algorithms for pose estimation using Lidar, stereo-camera and mono-camera are developed and presented in the following. The performance of these navigation systems obtained through an experimental campaign and real sensors, will be investigated in Chapter 7. Finally, this chapter develops the EKF for estimating the state vector $\boldsymbol{x}_t$ given the target pose.

6.2 Problem Statement

The sketch of the geometry of target and chaser spacecraft during proximity operations maneuvers is showed in Figure 6.2. As discussed in Section 3.1, the origin of the spacecraft principal coordinate system is located at the center of mass of the vehicle with the axes directed along the principal axes of inertia. The coordinate system S is centered at the chaser spacecraft's rendezvous sensor and it is defined in such a way that the z-axis is oriented along the outward direction from the chaser and the y-axis points to the bottom of the spacecraft. The x-axis completes the right-hand rule.

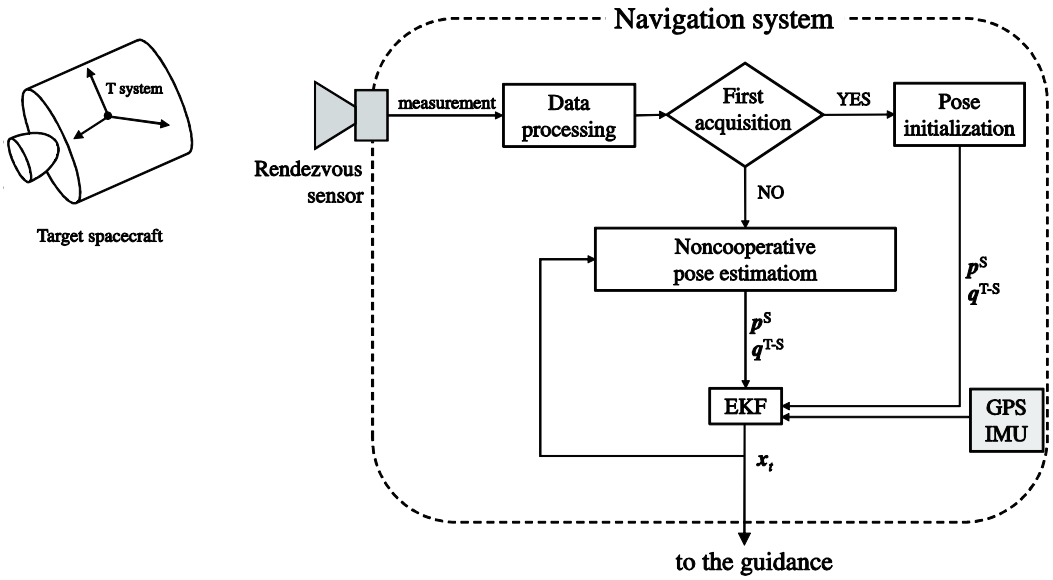

Figure 6.1: Block diagram of the navigation system for autonomous noncooperative docking.

The pose estimation problem during autonomous proximity operations for a noncooperative target spacecraft can be stated as follows: determine current position $\boldsymbol{p}^{\mathrm{S}}$ of the target center of mass and current orientation $\boldsymbol{q}^{\mathrm{T\text{-}S}}$ of the target principal coordinate system with respect to the system S using the measurements provided by the close-range rendezvous sensor and the known geometric model of the target. In fact, it is reasonable to assume that the geometric model of the target spacecraft is available either from the target constructor or from image reconstruction during the inspection phase [97]. Finally, in order to enable onboard trajectory optimization, compute the current system state vector $\boldsymbol{x}_t$ defined in Eq. (5.5) using the estimated pose $\boldsymbol{p}^{\mathrm{S}}$ and $\boldsymbol{q}^{\mathrm{T\text{-}S}}$ of the target.

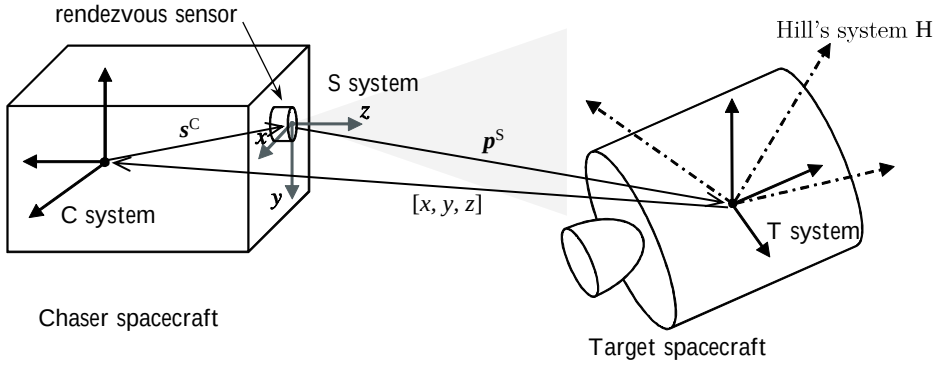

Figure 6.2: Spacecraft geometry and sensor coordinate system.

As can be observed in Figure 6.2, the position of the chaser center of mass with respect to the Hill's coordinate system is related to $\boldsymbol{p}^{\mathrm{S}}$ according to

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = -\boldsymbol{R}_{\mathrm{ECI}}^{\mathrm{H}} \boldsymbol{R}_{\mathrm{C}}^{\mathrm{ECI}} \left[\boldsymbol{R}_{\mathrm{S}}^{\mathrm{C}} \boldsymbol{p}^{\mathrm{S}} + \boldsymbol{s}^{\mathrm{C}} \right], \tag{6.1}$$

where $\boldsymbol{R}_{\mathrm{S}}^{\mathrm{C}}$ and $\boldsymbol{s}^{\mathrm{C}}$ are given geometric characteristics of the chaser spacecraft whereas $\boldsymbol{R}_{\mathrm{C}}^{\mathrm{ECI}}$ is provided by the IMU. The orientation of the Hill's coordinate system with respect to ECI can be obtained as follows. Given the position $\boldsymbol{r}_{\mathrm{chaser}}$ of the chaser spacecraft in ECI (provided by the GPS), the position of the target can be expressed as

$$\boldsymbol{r}_{\mathrm{target}} = \boldsymbol{r}_{\mathrm{chaser}} - \boldsymbol{R}_{\mathrm{H}}^{\mathrm{ECI}} \begin{bmatrix} x \\ y \\ z \end{bmatrix}. \tag{6.2}$$

In the particular case of close-range docking, the chaser is situated few meters away from the target and, therefore, $\boldsymbol{r}_{\mathrm{chaser}} \gg [x, y, z]^{\mathrm{T}}$, which implies $\boldsymbol{r}_{\mathrm{target}} \cong \boldsymbol{r}_{\mathrm{chaser}}$. Assuming the angular momentum $\boldsymbol{h}$ of the target orbit to be known, the orientation of the Hill's coordinate system with respect to ECI is given by

$$\boldsymbol{R}_{\mathrm{H}}^{\mathrm{ECI}} = \begin{bmatrix} \boldsymbol{e}_1 & \boldsymbol{e}_2 & \boldsymbol{e}_3 \end{bmatrix} \tag{6.3}$$

where

$$\boldsymbol{e}_1 = \boldsymbol{r}_{\mathrm{target}} \Big/ \left\| \boldsymbol{r}_{\mathrm{target}} \right\|,$$

$$\boldsymbol{e}_3 = \boldsymbol{h} \Big/ \left\| \boldsymbol{h} \right\|, \tag{6.4}$$

$$\boldsymbol{e}_2 = \boldsymbol{e}_3 \times \boldsymbol{e}_1.$$

In particular, when this routine is executed in closed loop, $\boldsymbol{r}_{\mathrm{target}}$ can be estimated with Eq. (6.2) using $[x, y, z]^{\mathrm{T}}$ and $\boldsymbol{R}_{\mathrm{H}}^{\mathrm{ECI}}$ from the previous loop.

Finally, the orientation of the target principal system T with respect to H can be obtained from the following equation

$$\boldsymbol{R}_{\mathrm{H}}^{\mathrm{T}} = \boldsymbol{R}_{\mathrm{S}}^{\mathrm{T}} \boldsymbol{R}_{\mathrm{C}}^{\mathrm{S}} \boldsymbol{R}_{\mathrm{ECI}}^{\mathrm{C}} \boldsymbol{R}_{\mathrm{H}}^{\mathrm{ECI}}, \tag{6.5}$$

where $\boldsymbol{R}_{\mathrm{S}}^{\mathrm{T}}$ is calculated from the solution $\boldsymbol{q}^{\mathrm{T\text{-}S}}$ of the pose estimation algorithm using Eq. (3.6). Note that these equations are applicable for any rendezvous sensors and pose estimation algorithm employed.

6.3 Lidar Sensor

Lidar is an active sensor that estimates the distance of a target object by illuminating the scene with a laser beam. Distances are then estimated with the time-of-flight principle applied to the reflected signal. In general, this type of sensor requires significant resources in terms of mass and power.

The sensor considered in this thesis and employed in the experimental campaign (see Section 7) is the depth camera of the Microsoft Kinect v2 [98, 99]. Although this sensor is not a proper Lidar, it is based on the same time-of-flight principle applied to an infrared laser beam that illuminates the scene. Whereas a Lidar system uses precise clocks to estimate the time-of-flight of the beam, the Kinect v2 employs the phase shift $\Delta\varphi$ of the received signal to estimate, in real-time, the distance Z^{S} of each point of the scene [98]

$$Z^{\mathrm{S}} = \frac{\Delta\varphi}{4\pi f} c\,, \tag{6.6}$$

where c is the speed of light and f the modulation frequency. Notably, the depth values Z^{S} are automatically provided by the Microsoft Kinect v2's internal software as a matrix corresponding to the sensor resolution. The coordinates X^{S} and Y^{S} of each point of the acquired depth image can be calculated from Z^{S} using the standard pinhole camera model [100]:

$$\begin{aligned} X^{\mathrm{S}} &= \frac{u - c_x}{f_x} Z^{\mathrm{S}} \\ Y^{\mathrm{S}} &= \frac{v - c_y}{f_y} Z^{\mathrm{S}} \end{aligned}\,, \tag{6.7}$$

where (u,v) are the coordinates of the pixel that corresponds to the measured Z^{S}. The sketch of the standard pinhole camera model is showed in Figure 6.3. Since the sensor for acquiring the reflected beam uses standard lenses, the point cloud suffers from distortion effects in the (x, y) plane (Figure 6.4). The coordinate values obtained from Eq. (6.7) are corrected using the following Brown-Conrady model [101, 102, 103]

$$\begin{aligned} X_{\mathrm{u}}^{\mathrm{S}} &= X^{\mathrm{S}}\left(1 + K_1 r^2 + K_2 r^4 + K_3 r^6\right) + \left[P_2\left(r^2 + 2X^{\mathrm{S}^2}\right) + 2P_1 X^{\mathrm{S}} Y^{\mathrm{S}}\right] \\ Y_{\mathrm{u}}^{\mathrm{S}} &= Y^{\mathrm{S}}\left(1 + K_1 r^2 + K_2 r^4 + K_3 r^6\right) + \left[P_1\left(r^2 + 2Y^{\mathrm{S}^2}\right) + 2P_2 X^{\mathrm{S}} Y^{\mathrm{S}}\right] \end{aligned} \tag{6.8}$$

where X_u^S and Y_u^S denote the undistorted coordinates and

$$r = \sqrt{\left(X^S - u\right)^2 + \left(Y^S - v\right)^2} \,. \tag{6.9}$$

The coefficients K_i and P_i are obtained from the calibration of the Kinect v2 sensor. The output point cloud $\boldsymbol{D}$ that will be used for pose estimation is therefore defined by the coordinates (X_u^S, Y_u^S, Z^S).

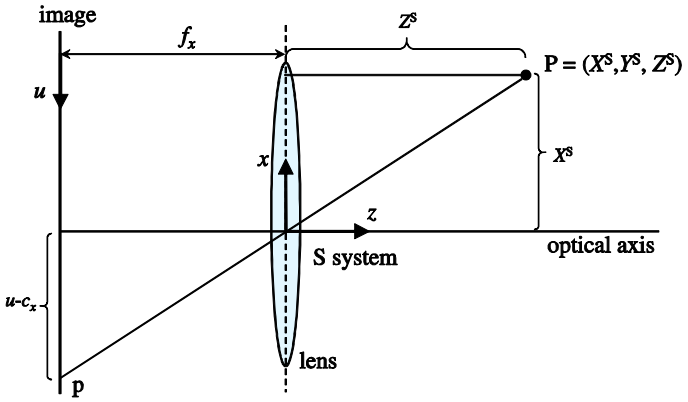

Figure 6.3: Pinhole camera model.

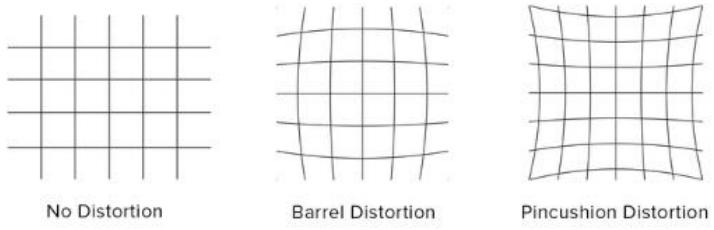

Figure 6.4: Lens distortion types.

The proposed pose estimation procedure employs the ICP algorithm to determine current attitude and position of the target spacecraft. The selected method calculates iteratively the rigid transformation that aligns the measured point cloud $\boldsymbol{D}$ of the target spacecraft with the known 3D model of the same vehicle, which is stored onboard the chaser as a set of points. The choice of employing ICP is motivated by its effectiveness for real-time matching of misaligned 3D shapes and robustness in real-world applications [104]. In fact, this method is broadly used in terrestrial robots for 3D surface reconstruction and robot localization. Furthermore, several variants of the original procedure have been introduced to improve the performance and to adapt ICP to different real-world scenarios [105].

To ensure fast convergence and onboard execution, the ICP algorithm must be applied to slightly misaligned shapes. Let now $\boldsymbol{X}$ be the $3 \times n$ set of

n points defining the reference geometry of the target spacecraft and expressed in the sensor coordinate system S. As illustrated in Figure 6.5, the point cloud $\boldsymbol{X}$ is constructed in such a way that the target model center of mass is located at the origin of the system S and the axes of T are aligned with the axes of S. Let $\boldsymbol{D}$ be the 3 x m set of m points measured by the chaser spacecraft rendezvous sensor. The sketch of the point cloud acquired by the sensor is showed in Figure 6.6. Originally, the point clouds $\boldsymbol{X}$ and $\boldsymbol{D}$ are significantly misaligned due to the relative distance $\boldsymbol{p}^{\mathrm{S}}$. In order to apply efficiently the ICP procedure, the position $\boldsymbol{p}^{\mathrm{S}}$ of the target spacecraft is decomposed as

$$\boldsymbol{p}^{\mathrm{S}} = \boldsymbol{\mu}_{\boldsymbol{D}}^{\mathrm{S}} + \boldsymbol{d}^{\mathrm{S}}, \tag{6.10}$$

where $\boldsymbol{\mu}_{\boldsymbol{D}}^{\mathrm{S}}$ is calculated from $\boldsymbol{D}$ and $\boldsymbol{d}^{\mathrm{S}}$ is the unknown distance of the target vehicle center of mass from the centroid of the measurement (see Figure 6.6). The vector decomposition in Eq. (6.10) allows converting the pose estimation problem posted in Section 6.2 into the equivalent problem of determining $\boldsymbol{d}^{\mathrm{S}}$ and $\boldsymbol{q}^{\mathrm{T\text{-}S}}$ that align $\boldsymbol{X}$ with the following point cloud

$$\hat{\boldsymbol{D}}_i = \boldsymbol{D}_i - \boldsymbol{\mu}_{\boldsymbol{D}}^{\mathrm{S}}, \quad i = 1,2,\ldots,m \tag{6.11}$$

which corresponds to $\boldsymbol{D}$ centered at the origin of S. The ICP routine is then executed using $\hat{\boldsymbol{D}}$ and $\boldsymbol{X}$ as input. Notably these clouds are slightly misaligned since the distance is $\boldsymbol{d}^{\mathrm{S}}$. The position $\boldsymbol{p}^{\mathrm{S}}$ is calculated from the solution of the ICP algorithm using Eq. (6.10).

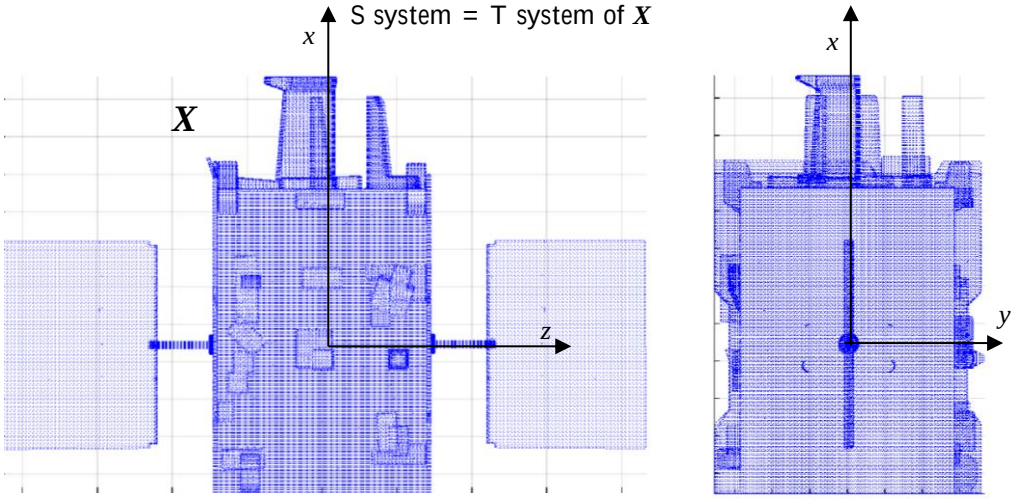

Figure 6.5: Reference point cloud model $\boldsymbol{X}$ of the target spacecraft. The spacecraft corresponds to the mockup in RACOON laboratory (see Section 7).

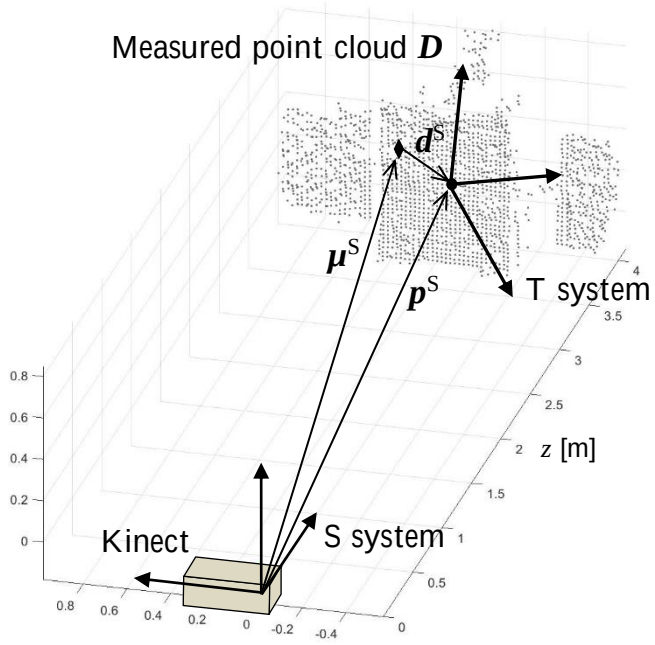

Figure 6.6: Kinect sensor and acquired point cloud used for noncooperative pose estimation.

In the proposed pose estimation procedure, the goal of the ICP algorithm is to find the rotation $\boldsymbol{q}^*$ and the translation $\boldsymbol{t}$ that move $\hat{\boldsymbol{D}}$ to be in the best correspondence with $\boldsymbol{X}$. This transformation can be determined by minimizing the following point-to-point Euclidean distance [106]

$$d = \sum_{i=1}^{m} \left\| \boldsymbol{X}_i^{\mathrm{cp}} - \boldsymbol{R}^* \hat{\boldsymbol{D}}_i - \boldsymbol{t} \right\|, \tag{6.12}$$

where $\boldsymbol{R}^*$ is obtained from $\boldsymbol{q}^*$ using Eq. (3.6). Notably the point-to-point metric has been chosen since it ensures superior performance in terms of computational time and accuracy if compared to the point-to-plane routine [96]. The parameters $\boldsymbol{q}^{\mathrm{T\text{-}S}}$ and $\boldsymbol{d}^{\mathrm{S}}$ that solve the decomposed pose estimation problem are obtained from the solution of the ICP procedure using the following equations:

$$\boldsymbol{q}^{\mathrm{T\text{-}S}} = (\boldsymbol{q}*)^{-1} \tag{6.13}$$

$$\boldsymbol{d}^{\mathrm{S}} = -\boldsymbol{R}^{*\mathrm{T}} \boldsymbol{t}\,. \tag{6.14}$$

The minimization of the point-to-point distance through the ICP routine consists of the following steps summarized from Ref. [105, 107]:

1. Initialize the algorithm by providing a first guess for $\boldsymbol{q}^*$ and $\boldsymbol{t}$. Set the counter k = 0.
2. For each point of the transformed data $\hat{\boldsymbol{D}}_k$, compute the set $\boldsymbol{X}^{\mathrm{cp}}$ of closest point in $\boldsymbol{X}$.
3. Use a trimming procedure to eliminate the outliers [105].
4. Compute the rotation $\boldsymbol{q}^*_{k+1}$ and the translation $\boldsymbol{t}_{k+1}$ that minimize the point-to-point error metric in Eq. (6.12) using the algorithm described Ref. [106]. Use the corresponding points $\boldsymbol{X}^{\mathrm{cp}}$ calculated in Steps 2 and 3.
5. Apply the transformation

$$\hat{\boldsymbol{D}}_{k+1} = \boldsymbol{R}^*_{k+1} \hat{\boldsymbol{D}} + \boldsymbol{t}_{k+1}, \tag{6.15}$$

   where $\boldsymbol{R}^*_{k+1}$ is computed from $\boldsymbol{q}^*_{k+1}$ using Eq. (3.6).
6. Evaluate the stopping criteria: terminate the routine if the ration of distance d_{k+1}/d_k is below a certain threshold or k reaches the maximum number of iterations. Otherwise, repeat from Step 2.

Notably for Step 2, the closest points $\boldsymbol{X}^{\mathrm{cp}}$ are computed with the nearest neighbor algorithm using a k-d tree structure of $\boldsymbol{X}$. Although the nearest neighbor search is not fastest matching procedure, it has been chosen for its robustness to noise and efficacy to converge when applied to complex geometries [104]. Lastly, the ICP routine is executed in closed loop during the maneuver and the first guess for $\boldsymbol{q}^*$ and $\boldsymbol{t}$ is obtained from the solution of the previous loop.

Since the first guess on position and attitude depends on the previous ICP solution, the pose estimation procedure must be initialized during the first

loop execution. A template matching routine is thus executed in order to estimate attitude and position of the target spacecraft using the ICP algorithm without prior state information. First, a database of the possible attitude configurations of the target is generated onboard by discretizing the Euler angles in the interval [-180; 180] deg. The ICP algorithm is then executed using each attitude configuration as first guess. The first pose of the target corresponds to the ICP solution that minimizes Eq. (6.12). More specifically, the proposed template matching algorithm takes the following steps:

1. Decouple position and attitude using Eqs. (6.10, 6.11).
2. Define the sampling angular step $\Delta\theta$. Generate the database of the possible target attitude configurations by discretizing each Euler angle of a chosen sequence in the interval [-180; 180] deg.
3. For each attitude configuration of the database, evaluate the distance in Eq. (6.12) with $\boldsymbol{t} = \mathbf{0}$. Reject the attitude configurations corresponding to the largest distances.
4. Execute the ICP algorithm using each selected attitude configuration as first guess on $\boldsymbol{q}^*$. The guess on $\boldsymbol{t}$ is $\mathbf{0}$.
5. Among the solutions provided by ICP, select the one that minimizes Eq. (6.12).
6. Calculate $\boldsymbol{q}^{\text{T-S}}$ and position $\boldsymbol{p}^{\text{S}}$ using Eqs. (6.10, 6.13, 6.14).

In particular, the rejection of the worst attitude configurations in Step 3 is executed in order to speed-up the template matching procedure. In this way, indeed, the ICP algorithm is executed using the best attitude configuration candidate only (Step 4).

The Lidar-based navigation system consists of the closed-loop implementation of the pose estimation algorithm and of the template matching routine. The block diagram of such loop is represented in Figure 6.7. During the maneuver, the Lidar acquires the depth information Z^{S} of the scene at a certain frequency. The depth map block generates the point cloud from the measured Z^{S} using Eq. (6.7). The depth scene is then corrected for lens distortion using Eq. (6.8). The state prediction block propagates the dynamics model of the docking maneuver and estimates the expected current orientation $\boldsymbol{q}^{\text{T-S}}_{\text{predicted}}$ and position $\boldsymbol{p}^{\text{S}}_{\text{predicted}}$ of the target spacecraft using Eqs. (6.1, 6.5). The ICP algorithm is then executed to estimate the current target pose. Notably the first guesses on $\boldsymbol{q}^*$ and $\boldsymbol{t}$ are computed from the predicted state as

$$\boldsymbol{q}^*_{\text{guess}} = \left(\boldsymbol{q}^{\text{T-S}}_{\text{predicted}}\right)^{-1} \tag{6.16}$$

$$\boldsymbol{t}_{\text{guess}} = \boldsymbol{R}^*_{\text{guess}}\left[\boldsymbol{\mu}^{\text{S}}_{\boldsymbol{D}} - \boldsymbol{p}^{\text{S}}_{\text{predicted}}\right]. \tag{6.17}$$

The complete state vector x_t is provided by the EKF using the solution of the pose estimation algorithm and the data from GPS and IMU. The estimated current state is fed to the guidance system and to the state estimation block of the following loop execution. The template matching procedure is executed after the first data acquisition in order to initialize the navigation system.

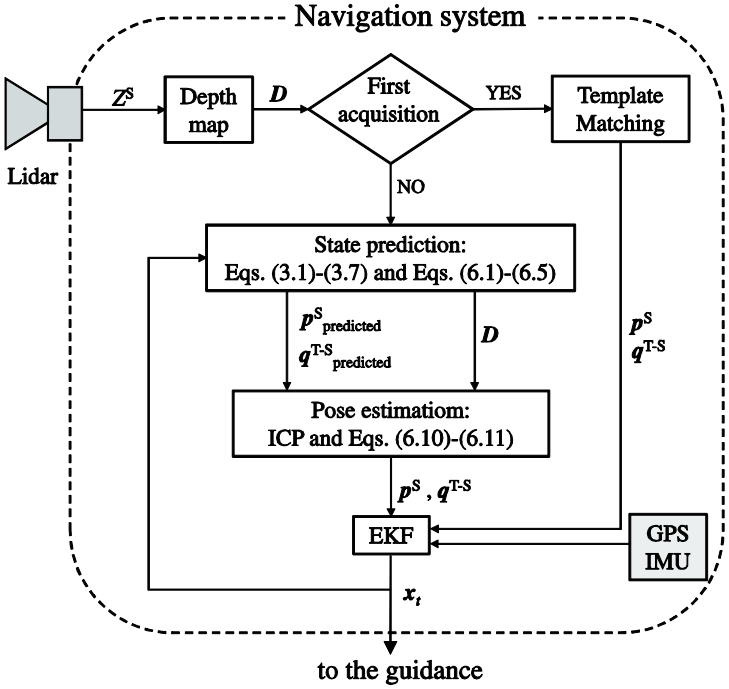

Figure 6.7: Block diagram of the Lidar-based navigation system.

6.4 Stereo Camera

Stereo vision technology allows depth measurement using images taken from slightly offset cameras mounted in the stereo camera sensor. Similarly to the human binocular vision, the depth of the scene is obtained from the feature correspondences in left and right images. This section provides the procedure to obtain the 3D point cloud from a stereo image, i.e. left and right images taken simultaneously by the stereo camera. The noncooperative pose estimation procedure for proximity operation maneuvers is then discussed.

The stereo camera sensor mounted on the chaser spacecraft provides two raw images of the target taken simultaneously from different views. In this thesis, these views are denoted as left and right. The first step in depth image reconstruction is to correct the raw images for radial and tangential distortions caused by non-ideal lenses using Eq. (6.8). In this way, the projection equations of an ideal pinhole camera can be applied for depth scene

reconstruction. The corrected images are then projected onto a common image plane to facilitate feature matching. This procedure is called image rectification [108].

The next step for depth reconstruction consists of calculating the disparity map from the pair of grayscale undistorted rectified images. The disparity is defined as the difference in pixel between the coordinates of the same feature in the left and right images. To compute the disparity of each pixel, several methods are available. In this thesis, the Sum of Absolute Differences (SAD) algorithm [109] is applied since it ensures the best trade-off between robustness and computational speed†. Given a pixel (u,v) of interest in the left image, the disparity d is computed by the SAD algorithm by minimizing the following error function:

$$\Phi_{\mathrm{SAD}} = \sum_{i=-\frac{M}{2}}^{i=\frac{M}{2}} \sum_{j=-\frac{M}{2}}^{j=\frac{M}{2}} \left\| I_{\mathrm{left}}(u+i+d, v+j) - I_{\mathrm{right}}(u+i, v+j) \right\|, \tag{6.18}$$

where M is the size of a square window centered at the pixel of interest (u,v) and $I(u,v)$ is the grayscale intensity value of the considered pixel. This procedure is repeated for each pixel in the left image. As can be observed in Eq. (6.18), the SAD algorithm searches the feature matches horizontally only. This can be done since the images are rectified.

The point cloud $\boldsymbol{D}$ is retrieved from the disparity map by applying the triangulation technique as showed in Figure 6.8. Without loss of generality, the coordinate system S that defines the rendezvous sensor is located at the left camera center. As can be observed from the figure, the ideal projections of a real point P = $(X^{\mathrm{S}}, Y^{\mathrm{S}}, Z^{\mathrm{S}})$ onto left and right images are $(u_{\mathrm{left}}, v_{\mathrm{left}})$ and $(u_{\mathrm{right}}, v_{\mathrm{right}})$ respectively. Using the disparity $d = u_{\mathrm{left}} - u_{\mathrm{right}}$ estimated by the SAD algorithm, the three dimensional coordinates of the point P can be obtained from the following equation derived from Figure 6.8:

$$\begin{aligned} Z^{\mathrm{S}} &= \frac{bf}{d} \\ X^{\mathrm{S}} &= \frac{u_{\mathrm{left}} - c_x}{f_x} Z^{\mathrm{S}} \\ Y^{\mathrm{S}} &= \frac{v_{\mathrm{left}} - c_y}{f_y} Z^{\mathrm{S}} \end{aligned}. \tag{6.19}$$

Notably the focal length f used to calculate Z^{S} is obtained through the calibration of the stereo camera.

The steps of stereo image processing are summarized in Figure 6.9. A test stereo image of a target spacecraft taken from RACOON laboratory is considered. Input images are already undistorted and rectified.

† http://vision.middlebury.edu/stereo/eval3/

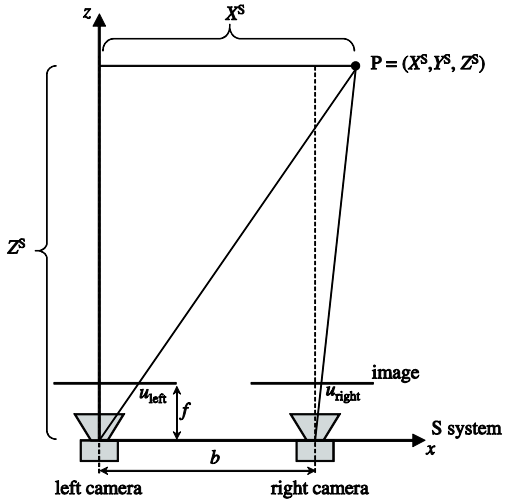

Figure 6.8: Triangulation model for depth scene reconstruction using stereo images.

The output of the depth image reconstruction from stereo image is a point cloud $\boldsymbol{D}$ of the target spacecraft defined in the sensor coordinate system S. Similarly to the Lidar-based system, the pose estimation of the target vehicle is executed using the ICP algorithm as described in Section 6.3.

The block diagram of the stereo camera-based navigation system is showed in Figure 6.10. The stereo camera acquires left and right images of the scene at a certain frequency during the maneuver. The stereo image is processed in the disparity & depth map block for generating the point cloud $\boldsymbol{D}$ of the target spacecraft. The state prediction block propagates the dynamics model of the docking maneuver and estimates the expected current orientation $\boldsymbol{q}_{\text{predicted}}^{\text{T-S}}$ and position $\boldsymbol{p}_{\text{predicted}}^{\text{S}}$ of the target spacecraft using Eqs. (6.1, 6.5). The ICP algorithm is then executed to estimate the current target pose as discussed in 6.3. Notably the first guesses on $\boldsymbol{q}^*$ and $\boldsymbol{t}$ are computed from the predicted state using Eqs. (6.16, 6.17). The complete state vector $\boldsymbol{x}_t$ is provided by the EKF using the solution of the pose estimation algorithm and the data from GPS and IMU. The estimated current state is fed to the guidance system and to the state estimation block of the following loop execution. The template matching procedure is executed after the first data acquisition in order to initialize the navigation system.

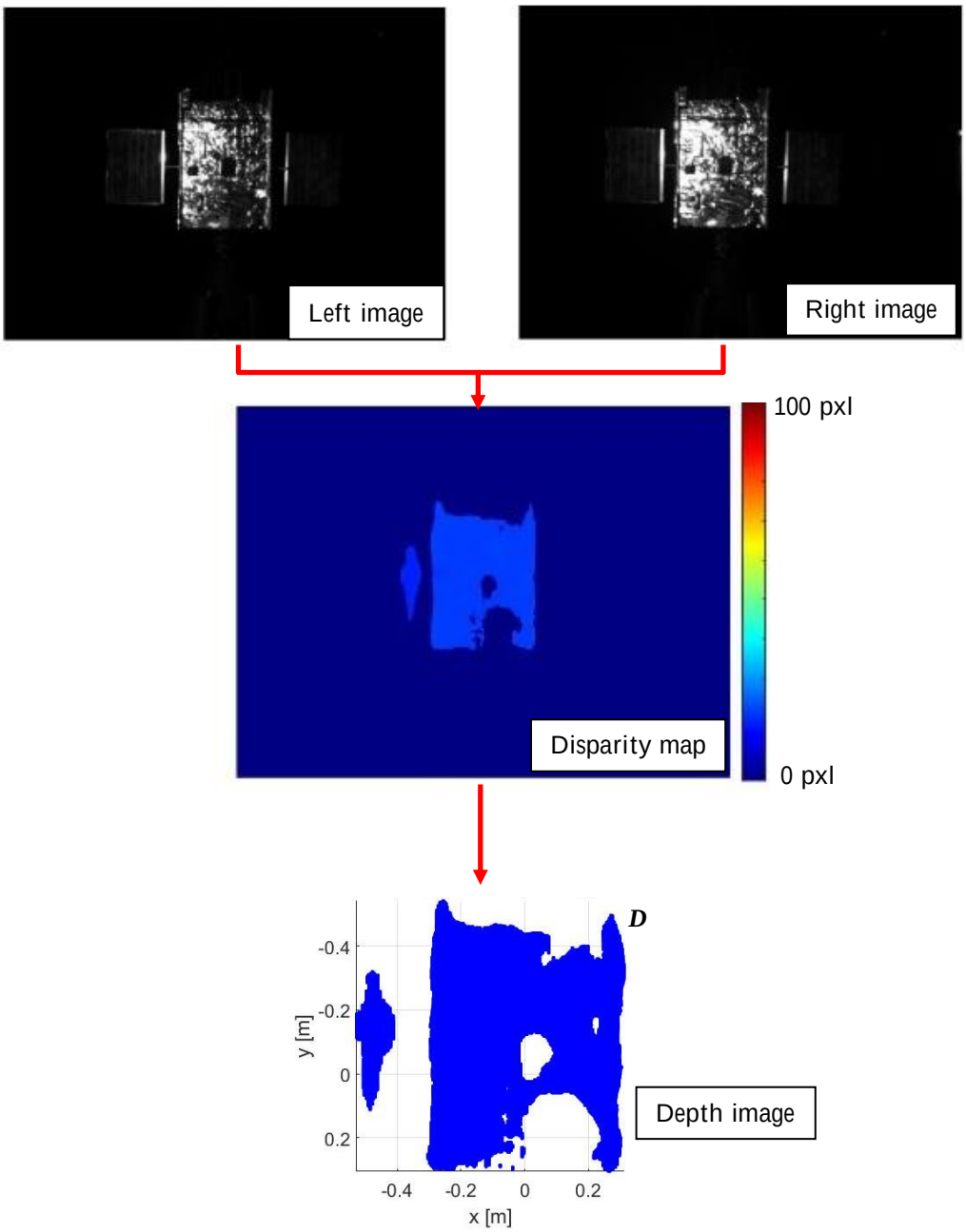

Figure 6.9: Steps of the stereo image processing for depth reconstruction.

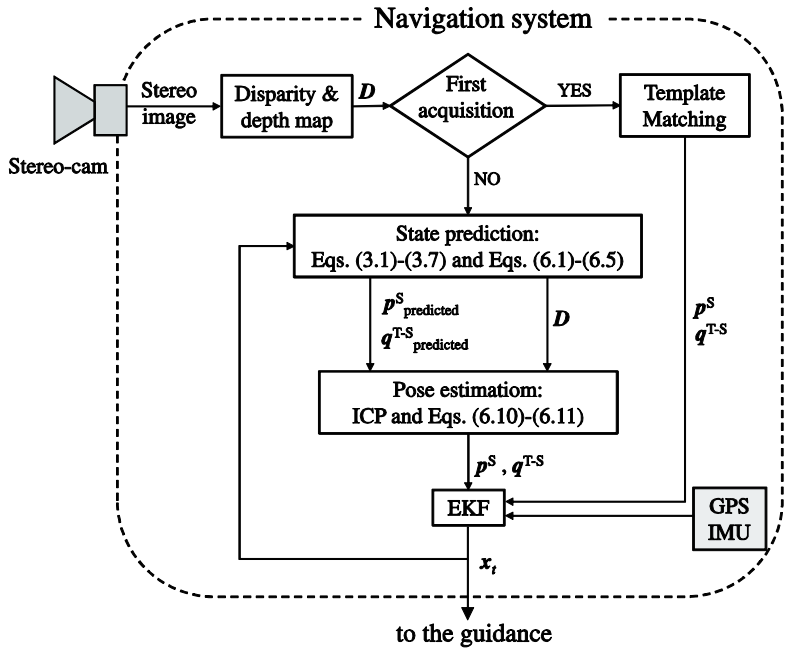

Figure 6.10: Block diagram of the stereo camera-based navigation system.

6.5 Mono Camera

Monocular vision navigation was previously employed in OOS test missions due to the low power and mass requirements of camera sensors. For instance, the Advance Video Guidance Sensor system of the Orbital Express performed cooperative pose estimation by detecting the visual marker pattern of the target spacecraft surface in the acquired image [17]. Noncooperative pose estimation from monocular imagery was employed in the PRISMA mission for angles-only navigation during the mid-range rendezvous phase [18]. Although monocular vision navigation has been considered a promising technology for future OOS and space debris removal missions, the application to noncooperative pose estimation during close-range proximity operations eventually until the docking hasn't been tested yet. The main challenges are indeed the lack of visual markers on the target satellite and the uncontrolled motion of the vehicle. This section describes a complete monocular-based pose estimation system for proximity operations with a noncooperative target spacecraft. The routine includes both tracking procedure and pose initialization algorithms.

The pose estimation algorithm used in the Orbital Express mission was based on the comparison between the visual markers on the target spacecraft detected in the image and the known model of the same pattern. The procedure for noncooperative pose estimation using monocular imagery proposed in this thesis follows the same approach. Instead of visual markers,

the image processing algorithm aims to identify key geometric features of the target spacecraft in the acquired image. The detected features are compared with the known geometric model of the same vehicle to estimate distance and attitude. Notably the reference geometry is a real scale wireframe model of the target spacecraft, which is defined in the target principal coordinate system T and stored onboard the chaser. The spacecraft considered in this section is the mockup model of the target spacecraft of RACOON laboratory. The wireframe model of such satellite is showed in Figure 6.11. Unlike the navigation system of Orbital Express, the proposed procedure is designed to cope with the uncontrolled motion of the noncooperative target vehicle, which may tumble about the principal axes. The noncooperative pose estimation procedure consists of three steps:

1. image processing to extract the key geometric features of the target from the image;
2. feature matching to pair the detected features with the corresponding details of the target model;
3. pose estimation.

Similarly to the stereo camera sensor, the mono camera provides raw images of the target vehicle. The procedure in Eqs. (6.8, 6.9) is therefore applied in order to remove radial and tangential distortions caused by non-ideal lenses. The Gaussian filter [110] is then employed to reduce the noise on the image.

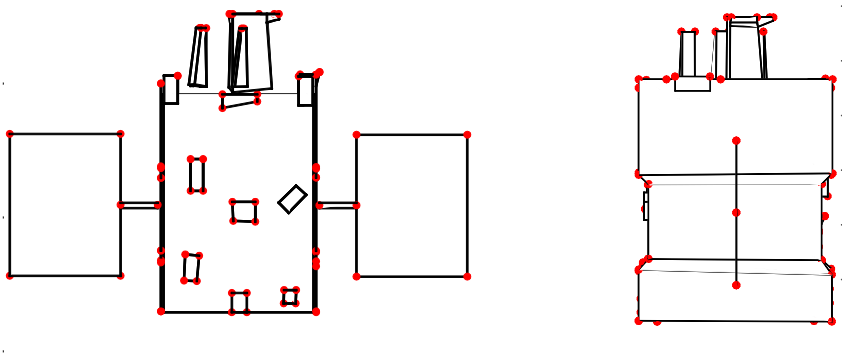

Figure 6.11: Real scale wireframe model of the target spacecraft (mockup model of RACOON laboratory). The satellite's features are denoted by the red dots.

The geometric features of the target spacecraft are identified using the Harris corner detector algorithm [111]. This method has been preferred among others since it ensures robustness to feature outliers along with high computational speed [112]. Note that the computational speed of the navigation system is critical to ensure robustness to sensor noise and to model uncertainties during the docking maneuver (see Section 5.7). Figure 6.12 shows a simplified example of a corner in an image. As can be observed from the figure, a corner is defined by the intersection of two edges and it represents the point where the directions of these two edges change drastically.

Hence, a corner in the image is a point having a high variation of gradient. The Harris matrix operator of the algorithm is therefore defined as [111]

$$\boldsymbol{A} = \begin{bmatrix} \sum G_u^2 & \sum\sum G_u G_v \\ \sum\sum G_u G_v & \sum G_v^2 \end{bmatrix}, \tag{6.20}$$

where G_u and G_v are the intensities of the gradient along the u- and v- axes of the image respectively. The Harris matrix operator $\boldsymbol{A}$ is built on every pixel of the image over a circular region whose radius is defined by the user. The matrices having the largest eigenvalues correspond to the largest variation of gradient and, therefore, to the corners.

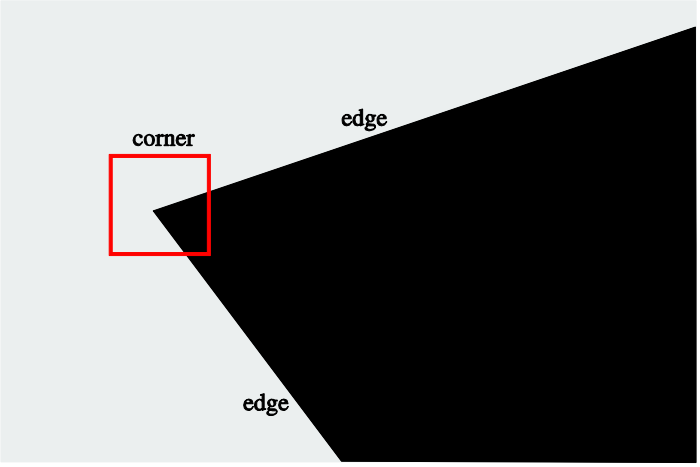

Figure 6.12: Corner feature in the image.

In order to couple the detected corners with the corresponding features of the target spacecraft, the 3D wireframe model of the target is projected onto the image using the 3D-2D true perspective projection equations:

$$\boldsymbol{\alpha}^{\mathrm{S}} = \boldsymbol{R}_{\mathrm{T}}^{\mathrm{S}} \boldsymbol{\alpha}^{\mathrm{T}} + \boldsymbol{p}^{\mathrm{S}} \tag{6.21}$$

$$\boldsymbol{\rho} = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \dfrac{\alpha_x^S}{\alpha_z^S} f_x + c_x \\ \dfrac{\alpha_y^S}{\alpha_z^S} f_y + c_y \end{bmatrix}, \tag{6.22}$$

where $\boldsymbol{\alpha}^{\mathrm{T}}$ is the feature point of the target model defined in the principal coordinate system T, $\boldsymbol{\alpha}^{\mathrm{S}}$ is the same point after applying the current pose $\boldsymbol{R}_{\mathrm{T}}^{\mathrm{S}}$ and $\boldsymbol{p}^{\mathrm{S}}$ of the target and $\boldsymbol{\rho}=(u,v)$ is the pixel of the image corresponding to the considered feature. In order to project only the visible features, the

wireframe model of the target vehicle is first rotated using Eq. (6.21). Note that this procedure requires a guess on the current pose of the target. The *z*-buffer algorithm [113] is then employed to detect the features visible from the camera. These features are subsequently projected onto the image using Eq. (6.22). The detected corners and projected visible features are finally matched using the nearest neighbor search [114]. In particular, the number of such matches is a parameter that must be set by the user. Figure 6.13 shows the corner detected by the Harris algorithm and the projected wireframe model of the target spacecraft. Notably the visible features are indicated by red points and are coupled with the corresponding detected corners using the nearest neighbor search.

The correspondences between detected corners and projected features of the target model are used to solve the 3D-2D true perspective projection Eqs. (6.21, 6.22) for $\boldsymbol{R}_{\mathrm{T}}^{\mathrm{S}}$ and $\boldsymbol{p}^{\mathrm{S}}$ using the Newton-Raphson technique as in Ref. [115]. For each match, the following fit error between detected corner and projected feature is defined:

$$\boldsymbol{E} = \boldsymbol{\rho}_{\mathrm{model}} - \boldsymbol{\rho}_{\mathrm{Harris}} = \begin{bmatrix} u_{\mathrm{model}} - u_{\mathrm{Harris}} \\ v_{\mathrm{model}} - v_{\mathrm{Harris}} \end{bmatrix}, \tag{6.23}$$

where $\boldsymbol{\rho}_{\mathrm{model}}$ is the projection of the visible geometric feature of the target model computed with Eqs. (6.21, 6.22), whereas $\boldsymbol{\rho}_{\mathrm{Harris}}$ is the corner point obtained with the Harris algorithm. The error in Eq. (6.23) has 6 unknown parameters being $\boldsymbol{x} = [\boldsymbol{p}^{\mathrm{S}}\ \boldsymbol{\theta}_{\mathrm{ST}}]$, where $\boldsymbol{\theta}_{\mathrm{ST}}$ is the Euler angles sequence that defines the rotation matrix $\boldsymbol{R}_{\mathrm{T}}^{\mathrm{S}}$. Since each feature correspondence provides two conditions, at least 3 matches between detected corners and corresponding projected features are required to solve the system of equations defined by the fit error in Eq. (6.23). Let us now assume that n matches are employed to determine the pose. The system of 2 x n equations that must be solved for $\boldsymbol{x}$ is given by

$$\boldsymbol{E}_{\mathrm{system}} = \begin{bmatrix} \boldsymbol{E}_1 \\ \vdots \\ \boldsymbol{E}_n \end{bmatrix}. \tag{6.24}$$

The Newton-Raphson algorithm solves this system of equations by iteratively updating the solution as

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \left(\boldsymbol{J}^{\mathsf{T}}\boldsymbol{J}\right)^{-1}\boldsymbol{J}^{\mathsf{T}}\boldsymbol{E}_{\mathrm{system}}(\boldsymbol{x}_k), \tag{6.25}$$

where $\boldsymbol{E}_{\mathrm{system}}(\boldsymbol{x}_k)$ is evaluated using Eq. (6.24) at $\boldsymbol{x}_k$ and $\boldsymbol{J}$ is the Jacobian of $\boldsymbol{E}_{\mathrm{system}}$:

$$J = \frac{\partial \boldsymbol{E}_{\text{system}}}{\partial \boldsymbol{x}} = \begin{bmatrix} \frac{\partial \boldsymbol{\rho}_{\text{model}_1}}{\partial \boldsymbol{\alpha}_1^{\text{S}}} \frac{\partial \boldsymbol{\alpha}_1^{\text{S}}}{\partial \boldsymbol{p}^{\text{S}}} & \frac{\partial \boldsymbol{\rho}_{\text{model}_1}}{\partial \boldsymbol{\alpha}_1^{\text{S}}} \frac{\partial \boldsymbol{\alpha}_1^{\text{S}}}{\partial \boldsymbol{\theta}_{\text{ST}}} \\ \vdots & \vdots \\ \frac{\partial \boldsymbol{\rho}_{\text{model}_n}}{\partial \boldsymbol{\alpha}_n^{\text{S}}} \frac{\partial \boldsymbol{\alpha}_n^{\text{S}}}{\partial \boldsymbol{p}^{\text{S}}} & \frac{\partial \boldsymbol{\rho}_{\text{model}_n}}{\partial \boldsymbol{\alpha}_n^{\text{S}}} \frac{\partial \boldsymbol{\alpha}_n^{\text{S}}}{\partial \boldsymbol{\theta}_{\text{ST}}} \end{bmatrix}. \tag{6.26}$$

The partial derivatives in Eq. (6.26) are obtained from Eqs. (6.21, 6.22):

$$\frac{\partial \boldsymbol{\rho}_{\text{model}}}{\partial \boldsymbol{\alpha}^{\text{S}}} = \begin{bmatrix} \frac{f_x}{\alpha_z^S} & 0 & -\frac{\alpha_x^S f_x}{\alpha_z^{S2}} \\ 0 & \frac{f_y}{\alpha_z^S} & -\frac{\alpha_y^S f_y}{\alpha_z^{S2}} \end{bmatrix} \tag{6.27}$$

$$\frac{\partial \boldsymbol{\alpha}^{\text{S}}}{\partial \boldsymbol{p}^{\text{S}}} = \boldsymbol{R}_{\text{T}}^{\text{S}} \boldsymbol{\alpha}^{\text{T}} \tag{6.28}$$

$$\frac{\partial \boldsymbol{\alpha}^{\text{S}}}{\partial \boldsymbol{\theta}_{\text{ST}}} = \frac{\partial \boldsymbol{R}_{\text{T}}^{\text{S}}}{\partial \boldsymbol{\theta}_{\text{ST}}} \boldsymbol{\alpha}^{\text{T}}. \tag{6.29}$$

The iterative routine stops when either the improvement of the solution achieves the tolerance or the number of iterations reaches the maximum number.

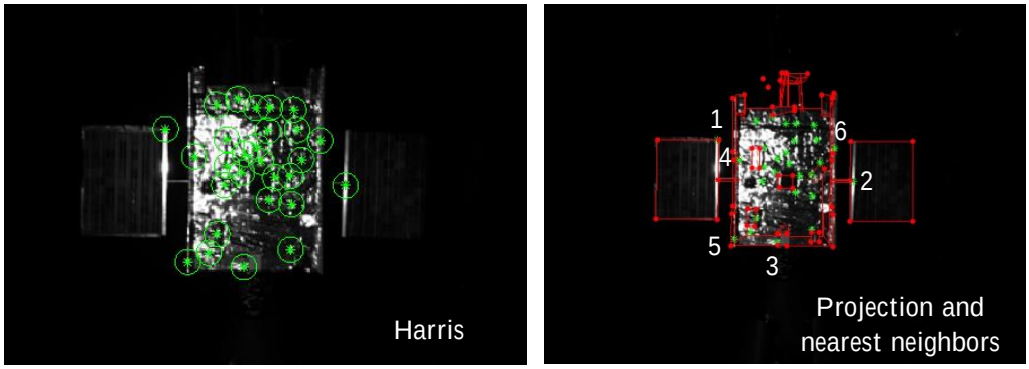

Figure 6.13: Harris corner detection algorithm and nearest neighbors (best 6 matches, see numbers) between corners and projected wireframe model.

As already mentioned, a guess on the current pose is required to project the wireframe model onto the image and to start the iterative Newton-

Raphson routine to solve the 3D-2D perspective Eqs. (6.21, 6.22). The monocular-based navigation system must be therefore initialized through a pose initialization algorithm. As already discussed in previous researches, pose initialization represents the most challenging part of the monocular vision navigation system since depth information are missing [46]. The pose initialization algorithm used in this thesis is based on the work of Ref. [46], where a conceptual routine for initial pose estimation of the target spacecraft of the PRISMA mission is described. The routine consists of the following steps:

1. image processing to extract line segments corresponding to the true edges of the target spacecraft;
2. feature matching to pair the detected geometric shapes with the corresponding shapes of the target model;
3. pose estimation algorithm to solve Eqs. (6.21, 6.22) without pose guess.

The image processing part aims to identify in the image the line segments that correspond to the visible true edges of the target spacecraft such as the edges of the bus, solar panels and antennae. The motivation of detecting edges instead of corner is double. First, the pose initialization is assumed to be performed at the beginning of the proximity operation procedures, which start at approximately 20 m from the target vehicle [56]. At that distance, only significant geometric features such as edges of large components can be clearly detected on the image. Small surface details are not visible, especially in low-resolution cameras. The second reason is that the proposed pose estimation routine requires at least four pairs of corresponding image and model features to solve the 3D-2D perspective equation. Since the solution guess is unknown, all the possible combinations between detected features and model points are considered. Clearly, combining the endpoints of significant geometric shapes instead of surface features drastically reduces the number of combinations and, therefore, the computational time of the algorithm. This aspect will be discussed again in the following.

The state-of-art procedure described in Ref. [46] is employed to detected line segments of the image that correspond to the visible true edges of the target spacecraft. Fist, the Gaussian filter is applied to the image after correction from lens distortions (Section 6.3) in order to reduce the noise. The Sobel operator is then executed to highlight the edges of the image [116]. Finally, the Hough transform is employed to extract straight lines from the detected edges [117]. This procedure is illustrated in Figure 6.14 on a test image taken from the RACOON laboratory. Whereas this routine is a standard approach for feature extraction, it requires the tuning of several parameters such as the kernel size and standard deviation of the Gaussian filter and thresholding, segment length and number of detected segments for the Hough transform. In general, these parameters are difficult to tune for broad applicability.

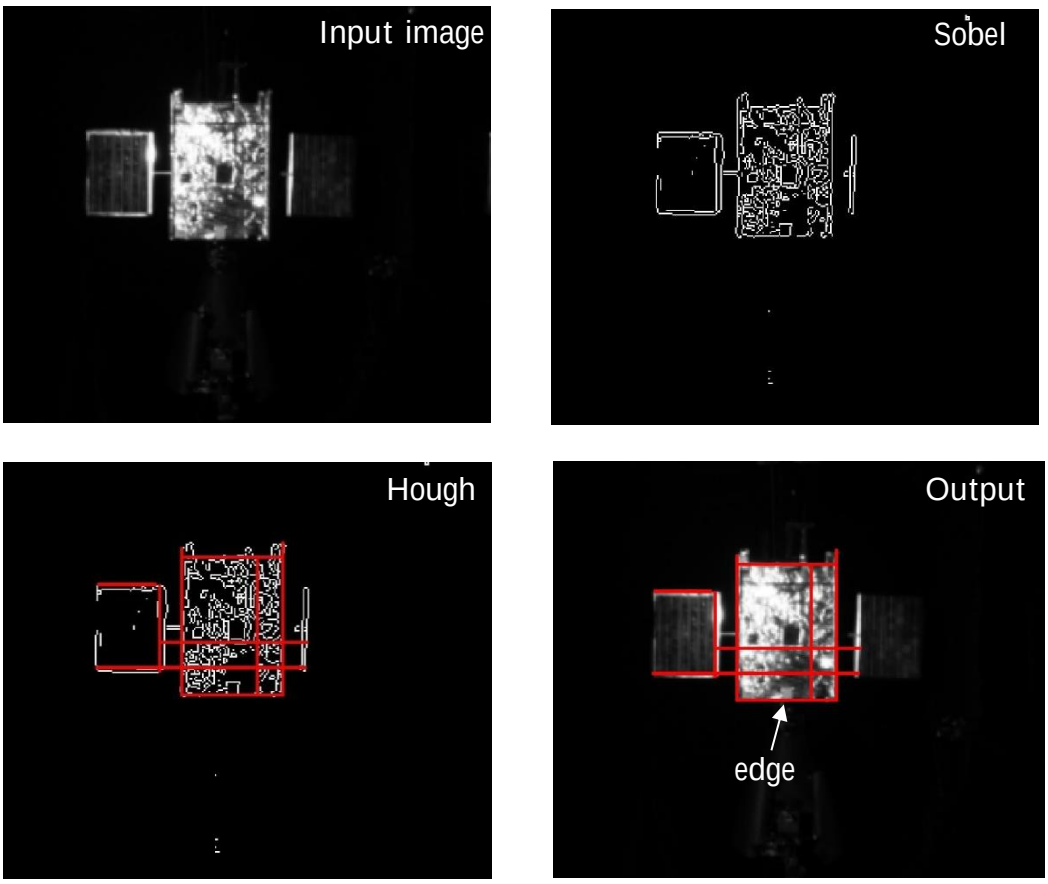

Figure 6.14: Steps of the image processing routine for monocular pose initialization: Sobel operator (top right) on the input image, Hough transform (bottom left) of the image edges and final output (bottom right).

The output of the Hough transform is a list of n segments identified by 2xn endpoints through their pixel coordinates. These segments are then combined into the geometric groups represented in Figure 6.15. The simplest groups that are identified in the edge population are proximity segments and parallel segments. As can be observed from Figure 6.15, two segments compose a proximity group if they satisfy the proximity condition

$$d_{12} \leq d_{\max}\,, \tag{6.30}$$

where d_{12} is the shortest distance between endpoints and $d_{\max}$ is the threshold value. Similarly, two segments form a parallel group if

$$\left\| m_1 - m_2 \right\| \leq m_{\max}\,, \tag{6.31}$$

where m_1 and m_2 are the segment slopes defined as

$$m_i = \frac{v_{Bi} - v_{Ai}}{u_{Bi} - u_{Ai}}, \quad i = 1,2 \tag{6.32}$$

and $m_{\max}$ is the threshold slope value. Parallelograms and parts of parallelogram are more complex structures that are obtained from the combination of the basic groups. In fact, four proximity groups with two parallel segments form a parallelogram, whereas three proximity groups whose endpoints satisfy the condition

$$(\boldsymbol{\rho}_{A1} - \boldsymbol{\rho}_{A3}) \cdot (\boldsymbol{\rho}_{B2} - \boldsymbol{\rho}_{B3}) > 0 \tag{6.33}$$

compose a part of a parallelogram. In particular, the previous condition ensures that the endpoints A and B of the shape lie on the same side with respect to the shared segment (see Figure 6.15). As for the wireframe model of the target spacecraft, the same geometric groups are precomputed using the same conditions in Eqs. (6.30–6.33) using zero thresholds.

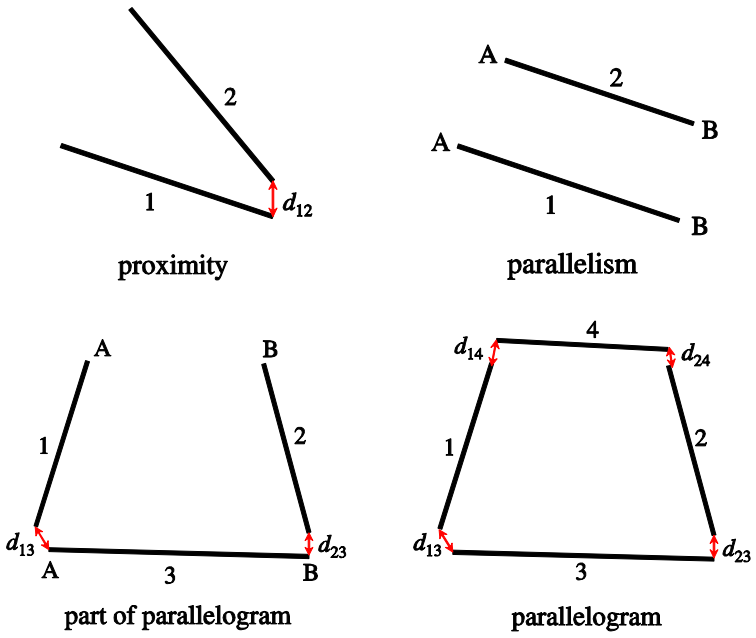

Figure 6.15: Geometric groups (adapted from [46]).

In order to solve the 3D-2D perspective projection equation for $\boldsymbol{R}_T^S$ and $\boldsymbol{p}^S$ without first guess, the EPnP algorithm is employed [118]. Among all the possible methods, EPnP ensures high computational speed and robustness to image noise and feature outliers [119]. The selected method requires n pairs of features points in the image and corresponding points in the 3D wireframe model. Four non-coplanar unknown control points $\boldsymbol{c}_1^S$, $\boldsymbol{c}_2^S$, $\boldsymbol{c}_3^S$ and

$\boldsymbol{c}_4^S$ of the rotated 3D model are introduced to solve the pose estimation problem. Each of the n visible points of the model (see Eq. (6.21)) can be expressed in the system S as combination of the control points as

$$\boldsymbol{\alpha}_i^S = \sum_{j=1}^{4} \gamma_{ij} \boldsymbol{c}_j^S \qquad i = 1,2,...,n \,. \tag{6.34}$$

Following the projection Eq. (6.22), the feature points $\boldsymbol{\rho}_i = (u_i, v_i)$ detected in the image coincide to the projected points $\boldsymbol{\alpha}_i^S$ of the model in the ideal case:

$$\begin{aligned} u_i &= \frac{\alpha_x^S}{\alpha_z^S} f_x + c_x \\ v_i &= \frac{\alpha_y^S}{\alpha_z^S} f_y + c_y \end{aligned} \qquad i = 1,2,...,n \,. \tag{6.35}$$

By substituting Eq. (6.34) into Eq. (6.35), we obtain the following linear equations for each pair of features

$$\begin{aligned} \sum_{j=1}^{4} f_x \gamma_{ij} c_{x_j}^S + \sum_{j=1}^{4} (c_x - u_i) \gamma_{ij} c_{z_j}^S = 0 \\ \sum_{j=1}^{4} f_y \gamma_{ij} c_{y_j}^S + \sum_{j=1}^{4} (c_y - v_i) \gamma_{ij} c_{z_j}^S = 0 \end{aligned} \qquad i = 1,2,...,n \,. \tag{6.36}$$

This linear system of $2 \times n$ equations is solved for the unknown control points $\boldsymbol{c}_1^S$, $\boldsymbol{c}_2^S$, $\boldsymbol{c}_3^S$ and $\boldsymbol{c}_4^S$. Since the unknown parameters are 12, at least 4 pairs of image features and corresponding points in the 3D wireframe model are required to solve the system. The current pose is then retrieved using Eqs. (6.21, 6.34).

The correspondences between features in the image and corresponding points of target 3D wireframe model are obtained using the geometric groups. Each geometry detected in the image is coupled with the analogous geometry of a simplified wireframe model of the target, which is showed in Figure 6.16. Since the pose initialization will be performed at the beginning of the proximity maneuver, it is possible to assume that the small surface features can be hardly spotted in the image and, therefore, are not recognizable by the Hough algorithm. In addition, it is also possible to tune the parameters of Hough in such a way to detect the longest edges only, i.e. the contour of the target spacecraft. It makes therefore sense to define a simplified wireframe model without surface features as in Figure 6.16. As regard to the correspondences between geometric groups, a parallelogram detected in the image is paired with every parallelogram of the model. Similarly, each proximity group detected in the image is paired with every proximity group of the

model. For each pair, all the possible combinations between segment endpoints in the image and 3D line endpoints of the model are considered. The obtained combinations of image features and model points are stored in the match matrix, which is used to solve the pose estimation problem using EP*n*P. The pose solutions are then ranked according to the output fit error in Eq. (6.35). The wireframe model of the target spacecraft is then reprojected onto the image using the best solutions from EP*n*P. The nearest neighbor search is executed to find the best fit between projected model points and image features, being the endpoints of the detected segments.

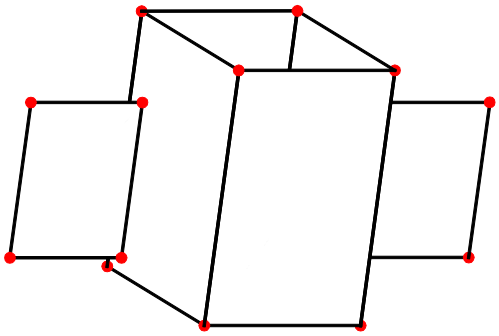

Figure 6.16: Simplified wireframe model of the target spacecraft used for monocular pose initialization.

The monocular vision navigation system is executed in closed-loop during the maneuver as showed in Figure 6.17. After the first image acquisition, the corrected image is employed for pose initialization using the proposed pose initialization routine described in this section. The first pose is fed into the EKF block. The estimated current state vector $\boldsymbol{x}_t$ is used as initial condition for state prediction, i.e. the numerical integration of the docking dynamics model. The estimated state is used in the state prediction block to propagate the dynamics model of the docking maneuver. The expected current orientation $\boldsymbol{q}^{\mathrm{T-S}}_{\mathrm{predicted}}$ and position $\boldsymbol{p}^{\mathrm{S}}_{\mathrm{predicted}}$ of the target spacecraft are then calculated using Eqs. (6.1, 6.5) and used as first guess for the pose estimation routine (Newton-Raphson).

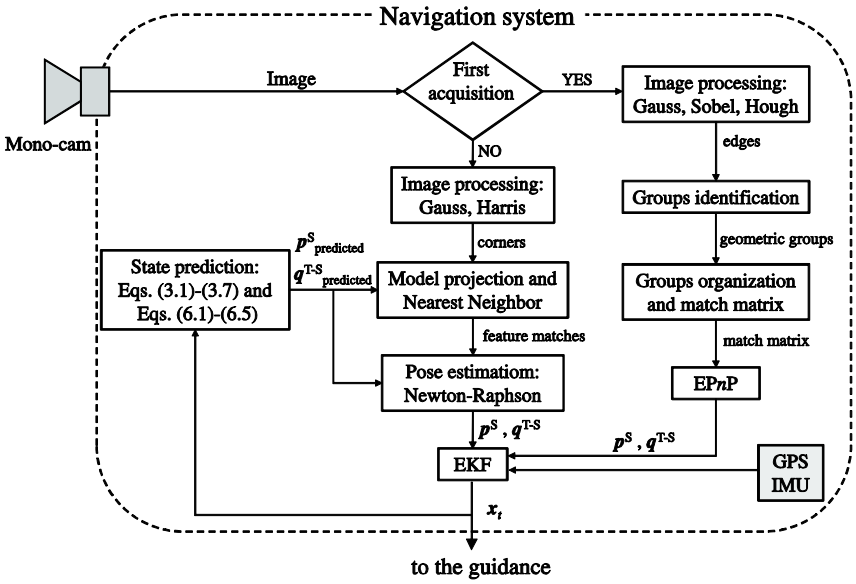

Figure 6.17: Block diagram of the mono camera-based navigation system.

6.6 Extended Kalman Filter

The EKF is a real-time near-optimal state estimator used for reducing measurement noise, estimating non-observable states and predicting future states. It employs the mathematical routine of the traditional Kalman filter applied to nonlinear systems [120]. The EKF has been used extensively for the last five decades in numerous technology applications, especially spacecraft GNC for satellite attitude and position estimation [21]. Previous space missions demonstrated that EKF is the basis of today's estimation algorithms, but requires numerous test simulations before the launch of the mission in order to tune the parameters and ensure rapid convergence.

In this thesis, a discrete-time EKF is developed since the dynamics model of proximity operations between chaser and target is time-continuous (see Section 3.1), whereas the measurements are time-discrete. The mathematical model of such system can be expressed in general form as

$$\dot{\boldsymbol{x}}(t) = f_x(\boldsymbol{x}(t),\boldsymbol{u}(t),t) + \boldsymbol{\varepsilon}(t) \qquad \boldsymbol{\varepsilon}(t) \sim \mathrm{N}[\boldsymbol{0},\boldsymbol{Q}(t)] \tag{6.37}$$

$$\boldsymbol{z}_k = h_z(\boldsymbol{x}(t_k)) + \boldsymbol{v}_k \qquad \boldsymbol{v}_k \sim \mathrm{N}[\boldsymbol{0},\boldsymbol{R}_k], \tag{6.38}$$

where $\boldsymbol{z}_k$ is the measurement vector, whereas $\boldsymbol{\varepsilon}(t)$ and $\boldsymbol{v}_k$ are independent zero-mean white noise having time-varying spectral density $\boldsymbol{Q}(t)$ and covariance $\boldsymbol{R}_k$ respectively.

The EKF is a recursive estimator that consists of two main steps: prediction and update. The conceptual scheme of the EKF is illustrated in Figure

6.18. In the first step, the filter predicts the covariance matrix $\boldsymbol{P}$ of the state estimate and the current state vector $\boldsymbol{x}$ using the dynamics model of the system and the previous state estimate as initial condition. In particular, the predicted state $\boldsymbol{x}_{k|k\text{-}1}$ relative to the instant of time t_k can be obtained from the previous estimation $\boldsymbol{x}_{k\text{-}1|k\text{-}1}$ using the following prediction equation in general form

$$\boldsymbol{x}_{k|k-1} = \boldsymbol{x}_{k-1|k-1} + \Delta t \dot{\boldsymbol{x}}_{k-1|k-1}, \tag{6.39}$$

with

$$\dot{\boldsymbol{x}}_{k-1|k-1} = f_x\left(\boldsymbol{x}_{k-1|k-1}, \boldsymbol{u}_{k-1}, t_{k-1}\right). \tag{6.40}$$

Similarly, the predicted covariance matrix $\boldsymbol{P}_{k|k\text{-}1}$ is calculated as

$$\boldsymbol{P}_{k|k-1} = \boldsymbol{P}_{k-1|k-1} + \Delta t \dot{\boldsymbol{P}}_{k-1|k-1}, \tag{6.41}$$

where

$$\dot{\boldsymbol{P}}_{k-1|k-1} = \boldsymbol{F}_{k-1}\boldsymbol{P}_{k-1|k-1} + \boldsymbol{P}_{k-1|k-1}\boldsymbol{F}_{k-1}^{\mathsf{T}} + \boldsymbol{Q}_{k-1}. \tag{6.42}$$

In the previous equation, $\boldsymbol{F}_{k\text{-}1}$ is the Jacobian matrix of the system model function f_x in Eq. (6.40) and it is given by

$$\boldsymbol{F}_{k-1} = \left.\frac{\partial f_x}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_{k-1|k-1}, \boldsymbol{u}_{k-1}}. \tag{6.43}$$

Note that Eqs. (6.39, 6.41) are linear and, therefore, valid for small propagation intervals Δt only. Long integration intervals require recursive application of these linear equations.

The predicted state estimate is corrected in the update step using the current measurement $\boldsymbol{z}_k$ taken at the particular instant of time t_k. The following Kalman gain $\boldsymbol{K}_k$ is first computed:

$$\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^{\mathsf{T}}\left(\boldsymbol{H}_k\boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^{\mathsf{T}} + \boldsymbol{R}_k\right)^{-1}, \tag{6.44}$$

where

$$\boldsymbol{H}_k = \left.\frac{\partial h_z}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_{k|k-1}} \tag{6.45}$$

is the Jacobian matrix of the measurement model in Eq. (6.38). The predicted state is updated using the acquired measurement $\boldsymbol{z}_k$. The final state estimate $\boldsymbol{x}_{k|k}$ is given by

$$\boldsymbol{x}_{k|k} = \boldsymbol{x}_{k|k-1} + \boldsymbol{K}_k\left(\boldsymbol{z}_k - h_z\left(\boldsymbol{x}_{k|k-1}\right)\right), \tag{6.46}$$

which corresponds to $\boldsymbol{x}_t$ of Figure 6.1. Finally, the covariance matrix $\boldsymbol{P}$ is also updated using the following equation:

$$\boldsymbol{P}_{k|k} = \left(\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{H}_k\right)\boldsymbol{P}_{k|k-1}, \tag{6.47}$$

where $\boldsymbol{I}$ the identity matrix.

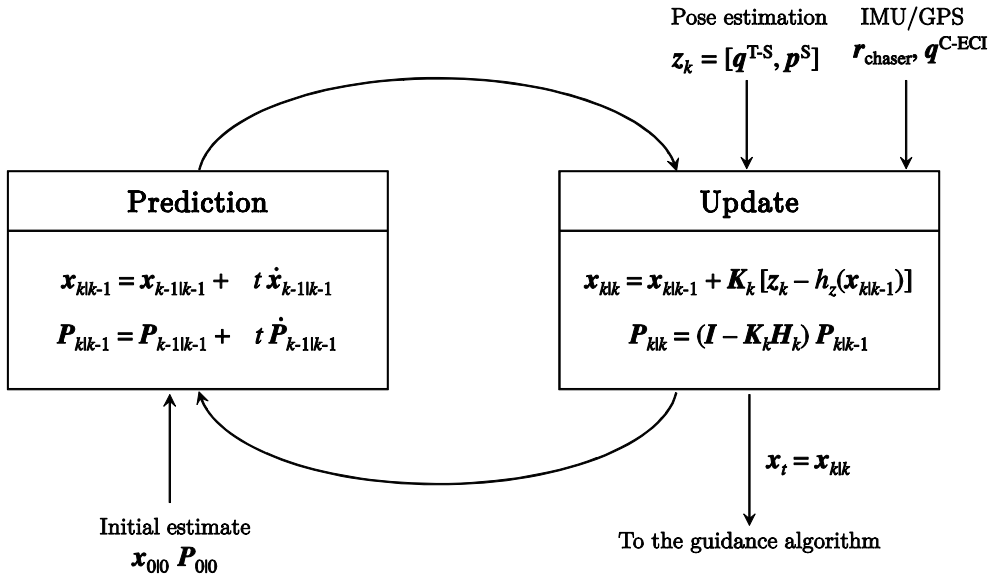

Figure 6.18: Conceptual scheme of the discrete-time EKF.

The discrete-time EKF is applied in the navigation scheme of Figure 6.1 in order to estimate the current state vector $\boldsymbol{x}$ (see Eq. (5.5)) of the proximity operations dynamics model using the output of the noncooperative pose estimation algorithm along with the data from the GPS and IMU. In order to simplify the procedure, two different EKFs have been developed. One filter estimates relative position and translational velocity of the chaser spacecraft in Hill's system in C-W dynamics, whereas the other provides attitude and angular velocity of the target vehicle. Attitude and angular velocity of the chaser spacecraft are assumed to be available from IMU and GPS.

As regards the estimation of the relative distance and velocity in Hill's system, the state vector

$$\boldsymbol{x}_{\mathrm{C-W}} = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} \end{bmatrix}^{\mathrm{T}}$$

is introduced. By following the general form, the C-W dynamics model in Eq. (3.1) can be written as

$$\dot{\boldsymbol{x}}_{\mathrm{C-W}}(t)=\boldsymbol{F}_{\mathrm{C-W}}\boldsymbol{x}_{\mathrm{C-W}}(t)+\begin{bmatrix}\boldsymbol{0}\\ \boldsymbol{u}^{\mathrm{H}}(t)\end{bmatrix}, \tag{6.48}$$

where

$$\boldsymbol{F}_{\mathrm{C-W}}=\begin{bmatrix}0 & 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 0 & 0 & 1\\ 3\Omega^2 & 0 & 0 & 0 & 2\Omega & 0\\ 0 & 0 & 0 & -2\Omega & 0 & 0\\ 0 & 0 & -\Omega^2 & 0 & 0 & 0\end{bmatrix}, \tag{6.49}$$

is the Jacobian matrix of the model as defined in Eq. (6.43). The measurement model is obtained from Eq. (6.1) and it is given by

$$\boldsymbol{p}^{\mathrm{S}}=-\left(\boldsymbol{R}_{\mathrm{C}}^{\mathrm{S}}\boldsymbol{s}^{\mathrm{C}}+\boldsymbol{R}_{\mathrm{C}}^{\mathrm{S}}\boldsymbol{R}_{\mathrm{ECI}}^{\mathrm{C}}\boldsymbol{R}_{\mathrm{H}}^{\mathrm{ECI}}\begin{bmatrix}x\\ y\\ z\end{bmatrix}\right), \tag{6.50}$$

as the measurement $\boldsymbol{z}_k$ for the EKF corresponds to the solution $\boldsymbol{p}^{\mathrm{S}}$ from the pose estimation algorithm. In the previous equation, $\boldsymbol{R}_{\mathrm{ECI}}^{\mathrm{C}}$ is provided from the IMU, whereas $\boldsymbol{R}_{\mathrm{H}}^{\mathrm{ECI}}$ is calculated using Eqs. (6.2–6.4) applied to the predicted state. Finally, the Jacobian of the measurement model is

$$\boldsymbol{H}_{\mathrm{C-W}}=\begin{bmatrix}-\boldsymbol{R}_{\mathrm{H}}^{\mathrm{S}} & \boldsymbol{I}_{3\mathrm{x}3}\end{bmatrix}. \tag{6.51}$$

Attitude and angular velocity of the target spacecraft are estimated using a second EKF. For this application, the following state vector

$$\boldsymbol{x}_{\mathrm{attitude\_T}}=\begin{bmatrix}\boldsymbol{q} & \boldsymbol{\omega}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}$$

is defined. In order to simplify the filter, the quaternion $\boldsymbol{q}$ of the attitude of the target principal system T with respect to ECI is estimated rather than the attitude $\boldsymbol{q}^{\mathrm{T}}$ of T with respect to the Hill's system. The quaternion $\boldsymbol{q}^{\mathrm{T}}$ required by the guidance algorithm is obtained by combining the estimated $\boldsymbol{q}$ with $\boldsymbol{q}^{\mathrm{H\text{-}ECI}}$, which is calculated from the output of the EKF related to the C-W dynamics using Eqs. (6.2–6.4). The rotational dynamics model of the target spacecraft (assumed in torque-free motion) is given by

$$\dot{\boldsymbol{q}}(t)=\frac{1}{2}\boldsymbol{\Omega}\left(\boldsymbol{\omega}^{\mathrm{T}}(t)\right)\boldsymbol{q}(t) \tag{6.52}$$

along with Eq. (3.3). The matrix $\boldsymbol{\Omega}(\boldsymbol{\omega}^{\mathrm{T}})$ in Eq. (6.52) is defined as

$$\boldsymbol{\Omega}\left(\boldsymbol{\omega}^{\mathrm{T}}\right)=\begin{bmatrix} 0 & \omega_z^{\mathrm{T}} & -\omega_y^{\mathrm{T}} & \omega_x^{\mathrm{T}} \\ -\omega_z^{\mathrm{T}} & 0 & \omega_x^{\mathrm{T}} & \omega_y^{\mathrm{T}} \\ \omega_y^{\mathrm{T}} & -\omega_x^{\mathrm{T}} & 0 & \omega_z^{\mathrm{T}} \\ -\omega_x^{\mathrm{T}} & -\omega_y^{\mathrm{T}} & -\omega_z^{\mathrm{T}} & 0 \end{bmatrix}. \tag{6.53}$$

The Jacobian matrix $\boldsymbol{F}_{\text{attitude\_T}}$ of the model is given by

$$\boldsymbol{F}_{\text{attitude\_T}}=\begin{bmatrix} 0.5\boldsymbol{\Omega}\left(\boldsymbol{\omega}^{\mathrm{T}}\right) & -0.5\boldsymbol{S}(\boldsymbol{q}) \\ \boldsymbol{0}_{3\mathrm{x}4} & \boldsymbol{L}\left(\boldsymbol{\omega}^{\mathrm{T}}\right) \end{bmatrix}, \tag{6.54}$$

where

$$\boldsymbol{S}(\boldsymbol{q})=\begin{bmatrix} q_4 & -q_3 & q_2 \\ q_3 & q_4 & -q_1 \\ -q_2 & q_1 & q_4 \\ -q_1 & -q_2 & -q_3 \end{bmatrix} \tag{6.55}$$

$$\boldsymbol{L}\left(\boldsymbol{\omega}^{\mathrm{T}}\right)=\begin{bmatrix} 0 & \dfrac{(I_{yy}^{\mathrm{T}}-I_{zz}^{\mathrm{T}})}{I_{xx}^{\mathrm{T}}}\omega_z^{\mathrm{T}} & \dfrac{(I_{yy}^{\mathrm{T}}-I_{zz}^{\mathrm{T}})}{I_{xx}^{\mathrm{T}}}\omega_y^{\mathrm{T}} \\ \dfrac{(I_{zz}^{\mathrm{T}}-I_{xx}^{\mathrm{T}})}{I_{yy}^{\mathrm{T}}}\omega_z^{\mathrm{T}} & 0 & \dfrac{(I_{zz}^{\mathrm{T}}-I_{xx})}{I_{yy}^{\mathrm{T}}}\omega_x^{\mathrm{T}} \\ \dfrac{(I_{xx}^{\mathrm{T}}-I_{yy}^{\mathrm{T}})}{I_{zz}^{\mathrm{T}}}\omega_y^{\mathrm{T}} & \dfrac{(I_{xx}^{\mathrm{T}}-I_{yy}^{\mathrm{T}})}{I_{zz}^{\mathrm{T}}}\omega_x^{\mathrm{T}} & 0 \end{bmatrix}. \tag{6.56}$$

The measurement model is obtained from Eq. (6.5) and it is given by

$$\boldsymbol{q}^{\mathrm{T-S}}=\boldsymbol{q}^{\mathrm{ECI\text{-}S}}\otimes\boldsymbol{q}, \tag{6.57}$$

as the measurement $\boldsymbol{z}_k$ for this EKF corresponds to the solution $\boldsymbol{q}^{\mathrm{T\text{-}S}}$ from the pose estimation algorithm. In the previous equation, $\boldsymbol{q}^{\mathrm{ECI\text{-}S}}$ is the quaternion of the rotation matrix $\boldsymbol{R}_{\mathrm{S}}^{\mathrm{ECI}}=\boldsymbol{R}_{\mathrm{C}}^{\mathrm{ECI}}\boldsymbol{R}_{\mathrm{S}}^{\mathrm{C}}$ and it is provided by the IMU. Finally, the Jacobian of the measurement model is

$$\boldsymbol{H}_{\text{attitude\_T}} = \begin{bmatrix} q_4^{\text{ECI-S}} & -q_3^{\text{ECI-S}} & q_2^{\text{ECI-S}} & q_1^{\text{ECI-S}} & 0 & 0 & 0 \\ q_3^{\text{ECI-S}} & q_4^{\text{ECI-S}} & -q_1^{\text{ECI-S}} & q_2^{\text{ECI-S}} & 0 & 0 & 0 \\ -q_2^{\text{ECI-S}} & q_1^{\text{ECI-S}} & q_4^{\text{ECI-S}} & q_3^{\text{ECI-S}} & 0 & 0 & 0 \\ -q_1^{\text{ECI-S}} & -q_2^{\text{ECI-S}} & -q_3^{\text{ECI-S}} & q_4^{\text{ECI-S}} & 0 & 0 & 0 \end{bmatrix}. \tag{6.58}$$

6.7 Summary

This chapter described the navigation algorithms for onboard pose estimation of a noncooperative space target during proximity operation maneuvers. The proposed methodology consists of three software modules: data processing of the data acquired by the rendezvous sensor, noncooperative pose estimation algorithm and time-discrete EKF. The software is initialized by the pose initialization module, which estimates the target pose using the first sensor data without prior state information. Detailed software implementation for noncooperative pose estimation using Lidar, stereo camera and mono camera was discussed.

As regards the Lidar-based software, the point cloud of the scene is first reconstructed from depth measurement using the pinhole camera model. The ICP algorithm is then applied to estimate the current pose of the target by comparing the acquired point cloud with a reference point cloud of the same vehicle stored onboard. Finally, the EKF provides the current position, attitude and velocities of the spacecraft with respect to the Hill's coordinate system. The same approach is used with stereo camera, since the point cloud of the scene can be obtained from the disparity of the stereo image. Concerning the monocular navigation software, the image processing extracts the features of the target spacecraft from the monocular image. The wireframe model of the target is projected onto the image and a nearest neighbor search determines the correspondences between image and model features. A Newton-Raphson technique solves the 3D-2D perspective projection equation using these correspondences. The performance of the proposed navigation algorithms will be discussed in Chapter 7, where the experimental validation campaign using the proximity operations testbed and data from real sensors is presented.

7 Test Cases in RACOON Laboratory

7.1 Proximity Operations Testbed

The experimental setup used for the validation campaign is the RACOON laboratory, which provides a real-time simulation of proximity operations between a controlled chaser spacecraft and a noncooperative uncontrolled target vehicle in full DoF under realistic lighting conditions [121]. As illustrated in Figure 7.1 and Figure 7.2, a target mockup is mounted on a small platform that runs on a C-shape rail structure. This curved structure can rotate about its vertical axis, whereas step motors inside the model allow the 360 deg rotation about the holding arm (see lateral view in Figure 7.2). Notably the system is designed in such a way that the geometric center of the target remains always at the center of the C-shape. As a result, the target satellite can rotate about its principal axes and, therefore, any attitude configuration and rotational motion can be simulated. The chaser mockup consists of a rendezvous sensor suite and an onboard computer mounted on a moving platform. The structure that holds this platform allows simulating the chaser rotational motion and the relative translational motion between spacecraft centers of mass. A dedicated orbital simulator drives the system in real-time. The dynamics model of each spacecraft consists of the two-body problem in Cowell's formulation as in Eq. (5.41) for the translational motion along with Eqs. (3.4, 5.42) for the rotational motion. The estimated position accuracy is about 5 mm for each translational axis, whereas the orientation accuracy is 0.5 deg for each rotational axis. As regards the illumination conditions, a compact fluorescent lamp is located near the target to simulate the albedo, whereas a Xenon-arc lamp is used as Sun simulator (see Figure 7.1 and Figure 7.2). The laboratory simulates a proximity operation maneuver with a scale 1:4.

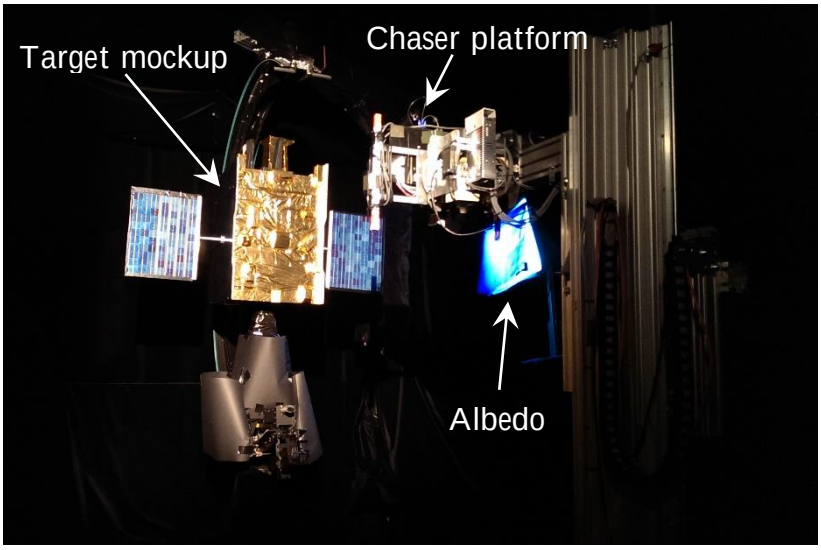

Figure 7.1: Testbed for real-time proximity operation simulations (RACOON laboratory) seen from the Sun simulator perspective. Notably this is the initial configuration of the considered test cases.

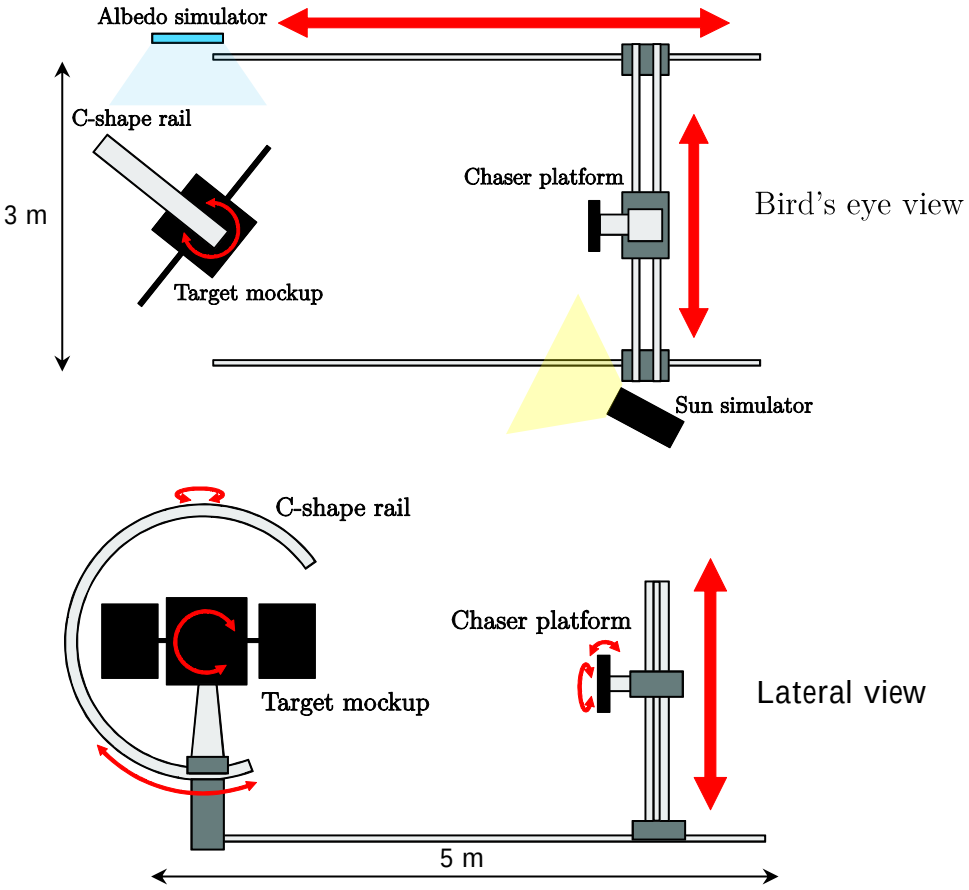

Figure 7.2: Schematic representation of the proximity operation testbed.

The simulated chaser spacecraft consists of a rendezvous sensor suite composed by the Microsoft Kinect v2 sensor and a Bumblebee2 stereo camera connected to an Intel i3 onboard computer as in Figure 7.3. In particular, the IR sensor of the Kinect is used to acquire the depth image of the scene using the time-of-flight principle as discussed in Section 6.3. The left camera of the Bumblebee2 provides the monocular images. The coordinate systems $\mathcal{S}$ are defined on the Kinect's IR sensor and on the Bumblebee's left camera as showed in Figure 7.3. The specifications of the sensors are summarized in Table 7.1.

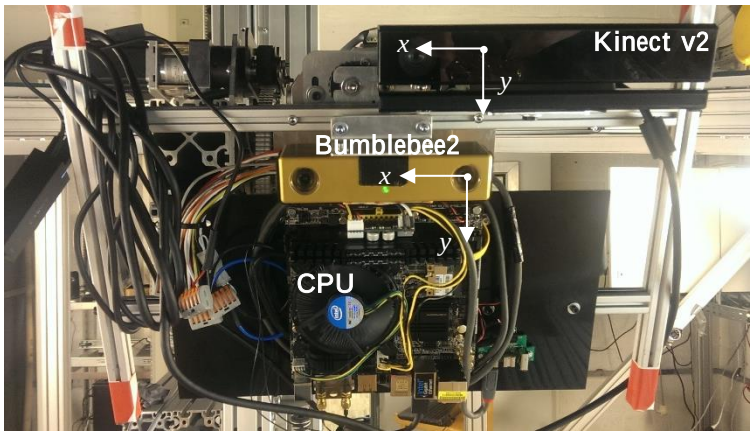

Figure 7.3: Rendezvous sensor suite of the simulated chaser spacecraft and sensor coordinate systems.

Table 7.1: Sensor specifications.

	Kinect v2 (IR sensor)	Bumblebee2	
Resolution	512 x 424	648 x 488	px
FOV (h x v)	70.6 x 60.0	65.0 x 65.0	deg
Focal length f_x	364.7 ± 1.6	523.5 ± 1.6	px
Focal length f_y	366.1 ± 1.5	523.5 ± 1.6	px
Principal point c_x	255.8 ± 2.1	321.0 ± 1.6	px
Principal point c_y	206.5 ± 2.7	241.9 ± 1.6	px
Baseline	-	120 ± 1	mm
Frame rate	30	48	FPS
Operating range	0.5 to 4.5	-	m
Mass	970	342	g
Dimension (w x d x h)	249 x 66 x 67	157 x 36 x 47	mm
Power consumption	15W at 12V	2.5W at 12V	-
Connection type	USB 3.0	FireWire 1394a	-

7.2 Test Cases

Three different test cases were simulated in RACOON laboratory in order to evaluate the performance of the navigation algorithms proposed in Chapter 6. The considered maneuvers are listed in Table 7.2 are V-bar approaches starting from the same initial distance and translational velocity defined in the Hill's coordinate system. The scenarios differ in the angular velocity of the target object. In scenario 1, the chaser spacecraft approaches a nonrotating space vehicle. In scenarios 2 and 3, the target rotates in flat spin mode at 1.6 deg/s and 3.2 deg/s, respectively. The principal axes of the target are defined as in Figure 6.5.

Table 7.2: Initial states and spacecraft specifications for Scenarios 1-3 of the validation campaign.

Parameter	Value	Unit of measure
x_0, y_0, z_0	0, -4.5, 0	m
$\dot{x}_0,\ \dot{y}_0,\ \dot{z}_0$	-0.5, 3, 0	cm/s
$\boldsymbol{\omega}_0^{\mathrm{C}}$	[0, 0, 0]	deg/s
$\boldsymbol{q}_0^{\mathrm{C}}$	[0, 0, -0.71, 0.71]	-
$\boldsymbol{\omega}_0^{\mathrm{T}}$ (Scenario 1)	[0, 0, 0]	deg/s
$\boldsymbol{\omega}_0^{\mathrm{T}}$ (Scenario 2)	[1.6, 0, 0]	deg/s
$\boldsymbol{\omega}_0^{\mathrm{T}}$ (Scenario 3)	[3.2, 0, 0]	deg/s
$\boldsymbol{q}_0^{\mathrm{T}}$	[-0.71, 0, 0.71, 0]	-
Orbit altitude	500	km
Eccentricity	0	-
$I_{xx}^{\mathrm{C}}, I_{yy}^{\mathrm{C}}, I_{zz}^{\mathrm{C}}$	608, 587, 373	kg m^2
mass, chaser	600	kg
$I_{xx}^{\mathrm{T}}, I_{yy}^{\mathrm{T}}, I_{zz}^{\mathrm{T}}$	129 180, 16 980, 124 800	kg m^2
mass, target	7828	kg

During these maneuvers, the rendezvous sensors acquire the images at a rate of 0.1 Hz. However, the Kinect resolution was set to 256 x 212 in order to reduce the number of acquired point and, therefore, to speed-up the pose estimation procedure [96]. After data acquisition, a dedicated software in the onboard computer removes the laboratory structure from the acquired data. The reference point cloud of the target spacecraft geometry is defined by 4000 points equally sampled from the model depicted in Figure 6.5. The size of the body of the target is 50x70x85 cm. The wireframe model used for monocular navigation is showed in Figure 6.11 and it is composed by 106 feature points. The simplified wireframe model for monocular pose initialization is composed by 6 parallelograms, 24 parts of parallelograms and 24 proximity segments and it is illustrated in Figure 6.16. Tolerances for stopping criteria of ICP and Newton-Raphson are 1e-7 and 1e-13 respectively. The navigation algorithms were executed offline using the data collected by the sensors. The software was implemented in Matlab environment on an

Intel i7 3.4 GHz laptop with 8 Gb of RAM. The reference solution for the state vector $\boldsymbol{x}$ (see Eq. (5.5)) was obtained from the orbital simulator of RACOON laboratory. The computational time of the algorithms was evaluated using Matlab's *timeit* function.

7.3 Pose Initialization

After the first image acquisition, the pose initialization algorithm is executed in order to estimate attitude and position of the target spacecraft without prior knowledge of its state. The data used for pose initialization are showed in Figure 7.4. Notably, the Kinect v2 sensor acquired 1793 points. The stereo image processing, whose steps are illustrated in Figure 7.5, provided 1883 points. Finally, the monocular image processing detected one parallelogram and one additional proximity group as illustrated in Figure 7.6.

Table 7.3 summarizes the performance of the ICP-based pose initialization algorithm. Several angular steps were considered for attitude discretization. In all the considered cases, the pose of the target spacecraft is estimated with errors up to 5.2 cm and 3.8 deg obtained with the point cloud from the stereo image processing. Furthermore, the pose initialization using the depth data from the Kinect sensor is always more precise than the initialization based on stereo images. In fact, as can be observed from Figure 7.4 and Figure 7.5, the point cloud obtained from stereo images is noisier than the one captured by the Kinect. This noise is indeed introduced by the calculation of the disparity map, which suffers the high reflectivity of the MLI material. As a matter of fact, the SAD method requires that the intensity of a feature is the same in left and right images. Finally, as can be observed from Table 7.3, the most precise results are obtained with refined attitude discretization, but at the cost of higher computational time, since more attitude configuration must be tested. The performances of the pose initialization algorithm for monocular vision navigation are summarized in Table 7.4. Also in this case, the algorithm is capable to estimate position and attitude of the target with errors of 4.6 cm and 2.3 deg. The high computational time is due to the EP*n*P, since several combinations of one parallelogram and one proximity groups must be tested. Solutions are illustrated in Figure 7.7.

Table 7.3: Performance of the ICP-based template matching algorithm applied to the images from Kinect and Bumblebee2 in Figure 7.4.

Angular step [deg]	Position error [cm]		Attitude error [deg]		CPU time [s]	
	Kinect	Stereo	Kinect	Stereo	Kinect	Stereo
60	1.4	5.2	2.7	3.8	51.2	55.0
45	1.2	4.8	1.4	3.5	142.7	103.5
40	1.2	4.8	1.4	3.5	293.0	179.8
36	1.2	4.7	1.1	3.4	289.8	275.0
30	1.1	4.7	1.0	3.4	422.0	429.1

Table 7.4: Performance of the monocular pose initialization algorithm applied to the left image in Figure 7.4.

Parameter	Value	Unit of measure
Position error	4.6	cm
Attitude error	2.3	deg
CPU image processing	2e-2	s
CPU EP*n*P	114.1	s
CPU reprojection	2.1	s
CPU time total	116.2	s

Figure 7.4: Acquired data at the beginning of Scenario 1: depth image acquired by the Kinect v2 (left) and stereo image acquired by the Bumblebee2 camera (center, right).

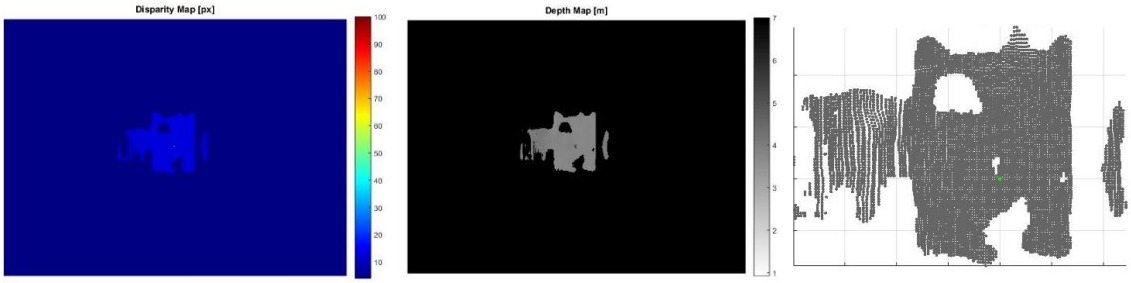

Figure 7.5: Steps of stereo image processing for pose initialization: disparity map (left), depth map (center) and final point cloud after disparity (right).

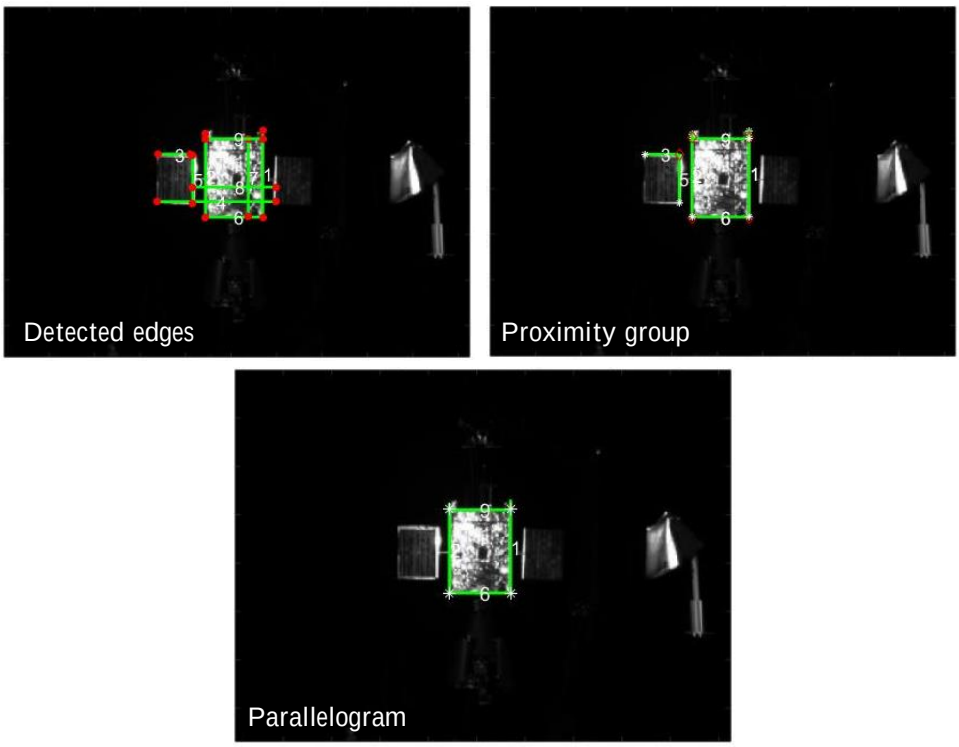

Figure 7.6: Image processing for monocular pose initialization: detected edges (top left), proximity groups (top right) and parallelogram (bottom).

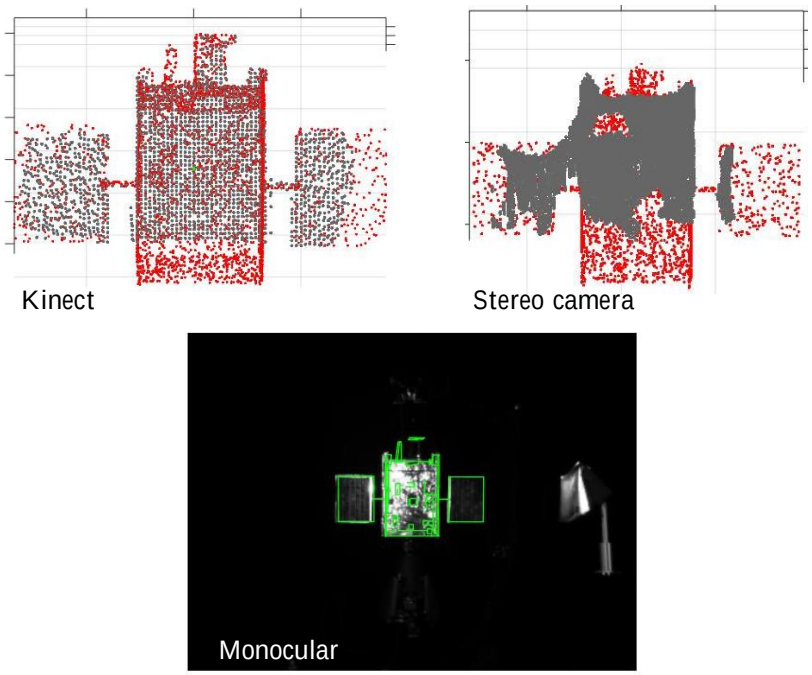

Figure 7.7: Solutions from pose initialization algorithms: ICP-based using data from Kinect (top left), ICP-based using data from stereo camera (top right) and monocular pose initialization (bottom).

The pose initialization algorithms have been also tested using the data showed in Figure 7.8 to further verify robustness and computational time. In particular, these images correspond to the second data acquisition during Scenario 3. Table 7.5 and Table 7.6 summarize the performance of the algorithms. The considerations made for the previous case in Figure 7.7 hold. As regards to the monocular pose estimation, the match matrix is composed by a part of a parallelogram (solar panel, see Figure 7.9) and parallel lines. Notably, the target pose was obtained using the part of the parallelogram along with segment 7 in Figure 7.9. Due to the fact that only a part of a parallelogram was detected, the computational time required in this case is higher than the previous test due to the larger size of the match matrix. The error in the estimated pose is larger than the previous test (see Table 7.6 with Table 7.4). In fact, the position of the endpoints of the detected segments does not always correspond to the true spacecraft corner due to reflections, smoothing effects and parameters of the Hough transform.

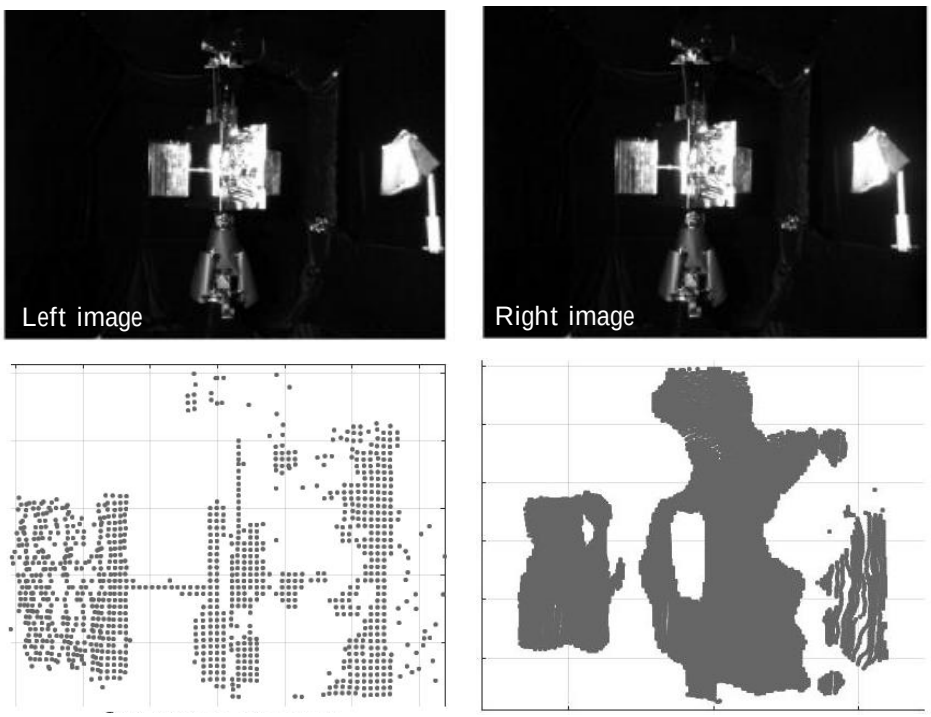

Figure 7.8: Second acquired data during Scenario 3: stereo image (top left and right), point cloud measured by Kinect (bottom left) and depth image reconstructed by image stereo processing algorithm (bottom right).

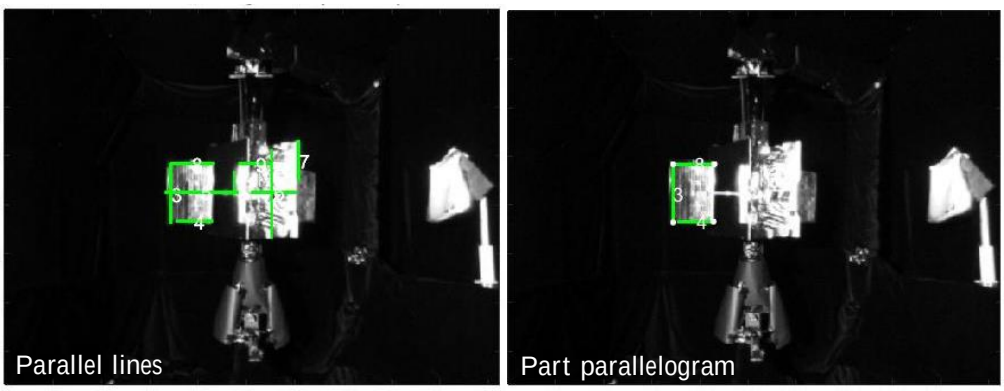

Figure 7.9: Detected geometric groups for monocular pose initialization: parallel lines and a part of a parallelogram.

Table 7.5 Performance of the ICP-based template matching algorithm applied to data from Kinect and Bumblebee2 in Figure 7.8

Angular step [deg]	Position error [cm]		Attitude error [deg]		CPU time [s]	
	Kinect	Stereo	Kinect	Stereo	Kinect	Stereo
60	6.9	9.3	4.8	6.3	42.7	66.5
45	6.7	9.1	4.8	6.2	87.9	134.5
40	5.6	6.9	4.2	4.6	159.2	249.0
36	5.0	6.9	4.1	4.6	239.1	386.4
30	4.1	6.9	3.7	4.6	380.5	574.6

Table 7.6: Performance of the monocular pose initialization algorithm applied to the left image in Figure 7.8. Detected geometric groups used for pose estimation are showed in Figure 7.9.

Parameter	Value	Unit of measure
Position error	10.7	cm
Attitude error	6.4	deg
CPU image processing	2e-2	s
CPU EP*n*P	221.2	s
CPU reprojection	2.3	s
CPU time total	223.5	s

7.4 Performance during Proximity Operations

This section discusses the performance of the proposed navigation algorithms during the considered three proximity operation test cases (see Table 7.2). These maneuvers were simulated in RACOON laboratory. The Sun

simulator illuminated the target in order to enable data acquisition with the Bumblebee2 camera. The reference translational trajectory and velocity of the chaser spacecraft center of mass with respect to the Hill's coordinate system that drive the laboratory are illustrated in Figure 7.10. The reference attitude trajectories of the target's principal coordinate system T with respect to the Hill's system are showed in Figure 7.11 for the three scenarios. For each simulated test case, the rendezvous sensors acquired data at a rate of 0.1 Hz. In order to provide a fair comparison among navigation algorithms, data were collected at the same instant of time exactly.

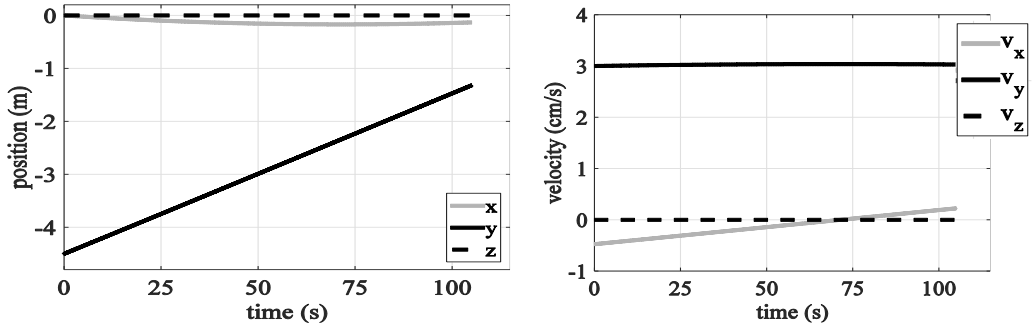

Figure 7.10: Position and velocity of the chaser center of mass with respect to the Hill's coordinate system during test Scenarios 1-3. Profiles obtained from the RACOON laboratory's orbital propagator.

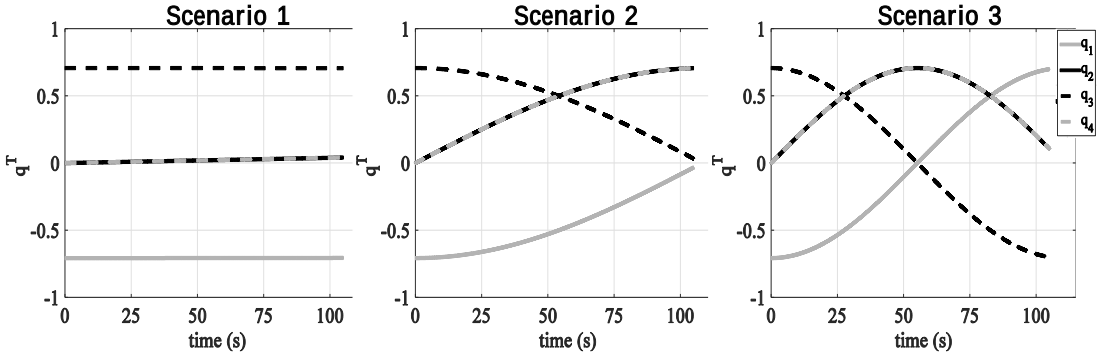

Figure 7.11: Attitude of the target spacecraft (coordinate system T) with respect to the Hill's coordinate system during test Scenarios 1-3. Profiles obtained from the RACOON laboratory's orbital propagator.

Figure 7.12 shows the final errors on the state vector (input to the guidance algorithm) estimated by the EKF. Errors on the position of the chaser center of mass with respect to the Hill frame are always limited to 3.7 cm. The stereo camera sensor offers the poorest performance, with an average

error of 2.9 cm. As already discussed for pose initialization, the disparity calculation algorithm suffers the high reflectivity of MLI, which causes different feature intensities in left and right images and, therefore, introduces large noise in the disparity map. Moreover, the calculation of the disparity is performed inside a small pixel box and the resulting disparity is influenced by all the pixels inside this area. This causes an averaging effect and, therefore, reduces the accuracy of the depth reconstruction procedure. As regards the performance of the Kinect v2-based navigation system, the estimation errors increase at a distance of 1.3 m from the target center of mass, which corresponds to 1.0 m to the target surface. In fact, this distance is close to the minimum measurable depth of 0.5 m and the quality of measurements starts to degrade. Moreover, the high reflectivity and irregular distribution of the MLI surface cause the diffuse reflection of the IR beam as illustrated in Figure 7.13. At close distances to the target surface, these diffuse rays are detected by the IR sensor and cause further noise in the acquired data. Figure 7.14 shows a comparison between the point clouds acquired at 4.0 m and at 1.3 m. Notably the latter is affected by relevant noise due to the close distance to the target surface. The best performance in terms of estimation errors are ensured by the monocular navigation system, having average estimation errors of 1.4 cm and 1.2 deg. However, at shorter distances the performance starts to worsen also in this case. At this distance, in fact, only the target body's surface is visible and the high reflectivity of MLI introduces false features that are detected by the Harris algorithm. This also requires a new tuning of the algorithm's parameters. Clear features such as corners of antennae and solar panels are visible at long distances only.

The errors in estimated target angular velocity and relative translational velocity are also illustrated in Figure 7.12. The maximum achieved errors are 0.3 cm/s and 0.1 deg/s. In particular, these values are lower than the maximum allowable noise levels obtained in Section 5.7. The average norm error on estimated translational velocity $\dot{x}_0$, $\dot{y}_0$, $\dot{z}_0$ is 0.1 m/s obtained by using the Kinect v2 IR sensor and by the mono camera, whereas the average error in target angular velocity is 0.05 deg/s obtained with mono camera and stereo camera.

Figure 7.15 illustrates the computational time required by the navigation system to calculate the complete state vector during Scenario 1. The monocular navigation system ensures the shortest computational time at every distance, with an average of 185 ms. notably this value is below the frequency of 0.5 s with which the trajectory is reshaped in the numerical experiments in Section 5.7. The most demanding part of the algorithm is the Matlab's z-buffer algorithm in the image processing procedure, which requires about 130 ms. The time required by the Harris algorithm is 50 ms, whereas the Newton-Raphson method converges in about 5 ms using 6 feature correspondences. The average number of iterations required by the Newton-Raphson routine to provide a solution is 7. The ICP routine is generally more computational demanding than Newton-Raphson, due to the

higher number of points employed and to the nearest neighbor search performed at each iteration. The shortest computational time required by ICP is 110 ms using 1762 measured points. Moreover, the shorter the distance to the target surface, the higher the number of measured points. This increases the computational time required by ICP up to 3 s (see Figure 7.17). The pose estimation procedure applied to the date from the stereo camera requires the highest total amount of time due to the disparity calculation, which takes at least 0.5 s.

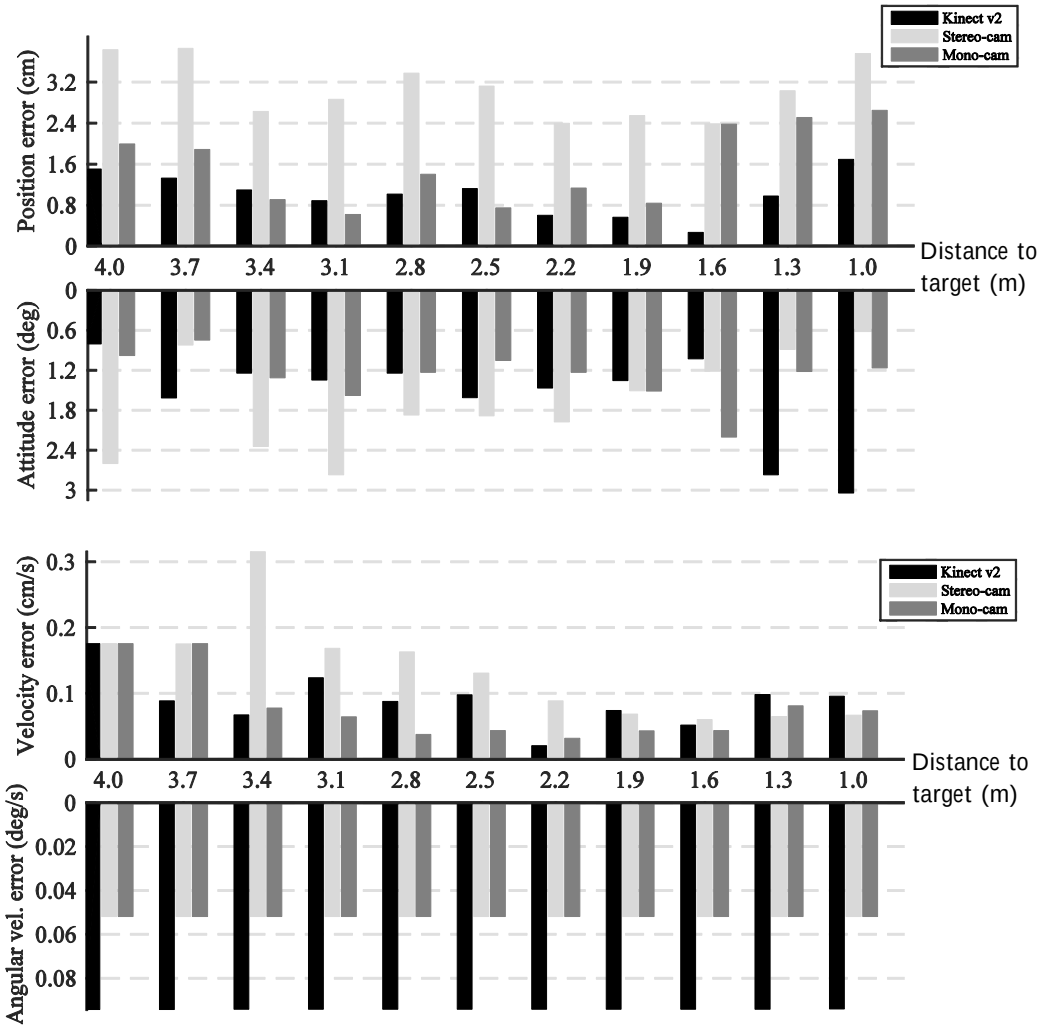

Figure 7.12: Performance comparison of the navigation algorithms for Scenario 1: norm errors on estimated position (x,y,z) of the chaser in H system (top), attitude of T system with respect to H system (top), relative translational velocity in H system (bottom) and target angular velocity (bottom). The distance to target is intended as distance between sensor platform and target center of mass.

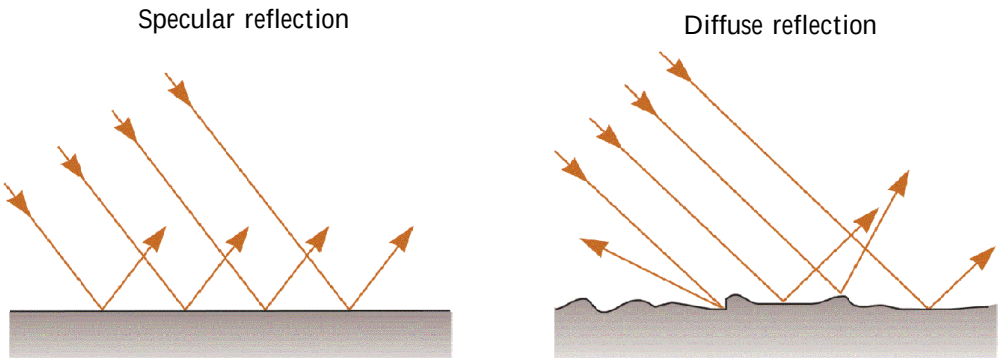

Figure 7.13: Specular reflection (left) and diffuse reflection (right) of IR rays.

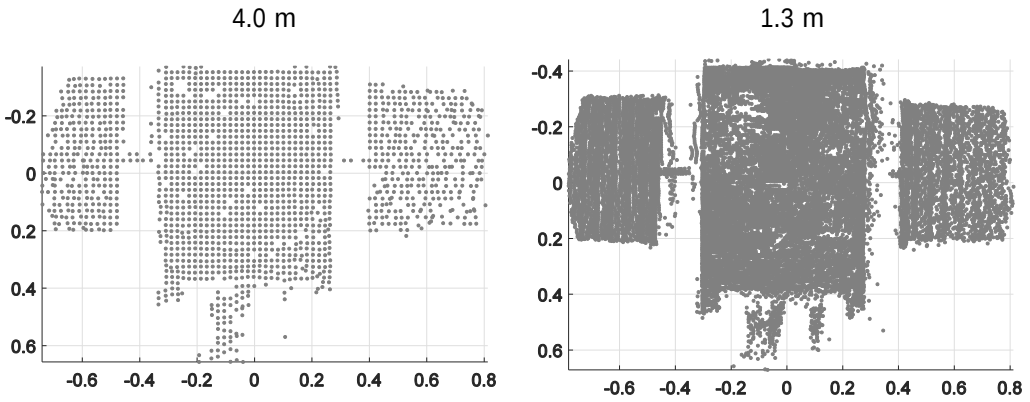

Figure 7.14: Point cloud acquired by the Microsoft Kinect v2 IR sensor at of 4.0 m (left) and 1.3 m (right) from the target center of mass during Scenario 1.

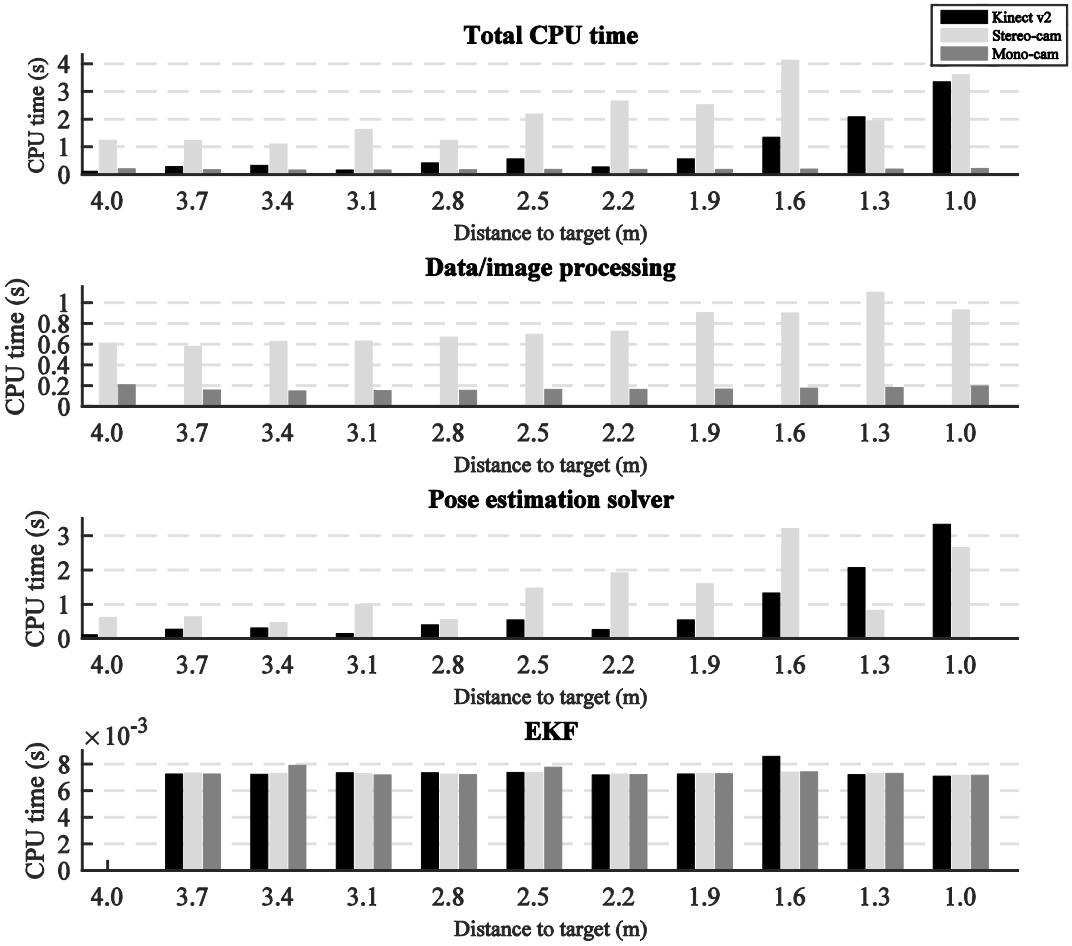

Figure 7.15: Computational time required by the navigation algorithms for Scenario 1. The distance to target is intended as distance between sensor platform and target center of mass.

The performances of the navigation systems during scenarios 2 and 3 are illustrated in Figure 7.16 and Figure 7.17 respectively. In these cases, the chaser approaches a rotating target (see Table 7.2). Interestingly, the monocular navigation system ensures the most accurate estimations, with average estimation errors of 1.7 cm, 1.8 deg, 0.11 cm/s and 0.05 deg/s. Peaks of errors appear for Kinect v2-based and stereo vision-based navigation systems. As can be observed in Figure 7.18, the target spacecraft's solar panel is reflected on the body's surface in certain attitude configurations. The algorithm for disparity map reconstruction identifies some features also in the reflected solar panel. As a consequence, the reflected feature appears in the disparity map and therefore in the reconstructed depth scene. As regards the data acquired with Microsoft Kinect v2, the IR beam is first deviated by the solar panel, then it is reflected by the target spacecraft's body surface and finally captured by the IR sensor. The time delay caused by this double reflection produces a reflected solar panel also in the resulting point cloud

(Figure 7.19). This imaginary feature produces significant errors in the estimated state. On the contrary, the monocular pose estimation is robust to this effect. As can be observed in Figure 7.20, the Harris algorithm detects some corners also in the reflected solar panel. However, none of the projected features of the wireframe model corresponds to the corners of the reflected panel. As a consequence, the nearest neighbor search couples model features with true corner only. This ensures robustness to reflected features.

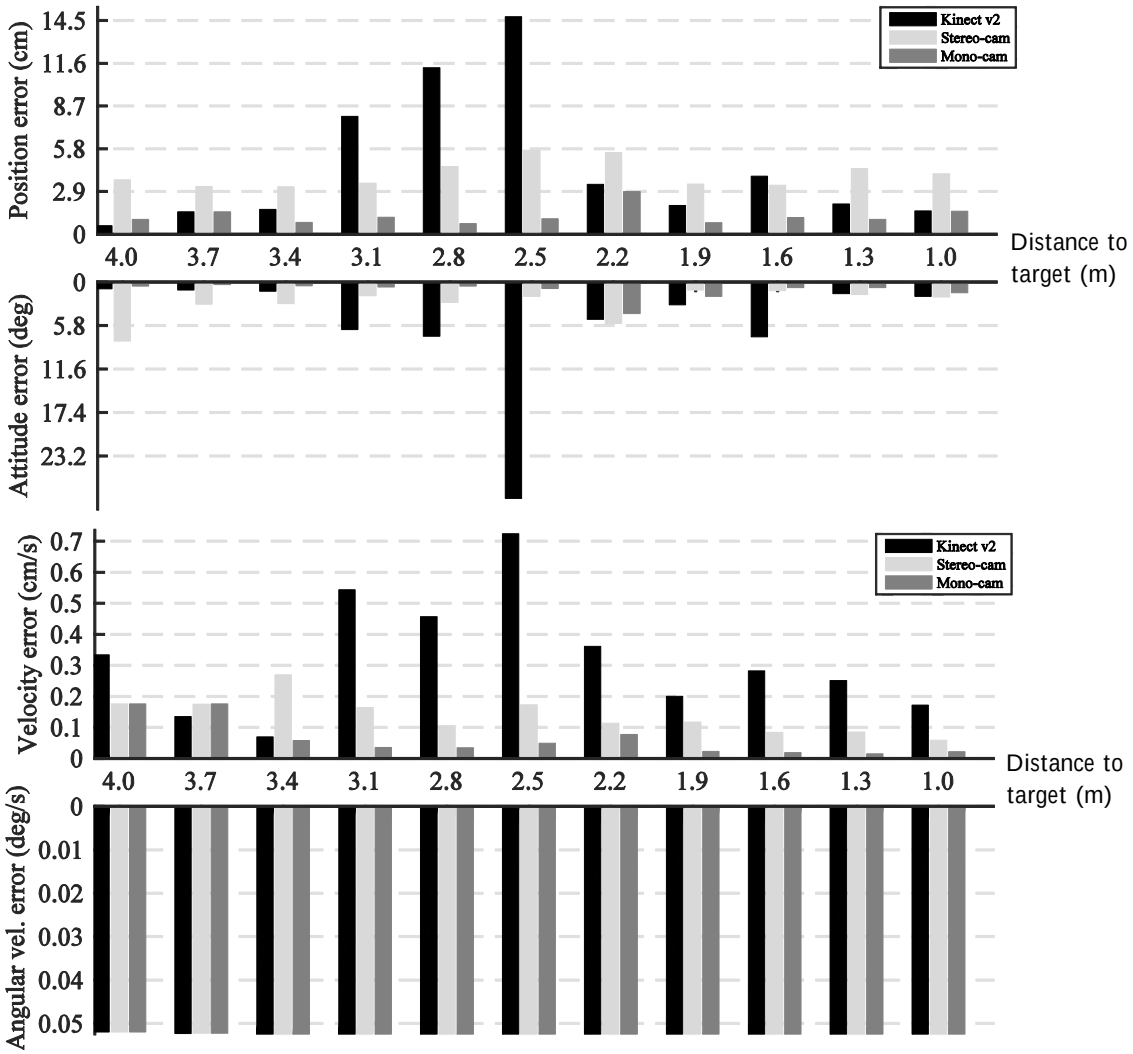

Figure 7.16: Performance comparison of the navigation algorithms for Scenario 2: norm errors on position (x,y,z) of the chaser in H system (top), attitude of T system with respect to H system (top), relative translational velocity in H system (bottom) and target angular velocity (bottom). The distance to target is intended as distance between sensor platform and target center of mass.

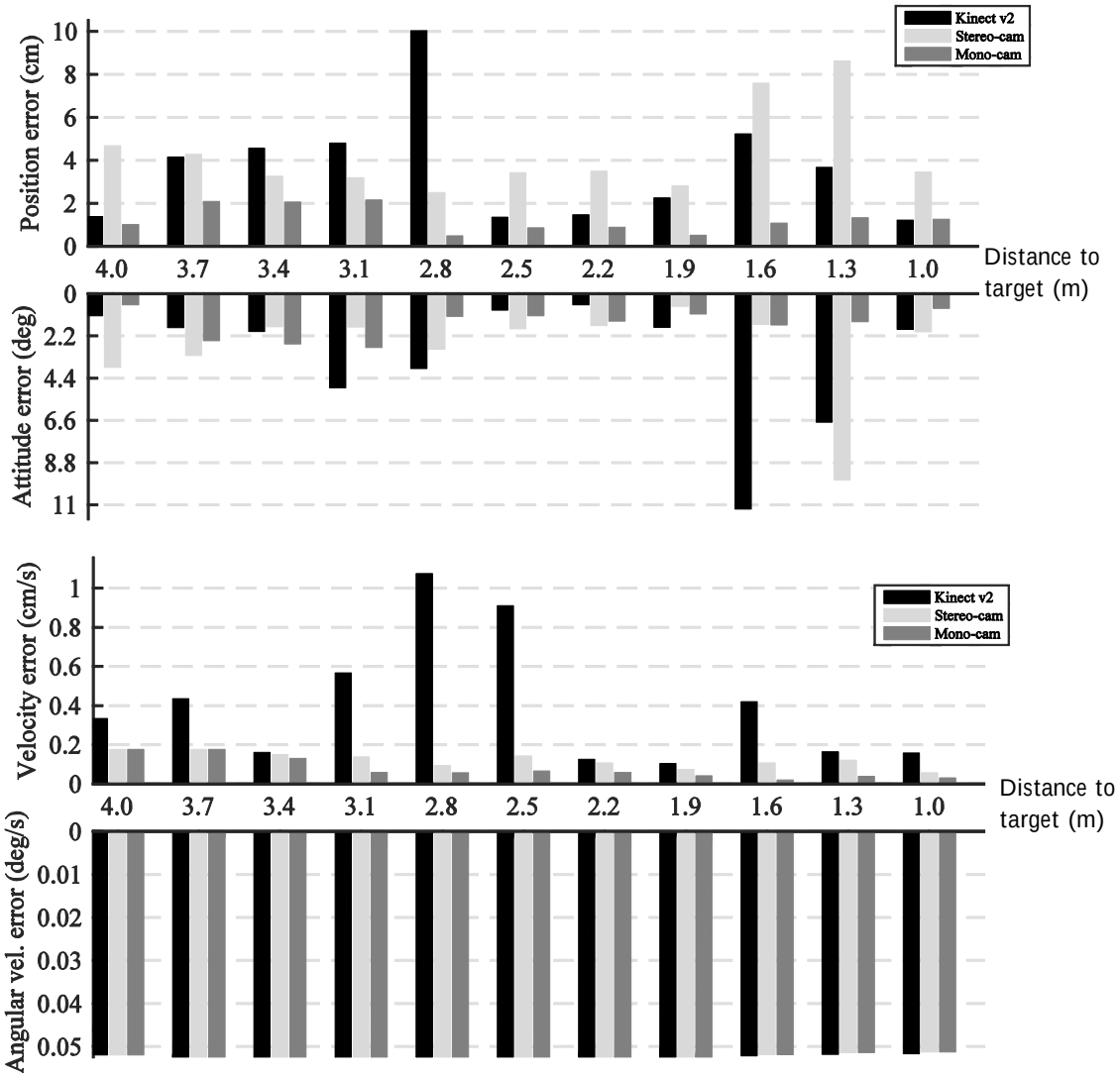

Figure 7.17: Performance comparison of the navigation algorithms for Scenario 3: norm errors on position (x,y,z) of the chaser in H system (top), attitude of T system with respect to H system (top), relative translational velocity in H system (bottom) and target angular velocity (bottom). The distance to target is intended as distance between sensor platform and target center of mass.

As can be observed from Figure 7.16 and Figure 7.17, the error in the estimated angular velocity of the target is constant. In fact, the target rotates at a constant rate and therefore the EKF converges to an accurate estimate.

Finally, as regard to the computational time required for data processing and pose estimation, the monocular navigation system ensures the fastest estimation with an average of 190 ms. Notably, the observations for the computational time made for Scenario 1 hold.

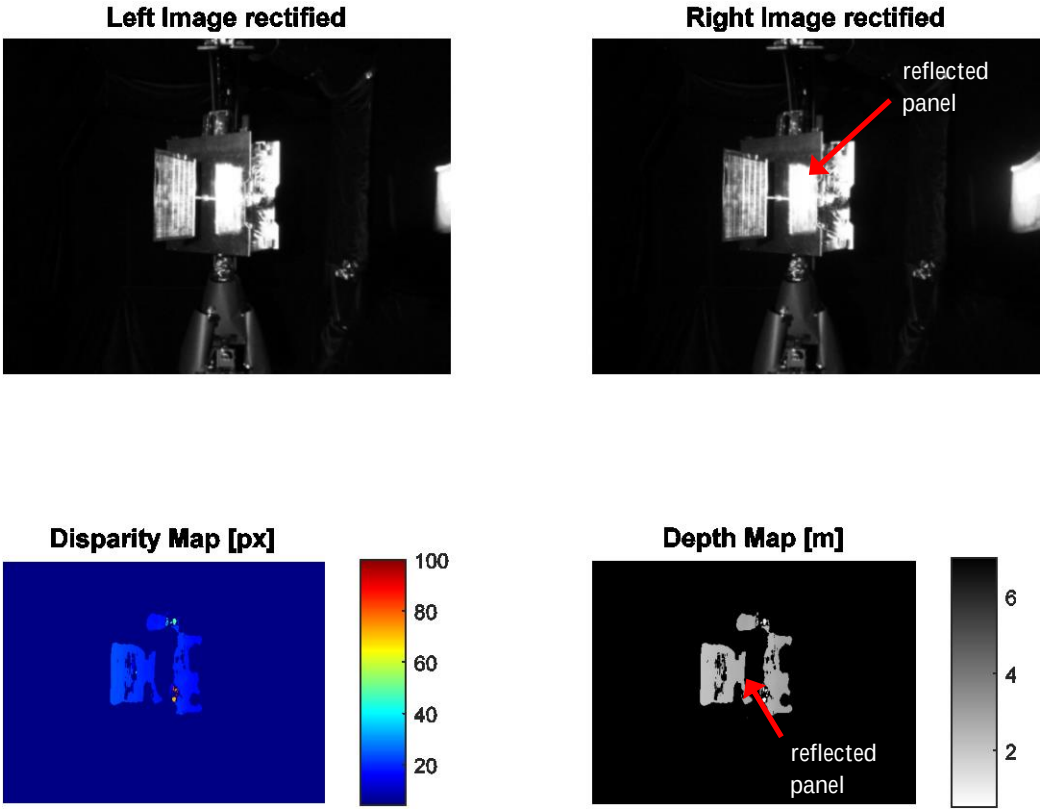

Figure 7.18: Reflection of the solar panel on the target spacecraft's body surface. The reflected feature is present in the depth map of the scene (red arrow) after stereo image processing.

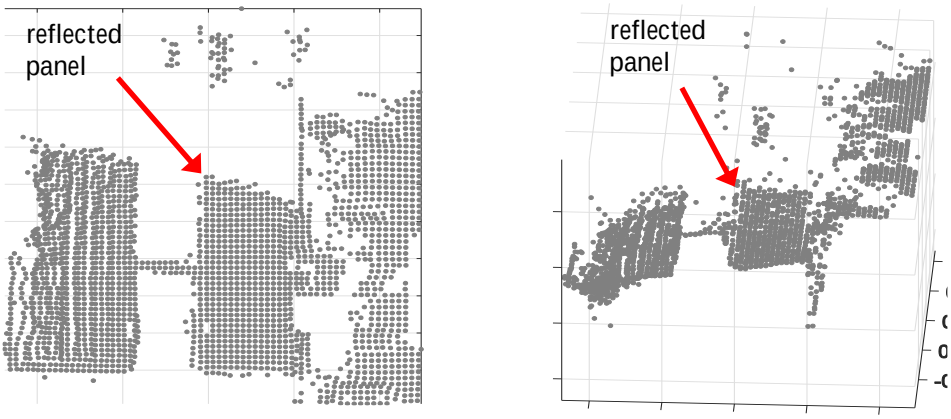

Figure 7.19: Reflection of the solar panel on the target spacecraft's body surface. The reflected feature is present in the depth map of the scene (red arrow) acquired by the Kinect v2 IR sensor.

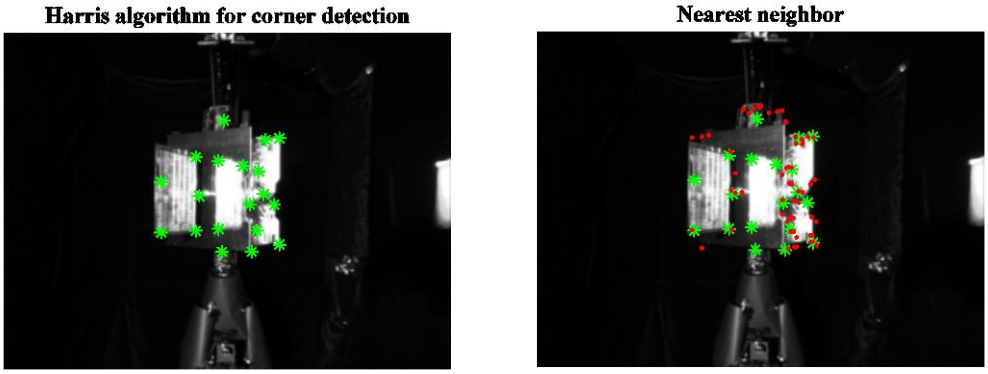

Figure 7.20: Effect of solar panel reflection on monocular image processing: corner detection (left) and nearest neighbor search (right). The projected visible features of the target spacecraft model are denoted with red dots in the right image.

7.5 Docking Precision

This section of the thesis discusses the feasibility of the docking to the target spacecraft using the state estimates provided by the navigation systems for Scenarios 1-3. In order to study the feasibility for different docking situations, several combinations of possible sizes of the target vehicle and remaining maneuver time after the latest state estimates have been considered. In particular, the position of the target docking port with respect to the target spacecraft center of mass was changed between 0.4 m (size of the target model in RACOON laboratory) and 4.6 m (size of Envisat) with a step of 0.2 m in order to cover most of catalogued space debris dimensions. The remaining maneuver time ranged between 0.5 s and 10.5 s with a step of 1 s. Notably, 0.5 s corresponds to the duration of the maneuver in the maneuver simulated in 5.7. The combinations of these cases resulted in 220 different docking situations. The latest state estimates obtained with the navigation algorithms for Kinect v2, stereo camera and mono camera in simulated Scenarios 1-3 were propagated using the two-spacecraft proximity operation model (Section 3.1). The expected docking conditions were evaluated using Eqs. (3.8-3.11). In the case the size of the target spacecraft resulted larger than the final relative distance of 1.0 m (see Figure 7.10 and Figure 7.12), data from RACOON laboratory were rescaled to the original size (scale factor 1:4). The original errors of the navigation algorithms were applied to the rescaled data. After this scaling, if the target dimension results larger than the final estimated distance, the second-last state estimate was employed for the propagation. The error on target inertia was set to 2 % of the real value according to the results of Section 5.7. The docking maneuver was considered successful the following conditions were satisfied:

1. the error in the estimated docking enabling conditions is lower than the docking tolerances defined in Section
2. the remaining maneuver duration is higher than the computational time required for the noncooperative pose estimation.

Figure 7.21 illustrates the success rates of the docking maneuvers using the data taken for Scenario 1. The mono camera-based navigation system ensures correct mechanical contact in the 39.0 % of the analyzed cases, resulting the highest rate for this scenario. Although the errors of the three navigation systems are comparable in the final state estimates (see Figure 7.12), the high computational time required by ICP and by stereo image processing limits the application to maneuvers lasting 3 s (Kinect v2) and 6 s (stereo camera).

The success rates for Scenario 2 and 3 are illustrated in Figure 7.22 and Figure 7.23 respectively. Also in this case, the mono camera-based navigation system ensures the highest success rates, which are 81.0 % for Scenario 2 and 60.0 % for Scenario 3. This result is motivated by the lower estimation errors during the maneuvers due to the robustness to feature reflection (see discussion in Section 7.4) and lower computational time. Moreover, the remaining maneuver duration to enable ICP-based pose estimation must be at least 3 s. However, due to the larger estimation errors, the success rates rapidly decrease with the maneuver time due to the propagation of the initial error.

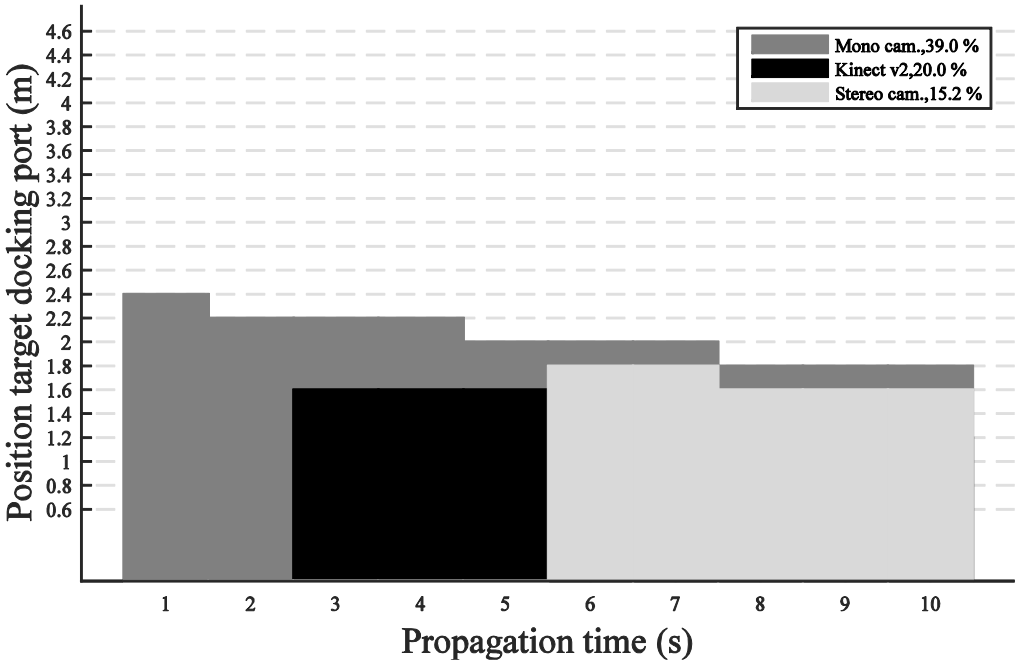

Figure 7.21: Success rate of the docking maneuver for different combinations of position of target docking port with respect to the target center of mass and remaining maneuver time after the latest state estimate of Scenario 1. Colored area indicates successful docking.

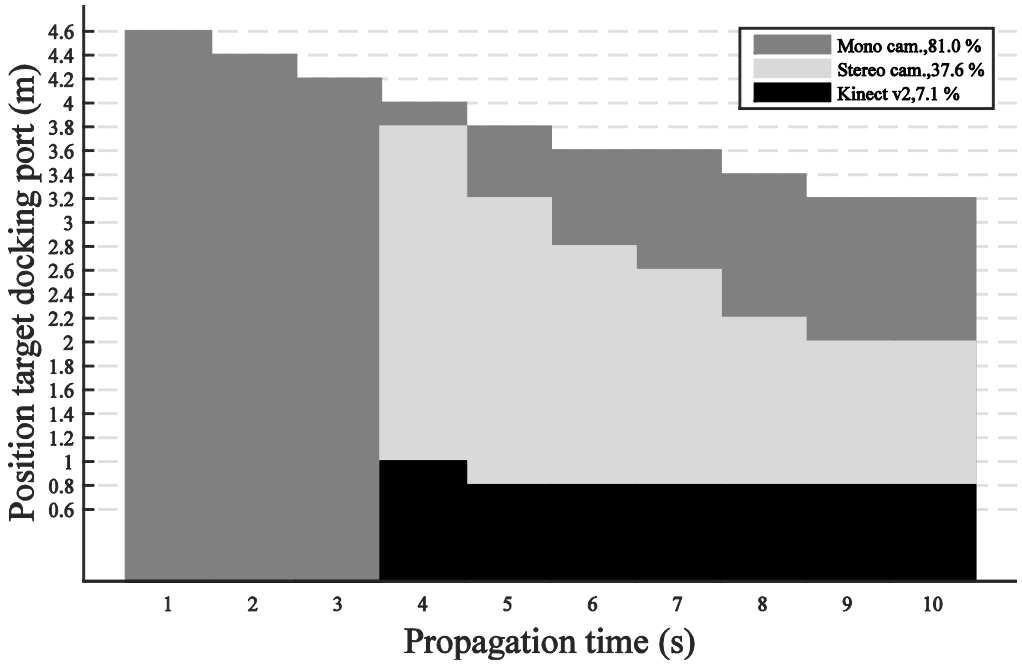

Figure 7.22: Success rate of the docking maneuver for different combinations of position of target docking port with respect to the target center of mass and remaining maneuver time after the latest state estimate of Scenario 2. Colored area indicates successful docking.

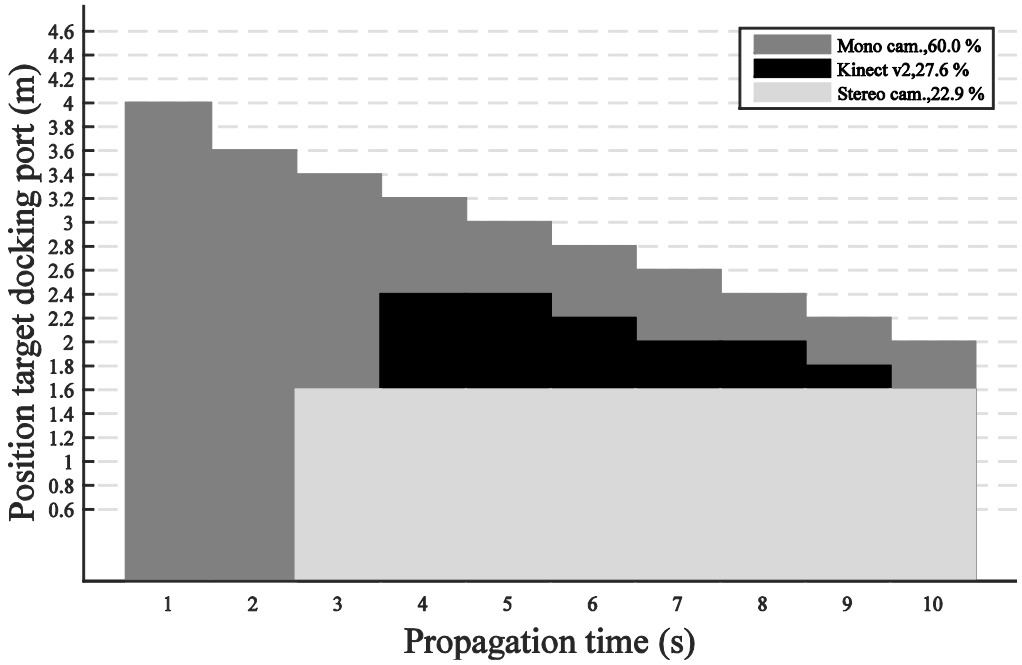

Figure 7.23: Success rate of the docking maneuver for different combinations of position of target docking port with respect to the target center of mass and remaining maneuver time after the latest state estimate of Scenario 3. Colored area indicates successful docking.

7.6 Summary

This chapter presented the test campaign on the navigation algorithms introduced in Chapter 6. Three proximity operation test cases were simulated using the RACOON laboratory, which provides real-time hardware simulation of proximity maneuvers in full DoF under realistic lighting conditions. In one test case, the simulated chaser approached an attitude-stabilized target. In the other scenarios, the target vehicle tumbled in flat-spin mode with different rotation rates. Three sensors were used to acquire data regularly during the maneuver and, therefore, to test the navigation algorithms proposed in Chapter 6: the IR sensor of the Microsoft Kinect v2 for Lidar capability and the Bumblebee2 stereo camera for both stereo and monocular navigation.

As regard to the pose initialization, all the proposed techniques provide the first estimate of the target pose with errors of at least 1.1 cm and 1.0 deg obtained with Kinect v2. The ICP-based template matching technique requires small angular steps in order to provide an accurate estimate of the first pose. However, the algorithm may require several minutes to provide the solution. Coarse angular discretization ensures the solution within 2 minutes, but at the cost of lower accuracy. The performance of the monocular pose initialization algorithm depends on the geometric groups detected in the image. Complex groups such as parallelograms generate smaller match matrices, since the number of parallelograms that define the spacecraft wireframe model is limited. As a consequence, the EPnP algorithm is called fewer times, resulting in lower computational time. If only basic groups such as proximity or parallel lines are detected, the number of combinations increases dramatically and the algorithm is not suitable for onboard applications due to the large size of the match matrix. Furthermore, the capability of the image processing algorithm to detect complex shapes depends on the tuning of numerous parameters, on the attitude of the target vehicle and on the illumination conditions. This aspect limits the broad applicability of the method.

Regarding the tracking procedure during proximity operations, the 3D point clouds obtained from Microsoft Kinect v2 and from Bumblebee2 after stereo image processing may present reflected features for certain attitude configurations of the target spacecraft. This effect produces large errors on the state estimates. On the contrary, the monocular navigation algorithm is robust to feature reflection, since the nearest neighbor search matches the projected model features only with true spacecraft corners detected in the image.

The propagation of the latest pose estimates allows obtaining the final docking conditions that will be enforced by the guidance algorithm. Numerous different maneuver conditions were considered. Due to the high computational speed and lower errors in the state estimates, the monocular navigation system ensures successful docking in numerous maneuvering scenarios.

The success depends on the remaining duration of the maneuver and on the size of the target. Clearly, the longer the duration, the lower the success rate due to the propagation of the errors in the state estimates. Similarly, the larger the target, the lower the success rate of the docking, since the estimate errors are amplified by the size of the target in the docking enabling conditions (see Section 5.7).

Overall, the monocular pose estimation algorithm ensures the highest number of success rate, due to the robustness to feature reflections and to the low computational time, which allows docking to the target even 0.5 s after the data acquisition. In general, successful proximity operations and docking are guaranteed if the docking is performed shortly after the last state estimate to a small target.

8 Summary and Conclusions

8.1 Research Results

The research addressed in this thesis investigates the hypothesis that it is possible to develop a complete GNC system to enable autonomous proximity operations to noncooperative space targets. This hypothesis is motivated by the need to execute a new generation of space robotic missions such as debris removal and OOS to repair, refuel and deorbit defunct or end-of-life satellites. The desired outcome of performing autonomous missions is the increment of mission robustness and reliability with respect to the existing ground-in-the-loop guidance methods. Furthermore, the capability to autonomously rendezvous and dock allows retrieving and servicing a variety of space targets being either end-of-life satellites, nonfunctional spacecraft or upper stage rockets. Finally, such GNC technology for autonomous proximity operations can be potentially applied to deep space rendezvous and docking procedures since autonomous systems do not require real-time support from ground.

Regarding the mission requirements, the chaser is assumed mid-size spacecraft designed to rendezvous, capture and deorbit a noncooperative target. The chaser of the future e.Deorbit mission to deorbit Envisat is considered as possible reference spacecraft. The target satellite is orbiting in LEO, is not equipped with dedicated visual markers for distance and attitude estimation, does not communicate its state to the ground and tumbles about the principal axes due to its nonfunctional attitude control system. The technologies developed in this thesis must ensure autonomous approach to the target from an initial distance of about 10 m being the beginning for standard proximity operations and docking maneuvers. The approach terminates at few centimeters from the target in order to enable either docking or berthing. The GNC system for these proximity operations must ensure safe and successful approaches, meaning that the path constraints for collision avoidance and endpoint conditions must be enforced with high precision even under measurement noise and model uncertainties. The geometric model of the target spacecraft is assumed available either from the constructor or from shape reconstruction performed during the inspection phase.

In order to investigate the research hypothesis, an innovative GNC system to enable autonomous proximity operations for noncooperative space targets was developed and tested step-by-step. The detailed scheme of the proposed system is illustrated in Figure 8.1. The input of the entire loop is the raw image of the target object taken by the mono camera sensor, which regularly

acquires data during the maneuver. After image acquisition, the pose initialization procedure is executed in order to estimate distance and attitude of the target without prior information about its state. To do this, standard Sobel and Hough algorithms are employed to detect the line segments of the image corresponding to the true edges of the target spacecraft. These segments are then organized in geometric groups such as pair of parallel, proximity, parallelograms and parts of a parallelogram. The detected groups are matched with the corresponding groups that define the (available) wireframe model of the target. Since no information on the target state are available, all the possible correspondences between groups in the image and the same groups in the model are considered. The EP*n*P solver determines attitude and position of the target spacecraft using the group correspondences stored in the match matrix. This routine for pose initialization is the most computationally expensive part of the entire pose estimation procedure since numerous combinations of group correspondences must be processed by EP*n*P. The EKF estimates current position, attitude and velocities of both spacecraft using the estimated pose and the measurements from GPS and IMU. Once initialized, the pose estimation procedure is executed in closed-loop. The Harris algorithm in the image processing block extracts the feature points of the target object from the image. The wireframe model of the target is projected onto the image using a guessed current pose obtained by propagating the previous pose estimate. Using the nearest neighbor search, the features detected in the image are paired with the corresponding model features just projected. At this point, the current target pose is estimated with a Newton-Raphson method that solves the 3D-2D true perspective projection equations using feature correspondences as initial condition. The EKF estimates current position, attitude and velocities of the vehicles with respect to the Hill's coordinate system.

The estimate of the current state is fed into the guidance part of the system. Knowing the dynamics model of the maneuver, the trajectory of the chaser with respect to the Hill's coordinate system is parameterized with a set of polynomial functions. Some of the polynomial coefficients are constrained to satisfy the endpoint conditions being the current state and the predicted final state. The free coefficients are optimized through the nonlinear constrained optimization solver SNOPT. The path constraints of the maneuver are enforced by introducing nonlinear constraints in the optimization problem. To ensure fast optimization, fifth-order polynomials are applied to represent the translational trajectory and the MRP-based attitude trajectory. Since the output controls are smooth, the SDM-PWM technique is used to generate thrust pulses. This trajectory generation procedure is executed in closed-loop as illustrated in Figure 8.1.

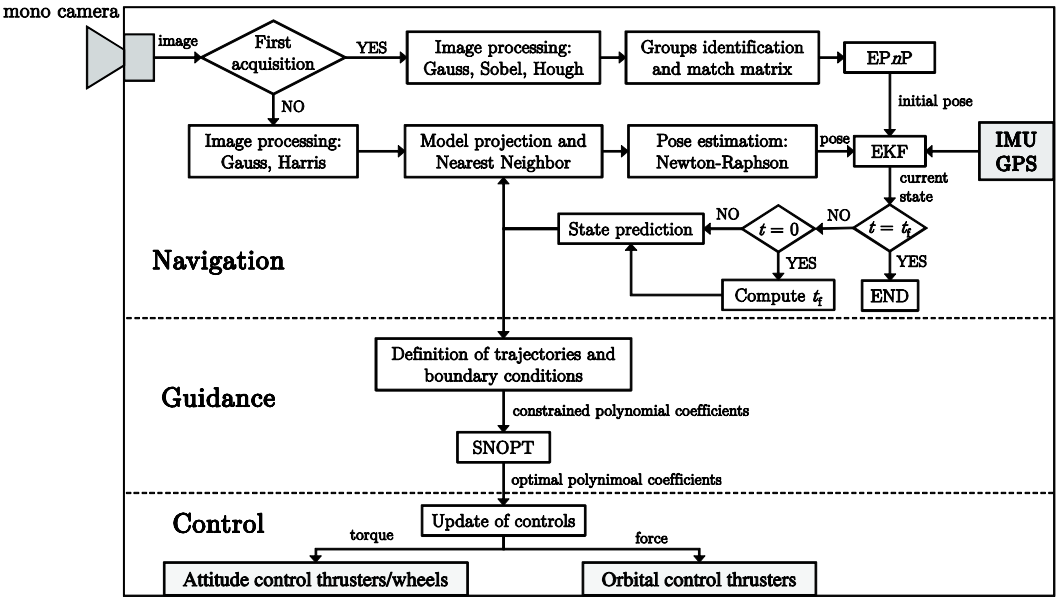

Figure 8.1: Scheme of the final GNC systems. Hardware payload are denoted with grey blocks, whereas software packages are indicated by white blocks.

The proposed system was subjected to test campaigns in order to evaluate the performance of the system and motivate the design choices. In particular, the trajectory optimization algorithm was implemented in Matlab / Simulink environment and tested through Monte Carlo analyses. In the ideal case of perfect knowledge of current state vector and target inertia, the proposed guidance algorithm is capable to generate near-optimal maneuvers while enforcing all the path constraints and achieving accurate docking conditions. The small difference in optimality is due to slight discrepancies in the chaser attitude trajectory, since piecewise polynomials can match the optimal trajectory up to a certain level. The results from the Monte Carlo simulation demonstrate that the guidance scheme, in the same ideal case of perfect knowledge of the target conditions, commands accurate trajectories in most maneuvering scenarios, also in presence of the orbital perturbations in LEO. The applicability of the guidance algorithm to noncooperative scenarios was demonstrated with a sensitivity analysis through another Monte Carlo simulation. Uncertainties on both target inertia and relative state were considered since these two parameters are subjected to the highest noise in noncooperative maneuvers. The sensitivity study showed that the proposed guidance system is particularly sensitive to noise in target angular velocity. In fact, this type of disturbance is amplified by the size of the target spacecraft in the docking-enabling conditions and causes significant errors in the final relative translational velocity between spacecraft docking points. As regard to the errors in the inertia of the target vehicle, the guidance is capable to dock successfully whenever the error is less than 4% of the nominal value. These results were obtained by updating the trajectory at a rate of 2

Hz in the last seconds before the docking using the latest optimized polynomial coefficients and the current state estimates.

The navigation subsystem was selected after an experimental campaign in RACOON laboratory, which provides real-time hardware simulation of proximity operations in full DoF under realistic lighting conditions. In this campaign, three different noncooperative pose estimation methods were tested. The first method employs a Lidar sensor to acquire the point cloud of the target. The related pose estimation algorithm employs the ICP method to estimate current distance and attitude of the target by comparing the acquired point cloud with a reference model of the same vehicle stored onboard. The second pose estimation method employs a stereo camera sensor to acquire two simultaneous images the target from different points of view. The point cloud of the scene is reconstructed from the disparity map of the stereo image. Similarly to the Lidar-based navigation system, the ICP algorithm is employed for noncooperative pose estimation using the 3D measurement and a reference point cloud. The third method considered for the test campaign is the monocular navigation system. Three test cases were simulated in RACOON laboratory. These maneuvers started from 4.5 m to the target, which rotated in flat-spin mode at 1.6 deg/s and 3.2 deg/s. In the third scenario, the target was attitude-stabilized. Data were acquired regularly during the maneuvers using the IR sensor of the Microsoft Kinect v2 for Lidar capability and the Bumblebee2 stereo camera for both stereo and monocular navigation. The pose initialization algorithms were able to estimate the target pose without prior state information. At least 1 or 2 minutes are required to estimate the pose, depending on the target attitude. Regarding the tracking procedure during proximity operations, the 3D point clouds obtained using Microsoft Kinect v2 and Bumblebee2 presented incorrect features due to the reflection of the target's solar panels on the vehicle's body surface for certain attitude configurations. These incorrect features caused large errors on the state estimates. On the contrary, the monocular navigation algorithm showed robustness to feature reflection, since the nearest neighbor search matches the projected model features only with true spacecraft corners detected in the image. The propagation of the latest pose estimates allows obtaining the final docking conditions that will be enforced by the guidance algorithm. Numerous test cases were considered by combining different remaining maneuver times with several possible sizes of the target object. Due to the fast computational speed and low errors in the estimated states, the monocular navigation system ensured successful docking in most of the considered scenarios. The success depends on the remaining duration of the maneuver and on the size of the target. Clearly, the longer the duration, the lower the success rate due to the propagation of the errors in the estimated states. Similarly, the larger the target, the lower the success rate of the docking, since errors are amplified by the size of the target in the docking enabling conditions. Overall, the monocular pose estimation algorithm ensures the highest number of success rate, due to the robustness

to feature reflections and to the low computational time, which allows docking to the target 0.5 s after the data acquisition. In general, successful proximity operations and docking are guaranteed if the docking is performed shortly after the last state estimate to a small target. For these reasons, the monocular navigation system has been selected.

8.2 Conclusions

As regard to the initial research hypothesis, a complete GNC system that enables autonomous proximity operations for noncooperative space targets was developed, implemented and tested step-by-step in this thesis. The proposed system is illustrated in Figure 8.1 and it consists of the following key components:

1. **Sensor suite** composed by mono camera, GPS receiver and IMU. The mono camera is used to acquire the picture of the target object, whereas GPS and IMU provide current inertial position, attitude and velocities of the chaser. An additional sensor such as Sun sensor or star tracker must be considered for chaser attitude initialization.
2. **Noncooperative pose estimation algorithm** that processes the acquired image and estimates distance and attitude of the target object without using visual markers, which are unavailable. This software package consists of an image processing procedure that identifies the target features in the image, a nearest neighbor search that pairs the detected features with the corresponding ones of the projected wireframe model and a Newton-Raphson routine that solves the projection equations for the pose using the features matches. The pose initialization procedure estimates distance and attitude of the target without prior information of its state using the line geometries detected in the image and the EPnP algorithm. In particular, this software package requires the real-size wireframe model of the target vehicle, which is stored onboard the chaser.
3. **EKF** that reduces the measurement noise and estimates the complete state vector, including the velocities, with respect to the Hill's coordinate system.
4. Rapid **trajectory optimization algorithm** that generates the approach trajectory and the required controls in closed-loop during the maneuver. The key feature of this method is that the trajectory of the chaser spacecraft with respect to the Hill's coordinate system is parameterized using a set of polynomials. The trajectory's endpoint conditions are enforced by constraining some of the polynomial coefficients. Knowing the dynamics model of the system, the optimal control problem of the maneuver can be converted into an equivalent nonlinear optimization problem, whose varied parameters are the free polynomial coefficients.

The path constraints are enforced by defining the nonlinear constraints of this optimization problem.

Numerical tests demonstrated that the proposed trajectory optimization algorithm is capable to generate precise and sub-optimal trajectories in few seconds between 0.2 s and 9.0 s. Despite the highly nonlinear dynamics of the maneuver, this algorithm ensures rapid convergence due to the limited number of variable parameters to be optimized. The highest computational times arise during the firsts loop executions since the free coefficients require some optimizations to converge to the optimal ones. Once converged, the trajectory can be just updated using the optimal polynomial coefficients and the latest state estimate. By updating at a rate of 2 Hz, the system ensures robustness to orbital perturbations, sensor noise and uncertainties on the target inertia.

The noncooperative pose estimation system was tested using the RACOON laboratory, which provides real-time full DoF hardware simulation of proximity operations under realistic lighting conditions. In all the test cases, the algorithm showed robustness to feature reflection on the target surfaces. On the contrary, other algorithms such as Lidar-based and stereo camera-based navigation systems did not ensure this robustness. Another important aspect of the monocular navigation system is that it requires about 0.3 s to estimate the current pose from the image of the target.

Other key advantages of the proposed GNC system are the low power and mass requirements. In fact, a typical mono camera of 500 g may require abut 2 W for operating at 48 FPS. More power can be therefore designated to the onboard computer, whose clock determines the computational time of the proposed algorithms. As demonstrated in Chapter 5, a low computational time is crucial to ensure robustness to system noise.

8.3 Technical Issues and Future Work

Despite promising test results, some considerations must be done in order to address the future work with the goal to apply the proposed technology in real proximity OOS and space debris removal missions.

Firstly, the trajectory optimization and the noncooperative pose estimation algorithms were implemented and tested in Matlab / Simulink environment. The required CPU times were estimated on an Intel Core i7 processor with 2.8 GHz and 8 Gb of RAM. Despite these algorithms will be implemented in faster software environments such as C/C++, the clocks of current state-of-the-art onboard processors are still limited between 300 MHz and 1 GHz. Therefore, it is reasonable to expect higher computational times on these processors. Further tests must be performed in C/C++ environment on an onboard-like processor in order to estimate more realistic computational speeds. However, the trajectory optimization software will remain the most demanding algorithm. To ensure onboard execution, the following

parameters can be tuned in order to speed-up the optimization: decrease the order of polynomials, reduce the nodes that discretize the time domain and reduce the maximum number of iterations of SNOPT. Notably, this last aspect may reduce the optimality of the output trajectory. As regard the path constraints during proximity operations, the plume impingement avoidance must be added. Moreover, failure detection conditions shall be considered to cope with mission contingencies such as thrust failure.

Another aspect to consider is that the low cost Bumblebee2 and Microsoft Kinect v2 commercial sensors were used for the test campaign in RACOON laboratory. Regarding the monocular navigation, the usage of space qualified sensor such the mono camera employed in PRISMA mission would possibly reduce the pose error due to the higher resolution with respect to Bumblebee2. Furthermore, the laboratory experiments employed real sensors and a scaled 50x70x85 cm model. At close distances, this model was still entirely visible due to its small size. On the contrary, in a real mission such the servicing of Envisat, the target satellite would not be entirely visible if located at the same distance to the chaser. This may create some problems for the monocular navigation, since the corners of the target's body will not be visible anymore. The efficacy of the algorithm must be verified with real scale models.

Finally, the monocular pose estimation algorithm requires the tuning of numerous parameters such as tolerances for line group definition, number of feature correspondences for Newton-Raphson, number of corners to be detected by the Harris algorithm and the covariance of the EKF. In order to tune these parameters, numerous ground tests are required before the launch of the mission. The pose initialization shall be improved too. In fact, if only simple line groups such proximity or pair of parallels are found, the computational time becomes too high to ensure onboard execution. It can be necessary, for example, to repeat the image acquisition until a complex group is found. At that point, the pose initialization is executed.

Despite the recommended future work for software improvement, the tools developed in this thesis find broad applicability to space robotics. For example, the proposed algorithm for rapid trajectory planning can modified in such a way to perform proximity operations to space objects in elliptical orbits. In fact, new linearized equations for relative motion between spacecraft in elliptical orbit were recently introduced [122]. With respect to the classical C-W equations, this model adds the eccentricity of the orbit and the rate change of the true anomaly. Interestingly, the C-W equations become a particular case of the more general linearized equations for relative motion. The trajectory generation algorithm presented in this thesis can be easily therefore applied to this new model. This algorithm would enable suboptimal proximity operations for space target in any closed orbit.

Bibliography

[1] Zimpfer, D., Kachmar, P., and Tuohy, S., "Autonomous Rendezvous, Capture and In-Space Assembly: Past, Present and Future," *1st AIAA Space Exploration Conference: Continuing the Voyage of Discovery*, NASA Johnson Space Center, Orlando, FL, 2005. doi: 10.2514/6.2005-2523

[2] Saleh, J.H., Lamassoure, E. and Hastings, D.E., "Space Systems Flexibility Provided by On-Orbit Servicing: Part I," *Journal of Spacecraft and Rockets*, Vol. 39, No. 4, pp. 551-560, 2002. doi: 10.2514/2.3844

[3] Hirzinger, G., et al., "DLR's robotics technologies for on-orbit servicing," *Advanced Robotics*, Vol. 18, Issue 2, pp. 139-174, 2004. doi: 10.1163/156855304322758006

[4] Woffinden, D.C., and Geller, D.K., "Navigating the Road to Autonomous Orbital Rendezvous," J*ournal of Spacecraft and Rockets*, Vol. 44, No. 4 (2007), pp. 898-909. doi: 10.2514/1.30734

[5] Kucharski, D., et al., "Attitude and Spin Period of Space Debris Envisat measured by Satellite Laser Ranging," *IEEE Transaction on Geoscience and Remote Sensing,* Vol. 52, No. 2, December 2014, pp. 7651-7657.

[6] Wilde, M., Harder, J. T., Ventura, J., Hörmann, J., and Walter, U., "Impact of space-to-ground video transmission constraints on teleoperated final approach and docking," *Journal of Aerospace Information Systems*, Vol. 12, Issue 7, 2015, pp. 441-454.

[7] Harder, J., Wilde, M., Ventura, J., and Dziura, M. "Acoustic telepresence for spacecraft proximity operations," *IEEE Aerospace Conference* , Big Sky, MT, March 2016, pp. 1-12.

[8] Starek, J.A., Acikmese, B., Nesnas, I. A. D. and Pavone, M., "Spacecraft Autonomy Challenges for Next Generation Space Missions," *Lecture Notes in Control and Information Sciences*, Ed. E. Feron, Vol. 460. Springer Berlin Heidelberg, 2016, pp. 1–48.

[9] Waltz, D. M., *On-orbit servicing of space systems*. Malabar, Fla., Krieger, 1993.

[10] Goodman, J. L., "History of Space Shuttle Rendezvous and Proximity Operations," *Journal of Spacecraft and Rockets*, vol. 43, no. 5, pp. 944–959, September-October 2006

[11] Hibbard, R. L., "Satellite On-Orbit Refueling: A Cost Effectiveness Analysis," Master's Thesis, Naval Postgraduate School, Monterey, California, 1996

[12] King, D., "Space Servicing: Past, Present and Future," Proceeding of the 6th International Sym-posium on Artificial Intelligence and Robotics & Automation in Space: i-SAIRAS 2001, 2001

[13] Oda, M., "ETS-VII: Achievements, Troubles and Future," Proceeding of the 6th International Symposium on Artificial Intelligence and Robotics and Automation in Space: i-SAIRAS, Saint-Hubert, Canada, 2001.

[14] Davis, T. M., and Melanson, D., "XSS-10 Micro-Satellite Flight Demonstration Program," *Proceedings of the 17th Annual AIAA/USU conference on Small Satellites*, Logan, UT, August 2003.

[15] Rumford, T.E., "Demonstration of Autonomous Rendezvous Technology (DART) Project Summary," *Proceedings of the Society of Photo-Optical Instrumentation Engineers: Space Systems Technology and Operations*, Vol. 5088, pp. 10-19, August 2003. doi: 10.1117/12.498811

[16] Howard, R.T., and Bryan, T.C., "DART AVGS flight results," Proceedings of the Society of Photo-Optical Instrumentation Engineers: Space Systems Technology and Operations, Vol. 6555, May 2007. doi: 10.1117/12.720242

[17] Weismuller, T., and Leinz, M., "GN&C technology demonstrated by the orbital express autonomous rendezvous and capture sensor system," *Proceedings of the 29th AAS Guidance and Control Conference*, AAS Paper 06-016, 2006.

[18] D'Amico, S., et al., "Noncooperative Rendezvous Using Angles-Only Optical Navigation: System Design and Flight Results," *Journal of Guidance, Control and Dynamics,* Vol. 36, No. 6, 2013, pp. 1576-1595. doi: 10.2514/1.59236

[19] Hablani, H., Tapper, M., and Dana-Bashian. D., "Guidance algorithms for autonomous rendezvous of spacecraft with a target vehicle in circular orbit," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, number AIAA-2001-4393, Montreal, Canada, August 2001.

[20] Pearson, D.J., "The glideslope approach," *Advances in the Astronautical Sciences*, Vol. 69, pp 109-123, 1989.

[21] Nolet, S., "Development of a Guidance, Navigation and Control Architecture and Validation Process Enabling Autonomous Docking to a Tumbling Satellite," Ph.D. dissertation, Aeronautics and Astronautics Dept., MIT, 2007.

[22] Richards, A., Schouwenaars, T., How, J.P., and Feron, E., "Spacecraft trajectory planning with collision and plume avoidance using mixed-integer linear programming," *Journal of Guidance, Control and Dynamics*, Vol. 25(Num.4), pp 755-764, July-August 2002.

[23] Richards, A.G., "Trajectory optimization using mixed-integer linear programming," Master of Science thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, June 2002

[24] Breger, L., and How, J.P., "Safe Trajectories for Autonomous Rendezvous of Spacecraft," *Journal of Guidance, Control and Dynamics*, Vol. 31, No. 5, 2008, pp. 1478-1489. DOI: 10.2514/1.29590

[25] Rawlings, J.B., and Mayne, Q.B., *Model Predictive Control: Theory and Design*, Nub Hill Publishing, 2009.

[26] Hartley, E.N., Trodden, P.A., Richards, A.G., and Maciejowski, J.M., "Model Predictive Control System design and implementation for spacecraft rendezvous," *Control Engineering Practice*, Vol. 20, Issue 7, pp. 695-713, 2012. doi: 10.1016/j.conengprac.2012.03.009

[27] Singh, L., Bortolami, S., and Page, L., "Optimal Guidance and Thruster control in orbital approach and rendezvous for docking using model predictive control," *AIAA Guidance, Navigation and Control Conference and Exhibit*, AIAA Paper 2010-7754, Toronto, Canada, 2010, pp. 1-15.

[28] Di Cairano, S., Park, H., and Kolmanovsky, I., "Model Predictive Control approach for guidance of spacecraft rendezvous and proximity maneuvering," *Int. J. Robust Nonlinear Control*, Vol. 22, 2012, pp. 1398-1427. doi: 10.1002/rnc.2827

[29] Virgili-Llop, J., Zagaris, C., Park, H., Zappulla, R., and Romano, M., "Experimental evaluation of model predictive control and inverse dynamics control for spacecraft proximity and docking maneuvers," *Proceedings of the 6th International Conference on Astrodynamics Tools and Techniques (ICATT)*, Darmstadt, March 2016,

[30] Wolf, C., "Onboard guidance for autonomous docking maneuvers with uncontrolled target spacecraft using Model Predictive Control," Master of Science thesis, Institute of Astronautics, Technical University of Munich, Munich, Germany, 2015.

[31] LaValle, S.M., and Kuffner, J. J., "Randomized Kinodynamic Planning," The International Journal of Robotics Research, Vol. 20(5), 2001, pp. 378-400.

[32] Karaman, S., and Frazzoli, E., "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, Vol. 30(7), 2011, pp. 846-894.

[33] Frazzoli, E., Dahleh, M.A., and Feron, E., "Real-Time Motion Planning for Agile Autonomous Vehicles", *Journal of Guidance, Control, and Dynamics*, Vol. 25, No. 1 (2002), pp. 116-129.

[34] Phillips, J.M., Kavraki, L.E., and Bedrossian, N, "Probabilistic Optimization applied to Spacecraft Rendezvous and Docking," *Proceedings of the 13th American Astronomical Society/AIAA - Space Flight Mechanics Meeting*, Puerto Rico, February 2003.

[35] Yakimenko, O.A., "Direct Method for Rapid Prototyping of Near Optimal Aircraft Trajectories." *Journal of Guidance, Control, and Dynamics*. Vol. 23, No. 5, 2000, pp. 865-875.

[36] Ventura, J., Romano, M., and Walter, U., "Performance Evaluation of the Inverse Dynamics Method for Optimal Spacecraft Reorientation,"

Acta Astronautica, Vol. 110, 2015, pp. 266-278. doi: 10.1016/j.actaastro.2014.11.041

[37] Bevilacqua, R., Romano, M., and Yakimenko, O., "Online generation of quasi-optimal spacecraft rendezvous trajectories", *Acta Astronautica*, Vol. 64, Issue 2-3, 2009, pp. 345-358.

[38] Ciarcia', M., and Romano, M., "Suboptimal Guidance for Orbital Proximity Maneuver with Path Constraints Capability." *AIAA Guidance, Navigation and Control Conference*. Minneapolis, Minnesota, 13-16 August 2012. DOI: 10.2514/6.2012-4920

[39] Ciarcia', M., Grompone, A., and Romano, M., " A near-optimal guidance for cooperative docking maneuvers," *Acta Astronautica*, Vol. 102, 2014, pp. 367-377.

[40] Boyarko, G., "Spacecraft guidance strategies for proximity maneuvering and close approach with a tumbling object," PhD dissertation, Mechanical and Aerospace Engineering Dept., Naval Postgraduate School, 2010.

[41] Kobilarov, M., and Pellegrino, S., "Trajectory Planning for CubeSat Short-Time-Scale Proximity Operations," *Journal of Guidance, Control, and Dynamics*, Vol. 37, No. 2, 2014, pp. 566-579. doi: 10.2514/1.60289

[42] Wilde, M., Ciarcià, M., Grompone, A., and Romano, M., "Experimental Characterization of Inverse Dynamics Guidance in Docking with a Rotating Target", *Journal of Guidance, Control, and Dynamics*, Vol. 39, No. 6 (2016), pp. 1173-1187.

[43] Kelsey, J. M., Byrne, J., Cosgrove, M., Seereeram, S., and Mehra, R. K., "Vision-Based Relative Pose Estimation for Autonomous Rendezvous and Docking," *2006 IEEE Aerospace Conference*, pp. 1–20, 2006.

[44] Oumer, N. W. and Panin, G., "3D point tracking and pose estimation of a space object using stereo images," *21st International Conference on Pattern Recognition (ICPR 2012) on. IEEE*, pp. 796 – 800, Tsukuba, Japan, 2012.

[45] Tweddle, B. E., and Saenz-Otero, A., "Relative Computer Vision-Based Navigation for Small Inspection Spacecraft," *Journal of Guidance, Control, and Dynamics*. Vol. 38, Issue 5, 2014, pp. 969-978.

[46] D'Amico, S., Benn, M., and Jørgensen, J. L., "Pose estimation of an uncooperative spacecraft from actual space imagery," *International Journal of Space Science and Engineering,* 2.2, 2014, pp. 171–189.

[47] Petit, A., Marchand, E., and Kanani, K., "Vision-based space autonomous rendezvous: A case study," *EEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 25-30 Sept. 2011, pp. 619-624

[48] Aghili, F., Kuryllo, M., Okouneva, G., and English, C., "Fault-Tolerant Position/Attitude Estimation of Free-Floating Space Objects Using a Laser Range Sensor," *IEEE Sensors Journal*, Vol. 11, No. 1, January 2011, pp. 176-185.

[49] English, C., Okouneva, G., Saint-Cyr, P., Choudhuri, A., and Luu, T., "Real-Time Dynamic Pose Estimation Systems in Space: Lessons Learned for System Design and Performance Evaluation", *Int. Journal of Intelligent Control and Systems*, Vol. 16, No 2, 2011, pp. 79-96.

[50] Boyarko, G., Yakimenko, O., and Romano, M., "Optimal Rendezvous Trajectories of a Controlled Spacecraft and a Tumbling Object," Journal of Guidance, Control and Dynamics, Vol. 34, No. 4, 2011, pp. 1239-1252. doi: 10.2514/1.47645

[51] Clohessy, W.H., and Wiltshire, R.S., "Terminal Guidance System for Satellite Rendezvous." *Journal of the Aerospace Sciences*, Vol. 27, No. 9, 1960, pp. 653-674.

[52] Wie, B., *Space Vehicle Dynamics and Control*, AIAA Education Series, 1998, Chap. 7, pp. 403-407.

[53] Shuster, M.D., "A survey of Attitude Representations," *The Journal of Astronautical Sciences*, Vol. 41, No. 4, 1993, pp. 439-517.

[54] Ventura, J., and Romano, M., "Exact analytic solution for the spin-up maneuver of an axially symmetric spacecraft. " *Acta Astronautica*, Vol. 104, Issue 1, 2014, pp. 324-340. Doi: 10.1016/j.actaastro.2014.07.038

[55] Tsiotras, P., and Longuski, J.M., "A New Parameterization of the Attitude Kinematics," The Journal of the Astronautical Sciences, Vol. 43, N. 3, 1995, pp. 243-262.

[56] Fehse, W., *Automated Rendezvous and Docking of Spacecraft*, Cambridge University Press, 2003, Chap. 2-7, pp. 21-282.

[57] Kirk, D.E., *Optimal control theory: an introduction*, Prentice-Hall, 1970.

[58] Boyarko, G.A., Romano, M., and Yakimenko, O.A., "Time-Optimal Reorientation of a Spacecraft using an Inverse Dynamics Optimization Method." *Journal of Guidance, Control, and Dynamics*. Vol. 34, No. 4, 2011, pp. 1197–1208.

[59] Yakimenko, O.A., "Optimization of Holonomic Attitude Dynamics using IDVD Method." *3rd International Symposium on Systems and Controls in Aeronautics and Astronautics (ISSCAA)*. Harbin, China, June 8-10, 2010.

[60] Bayir I., "IKONOS Collection Capability with the Direct Tasking from a Regional Operation Center." *Proceeding of the 2003 International Conference on Recent Advances in Space Technologies,* pp. 30-33, Istanbul, Turkey, 2003.

[61] Wie, B., Bailey, D., and Heiberg, C., "Rapid Multitarget Acquisition and Pointing Control of Agile Spacecraft." *Journal of Guidance, Control, and Dynamics*. Vol. 25, No. 1, 2002, pp. 96–104. DOI: 10.2514/2.4854

[62] Davis, T., and Straight, S., "Development of the Tactical Satellite 3 for Responsive Space Missions." *Proceedings of the 4th Responsive Space Conference,* AIAA 2006-4003, April 2006.

[63] Karpenko, M., Bhatt, S., Bedrossian, N., Fleming, A., and Ross, I.M., "First Flight Results on Time-Optimal Spacecraft Slews." *Journal of Guidance, Control, and Dynamics*. Vol. 35, No. 2, 2012, pp. 367–376. doi: 10.2514/1.54937

[64] Bai, X., and Junkins, J.L., "New Results for Time-Optimal Three Axis Reorientation of a Rigid Spacecraft." *Journal of Guidance, Control, and Dynamics*. Vol. 32, No. 4, 2009, pp. 1071–1076.

[65] Bilimoria, K.D., and Wie, B., "Time-Optimal Three Axis Reorientation of a Rigid Spacecraft." *Journal of Guidance, Control, and Dynamics*. Vol. 16, No. 3, 1993, pp. 446–452.

[66] Dixon, M.V., Edelbaum, T., Potter, J.E., and Vandervelde, W.E., "Fuel Optimal Reorientation of Axisymmetric Spacecraft." *Journal of Spacecraft and Rockets*. Vol. 7, Issue 11, 1970, pp. 1345–1351.

[67] Yutko, B.M., and Melton, R.G., "Optimizing spacecraft reorientation maneuvers using a pseudospectral method." *Journal of Aerospace Engineering, Sciences and Applications*. Vol. 2, No. 1, 2010.

[68] Melton, R.G., "Hybrid methods for determining time-optimal, constrained spacecraft reorientation maneuvers." *Acta Astronautica*. Vol. 94, 2014, pp. 294–301.

[69] Louembet, C., Cazaurang, F., Zolghadri, A., Charbonnel, C., and Pittet, C., "Design of Algorithms for Satellite Slew manoeuver by Flatness and Collocation." *Proceedings of the 2007 American Control Conference*. New York, USA, July 11-13, 2007.

[70] Shuster, M.D., "A survey of Attitude Representations." *The Journal of Astronautical Sciences*. Vol. 41, No. 4, 1993, pp.439-517.

[71] Kim, M., Kim, M., and Shin, S., "A general Construction Scheme for Unit Quaternion Curves with Simple High Order Derivatives." *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*. ACM, New York, 1995, pp. 369-376.

[72] Boehm, W., Farin, G., and Kahmann, J., "A survey of curve and surface methods in CAGD." *Computer Aided Geometric Design*, Vol. 1, Issue 1, July 1984, pp. 1-60. DOI: 10.1016/0167-8396(84)90003-7

[73] Farin, G.E., "Curves and Surfaces for Computer-Aided Geometric Design: A practical Code." Academic Press, Inc. Orlando, FL, USA, 1996, 4th Ed.

[74] MatLab Version 2013b User manual, The Mathworks, Natick, MA, 2013.

[75] Rao, A.V., Benson, D.A., Darby, C.L., Patterson, M.A., Francolin, C., Sanders, I., and Huntington, G.T., "GPOPS: A MATLAB Software for Solving Multiple-Phase Optimal Control Problems using the Gauss Pseudospectral Method." *ACM Transaction on Mathematical Software*. Vol. 37, No. 2, 2010.

[76] Ventura, J., Ciarcia', M., Romano, M., and Walter, U., "An Inverse Dynamics-Based Trajectory Planner for Autonomous Docking to Tumbling Targets", *AIAA Guidane, Navigation and Control conference*, *SciTech 2016*, San Diego, USA.

[77] Ventura, J., Ciarcia', M., Romano, M., and Walter, U., "Fast and Near-Optimal Guidance for Docking to Uncontrolled Spacecraft," *Journal of Guidance, Control, and Dynamics,* to appear.

[78] Liu, H., Lai, X., Wu, W., "Time-optimal and jerk-continuous trajectory planning for robot manipulators with kinematics constraints," *Robotics and Computer-Integrated Manufacturing*, Vol. 19, Issue 2, 2013, pp. 309-317. doi: 10.1016/j.rcim.2012.08.002

[79] Gill, P. E., Murray, W., and Saunders, M.A., "SNOPT: an SQP algorithm for large-scale constrained optimization", *SIAM Journal of Optimization*, Vol. 12, 2002, pp. 979-1006.

[80] Betts, J.T., Gablonsky, J. M., "A comparison of Interior Point and SQP methods on optimal control problems," Technical Report, Phantom Works, Mathematics and Computing Technology, March 2002.

[81] Colodro, F., Torralba, A., and Laguna, M., "Continuous-Time Sigma-Delta Modulator with an Embedded Pulsewidth Modulation," *IEEE Transaction on Circuits and Systems*. Vol. 55, No 3, 2008, pp. 775-785

[82] Colodro, F., and Torralba, A., "Pulse-Width Modulation in Sigma-Delta Modulators," *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, Paris, France, pp. 1081-1084

[83] Holtz, J., "Pulsewidth modulation for electronic power conversion," *Proc. IEEE*, Vol. 82, No. 9, pp. 1194-1214, 1994.

[84] Rupp, T., Boge, T., Kiehling, R., and Sellmaier, F., "Flight dynamics challenges of the German on-orbit servicing mission DEOS," *21st International Symposium on Space Flight Dynamics*, Toulouse, France, 2009.

[85] Virgili, B.B., Lemmens, S., and Krag, H., "Investigation on Envisat attitude motion," https://indico.esa.int/indico/event/46/material/slides/6.pdf [retrieved 01 July 2014].

[86] "Envisat-1: Mission & System Summary", European Space Agency, March 1998.

[87] Matlab Version 2013b User manual, The Mathworks, Natick, MA, 2013.

[88] Vallado, D.A., *Fundamentals of Astrodynamics and Applications*, Microcosm Press/Springer, New York, 2007, Chap. 8.

[89] Patterson, M.A., and Rao, A.V., "GPOPS II: a MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using hp-adaptive Gaussian Quadrature Collocation Methods and Sparse Nonlinear Programming," *ACM Transactions on Mathematical Software,* Vol. 39, No. 3, Article 1, pp. 1-41, 2013. Doi: 10.1145/0000000.0000000

[90] Walter, U., *Astronautics, The Physics of Space Flight*, 2nd edition, Wiley-VCH, 2008, Chap. 15, pp. 536-538.

[91] Hillenbrand, U., and Lampariello, R., "Motion and Parameter Estimation of a Free-Floating Space Object from Range Data for Motion Prediction," *The 8th International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS)*, August 2005.

[92] Woodman, O.J., "An introduction to inertial navigation," Technical Report UCAM-CL-TR-696, University of Cambridge, Computer Laboratory, 2007.

[93] Tweddle, B. E., "Computer Vision-Based Localization and Mapping of an Unknown, Uncooperative and Spinning Target for Spacecraft Proximity Operations," Ph. D. Thesis, Dept. of Aeronautics and Astronautics, Massachusetts Inst. of Technology, Cambridge, MA, 2013.

[94] Yu, F., He, Z., Qiao, B., and Yu, X., "Stereo-Vision-Based Relative Pose Estimation for the Rendezvous and Docking on Noncooperative Satellites," *Mathematical Problems in Engineering*, Vol. 2014, Article ID 461283, pp 1-12, doi: 10.1155/2014/461283

[95] Christiansen, S., and Nilson, T., "Docking System Mechanism Utilized on Orbital Express Program," *Proceedings of the 39th Aerospace Mechanisms Symposium*, NASA Marshall Space Flight Center, May 2008.

[96] Ventura, J., Fleischner, A., and Walter, U., "Pose Tracking of a Noncooperative Spacecraft During Docking Maneuvers Using a Time-of-Flight Sensor," *AIAA Guidance, Navigation and Control Conference, AIAA SciTech*, San Diego, January 2016. (AIAA 2016-0875).

[97] Dissanayake, M.W.M.G., Newman, P., Clark S., and Durrant-Whyte, H.F., "A Solution to the Simultaneous Localization and Map Building (SLAM) Problem", *Trans. IEEE Robotics and Automation*, Vol. 17, No. 3, 2001

[98] Lachat, E., Macher, H., Mittet, M.A., Landes, T., and Grussenmeyer, P. "First experience with Kinect v2 sensor for close range 3d modelling," *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Vol. XL-5/W4, 2015, pp.93-100.

[99] Fankhauser, P. et al., "Kinect v2 for Mobile Robot Navigation: Evaluation and Modeling," *Proceedings of the 17th International Conference on Advanced Robotics*, July 2015.

[100] Forsyth. D.A., and Ponce, J., "Computer Vision, A Modern Approach," Prentice Hall, 2003.

[101] Keikkilä, J., and Silvén, O., "A Four-step Camera Calibration Procedure with Implicit Image Correction." *Proceedings of IEEE Computer Vision and Pattern Recognition*, 1997, pp.1106-1112.

[102] Conrady, A. E., "Decentred Lens-Systems," *Monthly notices of the Royal Astronomical Society*, Vol. 79 (1919), pp. 384–390.

[103] Brown, D.C. (May 1966). "Decentering distortion of lenses," *Photogrammetric Engineering*, Vol. 32 (3), pp. 444–462.

[104] Pomerleau, F., Colas, F., Siegwart, R., and Magnenat, S.,"Comparing ICP Variants on Real-World Data Sets," Autonomous Robots, Vol. 34, Issue 3, 2013, pp. 133-148

[105] Rusinkiewicz, S., and Levoy, M., "Efficient variants of the ICP algorithm." *Proceedings of the Third International Conference on 3-D Digital Imaging and Modeling*, 2001, pp. 145-152.

[106] Horn, B.K.P, "Closed-form solution of absolute orientation using unit quaternions," *Journal of the Optical Society of America*, Vol. 4, Issue 4, 1987, pp. 629-642. doi: 10.1364/JOSAA.4.000629

[107] Besl, P. J., and McKay, N.D., "A Method for Registration of 3-D Shapes," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 14, Issue 2, pp. 239–256. doi:10.1109/34.121791

[108] Fusiello, A., Trucco, E., and Verri, A., "A compact algorithm for rectification of stereo pairs", *Machine Vision and Applications*, Vol. 12, 2000, pp. 16–22.

[109] Hamzah, R.A., and Ibrahim, H., "Literature Survey on Stereo Vision Disparity Map Algorithms," *Journal of Sensors*, Vol. 2016, Article ID 8742920, pp. 1-23, 2016.

[110] Nixon, M.S., and Aguado, A.S., *Feature Extraction and Image Processing*, Academic Press, 2008.

[111] Harris, C., and Stephens, M., "A Combined Corner and Edge Detector," *Proceedings of 4th AlveyVision Conference*, 1988.

[112] N. Dey, P. Nandi, N. Barman, "A Comparative Study between Moravec and Harris Corner Detection of Noisy Images Using Adaptive Wavelet Thresholding Technique," *International Journal of Engineering Research and Applications (TJERA)*, Vol. 2, Issue 1, 2012.

[113] Greene, N., Kass, M., and Miller, G., "Hierarchical z-buffer visibility", *Proc. SIGGRAPH*, pp. 231-240, 1993.

[114] Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R., and Wu, A. Y., "An Optimal Algorithm for Approximate Nearest Neighbor Searching in Fixed Dimensions," *Journal of the ACM*, vol. 45, no. 6, 1994, pp. 891–923.

[115] Ostrowski, A.M., *Solutions of Equations and Systems of Equations*, 2nd ed., New York:Academic Press, 1966.

[116] Gao, W., Zhang, X., Yang, L., and Liu, H., "An improved Sobel edge detection," *3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT),* Vol. 5, 2010, pp. 67-71.

[117] Duda, R.O., Hart, P.E., "Use of the Hough Transformation to Detect Lines and Curves in Pictures," *Communication of the ACM*, 15, 1, January 1972.

[118] Lepetit, V., Moreno-Noguer, F., and Fua, P., "EPnP: An Accurate O(n) Solution to the PnP Problem," *International Journal of Computer Vision*, Vol. 81(2), Feb. 2009, pp. 155–166.

[119] Sharma, S., and D'Amico, S., "Comparative assessment of techniques for initial pose estimation using monocular vision," *Acta Astronautica*, Vol. 123, 2016, pp. 435-445.

[120] Julier, S.J., and Uhlmann, J. K., "New extension of the Kalman filter to nonlinear systems," *AeroSense'97*, International Society for Optics and Photonics, 1997.

[121] Harder, J., Wilde, M., and Fleischner, A., "Technology development for real-time teleoperated spacecraft mission operations," *Aerospace Conference, 2013 IEEE*. IEEE, 2013.

[122] Melton, R. G., "Time-explicit representation of relative motion between elliptical orbits," *Journal of Guidance, Control, and Dynamics*, Vol.*23, Issue* 24, 2000, pp. 604-610.